The Analysis of South African Household Survey Data using R

Takwanisa Machemedze

2018-07-19

Contents

Welcome

This guide is a translation of the Southern Africa Labour and Development Research Unit (SALDRU) online course, The Analysis of South African Household Survey Data, from Stata to R. You will learn how use the R statistical package to investigate interesting policy issues in South Africa utilizing real household survey data.

This document is a work in progress and we welcome your comments and feedback.

Suggested citation: Southern Africa Labour and Development Research Unit, University of Cape Town. *The Analysis of South African Household Survey Data.* Translated from Stata to R by T. Machemedze. DataFirst, University of Cape Town.

Chapter 1

Introduction

This guide is a translation of the Southern Africa Labour and Development Research Unit (SALDRU) course, The Analysis of South African Household Survey Data, from Stata to R statistical package. The original course provides an introduction to using Stata to investigate interesting policy issues in South Africa utilizing real household survey data. This guide therefore compliments the Stata course to help students and interested analysts on how to investigate policy issues using survey data with the help of the R statistical package. This document will be useful to:

1. Those who do not have access to one of the traditionally used statistical software such as Stata, SPSS or SAS (as R is **free** of charge).
2. Analysts who are already familiar with the aforementioned statistical software but who would like to use R for data alaysis.

Since this is a translation of Stata code to R code, this document is **NOT** a proper introduction to R. The structure of the original course makes it possible to introduce Stata while carrying out the analysis. The learning curve is however different for R and those new to R should use this document together with other comphrensive guides for the relevant topic or package functionality that are available online. Links to relevant guides about R resources are provided at the end of this document.

Since R is diverse in functions that can perform the same task, this document provides opinionated approaches to common operations followed when conducting data analysis: import datasets, do data exploration, do basic data manipulation operations where necessary, produce descriptive statistics and test hypotheses.

Examples provided in this guide, use the National Income Dynamics Study (NIDS) training dataset used in the Stata course. Note that this dataset is for training purposes only and if you want the proper NIDS dataset, you need to apply for the data on the DataFirst website

1.1 What is R?

R is often described as a free software environment for statistical computing and graphics. It is a dialect of the S programming language that was developed in the 1970s. R can be downloaded from the Comprehensive R Archive Network (CRAN) website free of charge. Installation instructions as well as guides, tutorials and FAQ are available on the same website.

R has become increasingly a popular tool for data analysis (see this article by Robert A. Muenchen) within the statistical community and the academic world for teaching. One of its strengths is that R is extensible to implement cutting-edge analyses that are not yet available in proprietary packages. Muenchen's article and another article by Revolutions Analytics, for example, shows the exponential growth in user-written packages. The R packages are the equivalent of Stata's ado files that one can install and use for their analysis.

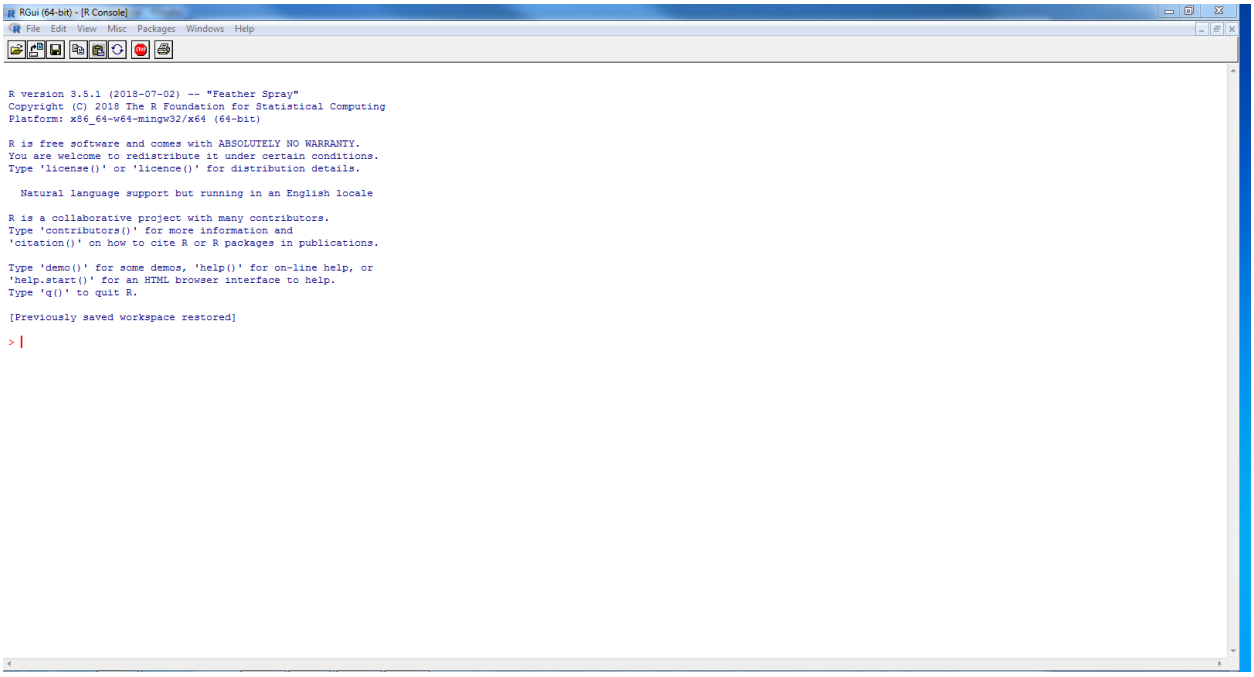


Figure 1.1: R interface

In addition to performing most of the tasks available in common statistical packages such as Stata, SPSS and SAS, R can be used for GIS map building, building interactive web applications and many more. R has multiple packages and functions to do the same analysis each with its own advantages and disadvantages. Therefore, users need to be flexible and to be aware of R's flexibility.

R, through various packages, interacts easily with data from several formats such as Stata, SPSS, SAS and many more. There are also R tools that supports several static and dynamic output formats that incudes HTML, PDF, MS Word, scientific articles, websites, and more.

A comprehensive introduction can be found on the R Foundation https://www.r-project.org/about.html.

Figure 1.1 shows a standard R interface that appears when the programme is launched.

1.2 Pros and cons of R

1.2.1 Pros

- R is free and open source software, allowing anyone to use and, importantly, to modify it.
  - R can be run and used for any purpose, commercial or non-commercial, profit or not-for-profit.
  - R's source code is freely available so you can study how it works and adapt it to your needs.
  - R is free to redistribute so you can share it with your friends.
  - R is free to modify and those modifications are free to redistribute and may be adopted by the rest of the community!
- R is a common tool among data experts at major universities.
  - Therefore, it reflects well on a very competent community of computational statisticians.
- R is cross-platform - runs on Linux, Mac, PC, server, desktop, etc.
- R connects easily with other languages - can import/export a variety of formats.
- R has active user groups to help with questions.

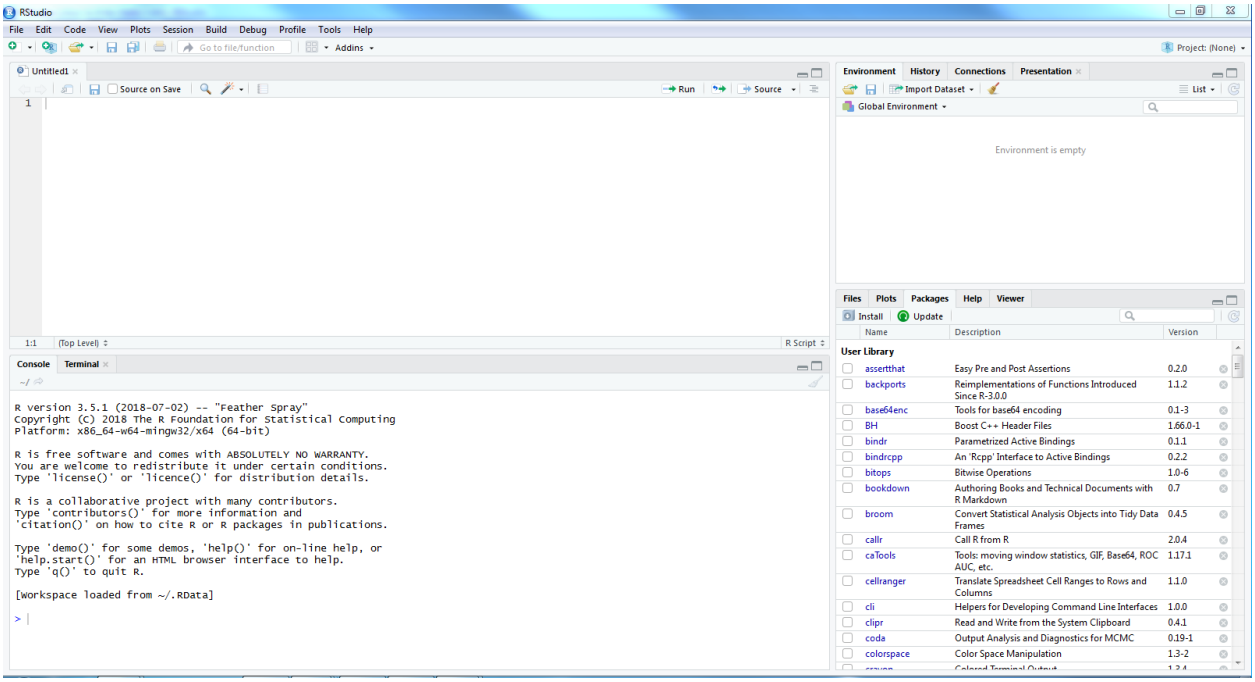


Figure 1.2: RStudio interface

- R supports extensions - developers can easily write their own software and distribute it in the form of add-on packages.
- R is flexible
  - can handle complex or standard statistical practices, bayesian modelling, GIS map building, building interactive web applications, and building interactive tests.

1.2.2 Cons

- R is based on S, which is now an old technology.
- R only has features that the community contributes.
  - If no one feels like implementing your favorite method, then it's your job!
- Not the ideal solution to all problems (drawback of all software packages).
- R is a programming language and not a software package - steeper learning curve.
- R can be much slower than compiled languages.

1.3 What is RStudio?

RStudio is an IDE (integrated Development Environment) which runs on top of R. R is installed as the underlying engine that powers RStudio's computations, while RStudio will provide command autocompletion, help files and an effective interface for getting things done quickly. RStudio provides rich user interface when compared to R and as a result has made R more accessible and easy to use. RStudio can be downloaded from the RStudio website: https://www.rstudio.com/.

Figure 1.2 shows shows a standard RStudio interface that appears when the programme is launched.

RStudio has mainly four panels with various tabs:

Top-left - where you type source code, like the script editor in Stata.

Bottom -left - is the console, where you see what R is doing.

Upper- right - where you see your list of R objects and things in your workspace (explained in the sections to follow). This is also where you can find your history of commands.

Bottom right is where you can see your file structure, graphical output, available packages, and help documents.

1.4 The R system

- The R system is divided into 2 conceptual parts:
  - The "**base**" R system that you download from CRAN.
  - User created packages that allows specialized statistical techniques
- R functionality is divided into a number of *packages*.
  - The "**base**" R system contains, among other things, the **base** package which is required to run R and contains the most fundamental functions.
  - The other packages contained in the "base" system include **utils, stats, datasets, graphics, grDevices, grid, methods, tools, parallel, compiler, splines, tcltk, stats4.**
  - There are also "Recommend" packages: **boot, class, cluster, codetools, foreign,KernSmooth, lattice, mgcv, nlme, rpart, survival, MASS, spatial, nnet, Matrix.**

1.4.1 Base system

The list of packages considered as `base` can be retrieved with some basic info calling:

```
#installed.packages(priority = "base")[, c(2, 3, 6)]
installed.packages(priority = "base")[, c(3, 6)] #less info than above to fit into window
```

```
##           Version Imports
## base      "3.5.1" NA
## compiler  "3.5.1" NA
## datasets  "3.5.1" NA
## graphics  "3.5.1" "grDevices"
## grDevices "3.5.1" NA
## grid      "3.5.1" "grDevices, utils"
## methods   "3.5.1" "utils, stats"
## parallel  "3.5.1" "tools, compiler"
## splines   "3.5.1" "graphics, stats"
## stats     "3.5.1" "utils, grDevices, graphics"
## stats4    "3.5.1" "graphics, methods, stats"
## tcltk     "3.5.1" "utils"
## tools     "3.5.1" NA
## utils     "3.5.1" NA
```

Chapter 2

Getting started with R

2.1 Introduction

This guide was prepared using RStudio and therefore you might need to launch RStudio if you want to follow the coding. **Note**: there are several other IDEs for R other than RStudio and you can as well use whichever IDE you are comfortable with.

Most R commands adopt the following syntax:

```
> command(argument1, argument2, ...,)
```

Any R command needs to be followed by brackets. Using RStudio, you can either type your commands at the console and press enter or type in the script editor (`Menu -> New File -> R Script`) and press the `Run` button to the top right corner of the same window. Alternatively, selected/highlighted parts of an R script file can be run pressing `Ctrl + Enter` and the whole script can be run by pressing `Ctrl + Alt + R`.

2.2 Organize your R session

2.2.1 Working directory

We check the current working directory by typing:

```
getwd()
```

```
## [1] "E:/stata2rcourse"
```

We set another working directory by typing:

```
setwd("path_to_directory"
```

Note that R uses forward slashes to separate directories or you can use double back slashes.

2.2.2 Load (and install) required libraries

To install a package, we type:

```
install.packages("package name")
```

Unlike in Stata where you just need to install an ado file and start running the programme, in R you need to load the package in each session where it is required. The command for loading a package is:

```
library(package name)
```

2.2.3 Comments

`#` denotes a comment. Anything after the `#` is not evaluated and ignored in R

2.3 Getting help

R has several help functions as described below:

- `help(solve)` or `?solve` - access help page for command `solve`.
  - useful only if you already know the name of the function that you wish to use.
- `apropos()` - searches for objects, including functions, directly accessible in the current R session that have names that include a specified character string. For example: `apropos("glm")`.
- `help.search("solve")` and `??solve` - scans the documentation for packages installed in your library for commands which could be related to string `solve`.
- `help.start()` - Start the hypertext (currently HTML) version of R's online documentation.
- `RSiteSearch()` - uses an internet search engine to search for information in function help pages. For example: `RSiteSearch("glm")`.
- `example(exp)` - examples for the usage of `exp`.
- `example("*")` - special characters have to be in quotation marks.
- Some packages provides a demonstration of the R functionality and typying `demo()` will return a list of all demos and the associated packages.
- For tricky questions, error messages and other issues, use Google (include "in R" to the end of your query).
- Thre is also a search engine just for `R`, RSeek.
- StackOverflow is also a great resource with many questions for many specific packages in R and a rating system for answers.

2.4 Output file

`sink()` function

For example:

```
sink(file="output_file_name.txt")
```

The `sink(file="output_file_name.txt")` function redirects output to the file `"output_file_name.txt"` and saves to your current working directory.By default, if the file already exists, its contents are overwritten. Include the option `append=TRUE` to append text to the file rather than overwriting it. Including the option `split=TRUE` will send output to both the screen and the output file. Issuing the command `sink()` without options will return output to the screen alone.

Unfortunately, compared to the Stata log file system, the `sink()` function only save the output, i.e. results with no code. It appears as if there is no straight forward way of having both "input" line and "output" results. One of the suggested solutions is to save output with `sink()` and input with `savehistory()` separately. We will show how to save command history in the section to follow.

2.5 Objects in R

It is possible and usual to have multiple pieces of data simultaneously open during an R session. These include datasets, summary tables, graphs, derived variables, functions, e.t.c. All these are referred to as objects. In fact, everything in R is an object and each can be manipulated independently in many ways.

For example, lets assign `x` to be 5 and `y` to be 3:

```
x<-5
y<-3
```

where `<-` (the less than sign plus the minus) is an assignment operator. `x` is assigned a value 5 and `y` a value 3.

`x` and `y` can be accessed by simply typing the object name:

```
x
```

```
## [1] 5
```

```
y
```

```
## [1] 3
```

A list of all objects loaded in the workspace can be retrieved by typing:

```
ls()
```

```
## [1] "x" "y"
```

There are several classes or types of objects in R which have distinct properties. By comparison, the Stata course introduced tabular datasets with variables and observations. There are also macros and scalars in Stata.

The R equivalent of a Stata dataset is called a `data frame`. This is the data structure we are going to be working with most of the time. A **factor** is another important data structure in R especially when working with survey data. Basically a factor is equivalent to a categorical variable (a variable with value labels) in Stata. Other common object classes include: vectors, matrices, lists, etc. Describing R objects and their properties (data types and structures) is well beyond the purpose of this guide and users interested should consult the online documentation for further explanations.

2.5.1 Removing objects from the R environment - rm() function

All R objects are stored in the memory of the computer, limiting the available space for calculation to the size of the **RAM** on your machine. To do more we need to learn how to manipulate the workspace and R makes organizing the workspace easy. The object `x` can be removed by typing:

```
rm(x)  # remove object x
```

Check:

```
ls()
```

```
## [1] "y"
```

We can see that we now only have `y` in the workspace.

To remove everything in in the workspace, you type:

```
rm(list=ls()) #or rm(object1,object2) to remove specific objects, object1 and object2
```

Check:

```
ls()
```

```
## character(0)
```

2.6 Saving in R

By default, R keeps track of all the commands you use in a session.You can browse the history from the command line by pressing the up-arrow and down-arrow keys. When you press the up-arrow key, you get the commands you typed earlier at the command line.

You can as well type the following commands to retrieve or save the history: (taken from statmethods website):

`history()` - work with previous commands and display last 25 commands

`savehistory(file="myfile")` - save your command history, default is ".Rhistory"

`loadhistory(file="myfile")` - recall your command history, default is ".Rhistory"

Objects in the current workspace can also be saved and loaded using the following commands:

`save.image()` - save the workspace to the file .RData in the current working directory

`save(objectlist,file="myfile.RData")` - save specific objects to a file, if you don't specify the path, the current working directory is assumed

`load("myfile.RData")` - load a workspace into the current session if you don't specify the path, the current working directory is assumed

You can quit the current workspace by simply typing:

`q()` - to quit R. You will be prompted to save the workspace. The are also options to tell R to/not to save, check help results for `?q()`

2.7 Importing NIDS data into R

In this document, we use a National Income Dynamics Study (NIDS) training dataset used in the corresponding Stata course. Therefore it is important to note that this is **NOT** the NIDS dataset that can be used to do serious analysis. If you need proper NIDS data, you need to apply for the data on the DataFirst website.

DataFirst provides microdata mainly in SAS, SPSS and Stata formats. It is possible to import either of these formats into R. For purposes of this course, we will import the Stata files with extension ".dta" and it is not difficult to import the other formats.

In order to import the data file, we need to employ the services of the `foreign` package (R Core Team 2017). The foreign package has functions to import, for example, Stata (`read.dta()`), SPSS (`read.spss()`) and other formats.

It is important to note that the `foreign` package can only read Stata files up to version 12. However, `haven` (Wickham and Miller 2018), a recently new package provides an alternative way to import Stata (up to version 14), SAS and SPSS data files into R.

Before we call functions from the `foreign` package (R Core Team 2017), we first need to load the package.

```
#install.packages("foreign") # to install if not installed
library(foreign)
```

The data is in a subfolder of the current working directory called "data".

```
nids<-read.dta("./data/nids.dta", convert.factors=FALSE)
```

You can check arguments of the `read.dta()` function by typing `?read.dta`. Note that I have instructed R not to use Stata value labels to create factors. The reason for doing this and for this specific dataset is that factors in R have underlying values which are natural numbers (1,2,3,...). NIDS used negative numbers for different missing types, e.g., -9 (Don't Know), -8 (Refused) and -3 (Missing). Setting `convert.factors=TRUE` or the default will not necessarily lead to losing data but the aforementioned value labels will be forced to be 1 (Don't Know), 2 (Refused) and 3 (Missing), i.e. natural numbers starting from the smallest. A factor is a very special and sometimes frustrating data type in R. While factors in R are comparable to categorical variables in Stata, the two programmes handle them differently. Factors in R look (and often behave) like character vectors but they are actually integers under the hood, and you need to be careful when treating them like strings or when converting to numeric.

The NIDS dataset is loaded in the workspace as a `nids` data frame, i.e. we have assigned (`nids<-`) the imported object to `nids` but you can give it any other name. Lets explore the data to get a better understanding.

2.8 Exploring the data

`dim()` - We can use the base R `dim()` function to get dimensions of tabular data by typing:

```
dim(nids)
```

```
## [1] 31170  2089
```

The dimensions are the number of rows and number of columns respectively. One can see that there are 31170 observations (rows) and 2089 variables (columns) in the dataset.

2.8.1 Subsetting data - part 1

Here we introduce the concept of subsetting data in R. To subset is to select specific elements of a data set. Since we have too many variables and we do not know anything about them yet, I will start by introducing the square bracket or matrix index notation.

We have a `nids` data frame with rows (observations) and columns (variables). Therefore, elements of nids are in the form:

```
nids[rows,columns]
```

if you want to think of this in matrix notation. The comma is crucial here since the dataset is organized by rows and columns.

Each row and column can be identified by its number between the square brackets. For example, if we type:

```
nids[2,7]
```

```
## [1] 1
```

we will get the value of the second observation of the 7th column in the data.

We can as well type:

```
nids[1:10,1:8]
```

```
##      hhqi   hhid    pid w1_r_b1 w1_r_phase w1_r_b1_1 w1_r_b3 w1_r_b4
## 1  201020 103034 301011       1          1         1       1       2
## 2  201015 103037 301012       1          1         1       1       2
## 3  201015 103037 301013       3          1         1       4       1
## 4  201015 103037 301014       2          1         1       8       2
```

```
## 5  201019 103050 301015       2          1         1       3       2
## 6  201018 103046 301016       6          1         1       4       1
## 7  201018 103046 301017       2          1         1       1       1
## 8  201022 103049 301018       1          1         1       1       2
## 9  201022 103049 301019       2          1         1       4       2
## 10 201022 103049 301020       3          1         1       8       1
```

which returns the first 10 observations `1:10` and the first 8 columns `1:8`. The `:` is an integer sequence operator, i.e. for example, `1:3` returns integers 1,2,3.

It is not always the case that you want to select everything in the range of certain integer, row or columns, and therefore we use the `c()` function to specifify the rows or columns that we need. Here, `c()` creates a vector based on the numbers you input. For example `c(1,2,3)` is equivalent to `1:3` as decribed above. However, `c()` allows to input non-sequential integers.

For example:

`nids[c(1:5,8),]` - to select rows 1 to 5 and 8

`nids[,c(1,2,3,7,8)]` - to select columns 1,2,3,7 and 8

To do the above in one line:

```
nids[c(1:5,8),c(1,2,3,7,8)]
```

```
##     hhqi   hhid    pid w1_r_b3 w1_r_b4
## 1 201020 103034 301011       1       2
## 2 201015 103037 301012       1       2
## 3 201015 103037 301013       4       1
## 4 201015 103037 301014       8       2
## 5 201019 103050 301015       3       2
## 8 201022 103049 301018       1       2
```

Given that our dataset has too many variables to print to screen (2089 variables) and that we are now able to subset specific elements of our dataset, we introduce other functions.

names()

`names` returns the names of the variables in the selected data frame. For example, We can see `names` of the first eight variables (columns) by typing:

```
names(nids[,1:8])
```

```
## [1] "hhqi"       "hhid"       "pid"        "w1_r_b1"    "w1_r_phase"
## [6] "w1_r_b1_1"  "w1_r_b3"    "w1_r_b4"
```

head(), tail()

The R functions `head()` and `tail()` return the first 6 and last 6 parts of a vector, matrix, table and data frame. For our dataset:

```
head(nids[,1:8])
```

```
##     hhqi   hhid    pid w1_r_b1 w1_r_phase w1_r_b1_1 w1_r_b3 w1_r_b4
## 1 201020 103034 301011       1          1         1       1       2
## 2 201015 103037 301012       1          1         1       1       2
## 3 201015 103037 301013       3          1         1       4       1
## 4 201015 103037 301014       2          1         1       8       2
## 5 201019 103050 301015       2          1         1       3       2
## 6 201018 103046 301016       6          1         1       4       1
```

```
tail(nids[,1:8])
```

```
##         hhqi   hhid pid w1_r_b1 w1_r_phase w1_r_b1_1 w1_r_b3 w1_r_b4
## 31165 209107 105648  NA       4          1         2      13       1
## 31166 207825 103699  NA       2          1         2       4       1
## 31167 201494 102647  NA       1          1         2       2       1
## 31168 201530 101684  NA       4          1         2      12       2
## 31169 202052 111168  NA       1          1         2       1       1
## 31170 207775 104193  NA       7          1         2       4       2
```

str() - compactly display the internal structure of an R object. Perhaps the most useful diagnostic function in R. `str` is short for structure and you can use it on any object. The structure of the first 8 variables are as follows:

```
str(nids[,1:8])
```

```
## 'data.frame':    31170 obs. of  8 variables:
##  $ hhqi      : int  201020 201015 201015 201015 201019 201018 201018 201022 201022 201022 ...
##  $ hhid      : int  103034 103037 103037 103037 103050 103046 103046 103049 103049 103049 ...
##  $ pid       : int  301011 301012 301013 301014 301015 301016 301017 301018 301019 301020 ...
##  $ w1_r_b1   : int  1 1 3 2 2 6 2 1 2 3 ...
##  $ w1_r_phase: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ w1_r_b1_1 : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ w1_r_b3   : int  1 1 4 8 3 4 1 1 4 8 ...
##  $ w1_r_b4   : int  2 2 1 2 2 1 1 2 2 1 ...
```

We see that the subsetted dataset is a data frame with `31170` observations, `8` variables (as expected), their names, all of integer variable type (`int`) and the first 10 observations.

2.8.2 Data documentation - value and variable labels

So far we have managed to import the data and view subset of its elements. However, we do not know what each of the variable names are or what the numbers represent. This is where we need the documentation about the data or a codebook with a description of the variable names, variable description and value labels, e.t.c

You can find some of this information in the questionnaires provided. We have also included a codebook with a list of variable names and their description. The codebook was prepared using an R package called `memisc` from the imported stata file. The Stata course made it easy by using the Stata `lookfor` function, but for this guide, you need to search in the documents (questionnaires and codebook) for more information.

Note: NIDS data from the DataFirst website has consistent standard names that are slightly different from the ones in this training dataset.

2.8.3 Operators in R

As we use these commands, we will often want to use qualifiers and operators. By using these options, we can restrict the specified R command to a specific sub-population.

Qualifiers and operators add more detail to the data exploration commands that we will be using. For instance, what if we wanted to explore information specific to a particular group or person only? This is where the qualifier and operators are used. Using qualifiers and operators allows us to apply R commands to specific observations in the data. To make things a bit more clear, here are some examples:

2.8.4 Subsetting data - part 2

Here we introduce another subset notation, the `subset` function. The basic structure of subset is like this:

`subset(x, condition, select=c(var1, var2))`

where x is the original dataset you want to subset, condition is a condition on your rows, and select is the columns you want to keep.

For example, if we want to examine the age (`w1_r_best_age_yrs`), race (`w1_best_race`) and highest educational qualification achieved (`w1_r_b7`) by all those aged 90 and above, we type:

```
head(subset(nids,
            subset = w1_r_best_age_yrs>90,
            select = c(w1_r_best_age_yrs, w1_best_race, w1_r_b7)))
```

```
##      w1_r_best_age_yrs w1_best_race w1_r_b7
## 575                101            1      25
## 925                 94            1      25
## 1293                92            1      25
## 1844                95            1      25
## 1887                99            1      25
## 1905                93            1      25
```

Where:

`x` is `nids`;

`condition` or `subset` is `w1_r_best_age_yrs>90` and

variable `selection` is `c(w1_r_best_age_yrs, w1_best_race, w1_r_b7)`

2.9 Tidyverse

2.9.1 Brief introduction to tidyverse

So far we have introduced the `foreign` package to import data and performing some basic data exploration using some base R functions. There are other base R functions that can perform a number of other operations. However, as mentioned in the introduction, the power of R is also derived from packages that make some of the processes easier to follow and efficient.

Here, we introduce `tidyverse` (Wickham 2017), a collection of modern R packages that share common philosophies, embed best practices, and are designed to work together. The tidyverse is the collective name given to a suite of R packages designed mostly by Hadley Wickham. All tidyverse packages share an underlying design philosophy, grammar, and data structures. At the time of writing this document, members of the `tidyverse` (or individual packages) include: `broom`, `dplyr`, `forcats`, `ggplot2`, `haven`, `httr`, `hms`, `jsonlite`, `lubridate`, `magrittr`, `modelr`, `purrr`, `readr`, `readxl`, `stringr`, `tibble`, `rvest`, `tidyr`, `xml2`.

2.9.2 Core tidyverse packages

The core tidyverse packages and their main functions are:

- `ggplot2` - for data visualisation (Wickham, Chang, et al. 2018).
- `dplyr` - for data manipulation - the data wrangling workhorse (Wickham, François, et al. 2018).
- `tidyr` - for data tidying (Wickham and Henry 2018).

- `readr` - for data import (Wickham, Hester, and Francois 2017).
- `purrr` - for functional programming (Henry and Wickham 2018).
- `tibble` - for tibbles, a modern reimagining of data frames (Müller and Wickham 2018).

You can install `tidyverse` packages by typing:

2.9.3 Install and load tidyverse packages

To install `tidyverse packages`, we type:

```
install.packages("tidyverse")
```

and to load the packages:

```
library("tidyverse")
```

2.9.4 dplyr verbs

`dplyr` (Wickham, François, et al. 2018) is one of the most important packages for data manipulation and it implements five fundamental verbs for data wrangling:

`filter()` – rows by their values

`arrange()` – reorder rows

`select()` – pick and/or exclude columns

`mutate()` – create new columns

`summarize()` – collapse rows with summaries

And any of the above may be scoped over a group of rows with:

`group_by()` – define groups based on categorical variables

2.9.5 Using dplyr functions

Arguments:

- First argument is the data frame to work on
- Following arguments define what to do on which named columns in the data frame. Result is always another data frame:

2.9.6 Pipe operator %>%

Pipes takes the result of the left hand side (LHS) and pushes into the first argument of the right hand side (RHS):

`%>%` is the pipe "operator". It takes what is on the left hand side and puts it in the right hand side's function.

2.10 More data exploration

Here, we demonstrate how to perform some tasks using both base R and tidyverse functions. However, this guide will use more of the tidyverse functions as we explore further.

The tidyverse packages are already installed and we load as follows:

```
library(tidyverse)
```

```
## -- Attaching packages ------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

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

For now you can ignore the warnings and messages.

2.10.1 Example 1: Subset variables

Our first task is to view the first few elements for age (`w1_r_best_age_yrs`), race (`w1_best_race`) and highest educational qualification (`w1_r_b7`) for every respondent in the dataset. We are going to use the `head()` function on a subset of the variables mentioned.

Option 1 - matrix index

```
head(nids[,c('w1_r_best_age_yrs', 'w1_best_race', 'w1_r_b7')])
```

```
##   w1_r_best_age_yrs w1_best_race w1_r_b7
## 1                -9            4      12
## 2                66            4      10
## 3                26            4       9
## 4                78            4       7
## 5                46            4      10
## 6                -9            4      12
```

Option 2 - subset

```
head(subset(nids, select=c(w1_r_best_age_yrs, w1_best_race, w1_r_b7)))
```

```
##   w1_r_best_age_yrs w1_best_race w1_r_b7
## 1                -9            4      12
## 2                66            4      10
## 3                26            4       9
## 4                78            4       7
## 5                46            4      10
## 6                -9            4      12
```

Option 3 - dplyr

```
nids %>%
  select(w1_r_best_age_yrs, w1_best_race, w1_r_b7) %>% # pick columns
  head()
```

```
##   w1_r_best_age_yrs w1_best_race w1_r_b7
```

```
## 1                -9           4       12
## 2                66           4       10
## 3                26           4        9
## 4                78           4        7
## 5                46           4       10
## 6                -9           4       12
```

The `dplyr` option takes the object `nids` then (`%>%`) select the variables then (`%>%`) list the first few observations

2.10.2 Example 2: Subset observations

Lets view contents of first few observations where race (`w1_best_race`) equals 3; the number 3 in this case refers to Indians, thus only information for Indian respondents will be displayed.

For this subset, we will suppress the command since it will print all variables to screen and fill several pages.

Option 1 - matrix index

```
#head(nids[nids$w1_best_race==3,])
```

The `$` sign allows you to extract elements of the data frame by name, i.e., `dataframe_name$column_name`.

Option 2 - subset

```
#head(subset (nids, subset=w1_best_race == 3))
```

Option 3 - dplyr

```
#nids %>%
#  filter(w1_best_race == 3) %>% # filter rows by values
#  head()
```

2.10.3 Example 3: Subset both observations and variables

In this example, we want to print all the observations for the race variable where age of the individual is greater than 90.

Option 1 - matrix index

```
nids[nids$w1_r_best_age_yrs > 90,c("w1_best_race")]
```

```
##  [1]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
## [24]  1  2  2  1  1  1  1  1  1  1  1  1  2  1  2  1  1  1  3  4  2  2 NA
## [47] NA
```

Option 2 - subset

```
subset(nids, subset=w1_r_best_age_yrs > 90,select=c(w1_best_race))
```

```
##       w1_best_race
## 575              1
## 925              1
## 1293             1
## 1844             1
## 1887             1
## 1905             1
## 3031             1
## 3300             1
## 3332             1
```

```
## 3430             1
## 4362             1
## 4954             1
## 6625             1
## 6627             1
## 7261             1
## 7734             1
## 7945             1
## 9203             1
## 9513             1
## 11194            1
## 11415            1
## 11593            1
## 12309            1
## 12374            1
## 12673            2
## 12726            2
## 12749            1
## 12930            1
## 12972            1
## 13849            1
## 13890            1
## 14469            1
## 14625            1
## 26620            1
## 26768            1
## 26907            2
## 27285            1
## 27394            2
## 27404            1
## 27587            1
## 27645            1
## 27792            3
## 27799            4
## 28117            2
## 28195            2
## 29246           NA
## 30220           NA
```

Option 3 - dplyr

```
nids %>%
  filter(w1_r_best_age_yrs > 90) %>%
  select(w1_best_race)
```

```
##    w1_best_race
## 1             1
## 2             1
## 3             1
## 4             1
## 5             1
## 6             1
## 7             1
## 8             1
## 9             1
```

```
## 10            1
## 11            1
## 12            1
## 13            1
## 14            1
## 15            1
## 16            1
## 17            1
## 18            1
## 19            1
## 20            1
## 21            1
## 22            1
## 23            1
## 24            1
## 25            2
## 26            2
## 27            1
## 28            1
## 29            1
## 30            1
## 31            1
## 32            1
## 33            1
## 34            1
## 35            1
## 36            2
## 37            1
## 38            2
## 39            1
## 40            1
## 41            1
## 42            3
## 43            4
## 44            2
## 45            2
## 46           NA
## 47           NA
```

When you try the above commands, you will note that two of the observations for race are recorded as a missing value, NA. If you wanted to list the race of all individuals older than 90 years old and did not want to list those for whom race was missing, you could type:

Option 1 - matrix index

```
nids[nids$w1_r_best_age_yrs > 90 & !is.na(nids$w1_best_race),c("w1_best_race")]
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1 1
## [36] 2 1 2 1 1 1 3 4 2 2
```

Option 2 - subset

```
subset(nids, subset=w1_r_best_age_yrs > 90 & !is.na(w1_best_race),select=c(w1_best_race))
```

```
##       w1_best_race
## 575              1
## 925              1
```

```
## 1293             1
## 1844             1
## 1887             1
## 1905             1
## 3031             1
## 3300             1
## 3332             1
## 3430             1
## 4362             1
## 4954             1
## 6625             1
## 6627             1
## 7261             1
## 7734             1
## 7945             1
## 9203             1
## 9513             1
## 11194            1
## 11415            1
## 11593            1
## 12309            1
## 12374            1
## 12673            2
## 12726            2
## 12749            1
## 12930            1
## 12972            1
## 13849            1
## 13890            1
## 14469            1
## 14625            1
## 26620            1
## 26768            1
## 26907            2
## 27285            1
## 27394            2
## 27404            1
## 27587            1
## 27645            1
## 27792            3
## 27799            4
## 28117            2
## 28195            2
```

Option 3 - dplyr

```
nids %>%
  filter(w1_r_best_age_yrs > 90 & !is.na(w1_best_race)) %>%
  select(w1_best_race)
```

```
##    w1_best_race
## 1             1
## 2             1
## 3             1
## 4             1
```

```
## 5              1
## 6              1
## 7              1
## 8              1
## 9              1
## 10             1
## 11             1
## 12             1
## 13             1
## 14             1
## 15             1
## 16             1
## 17             1
## 18             1
## 19             1
## 20             1
## 21             1
## 22             1
## 23             1
## 24             1
## 25             2
## 26             2
## 27             1
## 28             1
## 29             1
## 30             1
## 31             1
## 32             1
## 33             1
## 34             1
## 35             1
## 36             2
## 37             1
## 38             2
## 39             1
## 40             1
## 41             1
## 42             3
## 43             4
## 44             2
## 45             2
```

In the above, `is.na(x)` returns a logical value that is `TRUE` if x is missing and FALSE otherwise. By the same reasoning, `!is.na(x)` returns a logical value that is `TRUE` if `x` is not (`!`) missing and `FALSE` otherwise.

With NIDS being the National Income Dynamics Survey, there are obviously a large amount of variables dealing with various forms of income. We will be dealing mainly with the variable for imputed total household monthly income, `w1_hhincome`. This variable is calculated by aggregating all forms of income from the adult questionnaire. How do you think you would get R to list the incomes and races of all those households that earn over R75 000 per month as well as listing their race and age?

From now onwards, we will adopt `dplyr` functions to manipulate the data. The `dplyr` functions offers a more intuitive way of writing code by using the pipe `%>%` operator. To extract the income information, we can type:

```
nids %>%
  filter(w1_hhincome> 75000) %>%
```

```
  select(w1_best_race, w1_r_best_age_yrs, w1_hhincome)
```

```
##    w1_best_race w1_r_best_age_yrs w1_hhincome
## 1             4                57    87082.52
## 2             4                46    87082.52
## 3             1                29    89981.23
## 4             3                18    79500.00
## 5             3                56    79500.00
## 6             1                23   100070.00
## 7             1                39    93800.00
## 8             4                36   130000.00
## 9             1                58    84200.00
## 10            1                23    84200.00
## 11            1                50    84200.00
## 12            1                21    84200.00
## 13            1                65    89981.23
## 14            1                63    89981.23
## 15            1                21   100070.00
## 16            4                19   112000.00
## 17            4                58   112000.00
## 18            4                41   102033.33
## 19            4                17   102033.33
## 20            4                45   102033.33
## 21            1                17    89981.23
## 22            1                29    89981.23
## 23            4                70   137239.22
## 24            4                65   137239.22
## 25            3                44    79500.00
## 26            3                17    79500.00
## 27            4                52    86500.00
## 28            3                15    79500.00
## 29            4                21    77397.88
## 30            4                19    77397.88
## 31            4                41    84066.66
## 32            4                44    77397.88
## 33            4                44    77397.88
## 34            4                24    77397.88
## 35            4                54   106605.23
## 36            2                68   106605.23
## 37            4                57   106605.23
## 38            4                39    84066.66
## 39            4                57    99000.00
## 40            3                52    99000.00
## 41            4                56    75908.09
## 42            4                14   130000.00
## 43            4                14   102033.33
## 44            4                12   130000.00
## 45            4                10    84066.66
## 46            4                 4    84066.66
## 47            4                 5    75908.09
## 48            4                13    77397.88
## 49            3                14    99000.00
## 50            1                 0    89981.23
## 51            1                 5    89981.23
```

```
## 52            1                15    84200.00
## 53            4                61    75908.09
## 54            4                57    86500.00
## 55            4                59   112000.00
```

We see that there are some 4 and 5 years olds listed in the output. Why do you think this is? It can often be useful to also include the household identifier variable to see which respondents come from the same household. Use the "up-arrow" key to include the household identifier variable, `hhid`, in the variables that are listed.

```
nids %>%
  filter(w1_hhincome> 75000) %>%
  select(hhid,w1_best_race, w1_r_best_age_yrs,w1_hhincome)
```

```
##      hhid w1_best_race w1_r_best_age_yrs w1_hhincome
## 1  107006            4                57    87082.52
## 2  107006            4                46    87082.52
## 3  113860            1                29    89981.23
## 4  170283            3                18    79500.00
## 5  170283            3                56    79500.00
## 6  107242            1                23   100070.00
## 7  112961            1                39    93800.00
## 8  170611            4                36   130000.00
## 9  105215            1                58    84200.00
## 10 105215            1                23    84200.00
## 11 105215            1                50    84200.00
## 12 105215            1                21    84200.00
## 13 113860            1                65    89981.23
## 14 113860            1                63    89981.23
## 15 107242            1                21   100070.00
## 16 170533            4                19   112000.00
## 17 170533            4                58   112000.00
## 18 107067            4                41   102033.33
## 19 107067            4                17   102033.33
## 20 107067            4                45   102033.33
## 21 113860            1                17    89981.23
## 22 113860            1                29    89981.23
## 23 170593            4                70   137239.22
## 24 170593            4                65   137239.22
## 25 170283            3                44    79500.00
## 26 170283            3                17    79500.00
## 27 175112            4                52    86500.00
## 28 170283            3                15    79500.00
## 29 109426            4                21    77397.88
## 30 109426            4                19    77397.88
## 31 171093            4                41    84066.66
## 32 109426            4                44    77397.88
## 33 109426            4                44    77397.88
## 34 109426            4                24    77397.88
## 35 109253            4                54   106605.23
## 36 109253            2                68   106605.23
## 37 109253            4                57   106605.23
## 38 171093            4                39    84066.66
## 39 101257            4                57    99000.00
## 40 101257            3                52    99000.00
```

```
## 41 109186            4                56    75908.09
## 42 170611            4                14   130000.00
## 43 107067            4                14   102033.33
## 44 170611            4                12   130000.00
## 45 171093            4                10    84066.66
## 46 171093            4                 4    84066.66
## 47 109186            4                 5    75908.09
## 48 109426            4                13    77397.88
## 49 101257            3                14    99000.00
## 50 113860            1                 0    89981.23
## 51 113860            1                 5    89981.23
## 52 105215            1                15    84200.00
## 53 109186            4                61    75908.09
## 54 175112            4                57    86500.00
## 55 170533            4                59   112000.00
```

You should find that members of the same household are listed directly above one another. This is because the data has been sorted according to the household identifier. We can choose to change the way in which the data is sorted using the `arrange` command. For instance, we may wish to have the data ordered according to income.

```
nids<-nids %>% arrange(w1_hhincome)
```

The above command assign `nids` to be `nids` sorted by `w1_hhincome`.

```
nids %>%
  filter(w1_hhincome> 75000) %>%
  select(hhid,w1_best_race, w1_r_best_age_yrs,w1_hhincome)
```

```
##      hhid w1_best_race w1_r_best_age_yrs w1_hhincome
## 1  109186            4                56    75908.09
## 2  109186            4                 5    75908.09
## 3  109186            4                61    75908.09
## 4  109426            4                21    77397.88
## 5  109426            4                19    77397.88
## 6  109426            4                44    77397.88
## 7  109426            4                44    77397.88
## 8  109426            4                24    77397.88
## 9  109426            4                13    77397.88
## 10 170283            3                18    79500.00
## 11 170283            3                56    79500.00
## 12 170283            3                44    79500.00
## 13 170283            3                17    79500.00
## 14 170283            3                15    79500.00
## 15 171093            4                41    84066.66
## 16 171093            4                39    84066.66
## 17 171093            4                10    84066.66
## 18 171093            4                 4    84066.66
## 19 105215            1                58    84200.00
## 20 105215            1                23    84200.00
## 21 105215            1                50    84200.00
## 22 105215            1                21    84200.00
## 23 105215            1                15    84200.00
## 24 175112            4                52    86500.00
## 25 175112            4                57    86500.00
## 26 107006            4                57    87082.52
```

```
## 27 107006           4              46   87082.52
## 28 113860           1              29   89981.23
## 29 113860           1              65   89981.23
## 30 113860           1              63   89981.23
## 31 113860           1              17   89981.23
## 32 113860           1              29   89981.23
## 33 113860           1               0   89981.23
## 34 113860           1               5   89981.23
## 35 112961           1              39   93800.00
## 36 101257           4              57   99000.00
## 37 101257           3              52   99000.00
## 38 101257           3              14   99000.00
## 39 107242           1              23  100070.00
## 40 107242           1              21  100070.00
## 41 107067           4              41  102033.33
## 42 107067           4              17  102033.33
## 43 107067           4              45  102033.33
## 44 107067           4              14  102033.33
## 45 109253           4              54  106605.23
## 46 109253           2              68  106605.23
## 47 109253           4              57  106605.23
## 48 170533           4              19  112000.00
## 49 170533           4              58  112000.00
## 50 170533           4              59  112000.00
## 51 170611           4              36  130000.00
## 52 170611           4              14  130000.00
## 53 170611           4              12  130000.00
## 54 170593           4              70  137239.22
## 55 170593           4              65  137239.22
```

We can see that the level of income now increases as we scroll down the list. However, notice that members of the same household are still next to one another. This is because the income variable we are using is a household level variable and therefore has the same value for every member of the household.

Let's see if you have gotten the hang of this, try these quick exercises:

Exercises

1. What is the hhid value for the 1000th observation?

Question 1 Answer

2. What are the ages of the respondents with an hhid equal to 42011?

Question 2 Answer

3. How many 50 year old Indian respondents are there in the data? (Be careful this one is a bit tougher than the others)

Question 3 Answer

2.11 Managing the data

In this section, we'll learn some data management commands. In many circumstances, we will want to amend the original NIDS data set. We might want to add new variables that we create from the existing variables, we might want to drop variables that we will never use to free up memory, we might want to recode missing values, and/or we might want to create our own variable labels. While we will not deal with some of the

trickier data management issues (such as merging data sets) in this section, you will learn enough to get started.

We'll start by creating a new variable. To create a new variable in R using `dplyr` functions you use the `mutate` command. Suppose we want to create a new variable called "temp" and we would like this variable to be equal to 1 for every observation in the dataset. To create this new variable you need to type:

```
nids<-nids %>%
  mutate(temp=1)
```

Go ahead and enter this command into R. You will notice that our new variable temp has been added to the end of the variable list in the R variables window. How do we know for sure what this new variable is equal to? Did the command work correctly?

See if you can figure this out:

4. How can we check to make sure our new variable is equal to 1?

Question 4 Answer

5. How would you create a new variable called "temp2" that is equal to 50 for all of the observations in the data set?

Question 5 Answer

Now you may be asking yourself, why would I want to create a variable like temp that has the same value for each observation in the data set. Variables like temp can actually be quite useful at times, although a majority of the variables you create will not have the same value for all observations. Let's try another example. Suppose we want to create a variable for race that has only two values: white and non-white. This can be useful if we are investigating the consequences of past discrimination in South Africa. The variable `w1_best_race` in the NIDS data set has four values: 1 (African), 2 (Coloured), 3 (Asian/Indian) and 4 (White).

Our goal here is to have a variable `w1_best_race2` which is equal to 1 if the individual is white and equal to 2 if the individual is non-white. Here we suggest to use the `ifelse` command within the `mutate` function. The `ifelse` command works as follows:

```
ifelse(condition, value if condition is true, value if condition is false)
```

In the case of `w1_best_race2`:

```
nids<-nids %>%
  mutate(w1_best_race2=ifelse(w1_best_race==4,1,2))
```

We can tabulate results for the new variable and see the breakdown:

```
table(nids$w1_best_race2)
```

```
##
##     1     2
##  1432 26762
```

We can achieve the same results using base R functions.

```
nids$w1_best_race3<-NA # NA for all values of w1_best_race3
nids$w1_best_race3[nids$w1_best_race==4]<-1 # w1_best_race3 is 0 for all cases where w1_best_race==4
nids$w1_best_race3[nids$w1_best_race<4 & nids$w1_best_race>=1]<-2 # w1_best_race3 is 1 for all cases wh
table(nids$w1_best_race3)
```

```
##
##     1     2
##  1432 26762
```

The above example provides us with a good opportunity to introduce factor variables, the equivalent of Stata's categorical variables. Here, we want to assign the new variable, `w1_best_race2`, the respective value labels. We achieve this using the `factor` function:

```
nids<-nids %>%
  mutate(w1_best_race2=factor(w1_best_race2, levels=1:2, labels=c("White","Non-White")))
#nids$w1_best_race2<-factor(nids$w1_best_race2, levels=c(1,2), labels=c("White","Non-White"))
```

Tabulate labelled variable:

```
table(nids$w1_best_race2)
```

```
##
##     White Non-White
##      1432     26762
```

You will now notice that at the end of the Variable view window that our new variable `w1_best_race2` is listed. Alright now it's your turn. Try to answer the following questions:

6. How would you create a new variable called "head" that is equal to 1 if the individual is the resident head, and equal to 0 if the individual is not?

Question 6 Answer

~~**7. How would you label this new variable with the following label - "Resident head indicator"?**~~

Question 7 Answer

We can also generate variables using the mathematical operators. For example, if we wanted a variable for the total monthly household income from government we could add household monthly income from government grants (`w1_hhgovt`) and household monthly income from other government sources (`w1_hhother`) to create a new variable which we call `hhincome_govt`. Thus, we would type:

```
nids<-nids %>%
  mutate(hhincome_govt = w1_hhgovt + w1_hhother)
```

2.12 Worked example: creating a bmi variable

In this worked example, our aim is to create a body mass index (BMI) variable. The Body Mass Index (BMI) is a measure of the nutritional status of adults and is calculated by dividing weight in kilograms by the square of height in metres. As such, we first need to find and explore the height and weight variables in the dataset. Let's start with height.

You will see in the metadata document that there are 3 height variables for children and 3 height variables for adults. You should turn to the metadata and questionnaires to investigate why this is the case. One of the reasons for repeated measurement is to try to minimize the chance that heights are measured incorrectly. Ideally, we would collate the information contained in the three measures to obtain a variable containing the best estimate of every individual in the dataset. Here, we will instead use the first measure (`w1_a_n1_1`) as a crude estimate of the height of adults in the sample.

Let's have a look at this variable:

First six observations

```
nids %>%
  select(w1_a_n1_1) %>%
  head()
```

```
##   w1_a_n1_1
```

```
## 1     161.0
## 2     156.2
## 3     160.0
## 4     176.2
## 5     165.0
## 6     158.2
```

Last six observations

```
nids %>%
  select(w1_a_n1_1) %>%
  tail()
```

```
##       w1_a_n1_1
## 31165        NA
## 31166     187.1
## 31167        NA
## 31168        NA
## 31169      -8.0
## 31170      -8.0
```

Did you notice that the heights are measured in centimeters? In order to calculate BMI, we will need heights in meters. Also, since BMI is a ratio of weight to height squared, we will not want to include the negative nonresponse height or weight values. Lastly, BMI is only valid for people over the age of 20 years of age. Let's create a new height variable in a more useful form for our purposes:

```
nids<-nids %>%
  mutate(height = ifelse (w1_a_n1_1 >= 0 & w1_a_best_age_yrs >= 20, w1_a_n1_1/100, NA))
```

What is the height of the tallest person in our sample? What is the mean height of the sample?

```
summary(nids$height)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   0.453   1.553   1.610   1.614   1.680   2.074   19718
```

The `summary()` function is a generic function that recognizes objects belonging to different classes and treat them accordingly without returning an error message.

We see that applying the `summary()` function to a numerical variable above computes basic descriptive statistics (mean, median, quartiles, maximum and minimum). R automatically works out which type of summary is best suited to a given object class. Typing `summary(nids)` will return summary information about each variable in the dataset.

We can see that the range is [0.453, 2.074], which tells us that the shortest person in the sample has a reported height of less than half a metre, while the tallest person is 2.07 metres tall. The average height in the sample is 1.61 metres. We can get an idea of the distribution of heights in the sample by looking at the percentiles. Half the people in the sample are between 1.55 and 1.68 metres tall.

Returning to the shortest person in the population, recall that we have restricted the height variable to only include individuals over the age of 20. This means that a person who is 0.453 metres tall is either exceptionally short or that there was an error made in measurement or data capturing. Let's investigate people who have reported heights that are below 'normal'.

How many are there in the sample?

```
nids %>%
  filter(height<1) %>% #filter all height less than 1 meter
  select(w1_a_best_age_yrs, w1_a_b2, height) #pick to print data for these variables
```

```
##    w1_a_best_age_yrs w1_a_b2 height
## 1                 50       1  0.622
## 2                 22       2  0.637
## 3                 49       2  0.453
## 4                 48       2  0.994
## 5                 23       2  0.552
## 6                 26       2  0.641
## 7                 43       2  0.639
## 8                 34       1  0.654
## 9                 74       1  0.725
## 10                50       1  0.704
## 11                38       2  0.615
## 12                57       2  0.610
## 13                75       2  0.682
## 14                36       2  0.624
## 15                50       2  0.847
## 16                34       2  0.710
## 17                64       1  0.557
## 18                41       2  0.542
## 19                75       1  0.790
## 20                40       1  0.610
## 21                26       2  0.585
## 22                22       2  0.966
## 23                64       2  0.700
## 24                49       1  0.668
## 25                21       2  0.984
## 26                79       2  0.527
## 27                57       2  0.771
## 28                30       2  0.642
```

We see there are 28 people who are shorter than one meter tall in the sample. We also list the gender (`w1_a_b2`) and age of these individuals to give us more information to potentially identify a pattern. There does not appear to be a relationship with age. However, we see that the majority of individuals under one metre are females.

8. Create a new variable called "weight" that has no negative values and is missing for individual younger than 20. Investigate the weight distribution in the population as we have just done for the height distribution.

```
nids<-nids %>%
  mutate(weight = ifelse (w1_a_n2_1 >= 0 & w1_a_best_age_yrs >= 20, w1_a_n2_1, NA))

summary(nids$weight)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   19.20   57.20   66.80   69.59   79.60  150.00   19848
```

Generating a variable for BMI

The heights and weights of people in the South African population are interesting in themselves, but in order to assess the health of an individual we need a measure of weight given height. For instance, we might want to say that someone who weighs 100 kg and is 1.6 metres tall is overweight, while someone who weighs 100 kg, but is 2 metres tall is not. The Body Mass Index (BMI) is a commonly used measure of the relationship between weight and height and allows us to talk about whether an individual is overweight, underweight or of a healthy weight.

The BMI is calculated by dividing weight (kg) by height squared (m^2) for people over the age of 20 (i.e.

$BMI = \frac{weight}{height^2}$)

```
nids<-nids %>%
  mutate(bmi = weight/height^2)
```

Some points to note. Firstly, our `bmi` variable should have a missing value for individuals who are younger than 20 since both our `height` and `weight` variables are missing for these individuals. We will be ignoring them when we look at BMI in this course as calculating the BMI for individuals under the age of 20 involves a different formula.

It is always good to get a sense of what your new variable looks like. Have we coded it correctly? Does it contain the information we want it to contain? Is the variable missing for those who should not have a value and valid for those who should get a value?

Check that BMI is missing for people under age 20 and for those who have missing height for weight values.

```
nids %>%
  filter(w1_a_best_age_yrs<20) %>% # filter children younger than 20 years
  filter(!is.na(bmi)) %>% # filter children whose BMI is not missing
  summarise(n=n()) # count how many are they
```

```
##   n
## 1 0
```

```
nids %>%
  filter(is.na(height) | is.na(weight)) %>% #filter those with missing height and missing weight
  filter(!is.na(bmi)) %>% # filter not missing BMI
  summarise(n=n()) #count
```

```
##   n
## 1 0
```

How many respondents have non-missing BMI values?

```
nids %>%
  filter(!is.na(bmi)) %>%
  summarise(count = n())
```

```
##   count
## 1 11285
```

The World Health Organisation classifies a BMI under 18.5 as underweight, a BMI between 18.5 and 24.9 as normal, a BMI between 25 and 29.9 as overweight and a BMI of 30 or more as obese. In light of this, does our new variable appear to be reasonable?

In the later chapters of this series we will investigate the determinants of BMI and in doing so address the following questions. What happens to your BMI as you get older? Do richer people have a higher or lower BMI than middle income or poor people? Is the average BMI different across racial groups? But for now, try to put what you have learnt in this chapter into practice by completing the following question. Then use the exercises to revise the new tools you have learnt.

9. See if you can create a variable called `age_bins` that is equal to 1 if the individual is between ages 20 and 29, 2 for ages 30 - 39, 3 for ages 40 - 49, 4 for ages 50 - 59, 5 for ages 60 - 69, 6 for ages 70 - 120. Ensure that your variable is missing for other values of the age variable.

2.13 Question answers

Question 1 answer

What is the hhid value for the 1000th observation?

Answer:

```
nids[1000,c("hhid")]
```

```
## [1] 105886
```

The above command will display the value for the variable hhid for the 1000th observation in the data set. Now it is possible that you could get a different answer than above, it all depends on the order the observations have been saved in. If this is your first time using the NIDS data file then you should get the answer above.

Question 2 answer

What are the ages of the respondents with an hhid equal to 101041?

Answer:

8, 14, 15, 57, 59

```
nids[which(nids$hhid==101041),c("w1_r_best_age_yrs")]
```

```
## [1] 57 15 59 14  8
```

The above command displays the value for the variable `w1_r_best_age_yrs` if the variable `hhid` is equal to 101041.

Question 3 answer

How many 50 year old Indian respondents are there in the data?

Answer:

Two

```
nrow(nids[which(nids$w1_best_race == 3 & nids$w1_r_best_age_yrs==50),])
```

```
## [1] 2
```

```
#To see the hhid
nids[which(nids$w1_best_race == 3 & nids$w1_r_best_age_yrs==50),c("hhid")]
```

```
## [1] 170064 101113
```

Question 4 answer

How can we check to make sure our new variable is equal to 1?

Answer:

One way is to tabulate the new variable temp.

```
table(nids$temp)
```

```
##
##     1
## 31170
```

Question 5 answer

How would you create a new variable called "temp2" that is equal to 50 for all of the observations in the data set?

Answer:

```
nids<-nids %>%
  mutate(temp2=50)

#Or

nids$temp2<-50
```

Question 6 answer

How would you create a new variable called "head" that is equal to 1 if the individual is the resident head, and equal to 0 if the individual is not?

Answer:

The first step to creating the new variable head is to find a variable in the data set that will tell us who is and is not a resident head. To accomplish this, search for the variables and/or variable labels contain the key text "resident head" from the codebook.

The variable, `w1_r_b3` (Household member's relationship to Resident head), seems the most promising. To verify this we use the NIDS survey metadata pdf (page 56, contains the relevant question for variable `w1_r_b3` i.e. Wave 1, Full Roster Data, Question B3). It becomes clear after viewing the codelist relating to this question (Appendix A – Table 1 on page 120) that `w1_r_b3` is the variable we want. From this codelist we see that a resident head is given a value of 1, thus when the variable `w1_r_b3` is equal to 1, that individual is a resident head.

With this knowledge, we are able to create the new variable.

```
nids<- nids%>%
  mutate(head=ifelse(w1_r_b3 == 1,1,0))
#nids$head=ifelse(nids$w1_r_b3 == 1,1,0)
```

Here we used the `ifelse` command where if the condition `w1_r_b3 == 1` is met, the new variable is equal to 1 and zero for everyone else.

Question 7 answer

How would you label this new variable with the following label "Resident head indicator"?

Answer:

Not applicable

Question 8 answer

Answer:

Create a new variable called "weight" that has no negative values and is missing for individual younger than 20. Investigate the weight distribution in the population as we have just done for the height distribution.

The weight variable is `w1_a_n2_1`.

```
nids<-nids %>%
  mutate(weight = ifelse (w1_a_n2_1 >= 0 & w1_a_best_age_yrs >= 20, w1_a_n2_1, NA))

summary(nids$weight)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   19.20   57.20   66.80   69.59   79.60  150.00   19848
```

Question 9 answer

Create a variable called age_bins that is equal to 1 if the individual is between ages 20 and 29, 2 for ages 30 – 39, 3 for ages 40 – 49, 4 for ages 50 – 59, 5 for ages 60 – 69, 6 for ages 70 120. Ensure that your variable is missing for other values of the age variable.

Answer:

```
nids<-nids %>%
  mutate(age_bins=case_when(
    w1_a_best_age_yrs > 20 & w1_a_best_age_yrs<=29 ~ 1,
    w1_a_best_age_yrs > 29 & w1_a_best_age_yrs<=39 ~ 2,
    w1_a_best_age_yrs > 39 & w1_a_best_age_yrs<=49 ~ 3,
    w1_a_best_age_yrs > 49 & w1_a_best_age_yrs<=59 ~ 4,
    w1_a_best_age_yrs > 59 & w1_a_best_age_yrs<=69 ~ 5,
    w1_a_best_age_yrs > 69 & w1_a_best_age_yrs<=120 ~ 6)) %>%
  mutate(age_bins=factor(age_bins, labels = c("20-29","30-39","40-49","50-59","60-69","70-120")))

#OR
#nids$age_bins<-factor(cut(nids$w1_a_best_age_yrs, c(20,29,39,49,59,69,120),labels = c("20-29","30-39",

#OR
#nids$age_bins<-NA
# nids$age_bins[nids$w1_a_best_age_yrs>=20 & nids$w1_a_best_age_yrs<=29]<-1
# nids$age_bins[nids$w1_a_best_age_yrs>29 & nids$w1_a_best_age_yrs<=39]<-2
# nids$age_bins[nids$w1_a_best_age_yrs>39 & nids$w1_a_best_age_yrs<=49]<-3
# nids$age_bins[nids$w1_a_best_age_yrs>49 & nids$w1_a_best_age_yrs<=59]<-4
# nids$age_bins[nids$w1_a_best_age_yrs>59 & nids$w1_a_best_age_yrs<=69]<-5
# nids$age_bins[nids$w1_a_best_age_yrs>69 & nids$w1_a_best_age_yrs<=120]<-6
```

```
nids%>%
  group_by(age_bins)%>%
  summarise(count = n())
```

```
## # A tibble: 7 x 2
##   age_bins count
##   <fct>    <int>
## 1 20-29     3525
## 2 30-39     2921
## 3 40-49     2553
## 4 50-59     2018
## 5 60-69     1299
## 6 70-120     955
## 7 <NA>     17899
```

2.14 Exercises

Using R and the NIDS dataset, answer the following questions.

1. How many households are in the NIDS data file?

Exercise 1 Answer

2. What are some variables in the data set that are related to food?

Exercise 2 Answer

3. How old is the oldest person in the data set?

Exercise 3 Answer

4. How many 50 year old women are in the data set?

Exercise 4 Answer

5. How would you create a new variable that is equal to 1 for Africans and 0 for non-Africans?

Exercise 5 Answer

6. How would you drop every variable from the data set except the new variable you just created?

Exercise 6 Answer

2.15 Exercise answers

Exercise 1 answer

How many households are in the NIDs data file?

Answer: 7305

The first step is to determine what variable uniquely identifies each household. From the labels in the codebook, it is clear that the variable `hhid` is the variable needed as it is the "household identifier". We can confirm this by using the NIDS survey. One way to obtain the number of unique `hhid` values, is to use the commands `unique` and `length`:

```
length(unique(nids$hhid))
```

```
## [1] 7305
```

Exercise 3 answer

How old is the oldest person in the data set?

Answer:

```
max(nids$w1_r_best_age_yrs, na.rm=TRUE)
```

```
## [1] 105
```

Exercise 4 answer

How many 85 year old women are in the data set?

There are 10 women who are 85 years old in the data set. To obtain the answer to question 4, you could have used the following commands:

```
nrow(nids[which(nids$w1_r_best_age_yrs == 85 & nids$w1_r_b4 == 2),])
```

```
## [1] 10
```

```
#OR

nids %>%
  filter(w1_r_best_age_yrs == 85 & w1_r_b4 == 2) %>%
  nrow
```

```
## [1] 10
```

Exercise 5 answer

How would you create a new variable that is equal to 1 for Africans and 0 for non-Africans?

The first step in creating the new variable is to determine which value of the variable race identifies Africans. From the NIDS survey we know that the value 1 identifies an African respondent. With this knowledge, we are able to create the new variable.

```
nids<-nids %>%
  mutate(newvar=ifelse(w1_best_race==1,1,2))
```

Exercise 6 answer

How would you drop every variable from the data set except the new variable you just created?

```
#nids<-data.frame(nids$newvar)

#Or

#nids<-nids %>%
#  select(newvar)
```

2.16 Session information

The output below shows what R packages and versions where used.

```
print(sessionInfo(), locale = FALSE)
```

```
## R version 3.5.1 (2018-07-02)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 7 x64 (build 7601) Service Pack 1
##
## Matrix products: default
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] bindrcpp_0.2.2  forcats_0.3.0   stringr_1.3.1   dplyr_0.7.6
##  [5] purrr_0.2.5     readr_1.1.1     tidyr_0.8.1     tibble_1.4.2
##  [9] ggplot2_3.0.0   tidyverse_1.2.1 foreign_0.8-70
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_0.2.4 xfun_0.2         reshape2_1.4.3   haven_1.1.2
##  [5] lattice_0.20-35  colorspace_1.3-2 htmltools_0.3.6  yaml_2.1.19
##  [9] utf8_1.1.4       rlang_0.2.1      pillar_1.2.3     withr_2.1.2
## [13] glue_1.2.0       modelr_0.1.2     readxl_1.1.0     bindr_0.1.1
## [17] plyr_1.8.4       munsell_0.5.0    gtable_0.2.0     cellranger_1.1.0
## [21] rvest_0.3.2      psych_1.8.4      evaluate_0.10.1  knitr_1.20
## [25] parallel_3.5.1   broom_0.4.5      Rcpp_0.12.17     backports_1.1.2
## [29] scales_0.5.0     jsonlite_1.5     mnormt_1.5-5     hms_0.4.2
## [33] digest_0.6.15    stringi_1.1.7    bookdown_0.7     grid_3.5.1
## [37] rprojroot_1.3-2  cli_1.0.0        tools_3.5.1      magrittr_1.5
## [41] lazyeval_0.2.1   crayon_1.3.4     pkgconfig_2.0.1  xml2_1.2.0
## [45] lubridate_1.7.4  assertthat_0.2.0 rmarkdown_1.10   httr_1.3.1
## [49] rstudioapi_0.7   R6_2.2.2         nlme_3.1-137     compiler_3.5.1
```

Chapter 3

Understanding distributions

3.1 Introduction

Now that we have acquired some basic R skills, we are ready to begin analysing the data. The immediate problem is – Where do we begin? As we have seen, there are an enormous number of variables and observations at our disposal although none of this information in its raw form is especially useful. How can we summarize this massive amount of information simply and quickly to make it more accessible?

From Chapter 2, we now know how to open a data file and examine the file's contents in R. It turns out that we have more information than we can usefully process. To see this, import the data set (`nids.dta`) and open an output file (`chapter3.txt`). Now, simply type:

```
library(tidyverse)
```

```
## -- Attaching packages ------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

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

```
library(foreign)

nids<-read.dta("./data/nids.dta", convert.factors=FALSE)

#sink(file="./sink/chapter3.txt")
#table(nids$w1_hhincome)
#sink()
```

A large number of households scroll by. After seeing these numbers fly by, do you now know more about the distribution of income in South Africa? Probably not. We need a handy way to summarize lots of quantitative raw data, and R is very helpful here. For example, while we could, in principle, simply count the thousands of observations as they scroll by, R commands essentially do this in a much more efficient and sophisticated fashion. We will start by learning how to answer the following questions:

1. How many Africans are in the data?
2. What percentage of the sample is made up of Coloured respondents?

3. How many White men in the sample reside in rural areas?
4. Are there more men or women over the age of 60?
5. What percentage of the NIDS sample is made up of sons and daughters?

Using R, these questions are quickly and easily answered. First, though, we need to learn about different types of variables.

3.2 Variable types

In the NIDS dataset, there are several types of variables. If we load/import `nids.dta` into R and type `names(nids)`, we will see the list of variables names in the Console window. Economists, sociologists, and psychologists use different language to describe the multiple types of variables. Here, we will divide variables into two types: *continuous* variables and *categorical* variables. However, each of these types of variables can be applied to either the *individual* or the entire *household*. In this regard, we must understand the difference between an *individual-level variable* and a *household-level variable* as well. The statistical and graphical tools used to understand the distributions of the various types of variables are quite different, so it is important to understand the differences between these measures. We will now elaborate on what we mean by continuous versus categorical variables and then move on to explaining how these can be applied on an individual or household level.

Categorical Variables: Categorical variables, also known as nominal variables are made up of separate and distinct categories which do not have an inherent order. To code these variables, each category is typically assigned a value, but this assignment is arbitrary. Take for example the race variable, `w1_best_race`. Each racial group is assigned an arbitrary value. As we saw in chapter 2, in our data set if a person is African, the race variable for that person is set to 1. If the person is Coloured, the value is set to 2, Indian/Asian is 3, and White is 4. For the gender variable `w1_r_b4`, males are coded as 1 and females as 2. Other examples of what we will consider categorical variables include the household id number (`hhid`) the relationship to the head of household (`w1_r_b3`), and province (`w1_hhprov`).

A special type of a categorical variable is a dummy variable. A dummy variable is a variable that typically takes on a value of one if the observation meets specified criteria and a value of zero if otherwise. We will often want to create dummy variables ourselves. For example, if we wanted to create a dummy variable for whether a household was White, we could use the following `dplyr` commands.

```
nids<-nids %>%
  mutate(white = ifelse(w1_best_race==4,1,0))
```

Continuous Variables: Continuous variables have an infinite number of possible values that fall between any two observed values. For example, consider age. In our data, age is recorded in years. But it could have been recorded in months, days, minutes, or even seconds. A continuous variable is ordinal in the sense that its values have an inherent order. In the age example, an age of 16 years is one year older than the age of 15 years, thus the unit of measurement in between these two values is itself meaningful. (This may seem like common sense, but when we consider categorical variables, this will no longer be true.). Examples of continuous variables in the NIDS data set include age (`w1_r_best_age_yrs`), household monthly income (`w1_hhincome`), and the household expenditure measure (`w1_h_expenditure`) to name but a few.

We are actually not being terribly careful with our definitions. Consider, for example, a variable that counts members of the household (`w1_hhsizer`). Household size might be 5 or 6, but it will never be 5.49. Nonetheless, if we were told that the average number of members of a household was 5.49, this would be comprehendible. We would know that, on average, there are more than 5 members and less than 6. We are going to treat variables like household size and number of births as continuous variables. (Some disciplines refer to these as discrete variables.) Taking the average gives an answer that is readily interpreted. Taking the average of a categorical variable, on the other hand, yields nonsense (e.g. it doesn't make any sense to talk about the average race in the economy).

In general, it is important to know the types of variables you are using because some of the tools used to analyze variables differ depending on whether the variable is continuous or categorical. Another point to keep in mind is whether the variable you are using is an **individual level** or **household-level variable.**

Individual-level Variables: Individual-level variables are made up of values that are unique to each person in the household. An example is the variable for age (`w1_r_best_age_yrs`). To see an example of an individual-level variable, type the following:

```
nids %>%
  select(hhid, w1_r_best_age_yrs)%>%
  arrange(hhid) %>%
  head(10)
```

```
##      hhid w1_r_best_age_yrs
## 1  101012                51
## 2  101013                45
## 3  101013                32
## 4  101013                 9
## 5  101013                13
## 6  101013                11
## 7  101014                15
## 8  101014                25
## 9  101014                60
## 10 101014                22
```

As we can see, each person in the same household has an age value that is unique to them. Other examples in the NIDS data set would include the following variables: `w1_best_edu` (educational attainment level), and `w1_r_b4` (gender).

Household-level Variables: Household-level variables have the same value for every person in the household. An example would be the variable for total monthly household income (`w1_hhincome`). To see an example of a household-level variable, type the following commands:

```
nids %>%
  select(hhid, w1_hhincome)%>%
  arrange(hhid) %>%
  head(10)
```

```
##      hhid w1_hhincome
## 1  101012   1044.7892
## 2  101013    588.3008
## 3  101013    588.3008
## 4  101013    588.3008
## 5  101013    588.3008
## 6  101013    588.3008
## 7  101014   1307.4182
## 8  101014   1307.4182
## 9  101014   1307.4182
## 10 101014   1307.4182
```

As we can see, each person in the same household has the same value of total household monthly income. Other examples in the NIDS data set include the following variables: `w1_h_expf` (total household monthly food expenditure) and `w1_hhgovt` (total income from government grants)

3.3 Count observations - dim(),nrow()

So far we have introduced `dim()` which return the dimensions of a data frame, i.e. number of rows and columns, e.g.

```
dim(nids)
```

```
## [1] 31170  2090
```

There are other base R functions to achieve the same objective. These are `nrow` and `ncol` to return the number of rows and columns respectifully. For example, type:

```
nrow(nids)
```

```
## [1] 31170
```

The results should show that there are 31170 observations in this data set. However, try an example using a qualifier.

`dplyr` has a function that tallies observations as well.Suppose we want to count number of observations that satisfy a specific condition, for instance, suppose we want to count the number of females (`w1_r_b4==2`) in the data set:

```
nids %>%
  filter(w1_r_b4==2) %>% #filter females
  count
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1 16527
```

The results should show that there are 16527, females in the data set.

3.4 Using household level variables

We have seen above that household level variables have the same value for each member in the household. But it will often be the case where we want to count only one of these values per household. Suppose, for instance, that we wanted to count the number of households in the data set. In order to accomplish this task, we need to create a new variable which we will call `hhrestrict`.

```
nids<-nids%>% #assign the nids object to nids
  arrange(hhid, pid)%>% #sort by hhid then pid
  group_by(hhid) %>% #group by hhid
  mutate(hhrestrict = 1:n()) %>% # create hhrestrict that is numbered individuals within the group by v
  mutate(hhrestrict = ifelse(hhrestrict==1,1,0)) #make hhrestrict 1 for only the first observation
```

Lets try to understand what this variable looks like. So type the following commands to view the first 20 observations:

```
nids %>%
  select(hhid, hhrestrict) %>%
  head(20)
```

```
## # A tibble: 20 x 2
## # Groups:   hhid [7]
##      hhid hhrestrict
##     <int>      <dbl>
```

```
##  1 101012          1
##  2 101013          1
##  3 101013          0
##  4 101013          0
##  5 101013          0
##  6 101013          0
##  7 101014          1
##  8 101014          0
##  9 101014          0
## 10 101014          0
## 11 101014          0
## 12 101014          0
## 13 101015          1
## 14 101015          0
## 15 101016          1
## 16 101016          0
## 17 101017          1
## 18 101017          0
## 19 101017          0
## 20 101018          1
```

Alternatively

```
head(nids[,c("hhid", "hhrestrict")], n = 20L)
```

```
## # A tibble: 20 x 2
## # Groups:   hhid [7]
##      hhid hhrestrict
##     <int>      <dbl>
##  1 101012          1
##  2 101013          1
##  3 101013          0
##  4 101013          0
##  5 101013          0
##  6 101013          0
##  7 101014          1
##  8 101014          0
##  9 101014          0
## 10 101014          0
## 11 101014          0
## 12 101014          0
## 13 101015          1
## 14 101015          0
## 15 101016          1
## 16 101016          0
## 17 101017          1
## 18 101017          0
## 19 101017          0
## 20 101018          1
```

The hhrestrict variable is a dummy variable where a one is given to the first member of the household and a 0 for any consequent members. Using this variable we can count the number of households in the data set:

```
nids%>%
  filter(hhrestrict==1)%>%
  nrow
```

```
## [1] 7305
```

The 7305 that R gives us is the number of households in the data set.

We will be using the `hhrestrict` variable extensively in future chapters, so let's try one more example. Suppose we wanted to create a new type of variable that will only be applied to the first person in the household. Let's look at the household variable for total monthly income (`w1_hhincome`). First we sort by `hhid` and then list the `hhid`, `w1_hhincome` and `hhrestrict` variables. The output given below is the first piece of output you should obtain:

```
nids%>%
  select(hhid, w1_hhincome, hhrestrict)%>%
  arrange(hhid) %>%
  head(25)
```

```
## # A tibble: 25 x 3
## # Groups:   hhid [8]
##      hhid w1_hhincome hhrestrict
##     <int>       <dbl>      <dbl>
##  1 101012       1045.          1
##  2 101013        588.          1
##  3 101013        588.          0
##  4 101013        588.          0
##  5 101013        588.          0
##  6 101013        588.          0
##  7 101014       1307.          1
##  8 101014       1307.          0
##  9 101014       1307.          0
## 10 101014       1307.          0
## # ... with 15 more rows
```

We can see that everyone with the same household identification number (`hhid`) has the same total household monthly income. However, we want to create a variable that only assigns the total household monthly income to the first person in the household. One reason for doing this is that when we want to calculate the average household monthly income, we only want to count the income from each household once. If we calculate the average household income in South Africa using all the individuals in the dataset then the income level of large households will be weighted more heavily (i.e. a household with 27 members will be counted 27 times, while a household with 1 member will be counted once).

In order to create this new variable that we want, we tell R to create a new variable that is equal to the household income level whenever our `hhrestrict` variable is equal to 1 (i.e. only for the first person in every household).

```
nids<-nids%>%
  mutate(hhincome2 = ifelse(hhrestrict==1, w1_hhincome, NA))
```

Let's see what our new variable looks like:

```
nids %>%
  select(hhid, w1_hhincome,hhincome2, hhrestrict) %>%
  head(25)
```

```
## # A tibble: 25 x 4
## # Groups:   hhid [8]
##      hhid w1_hhincome hhincome2 hhrestrict
##     <int>       <dbl>     <dbl>      <dbl>
##  1 101012       1045.     1045.          1
##  2 101013        588.      588.          1
```

```
##  3 101013        588.       NA           0
##  4 101013        588.       NA           0
##  5 101013        588.       NA           0
##  6 101013        588.       NA           0
##  7 101014       1307.     1307.          1
##  8 101014       1307.       NA           0
##  9 101014       1307.       NA           0
## 10 101014       1307.       NA           0
## # ... with 15 more rows
```

From this output it seems that our new variable `hhincome2` is doing what we would like it to do and only assigning the household income level to the first person in the household. Why do this? As mentioned above, it will often be useful to only count a household level variable once per household, so that observations that come from households with more people are not given more weight than observations coming from households with less people. In doing this, we will hopefully avoid the problem of number bias. Note, it is a very common mistake for people to forget to take into account whether the variables they are working with are household level or individual level variables. It is very important that you are constantly paying attention to the type of variable you are using. If you do not, you can easily get very inaccurate results.

3.5 Frequency distribution tables

A frequency distribution table is simply a listing of all observed values for a given variable and the number of observations that fall under each of these values. To create a frequency distribution table in R base, we use the command `table()` ....

For example, to create a frequency distribution table for the categorical variable w1_best_race, you type:

```
table(nids$w1_best_race)
```

```
##
##     1     2     3     4
## 22157  4166   439  1432
```

The race variable does not have negative values for different missing types. We will change it to a factor variable:

```
nids<-nids %>%
  mutate(w1_best_race=factor(w1_best_race, levels=1:4,labels=c("African","Coloured","Asian_Indian","Whi
```

Tabulate again

```
table(nids$w1_best_race)
```

```
##
##      African     Coloured Asian_Indian        White
##        22157         4166          439         1432
```

We can also use `dplyr` group by function to tabulate:

First one:

```
nids%>%
  group_by(w1_best_race)%>%
  summarise(freq=n()) %>%
  mutate(percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(freq)/sum(freq)*100,2))
```

```
## # A tibble: 5 x 4
##   w1_best_race  freq percent cum_percent
```

```
##   <fct>        <int>   <dbl>       <dbl>
## 1 African      22157   71.1         71.1
## 2 Coloured      4166   13.4         84.4
## 3 Asian_Indian   439    1.41        85.9
## 4 White         1432    4.59        90.4
## 5 <NA>          2976    9.55       100
```

Second one (using `na.omit()` to remove those with missing race):

```
nids%>%
  group_by(w1_best_race)%>%
  summarise(freq=n()) %>%
  na.omit() %>% #remove category for missing
  mutate(percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(freq)/sum(freq)*100,2))
```

```
## # A tibble: 4 x 4
##   w1_best_race  freq percent cum_percent
##   <fct>        <int>   <dbl>       <dbl>
## 1 African      22157   78.6         78.6
## 2 Coloured      4166   14.8         93.4
## 3 Asian_Indian   439    1.56        94.9
## 4 White         1432    5.08       100
```

As we can see from the table, there are four distinct categories or values found within the `w1_best_race` variable. The four observed values being African, Coloured, Asian_India and White.

We computed three specific numbers related to each race category. The column with the header "freq" is the number of observations that fall within each category. Thus, we can now answer the question, "How many Africans are in the data?" The answer being that there are 22157 Africans in the NIDS data.

The second computed column with the header "percent" represents the percentage of the sample that falls within each observed category, the first computation including the missing and the second one excluding the missing observations for race. Thus, we can now answer the question "What percentage of the sample is made up of Coloured respondents?" Coloured respondents make up 14.8% of the NIDS sample.

Ok, now it is your turn to answer a few questions:

1. How many resident heads are there in the NIDS sample?

Question 1 Answer

2. What percentage of the sample is 70 years old and younger?

Question 2 Answer

3. How many Coloured respondents are there in the data?

Question 4 Answer

4. Are there more men or women over the age of 60?

Question 4 Answer

5. How would you create a new variable that is equal to 1 if the respondent is a women and equal to 0 if the respondent is a man (use the w1_r_b4 variable to identify the gender of the respondents)?

Question 5 Answer

6. How many White men in the sample reside in rural areas?

Question 6 Answer

D3 - Main material used for roof

Variable: *w1_h_dwlmatroof*

	Code list
-5	Not Applicable
-3	Missing
1	Bricks
2	Cement block/concrete
3	Corrugated iron/zinc
4	Wood
5	Plastic
6	Cardboard
7	Mixture of mud and cement
8	Wattle and daub
9	Tile
10	Mud bricks
11	Thatching
12	Asbestos/Cement roof sheeting
13	Stone and rock

Figure 3.1: Labels for main material used for roof

7. How many respondents in the sample have a missing value for the variable identifying the relationship to the head of household?

Question 7 Answer

3.6 Frequency distribution tables: household level data

Let's say that we are interested in investigating living standards in South Africa. In South Africa, one of the indicators of a household's living standard is the type of dwelling that people live in. In order to investigate this, we first need to know whether NIDS has any variables dealing with this topic. We should by now be starting to become familiar with the steps that we can take in looking for a variable.

The household questionnaire proves extremely helpful in this case as there is an entire section on living standards (Section D). Question D1 tells us about the type of dwelling and interestingly question D3 provides us with even more information relating to the building material used to make the dwelling walls and roof.

Please note that the above table is taken from the official NIDS documents on the DataFirst website, the naming convention is different from the training dataset.

Let's take a closer look at the corresponding variable that describes roof material. What proportion of households have roofs made of tiles? Corrugated iron? Wood?

```
nids%>%
  group_by(w1_h_d3_1)%>%
  summarise(freq=n()) %>%
  mutate(percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(freq)/sum(freq)*100,2))
```

```
## # A tibble: 15 x 4
##    w1_h_d3_1  freq percent cum_percent
##        <int> <int>   <dbl>       <dbl>
##  1        -5     1    0           0
##  2        -3   303    0.97        0.98
##  3         1   257    0.82        1.8
##  4         2   447    1.43        3.23
##  5         3 20453   65.6        68.8
##  6         4   186    0.6        69.4
```

```
##  7         5    31    0.1        69.6
##  8         6    62    0.2        69.8
##  9         7   104    0.33       70.1
## 10         8   187    0.6        70.7
## 11         9  3481   11.2        81.8
## 12        10   103    0.33       82.2
## 13        11  2235    7.17       89.4
## 14        12  3316   10.6       100.0
## 15        13     4    0.01      100
```

We see that there are 304 individuals who have a missing (-3) or not-applicable (-5) value for this variable. We will learn more about these in the next section. For now, let's ignore them by using qualifiers. We create a new variable `mat` that is equal to `w1_h_d3_1` for only positive values and missing otherwise.

```
nids<-nids%>%
  mutate(mat = ifelse(w1_h_d3_1>0 , w1_h_d3_1, NA))
```

```
nids%>%
  group_by(mat)%>%
  summarise(freq=n()) %>%
  na.omit() %>%
  mutate(cum_freq = cumsum(freq), percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(fre
```

```
## # A tibble: 13 x 5
##      mat  freq cum_freq percent cum_percent
##    <int> <int>    <int>   <dbl>       <dbl>
##  1     1   257      257    0.83        0.83
##  2     2   447      704    1.45        2.28
##  3     3 20453    21157   66.3        68.5
##  4     4   186    21343    0.6        69.2
##  5     5    31    21374    0.1        69.2
##  6     6    62    21436    0.2        69.4
##  7     7   104    21540    0.34       69.8
##  8     8   187    21727    0.61       70.4
##  9     9  3481    25208   11.3        81.7
## 10    10   103    25311    0.33       82
## 11    11  2235    27546    7.24       89.2
## 12    12  3316    30862   10.7       100.0
## 13    13     4    30866    0.01      100
```

Therefore 11% of households live in houses with tile roofs, 66% of households have corrugated iron roofs, and 0.6% have wooden roofs. Or do they? Can you see anything wrong with what we have done?

Recall that we began this chapter with a discussion about the difference between individual level and household level data. Essentially the table above gives us **individual** level percentages. It tells us that 66% of individuals in the sample live in houses with corrugated iron roofs. This is not the same as saying that 66% of households have corrugated iron roofs.

To see this, let's look at a simple example. Say you have a sample of 300 people. 150 live in houses with tile roofs and 150 live in houses with wooden roofs. The average household with a tile roof has 3 people, while the average household with a wooden roof has 5 people. Now you have 50 households with a tile roof and 30 households with a wooden roof. Therefore, 50% of **individuals** have a tiled roof, but 62.5% of **households** have a tile roof.

Basically, in the table above we are counting every large household more times than we are counting smaller households. Let's see how different our results are when we only count each household once. Recall that we can use the `hhrestrict` variable that we created earlier in this chapter to do this.

```
nids%>%
  filter(hhrestrict==1)%>%
  group_by(mat)%>%
  summarise(freq=n()) %>%
  na.omit() %>%
  mutate(cum_freq = cumsum(freq), percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(fre
```

```
## # A tibble: 13 x 5
##      mat  freq cum_freq percent cum_percent
##    <int> <int>    <int>   <dbl>       <dbl>
##  1     1    63       63    0.87        0.87
##  2     2   106      169    1.46        2.33
##  3     3  4793     4962   66.1        68.5
##  4     4    46     5008    0.63       69.1
##  5     5     8     5016    0.11       69.2
##  6     6    20     5036    0.28       69.5
##  7     7    18     5054    0.25       69.7
##  8     8    37     5091    0.51       70.2
##  9     9   929     6020   12.8        83.1
## 10    10    22     6042    0.3        83.4
## 11    11   349     6391    4.82       88.2
## 12    12   854     7245   11.8       100.0
## 13    13     3     7248    0.04      100
```

Notice that while we had 30866 individuals in the previous table, this table includes 7248 households. However, many of the percentages remain roughly the same size - for instance 66% of households have corrugated iron roofs. This suggests that the average household size for the whole sample is similar to the household size for households with corrugated iron roofs. One roof type that has a large change in percentage prevalence between individual level and household level is thatched roofed dwellings with 7.2% of individuals and 4.8% of households. This tells us that households with thatched roofs are on average larger than households in the whole sample.

8. What percentage of households consist of only 1 person? What percentage of individuals live in a household consisting of 1 person (i.e. they live alone)? What is the largest household size in the sample? How many households of this size are there?

Question 8 Answer

3.7 Missing data and non-responses

Question 2 brought to our attention the impact that recorded non-response values (missings, refusals, don't knows, not-applicables) can have on our results. In general, taking into account missing data and non-responses is essential to any data analysis. Therefore, it will be useful to gain a better understanding of the different ways in which missing data and non-responses come in the NIDS dataset. As mentioned previously, a good first port of call in trying to gain a better grasp of a dataset is always the metadata and questionnaires. Scrolling through the first few pages of the metadata immediately tells us what the potential values each of the variables in the dataset can take. We also see that a:

- -3 refers to a 'missing' value (3333 for years, 33 for months),
- -5 refers to a 'not-applicable' response (5555 for years, 55 for months),
- -8 refers to a 'refusal' (8888 for years, 88 for months), and
- -9 refers to a 'don't know' response (9999 for years, 99 for years).

M5 - Satisfaction level with life currently
Variable: *w1_a_wbsat*

Code list

-9	Don't Know
-8	Refused
-5	Not Applicable
-3	Missing
1	Satisfaction Scale: 1
2	Satisfaction Scale: 2
3	Satisfaction Scale: 3
4	Satisfaction Scale: 4
5	Satisfaction Scale: 5
6	Satisfaction Scale: 6
7	Satisfaction Scale: 7
8	Satisfaction Scale: 8
9	Satisfaction Scale: 9
10	Satisfaction Scale: 10

Figure 3.2: Labels for life satisfaction levels

However, you may find it puzzling that a -3 refers to missing value, but previously in the chapters we talked about a `NA` indicating a missing value. It is therefore useful for us to look at what the difference between these two types of missing data is and how missing data arises. Take a few seconds to think about why the dataset might be missing data for a specific variable. Is it because someone didn't want to answer the question? Perhaps the person wasn't asked the question. Perhaps the person didn't know the answer to a question. Or perhaps the field worker (person recording the answers to the questionnaire), the data capturer (person who transfers the answers from questionnaire to the computer), or the data cleaner (person who checks that the variables in the dataset have valid values and don't contradict one another) made a mistake!

It is clear that there are many different ways in which missing data can occur. Let's take a closer look at the variable `w1_a_m5` and to get an idea of how missing data and non-responses work in the dataset.

```
nids%>%
  group_by(w1_a_m5)%>%
  summarise(freq=n()) %>%
  mutate(cum_freq = cumsum(freq), percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(fre
```

```
## # A tibble: 15 x 5
##    w1_a_m5  freq cum_freq percent cum_percent
##      <int> <int>    <int>   <dbl>       <dbl>
##  1      -9  1433     1433    4.6          4.6
##  2      -8   163     1596    0.52         5.12
##  3      -5     2     1598    0.01         5.13
##  4      -3   240     1838    0.77         5.9
##  5       1  1007     2845    3.23         9.13
##  6       2   701     3546    2.25        11.4
##  7       3  1246     4792    4           15.4
##  8       4  1867     6659    5.99        21.4
##  9       5  2558     9217    8.21        29.6
## 10       6  1803    11020    5.78        35.4
## 11       7  1633    12653    5.24        40.6
## 12       8  1305    13958    4.19        44.8
## 13       9   455    14413    1.46        46.2
## 14      10  1217    15630    3.9         50.1
## 15      NA 15540    31170   49.9        100
```

How many observations of this variable are `NA`-missing? Notice that 15540 observations of this variable are NA-missing. This is almost half of the total number of respondents (31170) in the dataset. This is a huge portion of the respondents. Can you think of a reason why this is the case? It is immediately clear that if you ignore missing data without understanding why it came about, your results might be very, very biased. In this case, our clue as to the origins of the missing observations lies in understanding that the **a** in the variable name means that the variable corresponds to a question in the adult questionnaire. This means that it is an individual level variable that was only asked of adults who were present when the field worker went to the household to administer the questionnaire. Therefore, children and adult members of the household who were absent (represented by proxy questionnaire) would have no recorded answer for this question and therefore would be NA-missing in the dataset. This would probably explain a large portion of the 15540 NA-missing observations, however if you are really paying attention you may also notice that the range of this variable is [-9, 10]. We already know that in the NIDS survey, negative numbers are generally used to refer to non-responses. Let's see how many non-responses there are for this variable.

In addition to the 15 540 NA-missings, almost 6 percent of the respondents have non-response values with 4.6 % saying that they 'don't know' the answer to this question. This brings us back to the difference between a -3 and an NA-missing observation. An observation is recorded as `NA` if the respondent was not meant to answer the question or have a value recorded for that variable. For example, children are NA-missing on all adult questionnaire variables. In contrast the observation is recorded as a -3 if the respondent is meant to have a value recorded for the variable, but for some reason it is missing. For example, if the field worker forgot to write down the respondent's age or recorded their age as 312 (which is unlikely to be true). In general, all non-responses (negative numbers) occur when the respondent should have had a valid value for a given variable, but doesn't for some reason.

3.8 Three primary origins of NA-missing data in NIDS

Now that we understand the difference between non-responses and NA-missing data, we have to ask how we know whether a person was meant to answer a specific question or not. There are three primary ways in which people can be NA-missing in the NIDS dataset. We discuss these below, but remember that when working with data you can often come across situations different to those we describe here and always need to interrogate your data with an open, curious, questioning mind.

NA-missings: Merging Datasets

We have already seen above that one way in which people can be NA-missing is when there is more than one dataset merged together and then respondents will only have data from the household level datasets as well as one of the individual level datasets (child, adult, or proxy). They will be NA-missing for all variables from the other two.

NA-missings: Refused Questionnaires

Above when we were talking about the 15 540 NA-missing observations, we said that we could explain most of them through the children and absent adults in the dataset, but we didn't investigate this claim at all. Even though it is a good hypothesis, it is important to check exactly how many of the 15 540 are explained in this way. How would you go about doing this? One way is to check if there is a variable that gives the questionnaire type of an individual. We search for 'quest' since it is less restrictive than 'questionnaire', although in this specific case it doesn't make a difference to our results.

There is a variable called `w1_quest_typ`:

We see that the variable takes a value of '1' when the respondent was administered the adult questionnaire. So now we can check whether any of the adults who were present are NA-missing.

```
nids%>%
  filter(w1_quest_typ == 1) %>%
  group_by(w1_a_m5)%>%
```

Questionnaire type
Variable: *w1_quest_typ*
(Derived variable)

Code list	
1	Adult
2	Child
3	Proxy

Figure 3.3: Labels for questionnaire type

Refusals (if applicable)

A21	What is the main reason for refusal?	Too busy	1
		Not interested/waste of time	2
		Questionnaire too personal/too intrusive	3
		Don't trust surveys	4
		Never do surveys	5
		Too old	6
		Other (specify)	7

Figure 3.4: Labels for refusals

```
  summarise(freq=n()) %>%
  mutate(cum_freq = cumsum(freq), percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(fre
```

```
## # A tibble: 15 x 5
##    w1_a_m5  freq cum_freq percent cum_percent
##      <int> <int>    <int>   <dbl>       <dbl>
##  1      -9  1433     1433    8.49        8.49
##  2      -8   163     1596    0.97        9.45
##  3      -5     2     1598    0.01        9.46
##  4      -3   240     1838    1.42       10.9
##  5       1  1007     2845    5.96       16.8
##  6       2   701     3546    4.15       21
##  7       3  1246     4792    7.38       28.4
##  8       4  1867     6659   11.1        39.4
##  9       5  2558     9217   15.2        54.6
## 10       6  1803    11020   10.7        65.3
## 11       7  1633    12653    9.67       74.9
## 12       8  1305    13958    7.73       82.7
## 13       9   455    14413    2.69       85.4
## 14      10  1217    15630    7.21       92.6
## 15      NA  1255    16885    7.43      100
```

Notice how we filtered respondents based on `w1_quest_typ` to limit our sample to present adults. You will notice that the only value that has changed in this restricted table is the number of NA-missings which has decreased by 14285. This validates our hypothesis, but we are still left with 1255 NA-missings that we have not explained. This shows us why it is important to **ALWAYS** check that we FULLY understand where missing data comes from even when we have a plausible explanation. In this case, it is not easy to figure out where these missing observations come from, although we can get a clue from the metadata which tells us on page one of the metadata which tells us that 1246 adults refused to answer the entire adult questionnaire after section A. Let's check if this explains remaining NA-missings:

The corresponding variable is `w1_a_a21` and lets see its distribution:

```
nids%>%
  group_by(w1_a_a21)%>%
  summarise(freq=n()) %>%
  mutate(percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(freq)/sum(freq)*100,2))
```

```
## # A tibble: 10 x 4
##    w1_a_a21  freq percent cum_percent
##       <int> <int>   <dbl>       <dbl>
##  1        1   175    0.56        0.56
##  2        2   357    1.15        1.71
##  3        3    39    0.13        1.83
##  4        4    80    0.26        2.09
##  5        5    12    0.04        2.13
##  6        6    22    0.07        2.2
##  7        7    57    0.18        2.38
##  8        8   321    1.03        3.41
##  9        9   183    0.59        4
## 10       NA 29924   96         100
```

Since this variable takes on several different values for the different reasons of refusal to answer the questionnaire, let's recode it to give us an indicator variable which takes on the value of '1' when an adult refused to answer their questionnaire and '0' otherwise.

```
nids<-nids %>%
  mutate(refusal_dummy=ifelse(is.na(w1_a_a21),0,1))
```

```
nids%>%
  group_by(refusal_dummy)%>%
  summarise(freq=n())
```

```
## # A tibble: 2 x 2
##   refusal_dummy  freq
##           <dbl> <int>
## 1             0 29924
## 2             1  1246
```

We summarise the frequencies of the variable to check that the recode has done what we wanted it to – since we have 1246 one's (refusals), it would appear that it has.

Now we are in a position to check how many of our NA-missings we have explained:

```
nids %>%
  filter(w1_quest_typ == 1 & refusal_dummy == 0) %>%
  group_by(w1_a_m5) %>%
  summarise(freq=n()) %>%
  mutate(percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(freq)/sum(freq)*100,2))
```

```
## # A tibble: 15 x 4
##    w1_a_m5  freq percent cum_percent
##      <int> <int>   <dbl>       <dbl>
##  1      -9  1433    9.16        9.16
##  2      -8   163    1.04       10.2
##  3      -5     2    0.01       10.2
##  4      -3   240    1.53       11.8
##  5       1  1007    6.44       18.2
##  6       2   701    4.48       22.7
##  7       3  1246    7.97       30.6
```

C1.10	Are you currently pregnant?	Yes	1
		No	2
		Refused	8
		Don't know	9

Figure 3.5: Labels for question C1.10

```
##  8       4  1867   11.9        42.6
##  9       5  2558   16.4        58.9
## 10       6  1803   11.5        70.5
## 11       7  1633   10.4        80.9
## 12       8  1305    8.34       89.2
## 13       9   455    2.91       92.2
## 14      10  1217    7.78       99.9
## 15      NA     9    0.06      100
```

It seems we have hit the jackpot. All the values in our table remain unchanged, except that we now only have 9 NA-missing data points. This is a far more acceptable proportion of our data. Ideally, we would also want to explain these 9, but for our purposes we can move on – happy that we have managed to discover where almost all our NA-missing values come from.

NA-missings: Skip Patterns

So far we have seen that NA-missings can come from two types of situations in which an individual didn't answer any of the questions in an individual questionnaire – either because they refused or because they weren't meant to due to answering one of the other individual questionnaires. However there is a third common cause of NA-missings for individual questions within a questionnaire. Intuitively this occurs when it simply doesn't make sense for a set of people to answer a specific question. An illustrative example is given by question C1.10 of the adult questionnaire "Are you currently pregnant?". We know that the set of respondents who might have values for the corresponding variable (`w1_a_c1_10`) is already limited to those who answered the adult questionnaire (i.e. no children or questionnaire refusals). However it will make sense to limit the people who answer this question even further – for instance, it doesn't make sense to ask men or women past a certain age to answer this question. Here it is informative to look at the questionnaire to see exactly what restrictions are made. Firstly, we see that question C1.1 tells the interviewer to skip all the questions before D1 if the respondent is not female. Secondly, questions C1.3, C1.5, and C1.7 also have skips, but none of them affect whether the respondent answers the question we are interested in (C1.10). Thirdly, question C1.9 tells the interviewer to skip question C1.10 if the respondent is older than 49. Therefore, only adult respondents who are female and younger than 49 are required to answer this question. All others respondents will be NA-missing. We see below that this rules out a large proportion (79%) of the sample (as one would expect, since only a small proportion of the total population could potentially be pregnant).

```
nids %>%
  group_by(w1_a_c1_10) %>%
  summarise(freq=n()) %>%
  mutate(percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(freq)/sum(freq)*100,2))
```

```
## # A tibble: 6 x 4
##   w1_a_c1_10  freq percent cum_percent
##        <int> <int>   <dbl>       <dbl>
## 1         -9    19    0.06        0.06
## 2         -8    12    0.04        0.1
## 3         -3    82    0.26        0.36
## 4          1   247    0.79        1.15
## 5          2  6199   19.9        21.0
## 6         NA 24611   79.0       100
```

3.9 Missing data and qualifiers

In the previous chapter, we learnt how to use qualifiers to limit the sample we are working with. Now that we understand the different types of missing data and non-responses, we may want to look at the reported satisfaction levels of only the individuals who were supposed to answer the question.

```
nids %>%
  filter(!is.na(w1_a_m5)) %>%
  group_by(w1_a_m5) %>%
  summarise(freq=n()) %>%
  mutate(percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(freq)/sum(freq)*100,2))
```

```
## # A tibble: 14 x 4
##    w1_a_m5  freq percent cum_percent
##      <int> <int>   <dbl>       <dbl>
##  1      -9  1433    9.17        9.17
##  2      -8   163    1.04       10.2
##  3      -5     2    0.01       10.2
##  4      -3   240    1.54       11.8
##  5       1  1007    6.44       18.2
##  6       2   701    4.48       22.7
##  7       3  1246    7.97       30.7
##  8       4  1867   11.9        42.6
##  9       5  2558   16.4        59.0
## 10       6  1803   11.5        70.5
## 11       7  1633   10.4        81.0
## 12       8  1305    8.35       89.3
## 13       9   455    2.91       92.2
## 14      10  1217    7.79      100
```

We can see that we have indeed excluded all the NA-missing data. Now, we may want to browse/View all the entries of respondents who have a reported satisfaction level greater than 5

```
#nids %>%
#  select(w1_a_m5) %>%
#  filter(w1_a_m5>5) %>%
#  View
```

Now, how would you go about determining what percentage of your sample have a satisfaction level of 1 – excluding all NA-missings and non-responses?

```
nids %>%
  filter(!is.na(w1_a_m5) & w1_a_m5 >0) %>%
  group_by(w1_a_m5) %>%
  summarise(freq=n()) %>%
  mutate(percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(freq)/sum(freq)*100,2))
```

```
## # A tibble: 10 x 4
##    w1_a_m5  freq percent cum_percent
##      <int> <int>   <dbl>       <dbl>
##  1       1  1007    7.3         7.3
##  2       2   701    5.08       12.4
##  3       3  1246    9.03       21.4
##  4       4  1867   13.5        35.0
##  5       5  2558   18.6        53.5
##  6       6  1803   13.1        66.6
```

```
##  7       7  1633   11.8         78.4
##  8       8  1305    9.46        87.9
##  9       9   455    3.3         91.2
## 10      10  1217    8.82       100
```

Here we see that 7.3 % of the sample (excluding NA-missings and non-responses) has a satisfaction level of 1.

Now, the pertinent question is: Does this mean that 7.3 % of the total sample has a satisfaction level of one? Or more generally: "Is the sample for which I have information representative of the entire sample?" This is a very important and difficult question and one you should always ask of your results! You can only try to claim that your results accurately reflect South African society if you can answer 'yes' to this question. For instance, it requires a large leap of faith to say that 7.3 % of South Africans have a satisfaction level of 1. Recall that only adults who were present answered this question. Therefore, if children are generally happier than adults, then this percentage should be lower for South Africa as a whole. However, we might avoid this by saying "7.3 % of South African adults report a satisfaction level of 1". What about adults that were absent? What about the 1433 adults who said they 'don't know' or the 163 who refused? Will they have the same reported satisfaction distribution as those who were present and answered? It's hard to say without doing further research. In doing research, you will commonly face this problem. For now, let's return to acquiring a set of tools that will help you tackle these sorts of problems.

3.10 Question answers

3.11 Exercises

Now it is your turn to explore the distributions of variables using the commands from this chapter.

Using R and the NIDS data set, answer the following questions.

1. What percentage of the sample is made up of sons and daughters?

Exercise 1 Answer

2. Which of the racial groups makes up the largest proportion of urban formal residents?

Exercise 2 Answer

3. How many households are headed by females?

Exercise 3 Answer

3.12 Session information

```
print(sessionInfo(), locale = FALSE)
```

```
## R version 3.5.1 (2018-07-02)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 7 x64 (build 7601) Service Pack 1
##
## Matrix products: default
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
```

```
##  [1] bindrcpp_0.2.2  foreign_0.8-70  forcats_0.3.0   stringr_1.3.1
##  [5] dplyr_0.7.6     purrr_0.2.5     readr_1.1.1     tidyr_0.8.1
##  [9] tibble_1.4.2    ggplot2_3.0.0   tidyverse_1.2.1
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_0.2.4 xfun_0.2         reshape2_1.4.3   haven_1.1.2
##  [5] lattice_0.20-35  colorspace_1.3-2 htmltools_0.3.6  yaml_2.1.19
##  [9] utf8_1.1.4       rlang_0.2.1      pillar_1.2.3     withr_2.1.2
## [13] glue_1.2.0       modelr_0.1.2     readxl_1.1.0     bindr_0.1.1
## [17] plyr_1.8.4       munsell_0.5.0    gtable_0.2.0     cellranger_1.1.0
## [21] rvest_0.3.2      psych_1.8.4      evaluate_0.10.1  knitr_1.20
## [25] parallel_3.5.1   highr_0.7        broom_0.4.5      Rcpp_0.12.17
## [29] backports_1.1.2  scales_0.5.0     jsonlite_1.5     mnormt_1.5-5
## [33] hms_0.4.2        digest_0.6.15    stringi_1.1.7    bookdown_0.7
## [37] grid_3.5.1       rprojroot_1.3-2  cli_1.0.0        tools_3.5.1
## [41] magrittr_1.5     lazyeval_0.2.1   crayon_1.3.4     pkgconfig_2.0.1
## [45] xml2_1.2.0       lubridate_1.7.4  assertthat_0.2.0 rmarkdown_1.10
## [49] httr_1.3.1       rstudioapi_0.7   R6_2.2.2         nlme_3.1-137
## [53] compiler_3.5.1
```

Chapter 4

Understanding graphs

4.1 Introduction

Graphs are often an intuitive and easy-to-understand way of summarizing large amounts of information. In principle, descriptive statistics (e.g. the table command) and graphs (e.g. a pie graph) can convey the same information. Often, though, graphs are more readily interpreted. If we are trying to convey information to a colleague with limited (or no) quantitative training, a graph sometimes will be more effective than a table. But graphs are not only a good way to express information to those without statistical training! Even for those who are statistically sophisticated, graphs are often a wonderful way to get a quick feel for the information contained in the data set. In fact, open almost any newspaper, government publication or corporate report and the chances are good that you'll see graphs of some sort featured prominently.

R has an extensive set of graphing options that allow us to generate and customize a wide variety of graphs to suit our specific needs.

There are different ways of making graphs in R. Among the common ways is `base` or traditional R graphics package, `lattice` add-on package and `ggplot2` add-on package (Wickham, Chang, et al. 2018). In this guide, we made a choice to use `ggplot2`.

`ggplot2` was created by Hadley Wickham and is a family member of the tidyverse. It is based on "The Grammer of Graphics" - Leland Wilkinson. `ggplot2` uses layer elements to build a graphic, i.e., it uses an underlying "grammar" to build graphs component-by-component rather than providing premade graphs. It is easy enough to use without any exposure to the underlying grammar.

Useful resources for starting to learn `ggplot2` include:

- gglot2 Documentation (particularly the function reference).
- Data Visualization Chapter of R for Data Science Book
- RStudio's ggplot2 cheat sheet
- R Graphics Cookbook
- UCLA's introduction to `ggplot2`.

4.2 Getting ready

We begin this chapter by asking you to run the following lines of code. Please copy the following lines of code into an R script file and then run it before you begin the chapter. Make sure that you remember what each line of code is doing.

```
library(tidyverse)
library(foreign)
nids<-read.dta("./data/nids.dta", convert.factors=FALSE)

nids <- nids%>%
  arrange(hhid, pid)%>%
  #Tag hhid
  group_by(hhid) %>%
  mutate(hhrestrict = 1:n()) %>%
  mutate(hhrestrict = ifelse(hhrestrict==1,1,0))
```

```
#We rename some variables for ease of use.
nids<-nids %>%
 mutate(race = w1_best_race,
        age = w1_r_best_age_yrs,
        gender = w1_r_b4,
        province = w1_hhprov,
        hhincome = w1_hhincome) %>%
  mutate(gender = factor(gender, levels = 1:2, labels = c("Male", "Female")),
         race = factor(race, levels = 1:4, labels = c("African", "Coloured","Asian", "White")),
         province = factor(province, levels=1:9, labels = c("Western Cape","Eastern Cape","Northern Cap
         w1_hhgeo = factor(w1_hhgeo, levels = 1:4, labels = c("Rural formal", "Tribal authority areas",
```

4.3 ggplot2 and its elements

There is no graph menu for `ggplot2`, so we need to write code instructing R what to do. Since this is not an introduction to ggplot2, we will try to answer specific questions and less description on `ggplot2`. You can check with some of the highlighted resources for `ggplot2`.

`ggplot2` requires three essential elments to plot a graphic. These are:

- **data (data)**: variables mapped to aesthetic features of the graph.
- **aesthetic (aes)**: mapping between variables to their visualization.
- **geometric (geom)**: objects/shapes you add as layers to your graph.

The ggplot2 commands at the least looks like:

`ggplot(data=df, aes(x = ,y = )) + geom_*()`

where:

`df` - is your data frame

`x` and `y` - are variables to be mapped depening on graph type, and

`geom_*` - is for any one of:

- `geom_point` for drawing individual points (e.g., a scatter plot)
- `geom_line` for drawing lines (e.g., for a line charts)
- `geom_smooth` for drawing smoothed lines (e.g., for simple trends or approximations)
- `geom_bar` for drawing bars (e.g., for bar charts)
- `geom_histogram` for drawing binned values (e.g. a histogram)
- `geom_polygon` for drawing arbitrary shapes

- `geom_boxplot`: boxes-and-whiskers
- `geom_errorbar`: T-shaped error bars
- `geom_ribbon`: bands spanning y-values across a range of x-values

Each of these geometries will leverage the aesthetic mappings supplied although the specific visual properties that the data will map to will vary.

4.4 Histograms

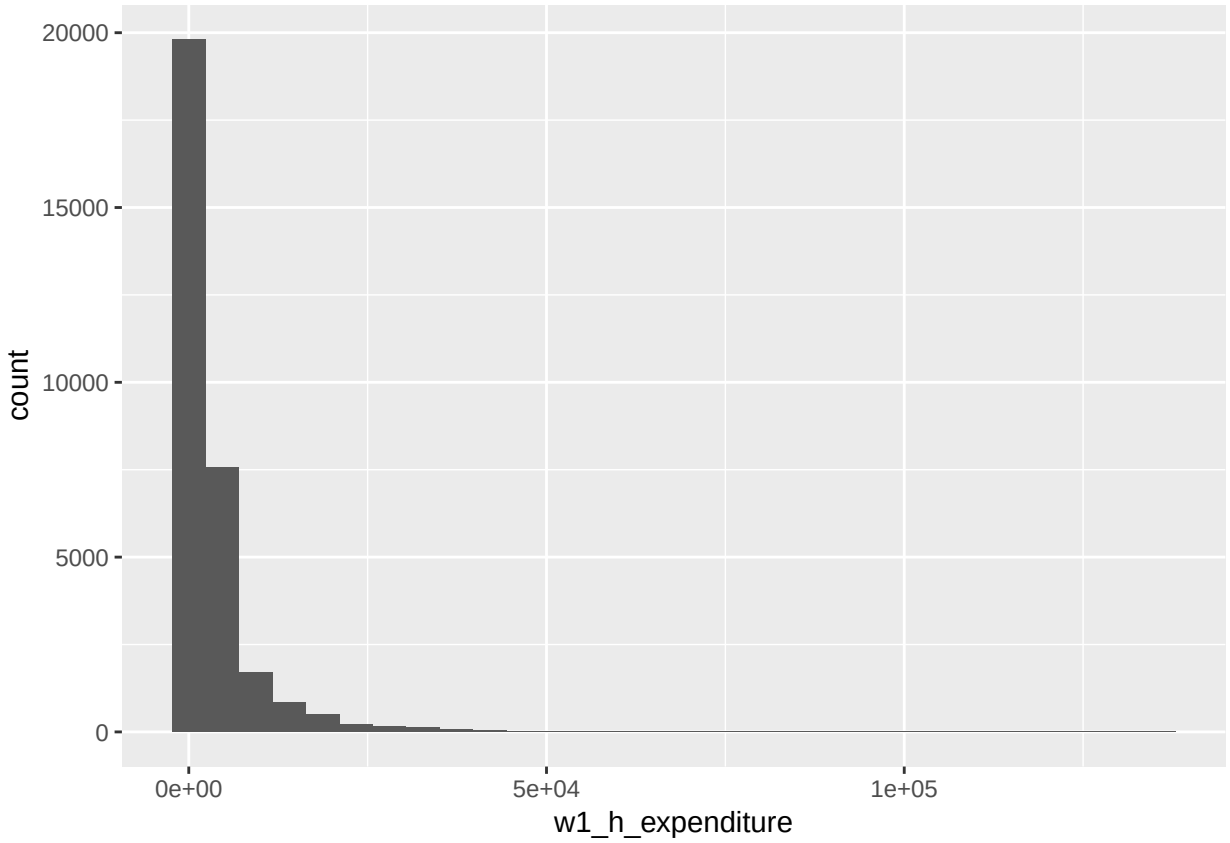
Histograms are a graphical tool that tell us the fraction (or percent) of observations, for any given variable, that fall within different ranges. Histograms are used for continuous variables. For example, if we have a continuous variable such as income, a histogram will tell us what percentage or fraction of the sample fall into different income 'bins' or 'groups'. We will present the basics of graphing histograms in R by working through the example we present below.

4.4.1 Graphing economic wellbeing in South Africa in 2008

Suppose we are interested in investigating the distribution of economic well-being in South African. The first question we need to answer is what variable should we use as a measure of economic well-being? Perhaps the first thoughts that come to mind are variables relating to household income or even assets owned? However when looking at survey data, economists tend to use a measure of expenditure rather than measures of income or assets. There are a number of reasons for this: firstly expenditure is far less variable than income, especially in rural areas (in fact, in rural areas income can vary greatly depending on when the survey is administered) and secondly, a good measure of assets is very hard to capture in a survey, for example one of the biggest assets that a household may have is its house, but it is very hard to get good data on the value of a house. So let's go ahead and graph a histogram of total monthly expenditure for the household.

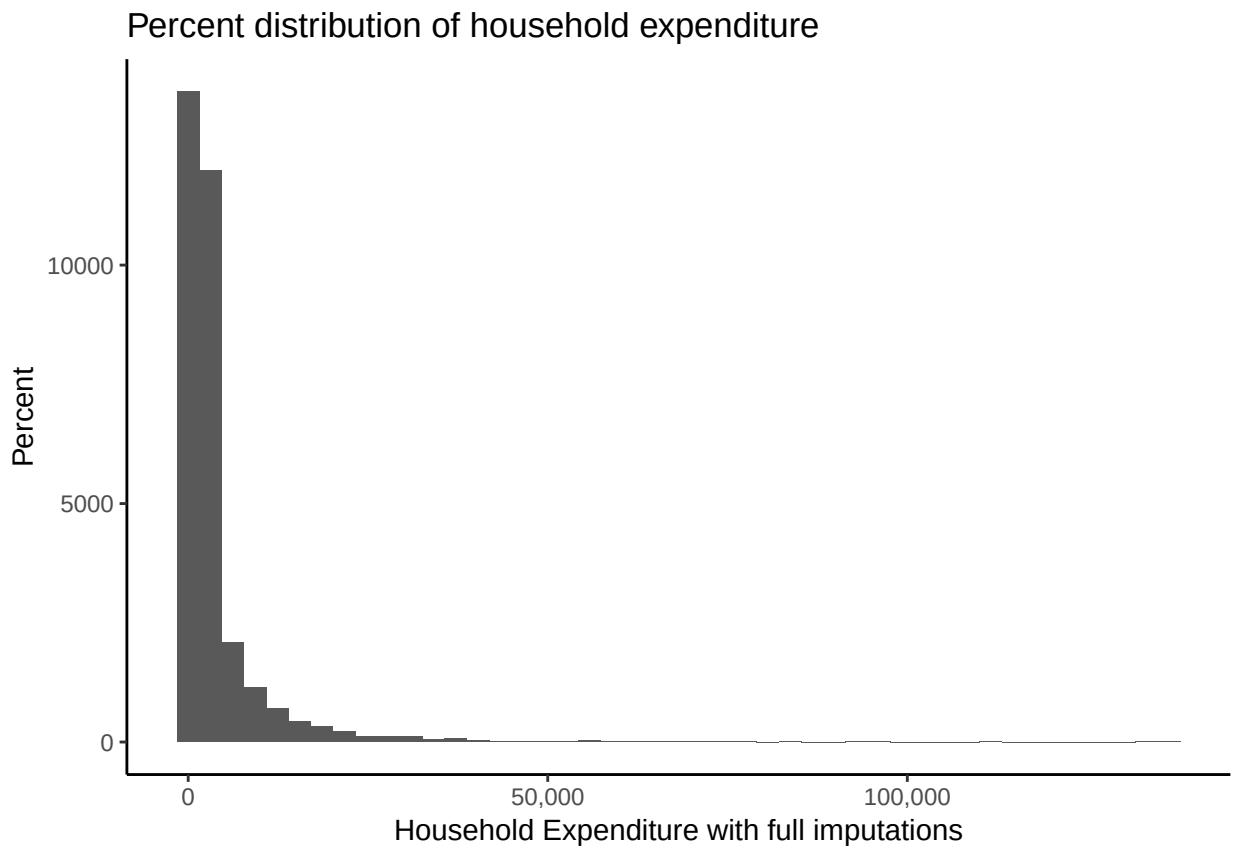
```
ggplot(data=nids, aes(x = w1_h_expenditure)) + geom_histogram()
```

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



The above commands match the basic commands introduced above. We can tidy the graph a little bit more, for example, change `binwidth`, x-axis scale and format, x-axis (`xlab`) and y-axis (`ylab`) titles, the main title (`ggtitle`) and plot background (`theme_classic`). Here, we are going to make use of the `scales` library to format the axis scales.

```
library(scales)
ggplot(data=nids, aes(x = w1_h_expenditure)) +
  geom_histogram(binwidth = 3100) +
  scale_x_continuous(breaks=seq(0,150000,50000), labels = comma) +
  xlab("Household Expenditure with full imputations") +
  ylab("Percent") +
  ggtitle("Percent distribution of household expenditure") +
  theme_classic()
```



Now let's analyze the graph we have produced. What does this graph tell us about expenditure and economic well-being? As it stands this graph doesn't seem very useful for interpretation; it has such a long x-axis (R0 to R150 000)! Why would this be the case? The answer is that we have a number of large outliers. A small number of observations report extremely large expenditure values which may have been due to extremely large once off purchases. To cope with this type of variability, economists tend to 'zone in' on food expenditure; monthly food expenditure tends not to vary greatly and is less likely to have any outliers (even Bill Gates can only eat so much!). In addition, because of its relatively small variability it is also easier to pick up the difference between a legitimate outlier and a miscode when it comes to food expenditure. Thus, let's proceed with our analysis but using food expenditure.

The food expenditure variable is `w1_h_expf`. For ease of reference we are going to rename this variable:

```
nids<-rename(nids, food = w1_h_expf)
```

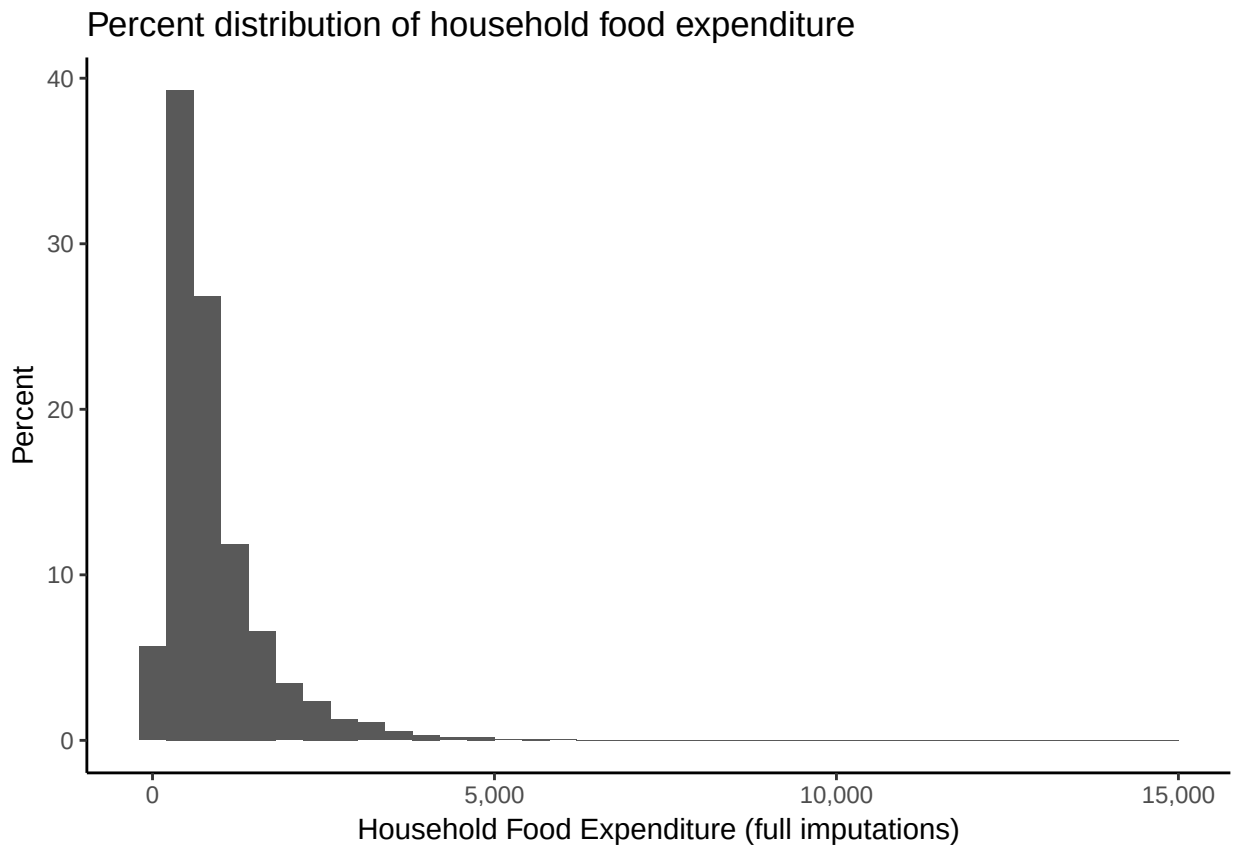
Let proceed to graphing our new variable, `food`.

When we graphed household expenditure we didn't control for the number of people in the household, this meant that we were counting large households more times than smaller ones. As we have discussed before this creates a bias in our analysis and we do not want to make the same mistake when graphing our food variable. We therefore need to add our `hhrestrict` qualifier to control for the household size (and thus, number bias). We need to control for household size by subsetting on `hhrestrict`, i.e. `hhrestrict==1`:

We can chain the `dplyr` functions to filter and then plot using `ggplot2` as follows:

```
nids%>%
  filter(hhrestrict==1) %>%
  ggplot(., aes(x = food, y = (..count..)/sum(..count..)*100)) +  # to create a density
  scale_x_continuous(breaks=seq(0,15000,5000), labels = comma) +
  geom_histogram(binwidth = 400) +
```

```
  xlab("Household Food Expenditure (full imputations)") +
  ylab("Percent") +
  ggtitle("Percent distribution of household food expenditure") +
  theme_classic()
```



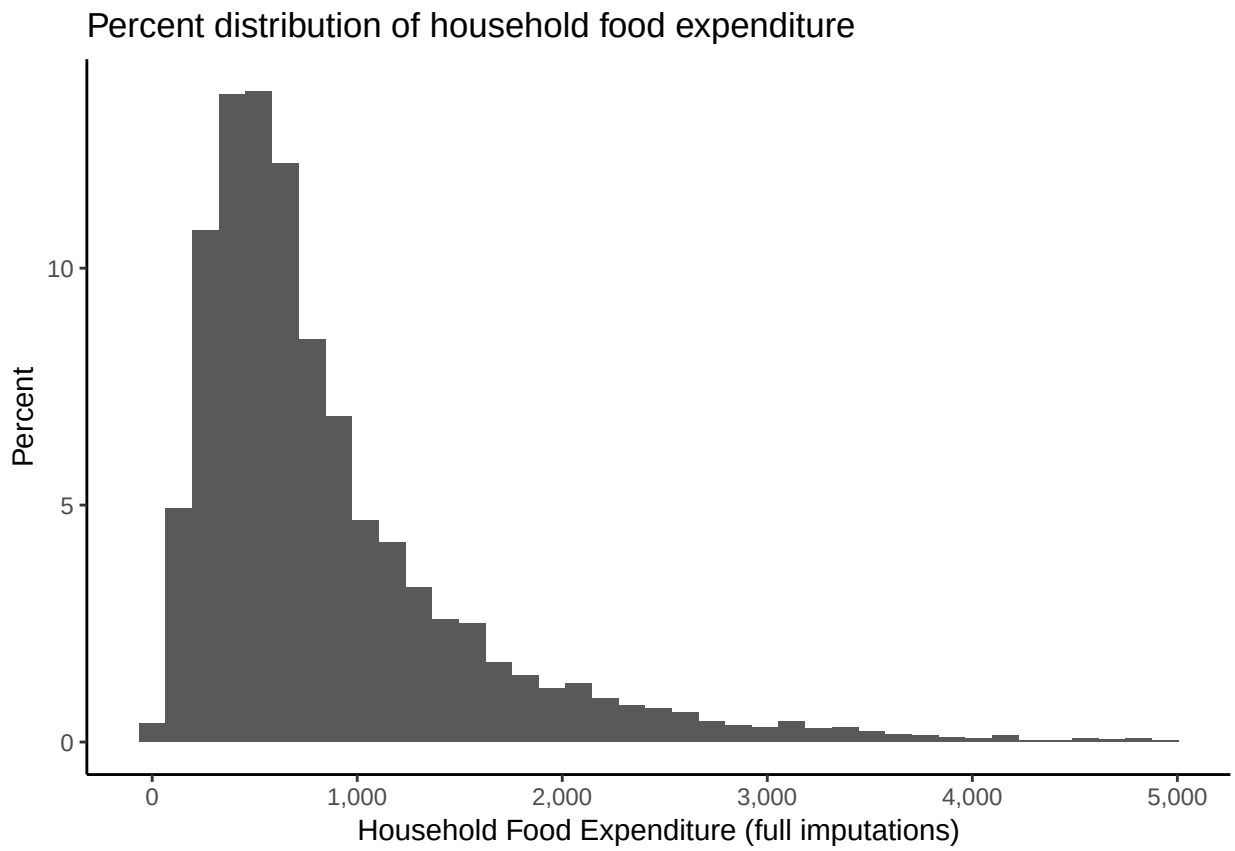
While this graph is more informative, we still seem to have some outliers. It would be interesting to find out how many households have expenditure above R5000. To do this we can type:

```
nids%>%filter(food>5000 & hhrestrict==1) %>%nrow
```

```
## [1] 25
```

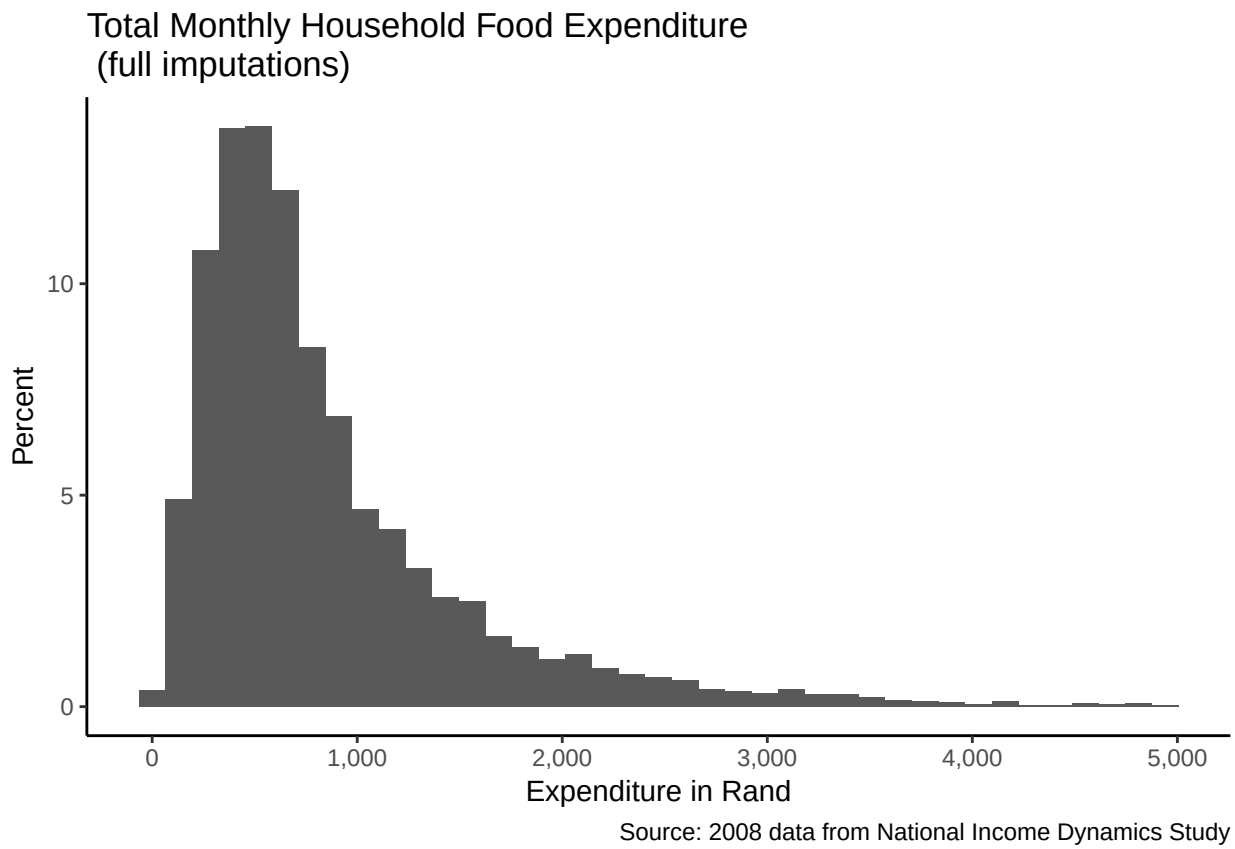
You should get an answer of 25. Since 25 is not too large a number out of the 7305 observations we have for food expenditure, let's restrict our graph to only those spending under R5000 on food.

```
nids%>%
  filter(food<5000 & hhrestrict==1)%>%
  ggplot(., aes(x = food, y = (..count..)/sum(..count..)*100)) +
  scale_x_continuous(breaks=seq(0,5000,1000), labels = comma) +
  geom_histogram(binwidth = 130) +
  xlab("Household Food Expenditure (full imputations)") +
  ylab("Percent") +
  ggtitle("Percent distribution of household food expenditure") +
  theme_classic()
```



We can customise the above graph even more to make the title more informative and to include important additional information such as the source of the data used.

```
nids%>%
  filter(food<5000 & hhrestrict==1)%>%
  ggplot(., aes(x = food, y = (..count..)/sum(..count..)*100)) +
  scale_x_continuous(breaks=seq(0,5000,1000), labels = comma) +
  geom_histogram(binwidth = 130) +
  labs(x = "Expenditure in Rand",
       y = "Percent",
       title = "Total Monthly Household Food Expenditure \n (full imputations)",
       caption="Source: 2008 data from National Income Dynamics Study") +
  theme_classic()
```



This is much better!

We can see that our graphing has already improved substantially. This graph tells us, among other things, that total monthly food expenditure is concentrated below R2000 and that values around and below R1000 per month are very frequent. We could be more specific if we estimated the fraction of the observations in each bin below the R1000 mark. For example, it appears that about percent 70% of the observations are below R1000 per month.

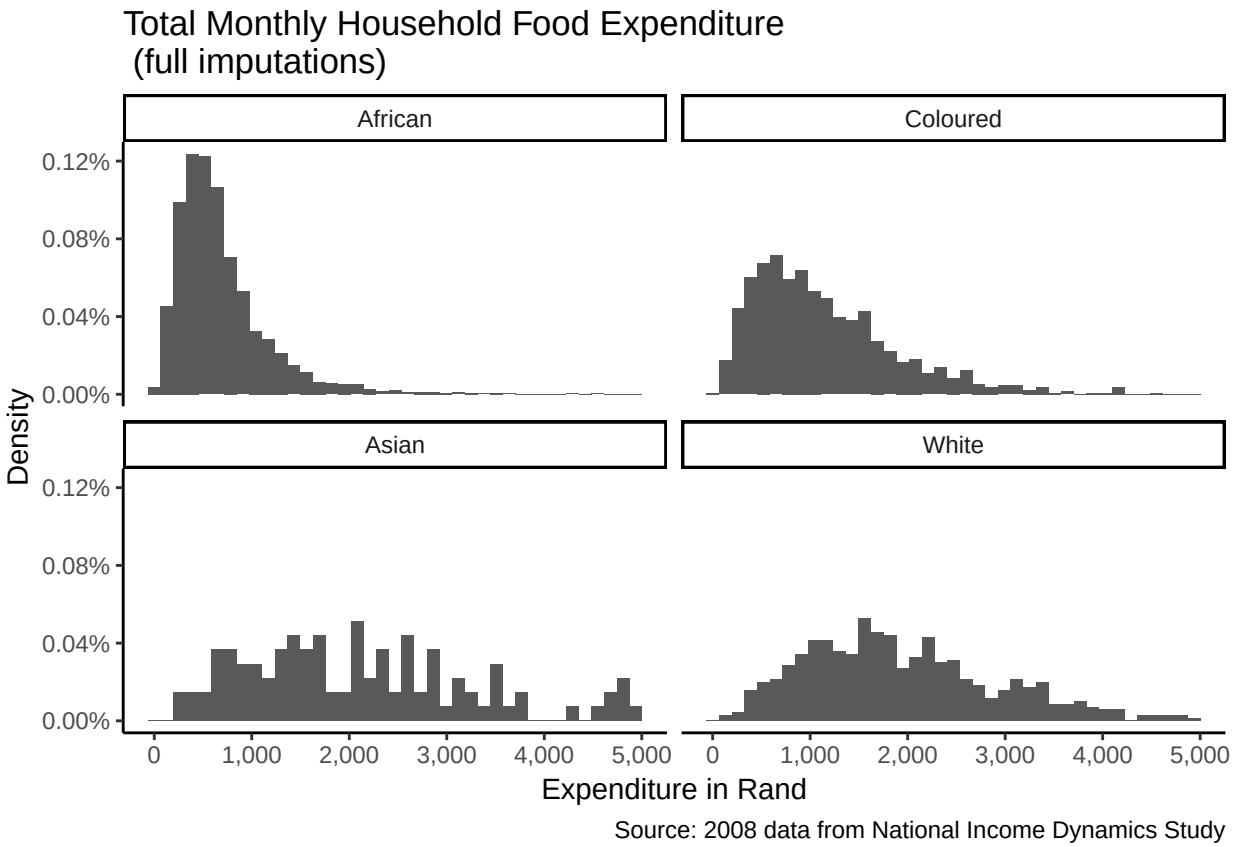
4.4.2 Facetting

Faceting splits the data into subsets to create multiple variations of the same graph (paneling).

Let's take our analysis a step further and ask the question: Does this graph look the same for Indian/Asians and Africans? We can establish the answer using the `facet_*` (`facet_wrap` or `facet_grid`). In the following commands, we filter food expenditure, restricting household size and non missing race (`filter(w1_h_expf<5000 & hhrestrict==1 & !is.na(w1_best_race))`) before plotting the previous graph and facet by race (`facet_wrap(~w1_best_race, ncol=2)`).

```
nids%>%
  filter(food<5000 & hhrestrict==1 & !is.na(race))%>%
  ggplot(., aes(x = food, y = ..density..)) +
  scale_x_continuous(breaks=seq(0,5000,1000), labels = comma) +
  geom_histogram(binwidth = 130) +
  labs(x = "Expenditure in Rand",
       y = "Density",
       title = "Total Monthly Household Food Expenditure \n (full imputations)",
       caption="Source: 2008 data from National Income Dynamics Study") +
```

```
  scale_y_continuous(labels = percent_format()) +
  facet_wrap(~race, ncol=2) +
  theme_classic()
```



We now see how the distribution of food expenditure varies by racial group. `ggplot2` repeats the graph for each of the categories of the variable that we specify (i.e. race), making comparison between the groups far easier!

4.5 Worked example 1: What is the most common age at which people start smoking?

Data analysis can be a tricky business. In most cases answering a research question will require careful and systematic exploration of the data before we can even begin to think about running regressions and using higher level statistical analysis. Graphs are often a good way to get to know your data! But you need to know which graph to use. The worked examples we present in this chapter are intended to not only refine your graphing skills, but also give you some idea of how we need to scrutinize variables before we go ahead and use them blindly. The worked examples share a common focus on issues of health. In this first example we want to establish the most common age at which people start smoking.

In the health themed examples we focus on adults, thus we want to restrict our sample to individuals aged 20 or older. To do this we create a new dummy variable which has a value of one for a respondent of age 20 and above, and a 0 otherwise.

```
nids<-nids %>%
  mutate(adult20 = ifelse(age>=20 & !is.na(age),1,0))
```

In the NIDS adult questionnaire, question J29 asks "How old were you when you first smoked cigarettes regularly?" In the data is called `w1_a_j29`.

```
nids%>%
  ungroup() %>%
  filter(adult20==1) %>%
  select(w1_a_j29) %>%
  summary
```

```
##     w1_a_j29
##  Min.   :-9.00
##  1st Qu.:15.00
##  Median :17.00
##  Mean   :16.14
##  3rd Qu.:20.00
##  Max.   :67.00
##  NA's   :13317
```

Let's dig a bit further and examine this question in the adult questionnaire. From the **a** in the variable name tells us that only people who answered the adult questionnaire should have a value for this variable. The questionnaire tells us that adult respondents were only meant to be asked question J29 if they currently smoke (J26==1) or have ever smoked (J27==1). Is the data we have in line with this? Do check we need to construct an ever-smoked variable which tells us which individuals currently smoke or have ever smoked?

```
nids<-nids%>%
  mutate(eversmoke = ifelse((w1_a_j26==1 | w1_a_j27==1) & adult20==1,1,0))
```

Thus, when `eversmoke==1` the person answered yes to J26 or J27. Now we want to check if there are there any respondents who answered J29 even though they said they had never smoked:

```
nids%>%
  filter(!is.na(w1_a_j29) & eversmoke!=1 & adult20==1) %>%
  nrow
```

```
## [1] 51
```

So there are 51 observations where respondents gave an answer to J29 even though they had previously said they never smoke. Next we check whether these 51 observations were valid answers or whether they were simply coded with negative non-response codes,

```
nids%>%
  filter(!is.na(w1_a_j29) & eversmoke!=1 & adult20==1)%>%
  select(w1_a_j29)
```

```
## Adding missing grouping variables: `hhid`
```

```
## # A tibble: 51 x 2
## # Groups:   hhid [49]
##      hhid w1_a_j29
##     <int>    <int>
##  1 101100       -8
##  2 101929       -3
##  3 101954       -3
##  4 102772       11
##  5 103120       -3
##  6 103151       -3
##  7 103210       51
##  8 103317       -3
##  9 103333       -3
```

```
## 10 103421       -3
## # ... with 41 more rows
```

Most of values are assigned a missing code. But there are 5 cases where a valid age is given. What would you do with these respondents?

Now let's consider the non-response categories. From the documents, we know what values are assigned to "Don't know", "Refused", "Not applicable" and "Missing". The non-responses are all given negative values. How many people did not give valid ages in response to this question of when they started smoking?

```
nids%>%
  filter(w1_a_j29 <0 & adult20==1)%>%
  nrow
```

```
## [1] 455
```

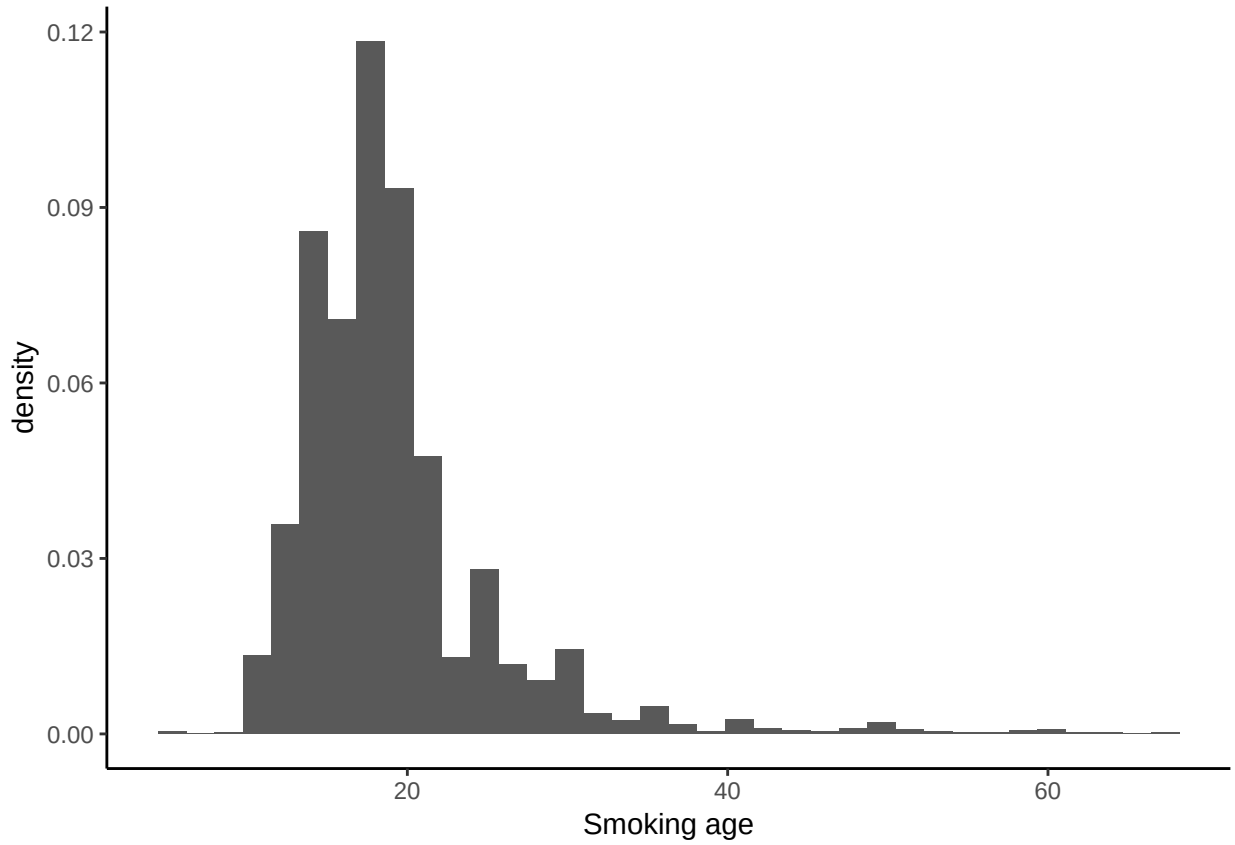
We will want to exclude these negative observations from our analysis, so let's create a new variable called `smoking_age` that only contains the non-negative responses. We also restrict the variable to only include adults over the age of 20.

```
nids<-nids%>%
  mutate(smoking_age = ifelse(w1_a_j29 > 0 & !is.na(w1_a_j29) & adult20==1,w1_a_j29,NA))
```

Now we can try and answer our question: What is the most common age at which people in the sample started smoking? While our `smoking_age` variable is categorical, it takes many different values and therefore a histogram might be best to illustrate the distribution. Go ahead and graph this variable, you will need to refine the x-axis labels to see the age where most people start smoking. You should get something like the graph below:

```
ggplot(nids, aes(x=smoking_age, y = ..density..)) +
  geom_histogram(binwidth = 1.7714286) +
  xlab("Smoking age") +
  theme_classic()
```

```
## Warning: Removed 27895 rows containing non-finite values (stat_bin).
```



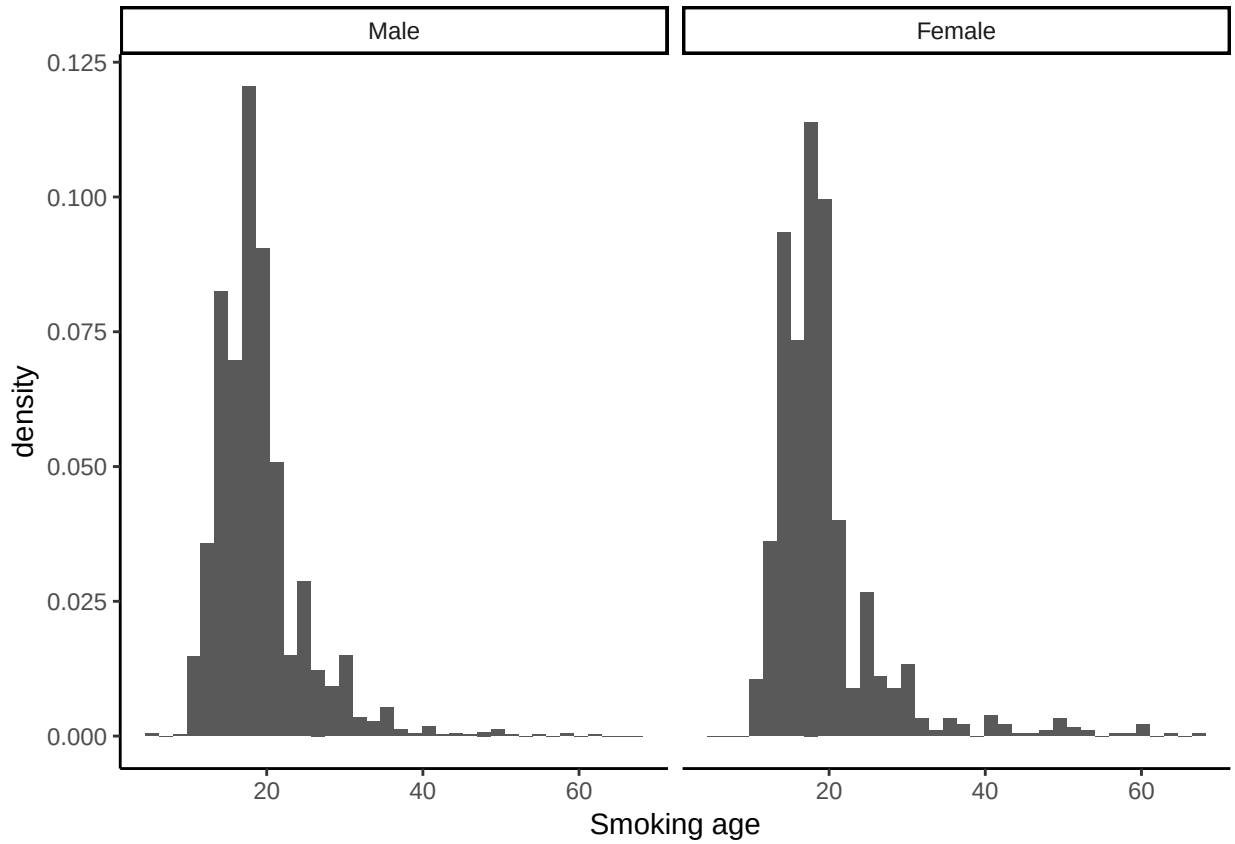
From the histogram it is evident that the most common ages for people to start smoking are those between 15 and 18.

Now we consider whether there are differences between the age at which males and females start smoking. The best way to see this is by generating a new histogram, facetting by gender (go ahead and to this). Your graph should look something like the one below:

Density

```
nids %>%
  ggplot(., aes(x=smoking_age, y = ..density..)) +
  geom_histogram(binwidth = 1.7714286) + #binwidth chosen to replicate the stata output
  xlab("Smoking age") +
  facet_wrap(~gender) +
  theme_classic()
```

```
## Warning: Removed 27895 rows containing non-finite values (stat_bin).
```



Proportion

```
#ggplot(nids, aes(x=smoking_age, y = (..count..)/sum(..count..))) +
#  geom_histogram(binwidth = 1.7714286) +
#  xlab("Smoking age") +
#  facet_wrap(~gender)
```

Let's inspect the histogram. The distributions don't seem to differ much by gender. One interesting thing to notice is that there are spikes at age 25 and age 30 in the female distribution on the right. Is it likely that substantially more females started smoking at age 25 than at age 24 or at age 30 instead of age 29?

No. While a few might find 30 a depressing age, the most likely explanation is that people can't remember the exact age at which they started smoking and so they approximate it with the closest age that is a multiple of 5 or 10 (Remember, the people getting asked when they started smoking might be 31 or they might by 91 years old). This is called 'age heaping' and is a common phenomenon in the reporting of ages in surveys. It is useful to be aware of it.

Now go ahead and try the following questions on your own!

1. Find a variable that gives the number of people that belong to each household. Use it to create a new variable for food expenditure per person, called foodpc and label this variable "per capita food expenditure".

Question 1 Answer

2. Generate a histogram of household per capita food expenditure. How many households spend above R2000 per person? Graph a histogram excluding households that spend above R2000 per person. What would this histogram have looked like if we had instead considered foodpc at the individual level? Question 2 Answer

3. Generate a histogram of household per capita food expenditure by gender of the resident household head (for foodpc $<$ 2000). Comment on your findings. Question 3 Answer

4.6 Pie charts

Similar to Histograms, pie graphs are graphical tools which inform the reader of the distribution of any particular variable. In contrast to histograms, however, Pie graphs are used for categorical variables only.

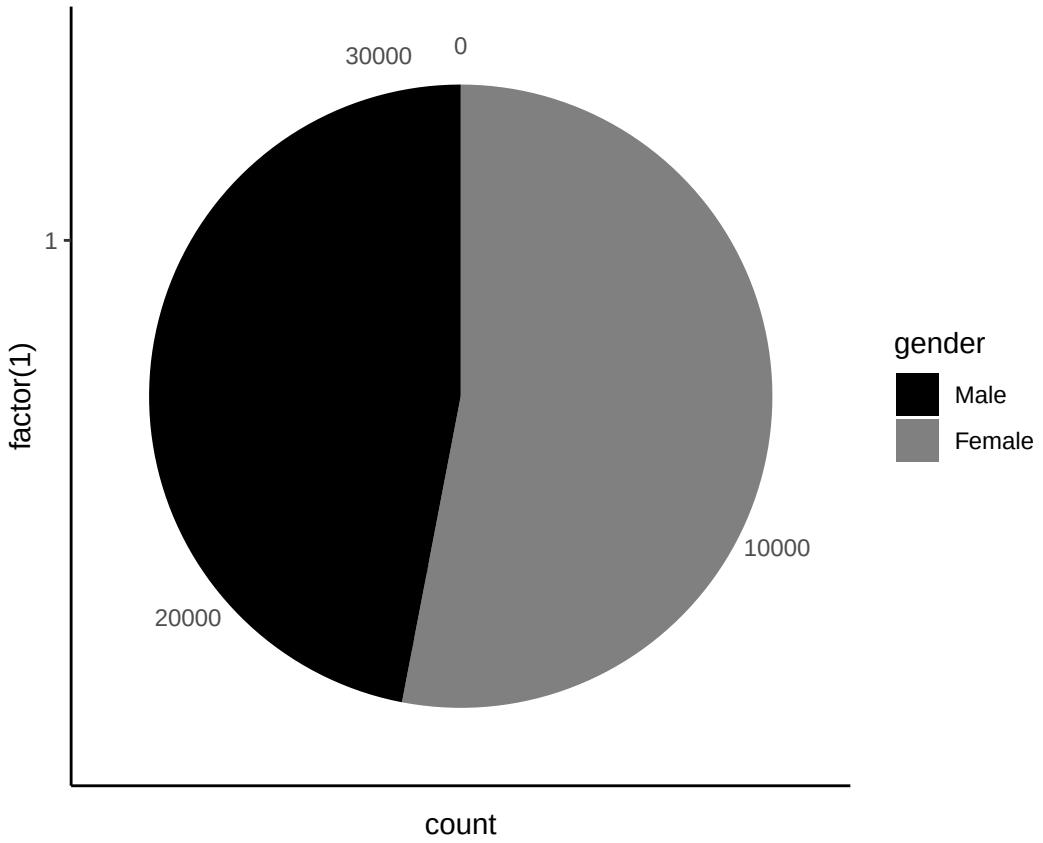
Here we introduce pie charts for completeness only as they are considered not to be a good way of presenting information.

4.7 Investigating newborn and infant gender

We will start our explanation of pie charts with an easy example: suppose we want to know whether there are an equal number of men and women in South Africa, one way we could find out is by producing a pie chart for gender. (Note, however, that we will need to ask ourselves whether our sample is perfectly representative of the South African population)

In ggplot2, a pie chart is emplemented as a bar chart on a polar coordinate system. We create the pie chart as follows:

```
ggplot(nids, aes(x = factor(1), fill = gender)) +
  geom_bar(width = 1) + coord_polar(theta = "y") +
  scale_fill_grey(start=0, end=0.5) +
  theme_classic()
```



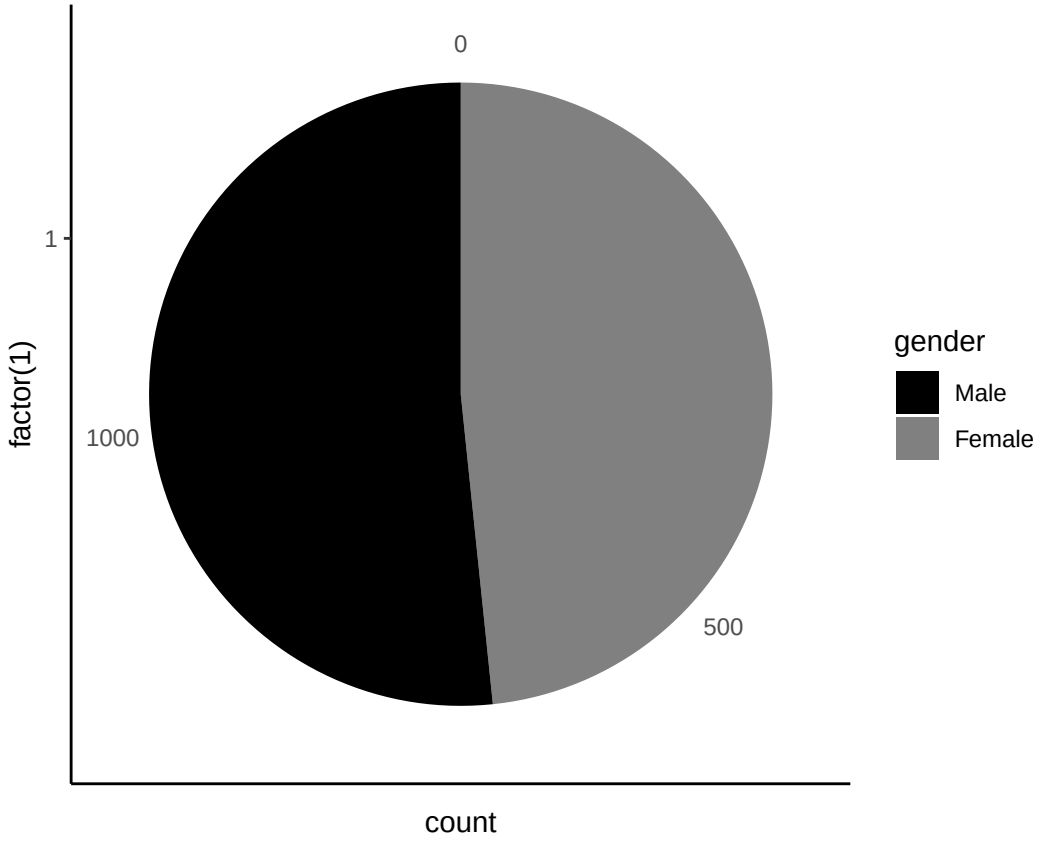
The pie chart tells us that there are more women than men in South Africa!

Subsetting

Now suppose we want to dig a bit deeper and investigate the gender distribution of new-borns and infants in South Africa. To do this we need to subset the `age` variable for being either 0 or 1.

```
nids%>%
  filter(age==0 | age==1)%>%
  select(gender)%>%
  ggplot(., aes(x = factor(1), fill = gender)) +
  geom_bar(width = 1) +
  coord_polar(theta = "y") +
  scale_fill_grey(start=0, end=0.5)+
  theme_classic()
```

```
## Adding missing grouping variables: `hhid`
```

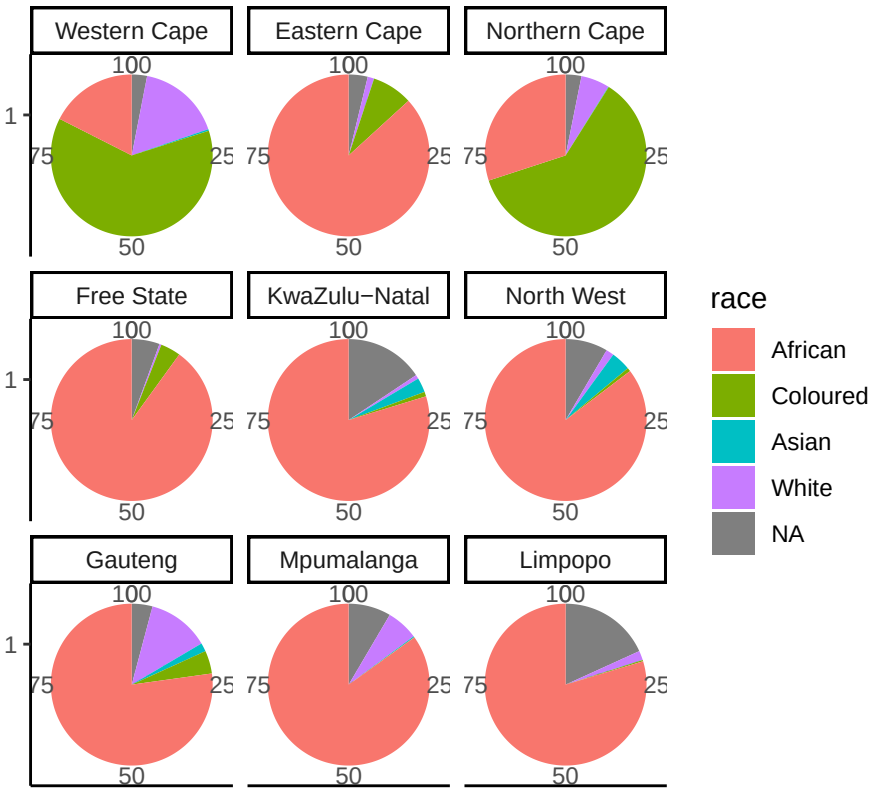


The pie chart reveals that the majority of infants are male. These results are rather interesting if we compare them to our first pie chart! What do you think might explain the difference between the two graphs? It might be that there is or has been a mortality differential between men and women, where more female infants grow into adults. Can you think of any other explanations?

4.7.1 Exploring the distribution of race across provinces

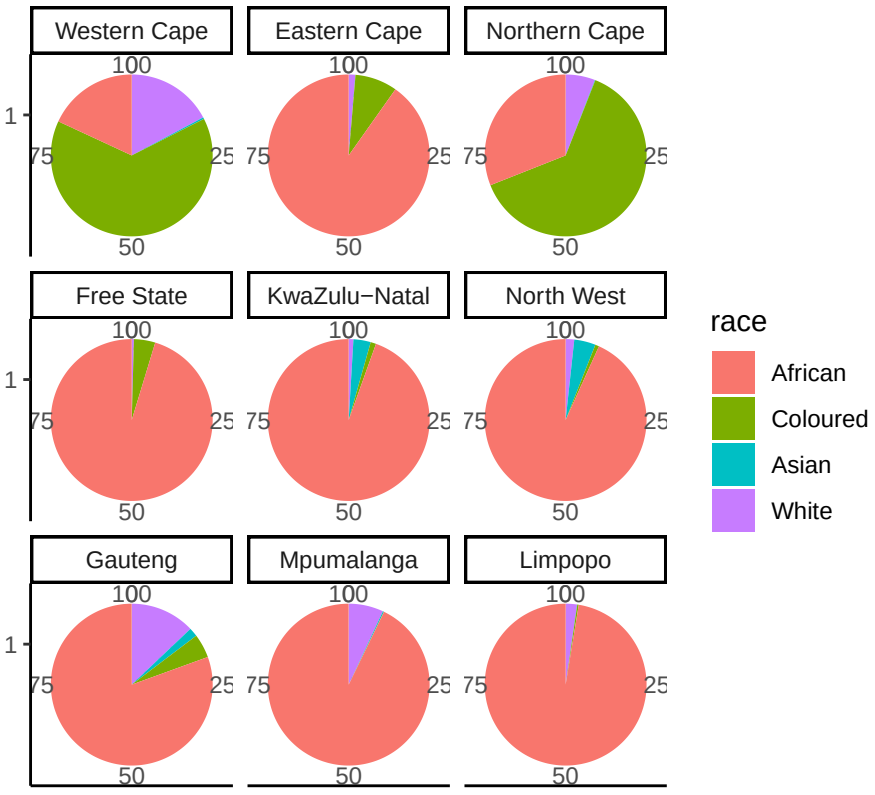
We now turn our attention to the race variable. Suppose we are interested in assessing whether the distribution of race varies significantly over different provinces or perhaps we would like to know which province has the largest Indian/Asian population. Pie charts can be particularly informative in this case since there are 9 separate categories (i.e. the provinces) for which we will need to compare distributions of race! We proceed as follows:

```
nids %>%
  group_by(province,race) %>%
  summarise(freq=n()) %>%
  group_by(province) %>%
  mutate(percent = round(freq/sum(freq)*100,2))  %>%
  ggplot(., aes(x=factor(1), y=percent, fill = race)) +
  geom_bar(stat = "identity",width = 1) +
  facet_wrap(~province) +
  labs(x="",y="") +
  coord_polar(theta = "y") +
  theme_classic()
```



There are some people with unknown race, exclude by filtering them out (`na.omit`) to produce:

```
nids %>%
  group_by(province,race) %>%
  summarise(freq=n()) %>%
  na.omit() %>%
  group_by(province) %>%
  mutate(percent = round(freq/sum(freq)*100,2))  %>%
  ggplot(., aes(x=factor(1), y=percent, fill = race)) +
  geom_bar(stat = "identity",width = 1) +
  facet_wrap(~province) +
  labs(x="",y="") +
  coord_polar(theta = "y") +
  theme_classic()
```



Examine the graph and try to answer the questions we posed above. It should be relatively easy to see the relevant answers. Firstly, we see that the Western and Northern Cape have rather different racial compositions in comparison to the other 7 provinces. In particular, they have many more coloured residents and a comparatively smaller percentage of African residents. Secondly, we see that either Kwa- Zulu Natal or the North West have the largest fraction of Indian/Asians.

4.7.2 Urban vs. rural, where are South Africans living?

We investigate the distribution of the urban or rural location of a given household (given by the `w1_hhgeo` variable) as a way to show how we can format our pie charts to make them more presentable. As with any other analysis, we must know what the variable we're working with "looks" like. In our case, we can do this by simply tabulating `w1_hhgeo`.

```
table(nids$w1_hhgeo)
```

```
##
##           Rural formal Tribal authority areas           Urban formal
##                   3010                  14114                  12048
##         Urban informal
##                   1998
```

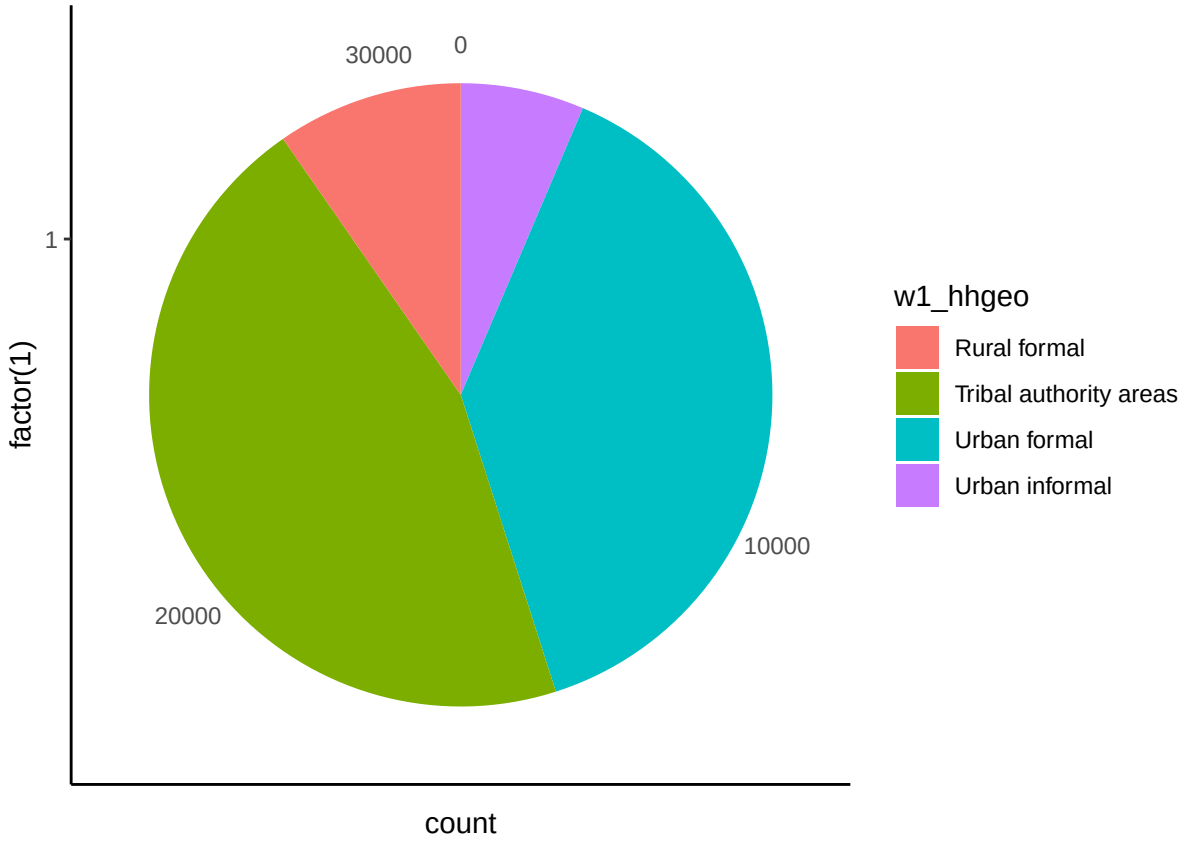
Or

```
nids %>%
  group_by(w1_hhgeo) %>%
  summarise(freq=n()) %>%
  mutate(percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(freq)/sum(freq)*100,2))
```

```
## # A tibble: 4 x 4
##   w1_hhgeo                freq percent cum_percent
##   <fct>                  <int>   <dbl>       <dbl>
## 1 Rural formal            3010    9.66        9.66
## 2 Tribal authority areas 14114   45.3         54.9
## 3 Urban formal           12048   38.6         93.6
## 4 Urban informal          1998    6.41       100
```

Now that we have a sense of what to expect we turn our attention to graphing this variable.

```
nids%>%
  ggplot(., aes(x = factor(1), fill = w1_hhgeo)) +
  geom_bar(width = 1) +
  coord_polar(theta = "y")+
  theme_classic()
```



We see that the greatest fraction of the NIDS households is located in Urban Formal settings.

4.8 Worked example 2: Are there differences between the age at which men and women start smoking?

We return to our earlier example where we considered the age at which respondents started smoking using the smoking_age variable. In the previous example we generated a histogram of the `smoking_age` variable. However, it might also be useful to look at more aggregated age categories to find out what percentage of smokers start smoking before age 16 (the legal age an individual can purchase tobacco), in their late teens,

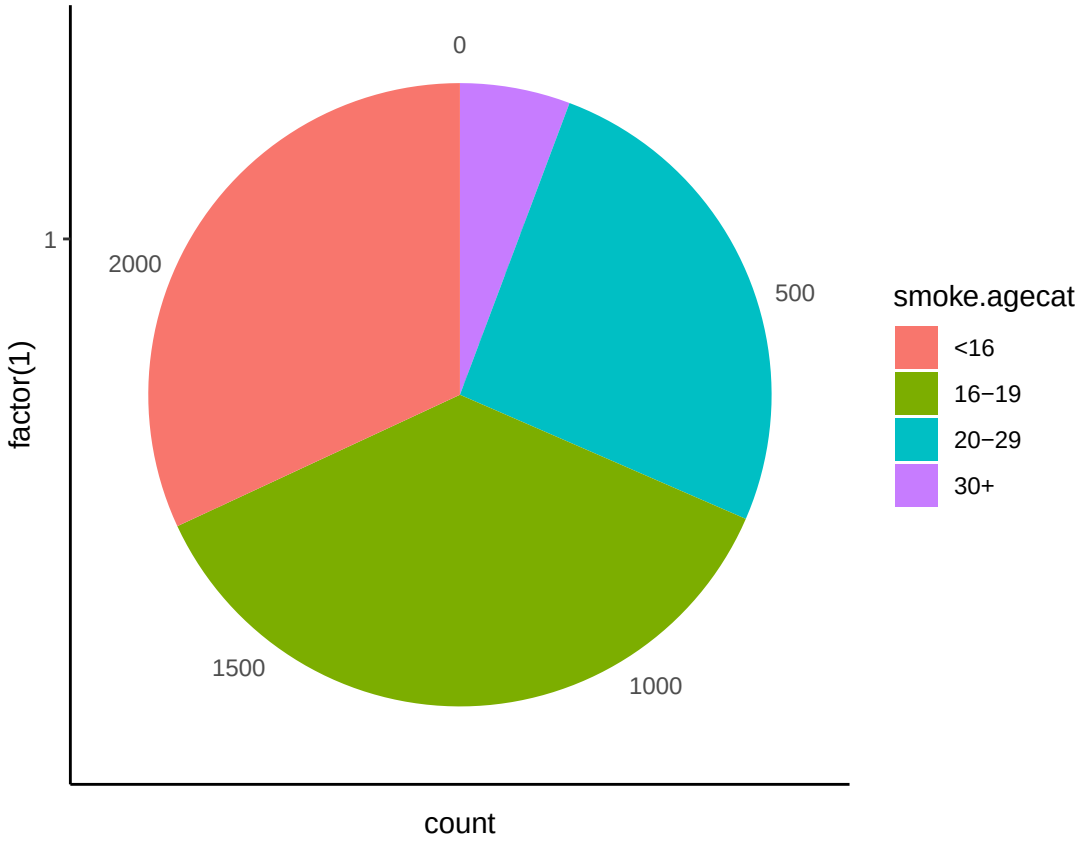
their twenties or later in life. To do this we need to generate a new variable separating the `smoking_age` variable into the categories <16, 16-19, 20-29 and 30+.

```
nids$smoke.agecat<-NA
nids$smoke.agecat[which(nids$smoking_age>=0 & nids$smoking_age<=15)]<-1
nids$smoke.agecat[which(nids$smoking_age>16 & nids$smoking_age<=19)]<-2
nids$smoke.agecat[which(nids$smoking_age>20 & nids$smoking_age<=29)]<-3
nids$smoke.agecat[which(nids$smoking_age>30 & nids$smoking_age<=67)]<-4

nids$smoke.agecat<-factor(nids$smoke.agecat, levels = 1:4, labels = c("<16","16-19","20-29","30+"))
```

Now we can graph this variable using a pie chart (go ahead and do it yourself in R). What is this graph telling us? Compare it to the histogram we produced in Worked Example 1 – which is more informative in this instance?

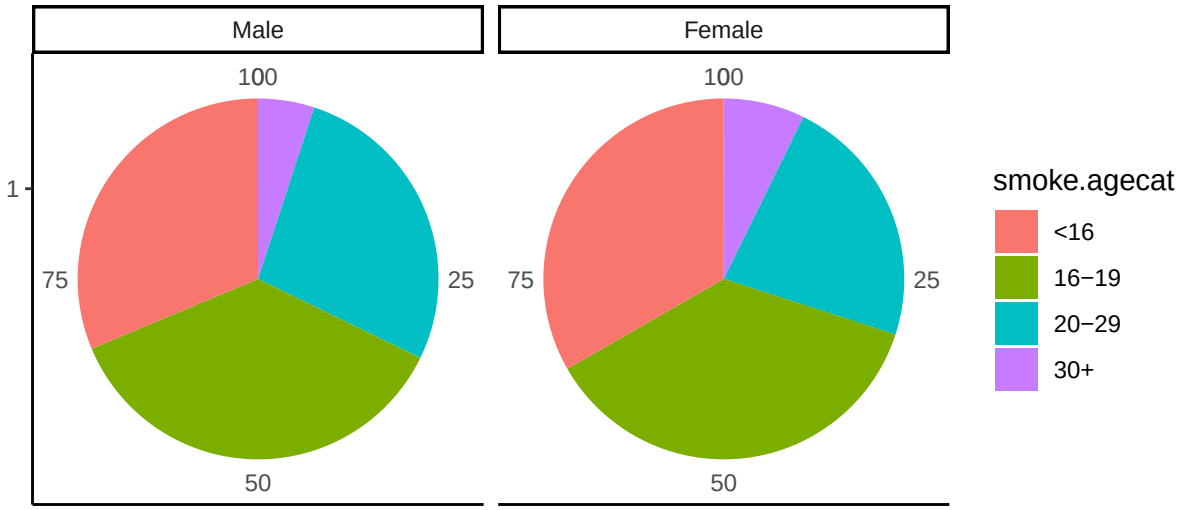
```
nids%>%
  filter(!is.na(smoke.agecat))%>%
  ggplot(., aes(x = factor(1), fill = smoke.agecat)) +
  geom_bar(width = 1) +
  coord_polar(theta = "y")+
  theme_classic()
```



Suppose, however, that we would also like to consider men and women separately. We do this by re-running the graph and faceting by gender. The code is once again presented below.

```
nids %>%
  group_by(gender,smoke.agecat) %>%
  summarise(freq=n()) %>%
```

```
  na.omit() %>%
  group_by(gender) %>%
  mutate(percent = round(freq/sum(freq)*100,2))  %>%
  ggplot(., aes(x=factor(1), y=percent, fill = smoke.agecat)) +
  geom_bar(stat = "identity",width = 1) +
  facet_grid(.~gender) +
  labs(x="",y="") +
  coord_polar(theta = "y")+
  theme_classic()
```



Looking at this pie chart we see that around a quarter of both male and female smokers started smoking before the age of 16 – in other words before the age at which they were legally allowed to buy tobacco. We also see that the majority of smokers start smoking by their late teens. Very few smokers started smoking after the age of 30. This is very useful in that it tells policy makers who wish to reduce the incidence of smoking in South Africa that they should target teenagers in their campaigns.

How many people in the sample started smoking before the legal age at which they were allowed to buy tobacco (16 in South Africa)?

```
nrow(subset(nids, subset=smoke.agecat=="<16"))
```

```
## [1] 787
```

Is there age heaping at age 18, the legal age of drinking in South Africa?

```
nrow(subset(nids, subset=smoking_age==17))
```

```
## [1] 251
```

```
nrow(subset(nids, subset=smoking_age==18))
```

```
## [1] 436
```

```
nrow(subset(nids, subset=smoking_age==19))
```

```
## [1] 214
```

Around double the number of people say they started smoking at 18 compared to either 17 or 19.

4.8.1 Exercise

4. Create a categorical variable called Poverty and set it equal to 1 if income is less than 800 and zero otherwise(i.e. we are looking at households in the lowest 10% of the income variable). Is Poverty a household or individual level variable? Graph the fraction of households in poverty by province. Now, can you make the "Income < 800" category explode from the rest of the pie chart?

Question 4 Answer

5. Find the satisfaction variable and rename it "Satisfaction". Is satisfaction an individual or household level variable? Create a pie chart with Satisfaction as the categorical variable across race, (excluding non-response values). Is this graph effective?

Question 5 Answer

4.9 Bar graphs

Question 5 above demonstrated why it may not always be optimal to use a pie chart, especially when we have a large number of categories. In this case the Bar Graph can be a useful alternative option. Bar charts can be used not only for categorical variables but also for continuous variables. The reason for this is that the bar chart is quite diverse in its applicability as it allows us to incorporate summary statistics in our graphs.

4.9.1 Investigating Satisfaction levels in South Africans

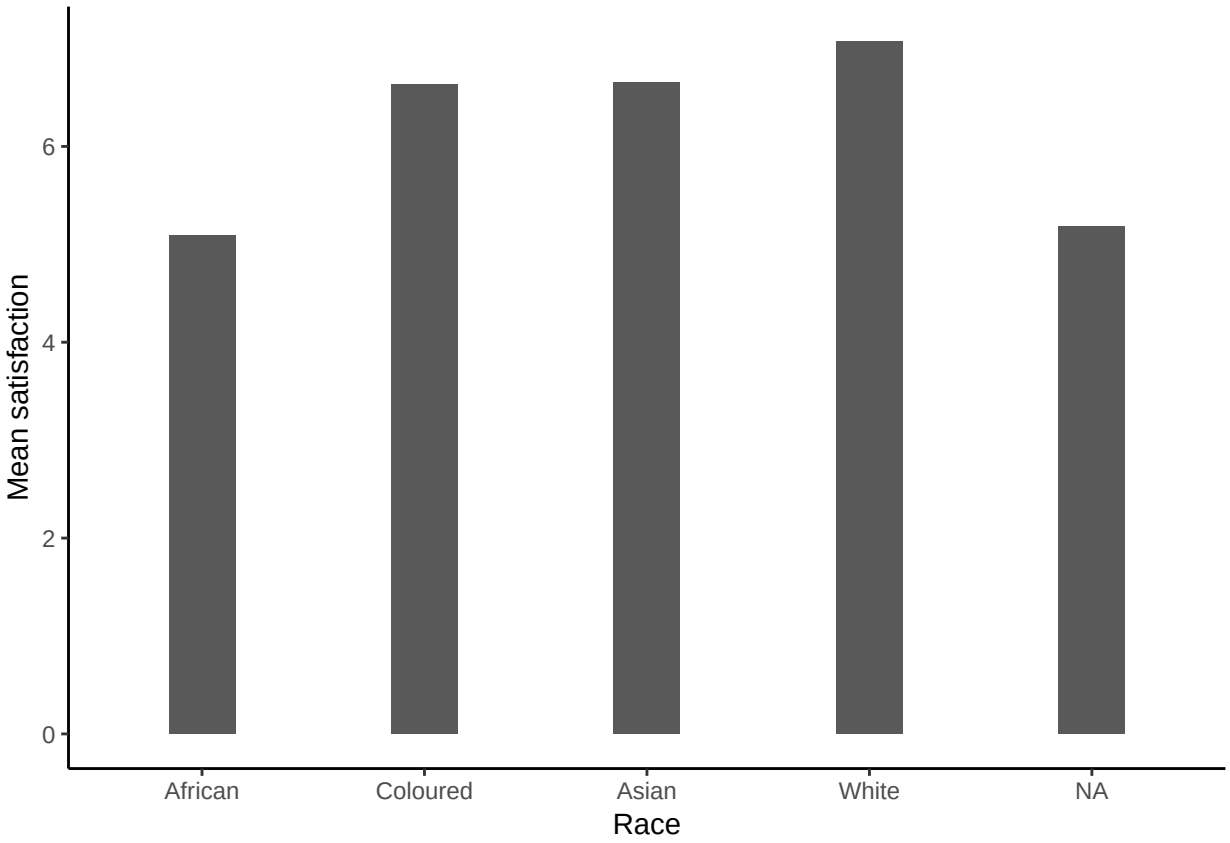
As an example let's generate a bar graph investigating how the `satisfaction` variable might differ by race (NB: Make sure that you have done question 5 or you won't have the `satisfaction` variable).

Q5

```
nids<-nids%>%
  mutate(satisfaction = ifelse(w1_a_m5>=1, w1_a_m5, NA))
```

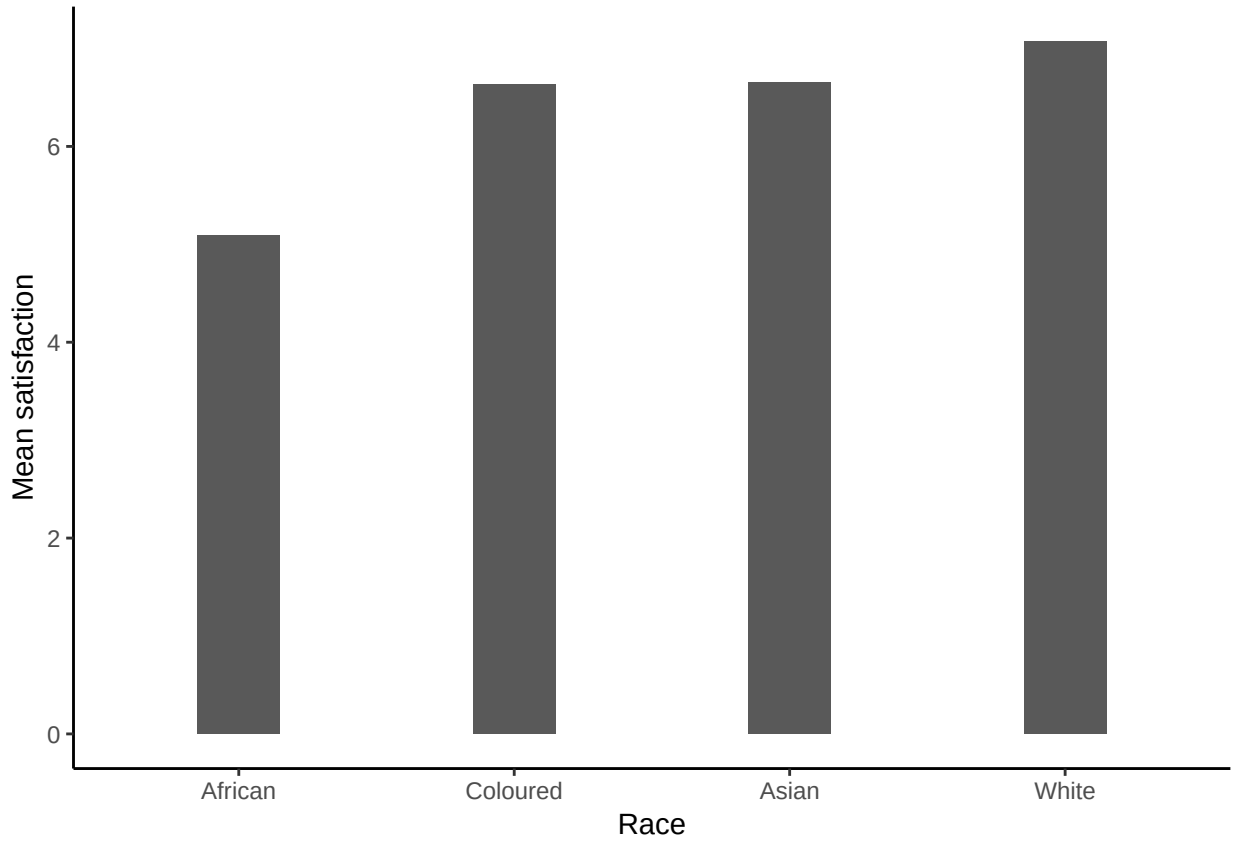
The code that generated this graph:

```
nids%>%
  group_by(race)%>%
  summarise(msatis = mean(satisfaction, na.rm=TRUE))%>%
  ggplot(., aes(x = race, y = msatis)) + geom_bar(stat = "identity", width=.3) +
  xlab("Race") + ylab("Mean satisfaction") +
  theme_classic()
```



Filter missing race:

```
nids%>%
  filter(!is.na(race))%>%
  group_by(race)%>%
  summarise(msatis = mean(satisfaction, na.rm=TRUE))%>%
  ggplot(., aes(x = race, y = msatis)) + geom_bar(stat = "identity", width=.3) +
  xlab("Race") + ylab("Mean satisfaction") +
  theme_classic()
```

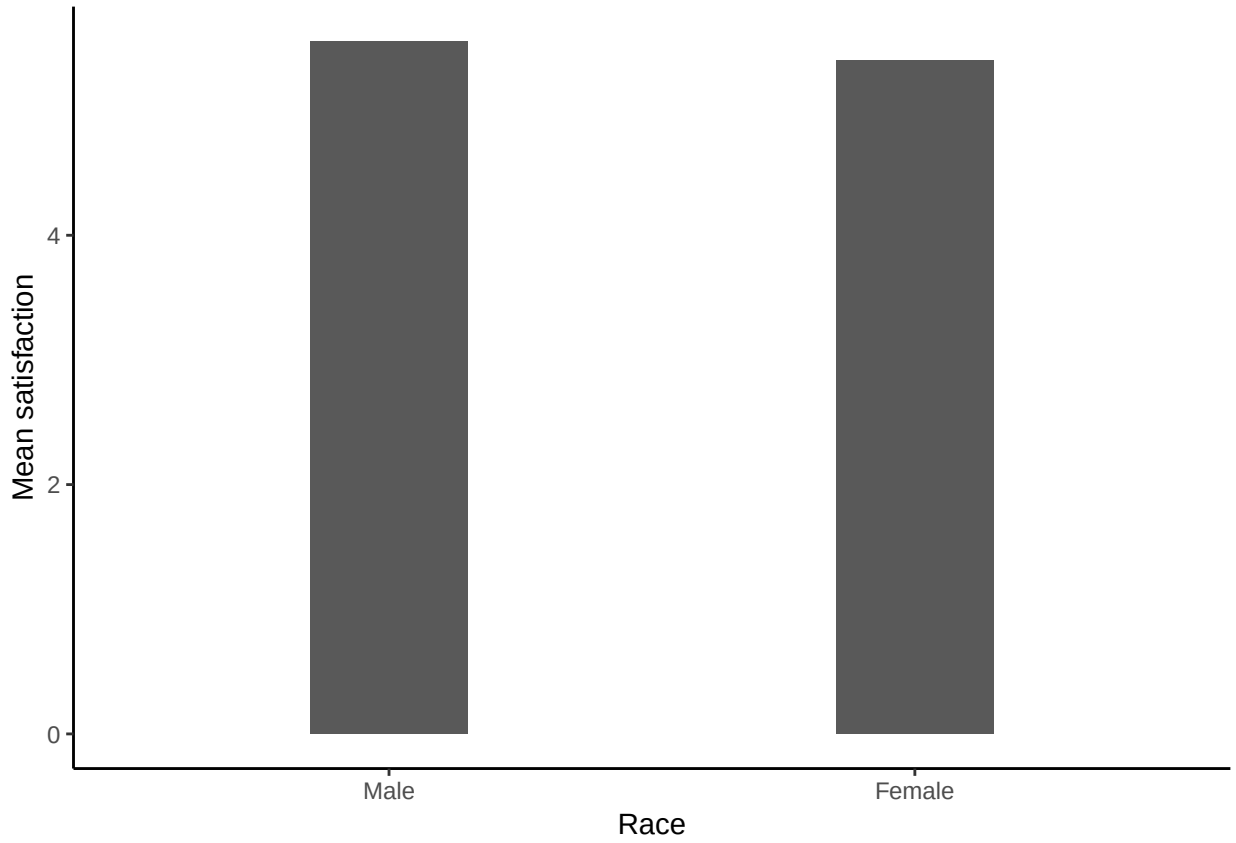


This graph is telling us the mean satisfaction level for each of the race groups. This gives us the information that we are looking for and we can conclude that Whites have the highest level of satisfaction on average, followed by Coloureds and Asian/Indians who share virtually the same mean satisfaction level, while the least satisfied are Africans.

Now let us consider satisfaction over gender. Do you think it is likely that one gender has a higher mean level of satisfaction than the other? Let's take a look:

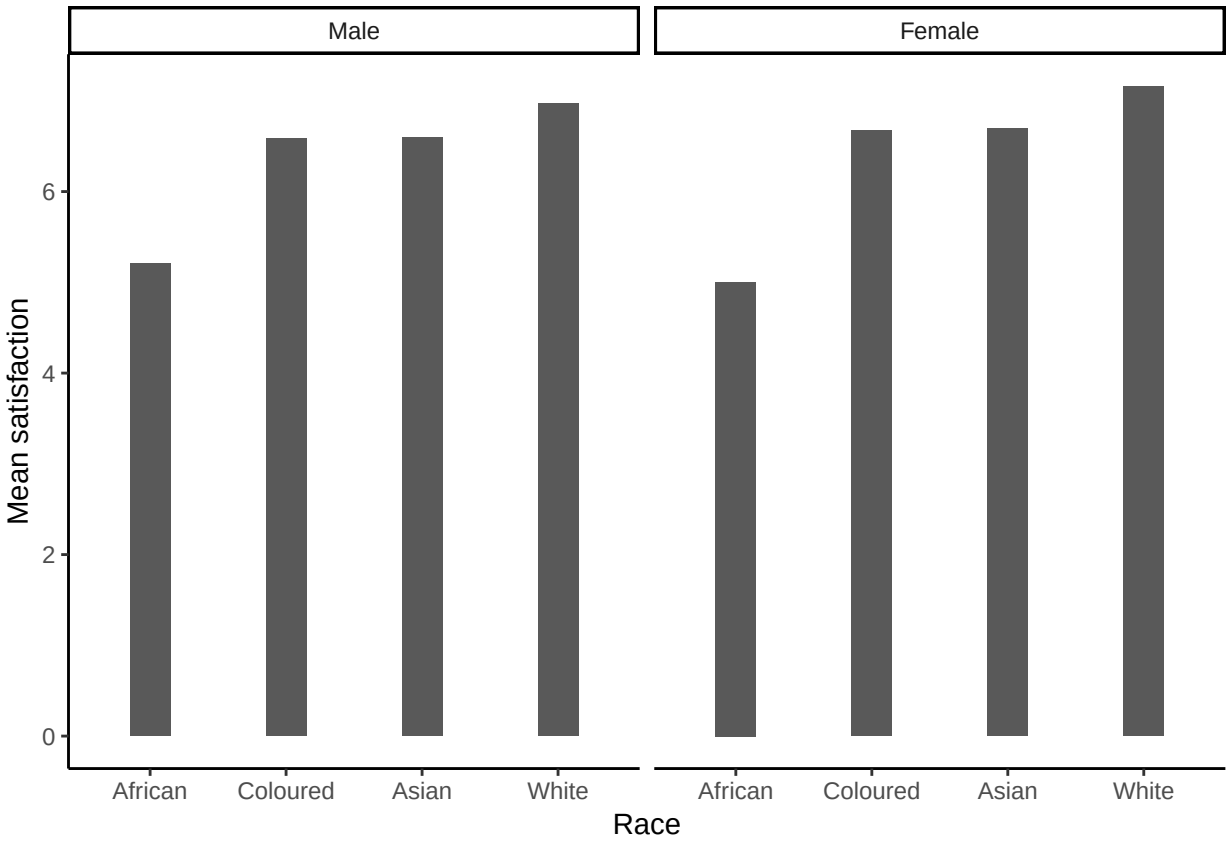
To graph satisfaction over gender we simply do the following:

```
nids%>%
  group_by(gender)%>%
  summarise(msatis = mean(satisfaction, na.rm=TRUE))%>%
  ggplot(., aes(x = gender, y = msatis)) + geom_bar(stat = "identity", width=.3) +
  xlab("Race") + ylab("Mean satisfaction") +
  theme_classic()
```



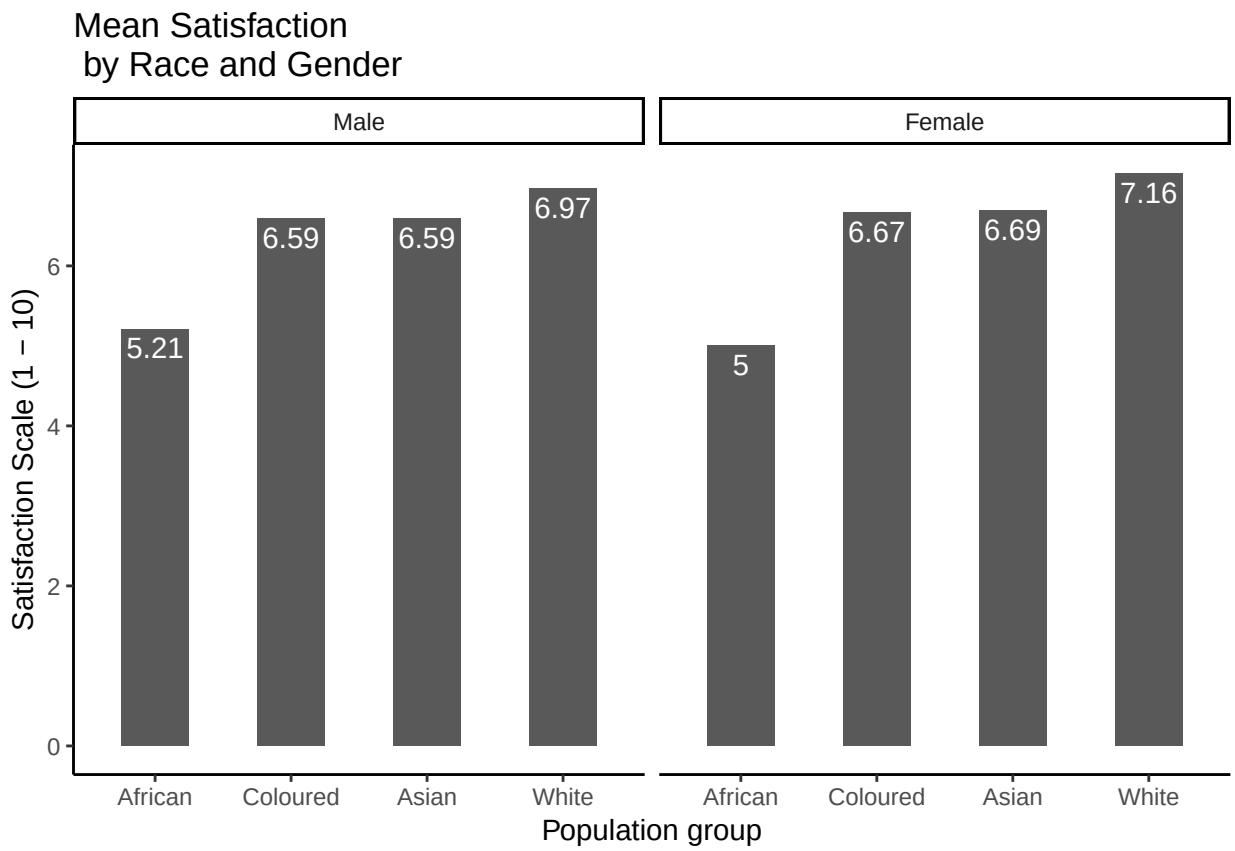
This graph implies that there is not a large difference between men and women when it comes to satisfaction. One of the most useful features of a bar graph is that we can compare across a variety of categories. For instance, let us now consider satisfaction across race and gender. To do this simply:

```
nids%>%
  filter(!is.na(race))%>%
  group_by(race, gender)%>%
  summarise(msatis = mean(satisfaction, na.rm=TRUE))%>%
  ggplot(., aes(x = race, y = msatis)) + geom_bar(stat = "identity", width=.3) +
  xlab("Race") + ylab("Mean satisfaction") +
  facet_wrap(~gender, ncol=2) +
  theme_classic()
```



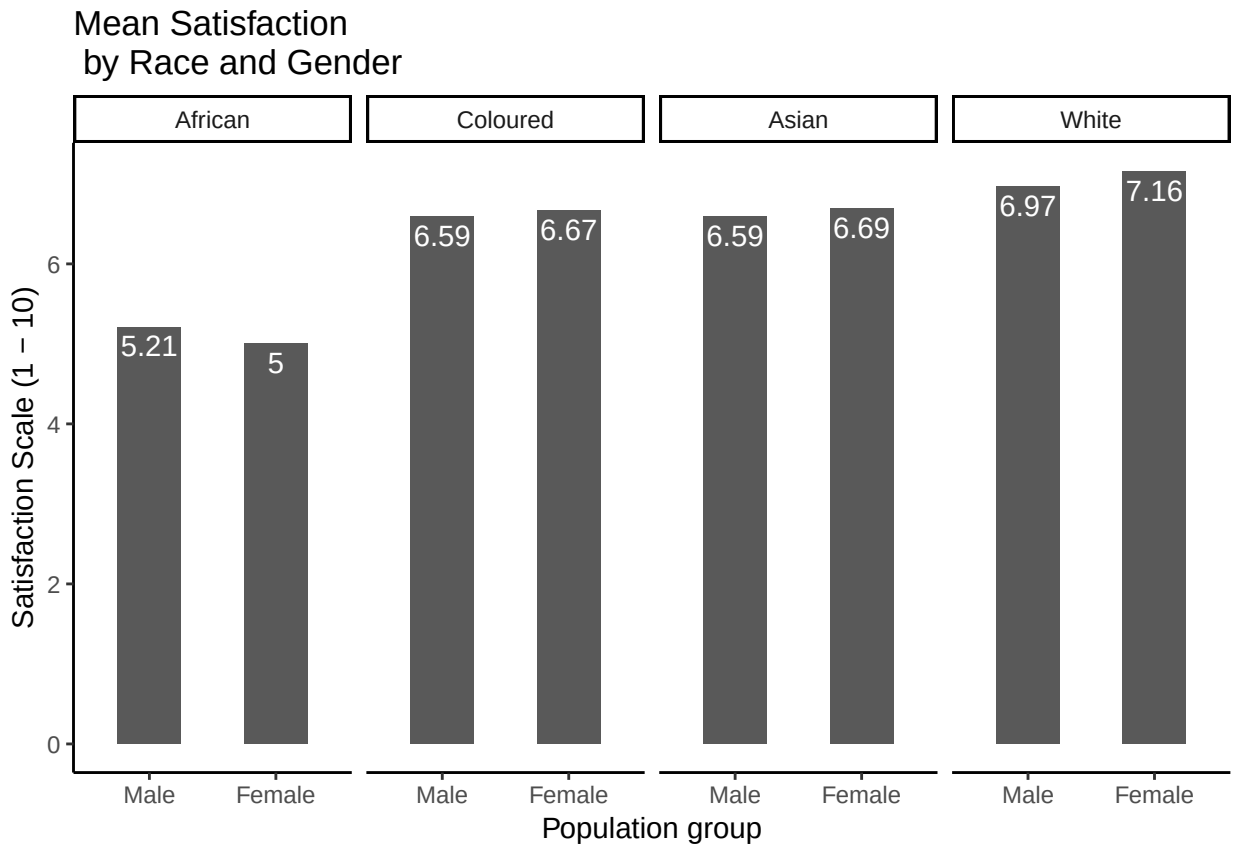
Clearly this graph is in need of some formatting. Below we change the angle of the group 1 labels, add titles, etc.

```
nids%>%
  filter(!is.na(race))%>%
  group_by(race, gender)%>%
  summarise(msatis = mean(satisfaction, na.rm=TRUE))%>%
  ggplot(., aes(x = race, y = msatis)) + geom_bar(stat = "identity", width=.5) +
  labs(x = "Population group", y = "Satisfaction Scale (1 - 10)",title = "Mean Satisfaction \n by Race a
  facet_wrap(~gender, ncol=2) +
  geom_text(aes(label=round(msatis,2)), vjust=1.4, color="white") +
  theme_classic()
```



This is much better! It seems as if the differences that we saw between races persist when we also categorize by gender. In fact, there seems to be very little difference between males and females within each racial category. Clearly race is more closely associated with reported satisfaction in South Africa than gender is. We could also put gender in group 1 and race in group 2 to get essentially the same graph, but represented in a different way:

```
nids%>%
  group_by(race, gender)%>%
  summarise(msatis = mean(satisfaction, na.rm=TRUE))%>%
  na.omit() %>%
  ggplot(., aes(x = gender, y = msatis)) + geom_bar(stat = "identity", width=.5) +
  labs(x = "Population group", y = "Satisfaction Scale (1 - 10)",title = "Mean Satisfaction \n by Race a
  geom_text(aes(label=round(msatis,2)), vjust=1.4, color="white") +
  ggtitle("Mean Satisfaction \n by Race and Gender") +
  facet_grid(~race) +
  theme_classic()
```



This graph places a greater emphasis on the comparison of males and females within racial categories, while the previous one placed a greater emphasis on comparing the satisfaction levels of different racial categories amongst men and women separately.

4.9.2 Worked example 3: Does the frequency of alcohol consumption vary by gender, race or religion?

The NIDS dataset includes questions on both the frequency and quantity of alcohol consumption. In this worked example we make use of bar graphs to find out whether the frequency of alcohol consumption differs by gender or religious affiliation.

From the questionnaire, this coresponds to question `j31`. Now, lets investigate the `w1_a_j31` variable (remember we want to restrict our sample to adults over the age of 20).

```
nids%>%
  filter(adult20==1)%>%
  group_by(w1_a_j31)%>%
  summarise(n=n())
```

```
## # A tibble: 11 x 2
##    w1_a_j31     n
##       <int> <int>
##  1       -8     5
##  2       -3    33
##  3        1  7890
##  4        2  1453
```

1 'I have never drunk alcohol'
2 'I no longer drink alcohol'
3 'I drink very rarely'
4 'Less than once a week'
5 'On 1 or 2 days a week'
6 'On 3 or 4 days a week'
7 'On 5 or 6 days a week'
8 'Every day'

Figure 4.1: Labels for alcohol consumption

```
##  5        3  1695
##  6        4   461
##  7        5   796
##  8        6   252
##  9        7    96
## 10        8   147
## 11       NA  4219
```

Examining the results from the summarise command above, how many different valid categories are there? What are these responses? If you had written the questionnaire, would you have used the same categories?

Suppose we want to plot a bar graph of this variable so that each category has a different bar in the graph. To do this we need to create 'dummy' variables for each category, representing each of the bars in the bar graph. We could do this by creating a dummy variable for each of the 8 possible values of the `w1_a_j31` variable, but we could also choose our own categories as we do here by grouping the 8 given categories into 4 broader categories.

```
#
nids$alc<-NA
#People who have never drunk alcohol
nids$alc[nids$w1_a_j31 ==1 & nids$adult20==1]<-1

#People who drink rarely (no longer, or less than once a week)
nids$alc[(nids$w1_a_j31 == 2 | nids$w1_a_j31 == 3| nids$w1_a_j31==4) & nids$adult20==1]<-2

#Frequent drinkers: between 1 and 4 days a week.
nids$alc[(nids$w1_a_j31 == 5 | nids$w1_a_j31 == 6) & nids$adult20==1]<-3

#Very frequent drinkers: more than 5 days a week.
nids$alc[(nids$w1_a_j31 == 7 | nids$w1_a_j31 == 8) & nids$adult20==1]<-4
```

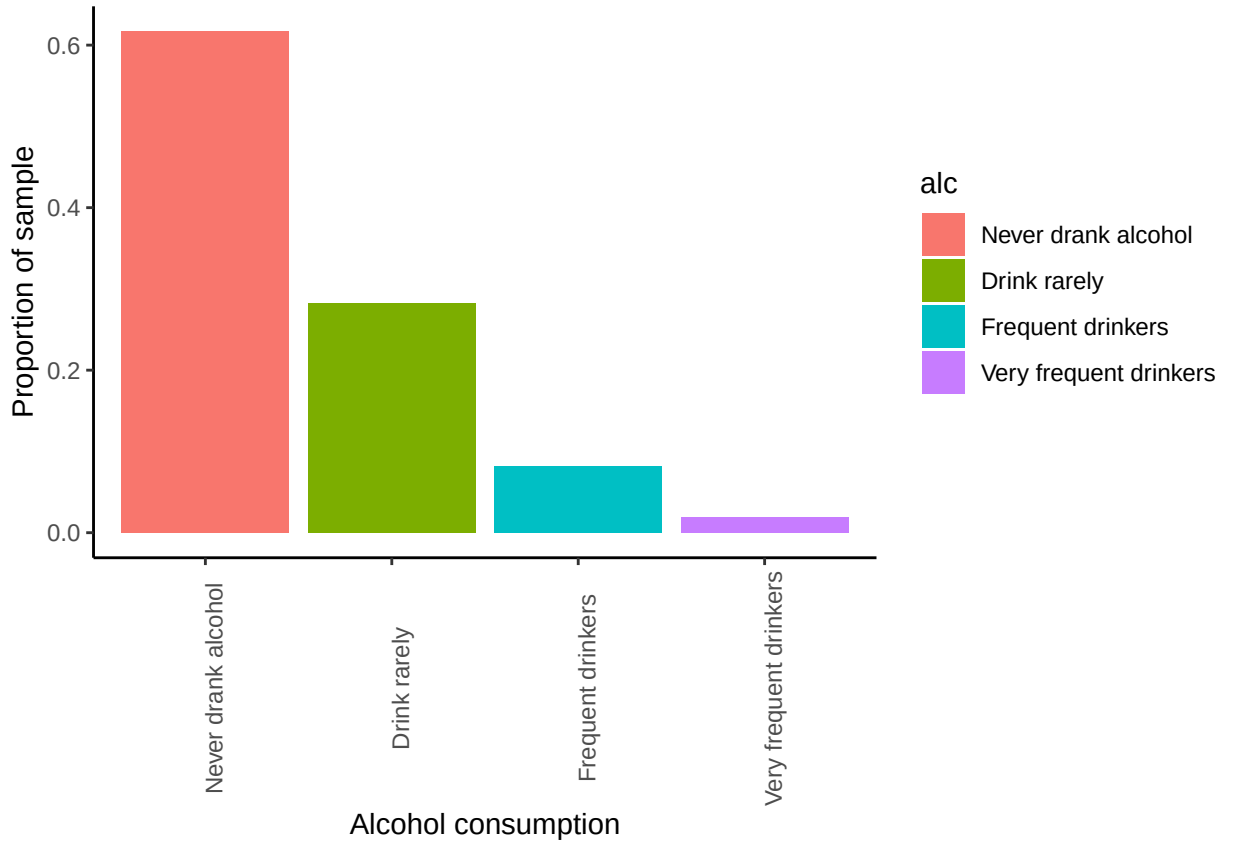
Adding labels:

```
nids<-nids %>%
  mutate(alc = ifelse(adult20==1, alc, NA)) %>%
  mutate(alc = factor(alc, levels = 1:4, labels = c("Never drank alcohol", "Drink rarely","Frequent dri
```

Graphing

```
nids%>%
  filter(!is.na(alc)) %>%
  ggplot(., aes(x = alc, fill=alc)) +
```

```
  geom_bar(aes(y = ..count../sum(..count..))) +
  xlab("Alcohol consumption") + ylab("Proportion of sample") +
  theme_classic() +
  theme(axis.text.x=element_text(angle=90))
```



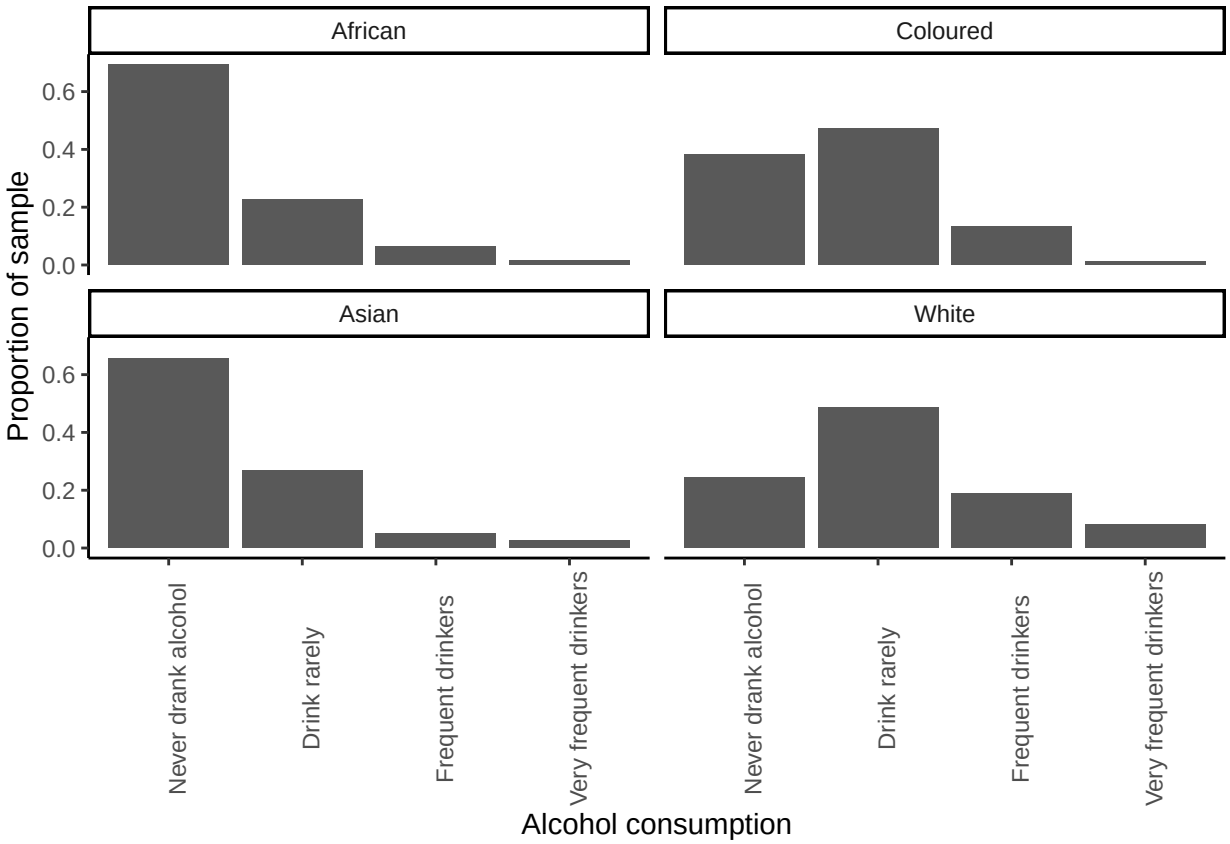
With this bar graph we can answer the following questions:

What proportion of respondents report that they have never drunken alcohol before? If we look at the blue column we see that over 60% of the respondents report they have never drunken alcohol before!

What does the green bar represent? It tells us that approximately 25% of the sample has either given up alcohol or consumes it very rarely.

Now, let's see whether alcohol consumption varies by race. Re-do the previous graph simply

```
nids%>%
  filter(!is.na(alc) & !is.na(race))%>%
  ggplot(.) +
  stat_count(aes(x= alc, y= ..prop.., group = race)) +
  xlab("Alcohol consumption") + ylab("Proportion of sample") +
  facet_wrap(~race, ncol=2) +
  theme_classic() +
  theme(axis.text.x=element_text(angle=90))
```



Are there substantial differences in reported alcohol consumption between racial groups? The majority of Africans and Asians/Indians report no alcohol consumption, while in the White and Coloured population groups, frequent alcohol consumption is far more common.

You may want to know whether alcohol consumption differs by some other characteristic instead of race. With a rich dataset such as the NIDS dataset, it is possible to investigate how various traits of the individual or household influence the variable we are interested in - in this case the frequency of alcohol consumption. In the following question we look at how alcohol consumption varies by gender.

6. Create a pair of bar graphs which show the frequency of alcohol consumption for men and women (use the alcohol frequency categories we created in the worked example). Is alcohol consumption similar between men and women in SA?

Question 6 Answer

7. Use bar graphs to investigate whether religious affiliation influences the frequency with which people consume alcohol.

Question 7 Answer

4.10 Question answers

4.11 Session information

```
print(sessionInfo(), locale = FALSE)
```

```
## R version 3.5.1 (2018-07-02)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 7 x64 (build 7601) Service Pack 1
##
## Matrix products: default
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] scales_0.5.0    bindrcpp_0.2.2  foreign_0.8-70  forcats_0.3.0
##  [5] stringr_1.3.1   dplyr_0.7.6     purrr_0.2.5     readr_1.1.1
##  [9] tidyr_0.8.1     tibble_1.4.2    ggplot2_3.0.0   tidyverse_1.2.1
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_0.2.4 xfun_0.2         reshape2_1.4.3   haven_1.1.2
##  [5] lattice_0.20-35  colorspace_1.3-2 htmltools_0.3.6  yaml_2.1.19
##  [9] utf8_1.1.4       rlang_0.2.1      pillar_1.2.3     withr_2.1.2
## [13] glue_1.2.0       modelr_0.1.2     readxl_1.1.0     bindr_0.1.1
## [17] plyr_1.8.4       munsell_0.5.0    gtable_0.2.0     cellranger_1.1.0
## [21] rvest_0.3.2      psych_1.8.4      evaluate_0.10.1  labeling_0.3
## [25] knitr_1.20       parallel_3.5.1   broom_0.4.5      Rcpp_0.12.17
## [29] backports_1.1.2  jsonlite_1.5     mnormt_1.5-5     hms_0.4.2
## [33] digest_0.6.15    stringi_1.1.7    bookdown_0.7     grid_3.5.1
## [37] rprojroot_1.3-2  cli_1.0.0        tools_3.5.1      magrittr_1.5
## [41] lazyeval_0.2.1   crayon_1.3.4     pkgconfig_2.0.1  xml2_1.2.0
## [45] lubridate_1.7.4  assertthat_0.2.0 rmarkdown_1.10   httr_1.3.1
## [49] rstudioapi_0.7   R6_2.2.2         nlme_3.1-137     compiler_3.5.1
```

Chapter 5

Measures of central tendency and variability

5.1 Getting ready

In the previous chapters we generated some variables and ran some commands that will influence the results that we get in this chapter. If you are starting a new session of R, please run the following lines of code before you begin the chapter. Make sure that you remember what each line of code is doing.

```
library(foreign)
library(tidyverse)
```

```
## -- Attaching packages ------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

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

```
nids<-read.dta("./data/nids.dta", convert.factors=FALSE)

nids<-nids%>%
  arrange(hhid, pid)%>%
  group_by(hhid) %>%
  mutate(hhrestrict = 1:n()) %>%
  mutate(hhrestrict = ifelse(hhrestrict==1,1,0))

###Creating a BMI variable - from chapter 2 ***

#Height
nids<-nids %>%
  mutate(height = ifelse (w1_a_n1_1 >= 0 & w1_a_best_age_yrs >= 20, w1_a_n1_1/100, NA))

#Weight
nids<-nids %>%
```

```
  mutate(weight = ifelse (w1_a_n2_1 >= 0 & w1_a_best_age_yrs > 20, w1_a_n2_1, NA))

#BMI
nids<-nids %>%
  mutate(bmi = weight/height^2)

#Age
nids<-nids%>%
  mutate(age_adult = ifelse(w1_a_best_age_yrs<0,NA, w1_a_best_age_yrs))

#Age bins
nids$age_bins<-NA
nids$age_bins[which(nids$w1_r_best_age_yrs>=20 & nids$w1_r_best_age_yrs<=29)]<-1
nids$age_bins[which(nids$w1_r_best_age_yrs>29 & nids$w1_r_best_age_yrs<=39)]<-2
nids$age_bins[which(nids$w1_r_best_age_yrs>39 & nids$w1_r_best_age_yrs<=49)]<-3
nids$age_bins[which(nids$w1_r_best_age_yrs>49 & nids$w1_r_best_age_yrs<=59)]<-4
nids$age_bins[which(nids$w1_r_best_age_yrs>59 & nids$w1_r_best_age_yrs<=69)]<-5
nids$age_bins[which(nids$w1_r_best_age_yrs>69 & nids$w1_r_best_age_yrs<=120)]<-6

nids$age_bins <- factor(nids$age_bins, levels = 1:6, labels = c("20 - 29 yrs","30 - 39 yrs", "40 - 49 y

nids <- nids%>%
  rename(race = w1_best_race,
         age = w1_r_best_age_yrs,
         gender = w1_r_b4,
         province = w1_hhprov,
         hhincome = w1_hhincome) %>%
  mutate(gender = factor(gender, levels = 1:2, labels = c("Male", "Female")),
          race = factor(race, levels = 1:4, labels = c("African", "Coloured","Asian", "White")),
          province = factor(province, levels=1:9, labels = c("Western Cape","Eastern Cape","Northern Cap
          w1_hhgeo = factor(w1_hhgeo, levels = 1:4, labels = c("Rural formal", "Tribal authority areas",
```

5.2 Introduction

In Chapter 3, we learned about the different variable types that exist in the NIDS data set, about different types of missing data, and about the commands that enable us to see frequency distribution tables. In chapter 4 we learned how to produce several types of simple graphs. Based on this knowledge, we are going to start learning how to do some basic statistical analysis, using measures of central tendency and variability.

As the name NIDS (National Income Dynamics Study) suggests, there are a great many questions that researchers wish to address regarding income levels. A few examples of questions we might wish to investigate using the data on individual monthly income (`w1_fwag - the monthly take home pay of main job`) are the following:

1. What is the average monthly individual income in South Africa?
2. Is the distribution of income in South Africa somewhat equal or are incomes much higher for the rich than for the poor? How much higher?
3. How does the average income in South Africa vary by racial group and how does income inequality vary within racial groups in South Africa?
4. How does income in South Africa vary by province? Do we live in a province that has above or below average income levels?

We will start this chapter considering continuous variables. Then we will learn some new commands that make analyzing data easier. Lastly, we will go through some of the key methods that will enable us to analyze categorical variables effectively.

5.3 Understanding distributions of continuous variables

For now, let's focus on monthly individual income (`w1_fwag`). Let's start by seeing what the distribution of income looks like. In the graph below, we find that almost 50 percent of the observations are in the first bar (which means income is roughly less than 1800 Rand per month.) Also note that only a very small fraction of income earners, earn monthly incomes above 40,000 Rand, in fact the bars themselves are very hard to see on the histogram! It turns out that only one person reports an income of 90,000. The next highest monthly income is 75,000 Rand, which is reported by 3 people and there is no one in NIDS that report a monthly income between these two figures. That is why, in the graph below, there are no bars between 90, 000 and 75,000. All in all, only 9 people report an income over 40, 000 Rand thus it is tempting to simply ignore these high values which seem out of keeping with the rest of the observations. Doing so would certainly produce a much more attractive histogram, that would allow us to focus on the where the bulk of the income values lie. This would, however, be a grave mistake; we can't just arbitrarily delete observations no matter how "unattractive" they may seem! Later we will experiment with different ways of handling these "outlying" data values, but for now, the data are what they are.

Below we give the code you could enter into your R Script in order to get the graph.

```
library(scales)
```

```
##
## Attaching package: 'scales'
```

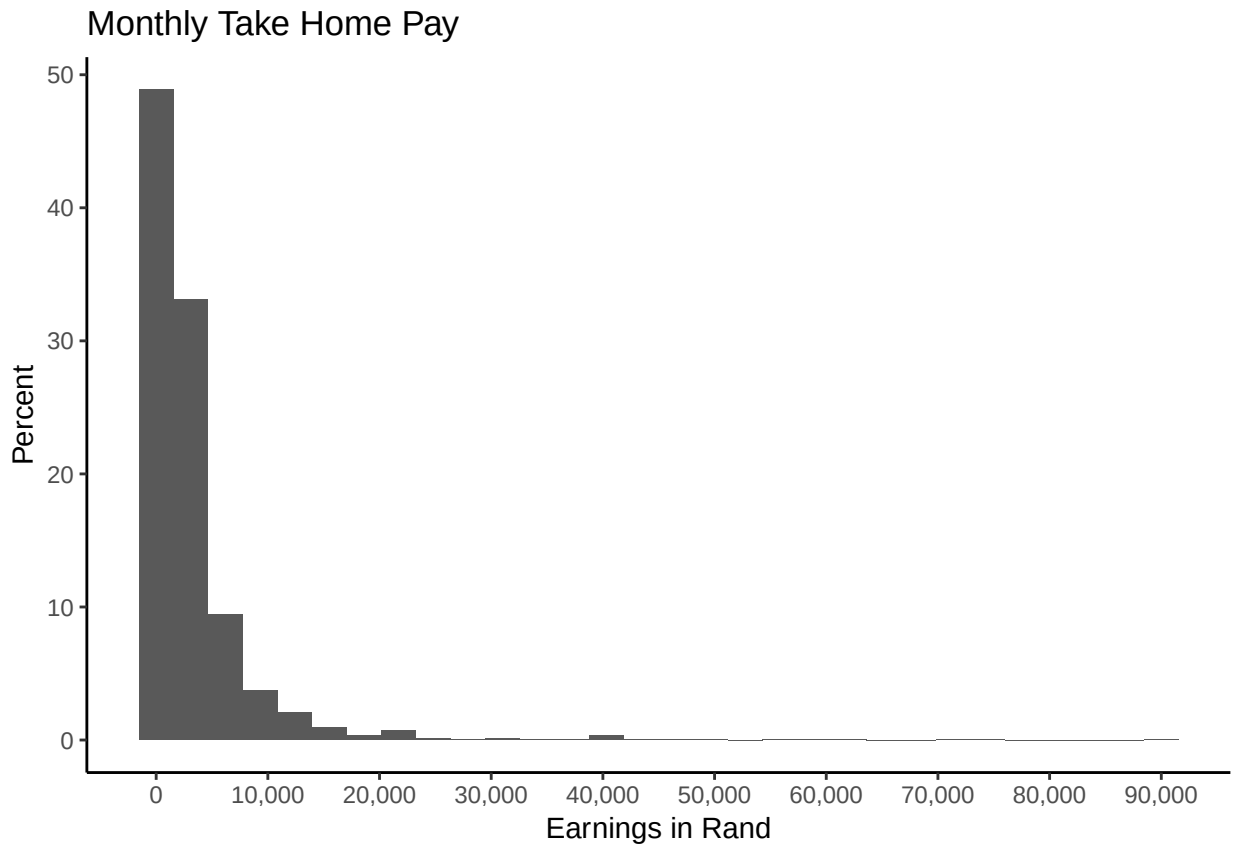
```
## The following object is masked from 'package:purrr':
##
##     discard
```

```
## The following object is masked from 'package:readr':
##
##     col_factor
```

```
ggplot(nids, aes(x = w1_fwag, y = (..count..)/sum(..count..)*100)) +
  geom_histogram() +
  scale_x_continuous(breaks=seq(0,90000,10000), labels = comma) +
  xlab("Earnings in Rand") +
  ylab("Percent") +
  ggtitle("Monthly Take Home Pay") +
  theme_classic()
```

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

```
## Warning: Removed 26671 rows containing non-finite values (stat_bin).
```



5.4 Continuous variables and recoding

While histograms are extremely useful in giving us a quick and clear overview of a particular variable, often we will want to delve a little deeper and obtain more precise results. For instance, we might wish to know exactly how many individuals have an income below R1500. Or we may want to know what percentage of the sample is 'middle class', as opposed to 'lower class' and 'upper class'. We learnt in chapter 3 that for categorical variables we can use `table` to get a good idea of the distribution of the variable, but if you try to use table for a continuous variable the result is not very useful. Try this for yourself!

We get the clue for how to deal with a continuous variable from the way a histogram works. Basically, a histogram divides the range of the variable up into bins and then counts how many values fall into each bin (for instance, a bin might be an income bracket of R2000). If we use this strategy and recode our income variable into bins then we will be able to tabulate the recoded variable and obtain meaningful results. Note that the choice of bin size is essentially arbitrary and depends on the context. Here, we will divide our income variable into only three bins for simplicity. We will use the following arbitrarily defined bins:

1. 0 - R1500 ("Lower Class")
2. R1500 - R4500 ("Middle Class")
3. R4500 + ("Upper Class") We generate a new variable that divides the sample into the three bins (and leaves those with dot-missing values as dot-missing).

```
summary(nids$w1_fwag)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    20.0   936.7  1600.0  3233.4  3550.0 90000.0   26671
```

```
nids<-nids %>%
  mutate(class = case_when(
    w1_fwag<=1500 ~ 1,
    w1_fwag>1500 & w1_fwag<=4500 ~ 2,
    w1_fwag>4500 ~ 3
  )) %>%
  mutate(class=factor(class, levels=1:3, labels = c("Lower Class","Middle Class","Upper Class")))
```

We need to be careful about given what we know about missing values and non-response codes in the NIDS dataset? Since we know that non-responses are coded as negative numbers, we have to be careful that we are not including all the non-responses in the lowest class category. However, when we used `summary` before running recode, we saw that the range of the `w1_fwag` variable is [20,90000]. This tells us that there are no non-responses recorded for this variable.

Let's have a look at the variable that we have created.

```
table(nids$class)
```

```
##
##  Lower Class Middle Class  Upper Class
##         2167         1503          829
```

or

```
nids %>%
  filter(!is.na(class)) %>%
  group_by(class) %>%
  summarise(freq=n()) %>%
  mutate(percent = round(freq/sum(freq)*100,2), cum_percent = round(cumsum(freq)/sum(freq)*100,2))
```

```
## # A tibble: 3 x 4
##   class         freq percent cum_percent
##   <fct>        <int>   <dbl>       <dbl>
## 1 Lower Class   2167    48.2        48.2
## 2 Middle Class  1503    33.4        81.6
## 3 Upper Class    829    18.4       100
```

Notice that the total number of individuals with a class value is 4 499. However, we know that our sample has over 30 000 individuals in it. We know that all the excluded individuals are NA-missing. You can check that the number of NA-missings are the same for the original income variable (`w1_fwag`) and our newly created `class` variable.

Recall that our objective in this section was to learn how to take a continuous variable and recode it in a way that made it easier to interpret. There are many ways of extracting information from a continuous variable - what we have done is one of them. We can now say that of the portion of the sample with recorded income levels, 48% earn below R1500, 33% between R1500 and R4500, and 18% above R4500. However, we must keep in mind the large proportion of the population with no recorded income value. Before thinking about publishing these results, we would need to investigate where all the NA-missings come from, however we will not do that here since you have already learnt how to go about conducting such an investigation.

5.5 Summarise: by - summary (base R) and group_by - summarise (dplyr)

We have just seen one of the ways that you can summarize the information contained in a continuous variable. Upon meeting a new variable, R has a several commands that enable you to get acquainted with your new

friend - we have already learnt `head`, `tail`, e.t.c. Here we do more exploration using the `summary()` command. In R, type:

```
summary(nids$w1_fwag)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    20.0   936.7  1600.0  3233.4  3550.0 90000.0   26671
```

We see that the average monthly income is R3233.40. There are two nice features of `summary`. First, `summary` tells us the range of the variable. For example, when we typed `summary(nids$w1_fwag)`, we learned that `w1_fwag` ranged between 20 and 90000. Second, summary works with the `by()` option. We can compute the mean by each of categories of another variable, e.g. race (`w1_best_race`), `w1_hhprov` (province variable), or any other distinguishing characteristic. To summarise `w1_fwag` by `race`, type:

```
by(nids[, c("w1_fwag")], nids$race, summary)
```

```
## nids$race: African
##     w1_fwag
##  Min.   :   20
##  1st Qu.:  850
##  Median : 1400
##  Mean   : 2494
##  3rd Qu.: 2995
##  Max.   :90000
##  NA's   :19227
## ------------------------------------------------------------
## nids$race: Coloured
##     w1_fwag
##  Min.   :   55.0
##  1st Qu.:  887.6
##  Median : 1400.0
##  Mean   : 2387.3
##  3rd Qu.: 2700.0
##  Max.   :42000.0
##  NA's   :3134
## ------------------------------------------------------------
## nids$race: Asian
##     w1_fwag
##  Min.   :  920
##  1st Qu.: 3000
##  Median : 4500
##  Mean   : 7550
##  3rd Qu.: 9338
##  Max.   :40000
##  NA's   :346
## ------------------------------------------------------------
## nids$race: White
##     w1_fwag
##  Min.   :  100
##  1st Qu.: 3766
##  Median : 6952
##  Mean   : 9241
##  3rd Qu.:11000
##  Max.   :75000
##  NA's   :992
```

We see that the average monthly income of Africans, Coloureds, Indians/Asians, and Whites, respectively are 2494, 2387, 7550, and 9241.4 Rand respectively.

Alternatively, you can use `group_by` and `summarise` functions from the library `dplyr` as follows:

```
nids %>%
  select(hhid,race, w1_fwag) %>%
  group_by(race) %>%
  summarise(min=min(w1_fwag, na.rm=TRUE),
            median=median(w1_fwag, na.rm=TRUE),
            mean=mean(w1_fwag, na.rm=TRUE),
            max=max(w1_fwag, na.rm=TRUE),
            std.dev=sd(w1_fwag, na.rm=TRUE))
```

```
## # A tibble: 5 x 6
##   race       min median  mean   max std.dev
##   <fct>    <dbl>  <dbl> <dbl> <dbl>   <dbl>
## 1 African     20  1400  2494. 90000   4104.
## 2 Coloured    55  1400  2387. 42000   3033.
## 3 Asian      920  4500  7550. 40000   7522.
## 4 White      100  6952. 9241. 75000   9504.
## 5 <NA>      1160  1622. 1576.  1900    348.
```

Sometimes, it will be helpful to compute means (or other statistics) by groups that we construct. For example, suppose we wanted to `summarise` to give income levels of different age groups - e.g. under 20, 21-40, 41-60, and over 60. We need to construct these age groups and then compute the means. As usual there are various ways to do this. We learnt how to use the square brackets to subset or `dplyr`'s `case_when`. Below is the code for generating an NA-missing variable and replacing it.

```
nids$agegrp<-NA
nids$agegrp[which(nids$age>=0 & nids$age<=20)]<-1
nids$agegrp[which(nids$age>20 & nids$age<=40)]<-2
nids$agegrp[which(nids$age>40 & nids$age<=60)]<-3
nids$agegrp[which(nids$age>60 & nids$age<=120)]<-4
```

To factor variable:

```
nids$agegrp<-factor(nids$agegrp, levels=1:4, labels=c("0 - 20 yrs", "21 - 40 yrs","41 - 60 yrs", "60+ y
```

```
nids%>%
  group_by(agegrp)%>%
  summarise(Mean=mean(w1_fwag, na.rm=TRUE),
       Std.Dev=sd(w1_fwag, na.rm=TRUE),
       Min=min(w1_fwag, na.rm=TRUE),
       Max=max(w1_fwag, na.rm=TRUE))
```

```
## # A tibble: 5 x 5
##   agegrp       Mean Std.Dev   Min   Max
##   <fct>       <dbl>   <dbl> <dbl> <dbl>
## 1 0 - 20 yrs  1357.   1057.    80  7000
## 2 21 - 40 yrs 3009.   4803.    20 90000
## 3 41 - 60 yrs 3851.   6141.    88 75000
## 4 60+ yrs     2773.   3116.   280 18000
## 5 <NA>        1226.   1678.   100  7000
```

This gives us the average income levels for each of our age categories. For instance, 41 - 60 year olds in the sample, who had a reported income level, earned R3851 on average. Notice that age category 0 - 20 years reports a mean of R1356, but it would be strange if the average person younger than 20 has an income

of R1356. What's going on here? If you look at the Obs category, you see that the group only has 153 respondents with reported income data. This is primarily because most people below the age of 20 do not work in the formal sector and therefore are NA-missing. The individuals who don't report any income are not included in calculating the mean.

5.6 Medians and modes of continuous variables

Up to now, the only measure of central tendency that we have examined is the mean. There are other measures and the two that we wish to examine now are the median and the mode of a distribution. The median of a distribution is the value for which half the observations are greater and half are less. If observations are symmetrically distributed, the median and the mean will be the same. If the distribution of a variable is quite skewed, however, the median and the mean will be quite different. In general, medians are sometimes used instead of means if one wants a measure that will be robust to large outliers like the ones we saw at the beginning of this chapter. That is, we want a measure that is not very sensitive to extraordinary values in the distribution.

Take for example the hypothetical situation in which you have ten people with an income of R100 and one with income of R1000 000. The mean will be R91 000 even though ten out of the eleven people earn less than one hundredth of this amount. Sometimes, it will be more informative to talk about the median of the distribution - in this case the median would be R100. While this example uses unrealistically extreme values, we will see the same principle holds in the actual data below.

Let's consider the NIDS income variable (`w1_fwag`) again. In case you have not noticed it in the previous outputs, the `summary()` command produces both the mean and median. Type:

```
summary(nids$w1_fwag)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    20.0   936.7  1600.0  3233.4  3550.0 90000.0   26671
```

Note that the median income is R1600. The median income is a lot less than the mean income. What is going on here? Have a look at the income histogram that we created in the first section of this chapter. The graph shows us that most of the income values are less than R20 000, with a couple of much larger values - particularly one of R90 000. This extremely large value is inflating the mean of the distribution. In contrast, the median treats it as just one more value above the "half way" mark. For many variables with skewed distributions, the median is a very useful statistic. The mode of a distribution is the value that appears most often in the distribution. The mode is a seldom used measure, but we should be aware of it. Let's consider the education variable - `w1_best_edu`.

The mode of this variable represents the most commonly reported educational level amongst NIDS respondents. There is no simple way to compute the mode in R. we can summarise the grouped categories of `w1_best_edu` and see which one has the most frequency.

```
nids%>%
  group_by(w1_best_edu) %>%
  summarise(n=n())
```

```
## # A tibble: 31 x 2
##    w1_best_edu     n
##          <int> <int>
##  1          -9   132
##  2          -8     1
##  3          -5     1
##  4          -3   710
##  5           0   924
##  6           1   883
```

```
##  7           2   905
##  8           3  1175
##  9           4  1340
## 10           5  1381
## # ... with 21 more rows
```

By examining the entire frequency distribution, we should note that the mode of `w1_best_edu` is 25. In other words a large number of respondents (5731) reported "no schooling" as their level of education, far more than for any other category. That's not very surprising since infants and toddlers are included in the sample. What is more surprising is the modal level of education for adults 21 and older. Type:

```
nids%>%
  filter(age >= 21 & !is.na(age))%>%
  group_by(w1_best_edu) %>%
  summarise(n=n())%>%
  print()
```

```
## # A tibble: 29 x 2
##    w1_best_edu     n
##          <int> <int>
##  1          -9    57
##  2          -3    15
##  3           0     4
##  4           1   101
##  5           2   241
##  6           3   394
##  7           4   537
##  8           5   626
##  9           6   755
## 10           7   996
## # ... with 19 more rows
```

As we see the table above, the modal value is still 25. Therefore, even amongst adults, the most common category reported by respondents over the age of 21 is having no education at all.

5.7 Example: recoding the education variable

Suppose we want to know the average level of education for individuals aged 21 or over in South Africa. This entails computing the mean of an education variable. When we compute means for continuous variables, we need to be wary of how the variables are coded. In this extended example, we will consider the education variable, `w1_best_edu`.

We need to consider how the education variable was coded. For this, we need to refer to the survey documents, i.e. questionaires.

From the table above we note that a code of "0" indicates Grade R/0, a code of 3 indicates Grade 3 and so on. The code usually increases by one for each year of schooling up to a value of 12. Codes greater than 12 do not necessarily indicate more years of schooling; for example, a code of 25 represents no schooling. We also observe that negative numbers are assigned for non-responses. In addition, there are several NA-missing values.

At this point you may be asking: how then, should we compute the average education level for households in the survey? What we really need here is a variable that ranks different educational levels on some numerical scale. There is no single answer on how to create this variable. Below is one answer, worked out in R.

We begin by creating a new education variable that we call `educ_new` and setting it equal to the original

Table 5 - Education Codes

-3	Missing
-9	Don't Know
-8	Refused
-5	Not Applicable
0	Grade R/0
1	Grade 1/Sub A/Class 1
2	Grade 2/Sub B/Class 2
3	Grade 3/Std. 1
4	Grade 4/Std. 2
5	Grade 5/Std. 3
6	Grade 6/Std. 4
7	Grade 7/ Std. 5
8	Grade 8/ Std. 6/Form 1
9	Grade 9/Std. 7/Form 2
10	Grade 10/ Std. 8/Form 3
11	Grade 11/ Std. 9/Form 4
12	Grade 12/Std. 10/Form 5/Matric/Senior Certificate
13	NTC 1
14	NTC 2
15	NTC 3
16	Certificate with less than Grade 12/Std 10
17	Diploma with less than Grade 12/Std 10
18	Certificate with Grade 12/Std 10
19	Diploma with Grade 12/Std 10
20	Bachelors degree - Level 4
21	Bachelors degree and Diploma
22	Honours degree
23	Higher degree (Masters Doctorate)
24	Other
25	No Schooling
55	No Higher Education

Figure 5.1: Education codes

education variable - w1_best_edu. We will then replace several of the values in order to make our mean meaningful.

We want to create an education variable that is able to rank in some way different educational qualifications. One of the easiest ways to do this is to consider the **number of years spent on education** up to the point of the level of the qualification. We know that starting from Grade 0 (Code 0) the code increases by one for each year of schooling up to a value of 12. So the education codes for Grade 0 to Grade 12 already have the intrinsic ordering we want.

Next we elect (rather arbitrarily) to combine those who have no reported education with those whose highest reported education level was Grade R/0.

```
nids$educ.new<-nids$w1_best_edu
nids$educ.new[which(nids$w1_best_edu == 25)]<-0
```

Next, we set the negative values to "missing".

```
nids$educ.new[which(nids$w1_best_edu <0)]<-NA
```

We also set the variable coded 24 ("other") to "missing" since we don't know how to interpret "other."

```
nids$educ.new[which(nids$w1_best_edu == 24)]<-NA
```

We still are not done. Next we look at categories where respondents have a diploma or certificate but did not complete their schooling up to Grade 12. Here we have to make some educated but essentially subjective decisions on how to code these educational qualifications. This would in practice require researching the South African education system in order to inform these decisions. The code 13 "NTC 1" (note NTC means National Technical Certificate) and code 16 "Certificate with less than Grade 12" are assessed as being more or less equivalent to a Grade 10 qualification and so we assign all of these individuals with a value of "10". In addition code 14 "NTC2" and code 17 "Diploma with less than Grade 12" are thought to be somewhat equivalent to a Grade 11 education and so we assign individuals in these categories a value of 11. Finally, code 15 "NTC 3" is regarded to be more or less equivalent to Grade 12 so NTC 3 individuals are recoded with a value of "12".

```
nids$educ.new[which(nids$w1_best_edu == 13 | nids$w1_best_edu == 16)]<-10
nids$educ.new[which(nids$w1_best_edu == 14 | nids$w1_best_edu == 17)]<-11
nids$educ.new[which(nids$w1_best_edu == 15)]<-12
```

But what about all individuals who have completed Grade 12 plus have additional qualifications? Once again we make some educated but essentially subjective assertions. Those who have a Certificate in addition to Grade 12 are assigned a code of 13, while those who have a diploma with Grade 12 are assigned a code of 14. A Bachelors Degree is assigned a value of 15 - Grade 12 plus 3 more years of education, while those with an Honours Degree or a Bachelors Degree and Diploma are assigned values of 16. Finally, Higher Degrees such as Masters of Doctorate are assigned a code of 17 (Although we could have just as easily assigned this last category a code of 18, if for instance you feel strongly that Masters is most commonly a two year course).

```
nids$educ.new[which(nids$w1_best_edu == 18)]<-13
nids$educ.new[which(nids$w1_best_edu == 18)]<-13
nids$educ.new[which(nids$w1_best_edu == 19)]<-14
nids$educ.new[which(nids$w1_best_edu == 20)]<-15
nids$educ.new[which(nids$w1_best_edu == 21 | nids$w1_best_edu == 22)]<-16
nids$educ.new[which(nids$w1_best_edu == 23)]<-17
```

To review where we stand, type:

```
nids%>%
  group_by(educ.new)%>%
  summarise(n=n())
```

```
## # A tibble: 19 x 2
```

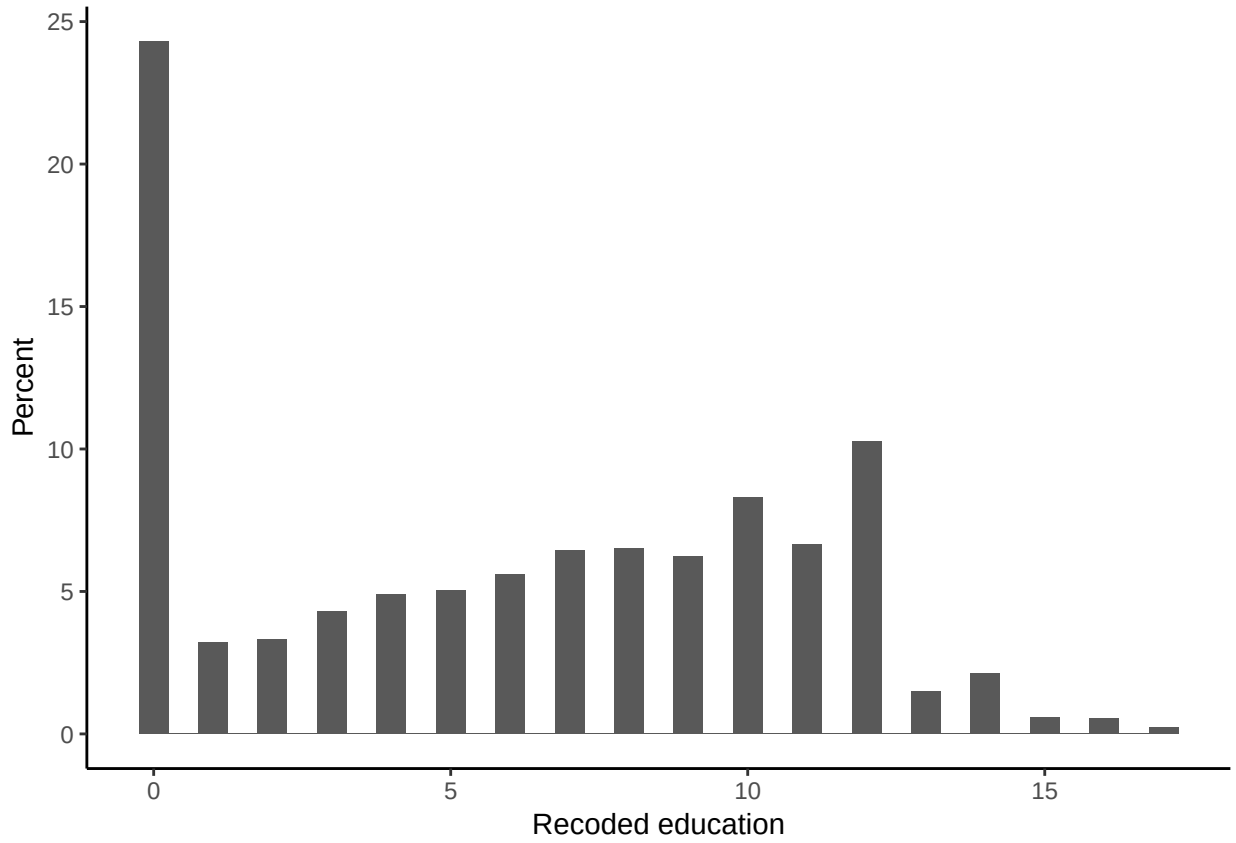
```
##    educ.new     n
##       <dbl> <int>
##  1        0  6655
##  2        1   883
##  3        2   905
##  4        3  1175
##  5        4  1340
##  6        5  1381
##  7        6  1531
##  8        7  1766
##  9        8  1782
## 10        9  1701
## 11       10  2270
## 12       11  1821
## 13       12  2806
## 14       13   407
## 15       14   582
## 16       15   158
## 17       16   145
## 18       17    67
## 19       NA  3795
```

We will see that we now have a variable whose values are much more meaningful than those of the original `w1_best_edu`.

The distribution of education levels is graphed below.

```
nids%>%
  ggplot(., aes(x=educ.new)) +
  geom_histogram(aes(y = ..count..*100/sum(..count..)), binwidth=0.5)+
  xlab("Recoded education") +
  ylab("Percent") +
  theme_classic()
```

```
## Warning: Removed 3795 rows containing non-finite values (stat_bin).
```



In the above histogram, we can see that about 24 percent of the respondents report having zero years of education. The mean education level is found by simply typing:

```
summary(nids$educ.new)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   0.000   1.000   6.000   6.037  10.000  17.000    3795
```

We see that respondents have, on average, an education level of 6.04, somewhere close to having completed Grade 4. The correction for the strange coding of `w1_best_edu` resulted in a lower mean level of education and this is what we would expect since the original variable coded no schooling with a higher value than a completed university degree.

A more interesting question is the following:

What is the mean level of education in South Africa for individuals 21 years old and older?

```
nids%>%
  filter(age >= 21 & !is.na(age))%>%
  select(educ.new)%>%
  ungroup() %>%
  summarise(Mean=mean(educ.new, na.rm=T))
```

```
## Adding missing grouping variables: `hhid`
```

```
## # A tibble: 1 x 1
##    Mean
##   <dbl>
## 1  7.68
```

You should have found that the average education level of adults was 7.68 - the average respondent over the age of 21 (adult) has completed grade 5. What do we think this statistic will be 20 years from now when those who are now infants reach the age of 21?

To get an indication of whether today's youth are getting more education on average, let's see what the average education level of 23 year olds is as most people have completed their education by this age.

```
nids%>%
  filter(age == 23 & !is.na(age))%>%
  select(educ.new)%>%
  ungroup() %>%
  summarise(Mean=mean(educ.new, na.rm=T))
```

```
## Adding missing grouping variables: `hhid`
```

```
## # A tibble: 1 x 1
##    Mean
##   <dbl>
## 1  10.3
```

Although this is a crude measure, it suggests that today's youth are on average getting a grade 8, which is over two years more than the average adult in the sample.

As an exercise, try the following question:

1. Compute the mean education level of adults (21 and older) by gender and of non-adults (under 21) by gender. How about the differences for children between the ages of 6 and 10?

What do the answers to this exercise suggest for the future?

Question 1 Answer

5.8 Measures of dispersion - variance and standard deviation

So far, we have learned something about the average level of monthly income among individuals in South Africa. Higher incomes are generally a good thing, but from a policy perspective, we will often care about how widely spread the distribution of income is. In an extreme situation, it could be the case that everyone in the country has exactly the same level of income (therefore every individual would have the "mean" as their level of income). That situation could be considered to be perfectly equal distribution of income. Of course, in the real world, individuals and households have vastly different levels of income.

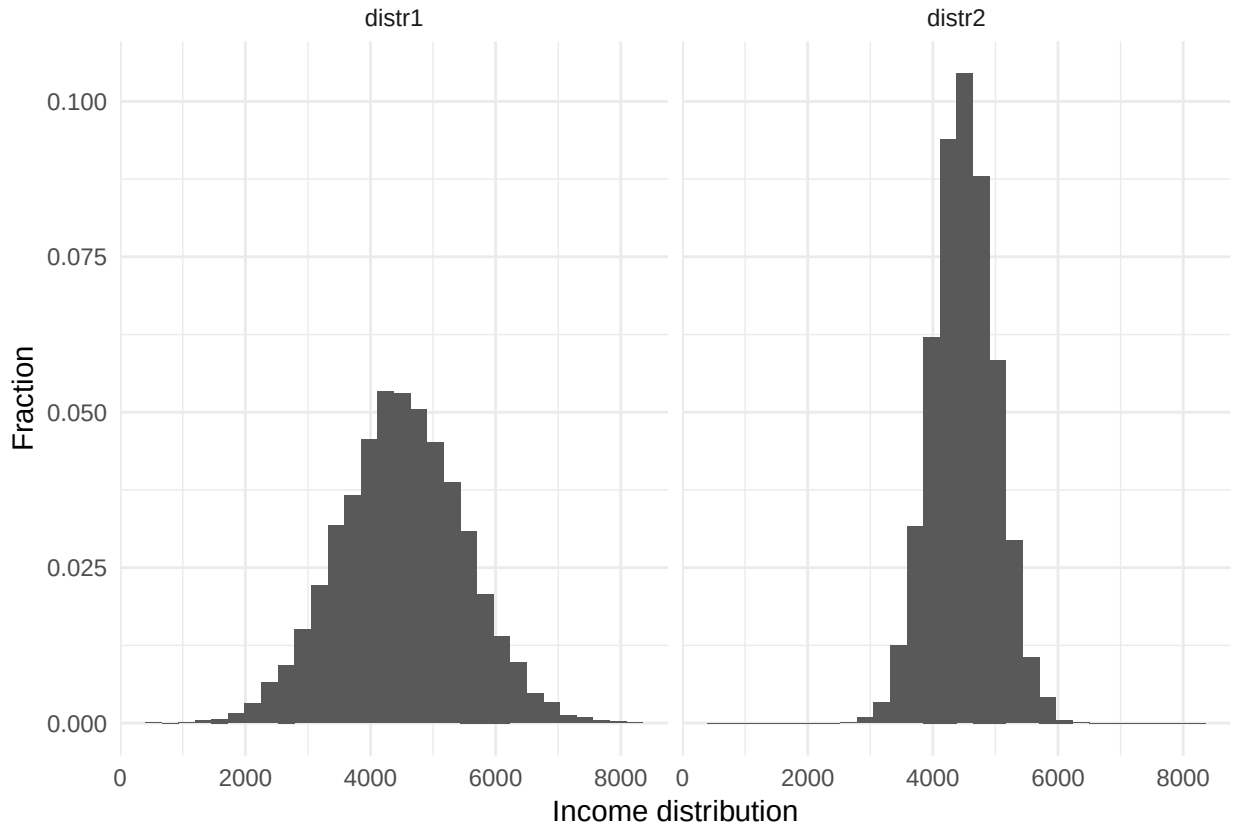
In an effort to clarify this notion of variance consider two theoretical distributions with the same mean, as shown below. Both of these distributions have a mean income of about 4500 Rand. Examine these two distributions.

```
set.seed(123)
d1<-data.frame(dist="distr1", value=rnorm(10000, 4500, 1000))
d2<-data.frame(dist="distr2", value=rnorm(10000, 4500, 500))

df<-rbind(d1,d2)

ggplot(df, aes(x=value, y=..count../sum(..count..))) +
  geom_histogram() +
  facet_grid(.~dist) +
  labs(x="Income distribution", y="Fraction") +
  theme_minimal()
```

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



Even though both distributions have the same average level of income, we can see that the income inequality is greater in the first graph. If we only computed the mean of each of these (made-up) income variables, we might conclude that these distributions were essentially the same even though the first distribution is clearly more dispersed. The variance is a useful measure of dispersion of a variable - basically, the variance gives an indication of how wide the distribution is. The square root of the variance is termed the standard deviation. A bigger variance always means a larger standard deviation. In the above two distributions, the standard deviation of Income Distribution #1 is 1000 while the standard deviation of Income Distribution #2 is 500. Since the standard deviation of the first distribution is twice as large as that of the second distribution, its variance is about four times as large.

One useful way to think about the standard deviation of a distribution is the following. Suppose that the distribution of a variable is bell-shaped (more formally termed a normal distribution.) If we picked randomly from the normal distribution, two thirds of the time we would pick a value that was within two standard deviations (two times the standard deviation) from the mean of the distribution. For the case of Income Distribution #1 above, this means that if we randomly picked a person with an income level from this distribution, two out of every three people we pick should have an income within 2000 Rand of the mean income of 4500 Rand - i.e. between R2500 and R6500. In contrast, two thirds of the people picked from distribution 2 should have an income level between R3500 and R5500.

In R, the standard deviation of a variable is obtained using the `sd()` command.

What is the standard deviation of individual monthly income by `w1_best_race` (now renamed `race`) in South Africa?

To find out, type:

```
nids%>%
  group_by(race)%>%
  summarise(Mean=mean(w1_fwag, na.rm=TRUE),
```

```
            Std.Dev=sd(w1_fwag, na.rm=TRUE),
            Min=min(w1_fwag, na.rm=TRUE),
            Max=max(w1_fwag, na.rm=TRUE))
```

```
## # A tibble: 5 x 5
##   race      Mean Std.Dev   Min   Max
##   <fct>    <dbl>   <dbl> <dbl> <dbl>
## 1 African  2494.   4104.    20 90000
## 2 Coloured 2387.   3033.    55 42000
## 3 Asian    7550.   7522.   920 40000
## 4 White    9241.   9504.   100 75000
## 5 <NA>     1576.    348.  1160  1900
```

We discover that the standard deviation for some of the groups is quite different from others. Put another way, income inequality appears to vary by racial group. For instance, the variation in income amongst Whites is far larger than amongst Africans.

As another exercise, let's investigate the average household size as well as the standard deviation of household size depending on whether a household lives in a Rural, Tribal Authority, Urban Formal or Urban Informal area. To eliminate the bias of household level variables, we need to create a new household-level household size variable.

```
nids<-nids%>%
  arrange(hhid)%>%
  mutate(w1_hhsizer2 = ifelse(hhrestrict == 1, w1_hhsizer, NA))
```

Overall mean:

```
nids%>%
  ungroup() %>%
  summarise(Mean=mean(w1_hhsizer2, na.rm=TRUE),
            Std.Dev=sd(w1_hhsizer2, na.rm=TRUE),
            Min=min(w1_hhsizer2, na.rm=TRUE),
            Max=max(w1_hhsizer2, na.rm=TRUE))
```

```
## # A tibble: 1 x 4
##    Mean Std.Dev   Min   Max
##   <dbl>   <dbl> <dbl> <dbl>
## 1  3.87    2.56     1    25
```

By `w1_hhgeo`:

```
nids%>%
  group_by(w1_hhgeo)%>%
  summarise(Mean=mean(w1_hhsizer2, na.rm=TRUE),
            Std.Dev=sd(w1_hhsizer2, na.rm=TRUE),
            Min=min(w1_hhsizer2, na.rm=TRUE),
            Max=max(w1_hhsizer2, na.rm=TRUE))
```

```
## # A tibble: 4 x 5
##   w1_hhgeo                Mean Std.Dev   Min   Max
##   <fct>                  <dbl>   <dbl> <dbl> <dbl>
## 1 Rural formal            3.24    2.37     1    16
## 2 Tribal authority areas  4.59    2.85     1    25
## 3 Urban formal            3.48    2.22     1    17
## 4 Urban informal          3.70    2.53     1    18
```

```
#by(nids[, c("w1_hhsizer2")], nids$w1_hhgeo, summary)
```

We note that there is a pretty big difference in average family size depending on whether a household lives in tribal authority or urban informal area. The average household size is about 4.6 persons for the former and about 3.7 overall for the latter. Notice also that the variation in household size is the smallest in urban formal areas and the largest in tribal authority areas.

Examining the median household size instead of the mean gives a similar picture. (Try it, just add the `median ()` function to the above command.)

5.9 Handling outliers

The mean can be very sensitive to data points that are very, very different from all the other observations in the distribution. We refer to these extreme data points as "outliers." There are several large literatures in statistics that deal with outliers. These include statistical measures that are not especially sensitive to outliers– robust statistics, as well as literatures on how to identify outliers and what to do once they are found. Reviewing this literature is beyond the scope of this chapter, but we want to be alert to the presence of outliers and aware of how they can impact some of our results. There are, in general, two ways we will deal with outliers. One is to use measures that are not sensitive to them, such as the median instead of the mean, and the other is to remove their impact (usually by setting them equal to a missing value.)

Sometimes it will be obvious when an outlier is simply miscoded data and hence should be set to missing. If for example, age was reported as 230, we would know that was a miscode. Most of the NIDS data set has been "cleaned" and we are not aware of many miscodes. (If you think you find some, email us.) This doesn't mean that outliers won't matter. Consider our initial example of income, `w1_fwag`. Examine what happens to mean income if we drop just the highest 8 income levels of the 4499 observations. Since we will want to compare the original income measure to our truncated measure, we want to create a new income measure without writing over the old one:

```
nids<-nids%>%
  mutate(w1_fwag2 = ifelse(w1_fwag>50000,NA, w1_fwag))
```

We can then `summary()` the two variables:

```
summary(nids[,c("w1_fwag", "w1_fwag2")])
```

```
##     w1_fwag           w1_fwag2
##  Min.   :   20.0   Min.   :   20.0
##  1st Qu.:  936.7   1st Qu.:  932.3
##  Median : 1600.0   Median : 1600.0
##  Mean   : 3233.4   Mean   : 3114.0
##  3rd Qu.: 3550.0   3rd Qu.: 3500.0
##  Max.   :90000.0   Max.   :50000.0
##  NA's   :26671     NA's   :26679
```

Simply by dropping these 8 observations out of 4499, our mean has been reduced by over R100.

And to see how the outliers change our results for how average income varies by race:

```
by(nids[, c("w1_fwag", "w1_fwag2")], nids$race, summary)
```

```
## nids$race: African
##     w1_fwag         w1_fwag2
##  Min.   :   20   Min.   :   20
##  1st Qu.:  850   1st Qu.:  850
##  Median : 1400   Median : 1400
```

```
##  Mean   : 2494   Mean   : 2382
##  3rd Qu.: 2995   3rd Qu.: 2946
##  Max.   :90000   Max.   :40000
##  NA's   :19227   NA's   :19232
## ------------------------------------------------------------
## nids$race: Coloured
##     w1_fwag          w1_fwag2
##  Min.   :   55.0   Min.   :   55.0
##  1st Qu.:  887.6   1st Qu.:  887.6
##  Median : 1400.0   Median : 1400.0
##  Mean   : 2387.3   Mean   : 2387.3
##  3rd Qu.: 2700.0   3rd Qu.: 2700.0
##  Max.   :42000.0   Max.   :42000.0
##  NA's   :3134      NA's   :3134
## ------------------------------------------------------------
## nids$race: Asian
##     w1_fwag         w1_fwag2
##  Min.   :  920   Min.   :  920
##  1st Qu.: 3000   1st Qu.: 3000
##  Median : 4500   Median : 4500
##  Mean   : 7550   Mean   : 7550
##  3rd Qu.: 9338   3rd Qu.: 9338
##  Max.   :40000   Max.   :40000
##  NA's   :346     NA's   :346
## ------------------------------------------------------------
## nids$race: White
##     w1_fwag         w1_fwag2
##  Min.   :  100   Min.   :  100
##  1st Qu.: 3766   1st Qu.: 3760
##  Median : 6952   Median : 6800
##  Mean   : 9241   Mean   : 8801
##  3rd Qu.:11000   3rd Qu.:11000
##  Max.   :75000   Max.   :50000
##  NA's   :992     NA's   :995
```

These are significant differences in the mean and median income, on the other hand, barely differs at all between `w1_fwag` and `w1_fwag2`.

It is always important to be aware of possible outliers in the data and there will be other variables in the NIDS datasets whose means are sensitive to a few very high values. For example, dropping just the top 5 of over 951 observations on monthly income from self employment decreases the mean of these income values by almost 10 percent.

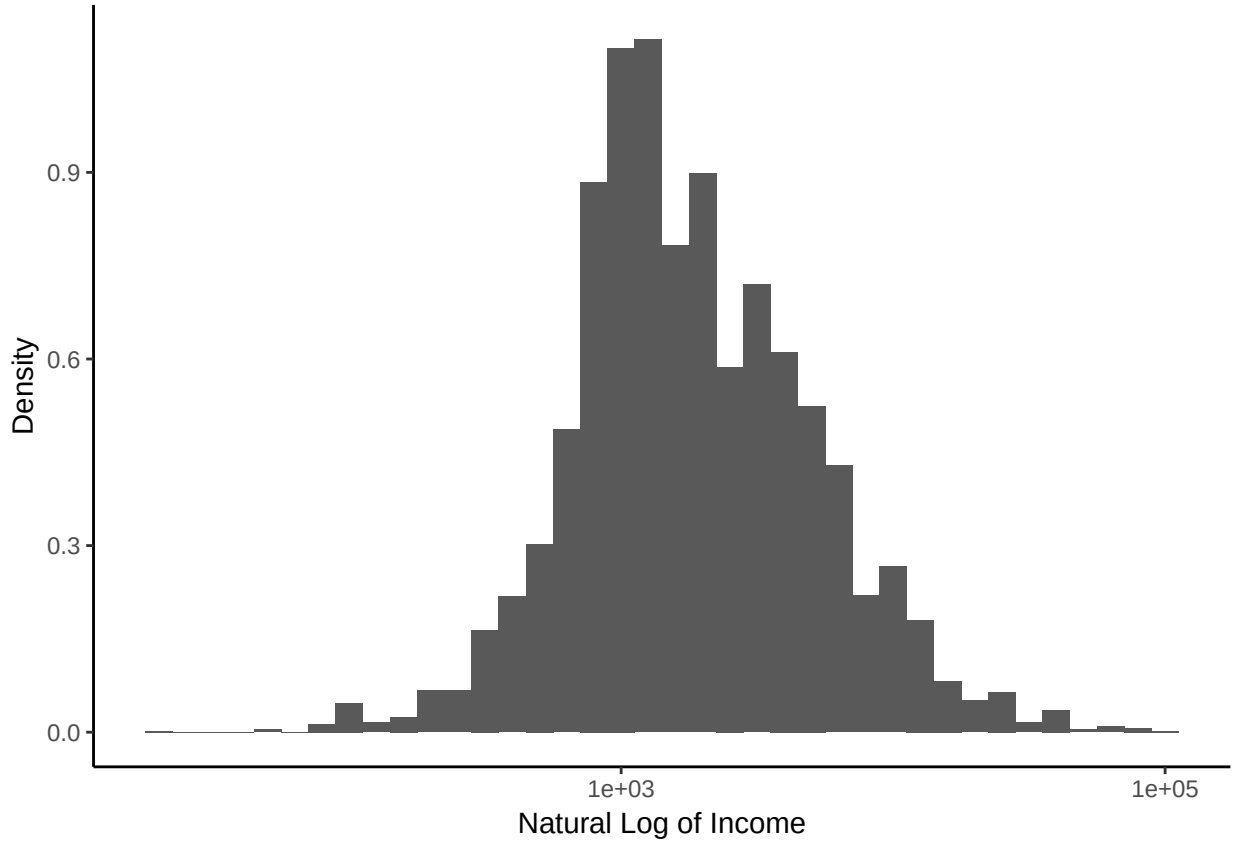
Comparing the mean to the median is a helpful way to gauge the presence of outliers. There are a number of more sophisticated tools in R for examining outliers. These include graphical methods. Graphical methods were introduced in Chapter 5 and we'll see an example of how useful they can be below.

If we are comfortable with the concept of logarithms, we can also look at the log of `w1_fwag`. Here is a graph of monthly income with the x-axis calibrated in logs.

```
nids%>%
  ggplot(., aes(x=w1_fwag)) +
  geom_histogram(aes(y = ..density..), binwidth=0.1, labels = comma) +
  scale_x_log10() +
  labs(x="Natural Log of Income",y ="Density") +
  theme_classic()
```

```
## Warning: Ignoring unknown parameters: labels
```

```
## Warning: Removed 26671 rows containing non-finite values (stat_bin).
```



An advantage of calibrating the x-axis in logs is that high income outliers don't have as large an impact on the graph's orientation. When we work on regression analysis later on this will become very useful method.

5.10 Understanding the distributions of categorical variables

Many variables in NIDS are categorical variables. Examples that we will work with in this subsection include occupation and race. In Chapter 3 we have already investigated frequency distributions of categorical variables. In this section, we ask whether there are statistics analogous to the mean and standard deviation (which we used to describe distributions of continuous variables) that we can use to describe distributions of categorical variables.

We begin with a cautionary note. Put simply, taking the mean of a categorical variable yields nonsense. Consider the race variable - R will let we compute the mean of it. Try the following:

```
summary(nids$race) #summary factor returns the frequencies
```

```
##  African Coloured    Asian    White     NA's
##    22157     4166      439     1432     2976
```

```
summary(as.numeric(nids$race))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   1.000   1.000   1.000   1.331   1.000   4.000    2976
```

We find the mean of the `w1_best_race` variable is 1.33. What does this mean? As near as we can tell, it means nothing. It certainly does not mean that the average respondent is about halfway between an African and a Coloured respondent. The coding of the race variable was arbitrary. NIDS dataset could just as easily have coded Whites as a 1, Indians & Asians as a 2, Africans as a 3, and Coloureds as a 4. If the mean of a categorical variable is nonsensical, are there other measures that do convey information? There are two - the mode and the range of the distribution.

The mode of the distribution, as we have already mentioned is that value that appears most often in the sample. In R, we can use `table()` or `summary()` on the factor variable. To find the mode of the race variable (`race`) and primary occupation variable (`w1_a_e4_code`), type:

```
summary(nids$race)
```

```
##  African Coloured    Asian    White     NA's
##    22157     4166      439     1432     2976
```

```
table(nids$w1_a_e4_code)
```

```
##
##  -9  -3   1   2   3   4   5   6   7   8   9
##  59  13 147 474 153 328 468 389 492 344 968
```

Or the `dplyr` `group_by` `summarise` approach:

```
nids %>%
  group_by(w1_a_e4_code) %>%
  summarise(freq=n()) %>%
  na.omit()
```

```
## # A tibble: 11 x 2
##    w1_a_e4_code  freq
##           <int> <int>
##  1           -9    59
##  2           -3    13
##  3            1   147
##  4            2   474
##  5            3   153
##  6            4   328
##  7            5   468
##  8            6   389
##  9            7   492
## 10            8   344
## 11            9   968
```

We find that the mode of race is "African" while the mode of the occupation variable is 9 - "Elementary Occupations."

Just as the mean of the distribution of a categorical variable does not make any sense, neither does the standard deviation. Still, it may be useful to know the range of the values of a categorical variable. That is, what values are spanned by the variable codes? To see the range of a variable, you can simply use the `summary()` or `range()` command:

```
summary(nids$w1_a_e4_code)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  -9.000   4.000   6.000   5.705   9.000   9.000   27335
```

The above result tells us that the occupation variable varies from -9 to 9.

Table 8 - Occupational Codes

-3	Missing
-9	Don't Know
1	Legislators senior officials and managers
2	Professionals
3	Technicians and associate professionals
4	Clerks
5	Service workers and shop and market sales workers
6	Skilled agricultural and fishery workers
7	Craft and related trades workers
8	Plant and machinery operators and assemblers
9	Elementary occupations
-7	Never worked

Figure 5.2: Occupational Codes

5.11 Worked example 1: Exploring the distribution of satisfaction variable

There are many variables that we will want to explore in the NIDS data set. One that we will work with often in our examples and exercises is the variable `w1_a_m5`. We have already met this variable in chapter 3. Recall that it measures adults' satisfaction level with their perceived current quality of life. If we refer to the adult questionnaire Section M, Question 2 we see that respondents were asked: 'Using a scale of 1 to 10, where 1 means "Very dissatisfied" and 10 means "Very satisfied", how do you feel about your life as a whole right now?'.

Unlike variables such as age, income, or number of children, the units of measurement for a variable measuring satisfaction may not have been that obvious. In this sense, the "Satisfaction" variable (as we will call it) is not a typical continuous variable. In the language of statistics, it is referred to as an ordinal variable (as opposed to nominal, which has no inherent order). To read more about the subjective wellbeing variables in the NIDS Dataset refer to Discussion Paper 14 of the NIDS survey. The discussion papers are available from the NIDS website.

We will use this variable to "test drive" some of the R commands we have learned. First it is always best to check how the variable is coded in R. To do this we type:

```
table(nids$w1_a_m5)
```

```
##
##   -9   -8   -5   -3    1    2    3    4    5    6    7    8    9   10
## 1433  163    2  240 1007  701 1246 1867 2558 1803 1633 1305  455 1217
```

As we expect a code of 10 indicates the highest level of satisfaction and a code of 1 indicates the lowest level. In addition, values where respondents didn't give a value on the 'satisfaction scale' were coded with negative numbers. Before analyzing the data, we recode these to "missing" and create a new variable to work with by typing:

```
nids<-nids%>%
  mutate(satisfaction = ifelse(w1_a_m5<0,NA,w1_a_m5))
```

These lines of code tells R to create a new variable called `satisfaction` that is exactly the same as `w1_a_m5`, except that all negative values (between -9 and -3) are recoded to NA-missings. Now we are ready to investigate how respondents perceive their level of satisfaction with life. We can compute the overall mean level of satisfaction and the mean by `w1_hhgeo` by typing:

```
summary(nids$satisfaction)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   1.000   4.000   5.000   5.464   7.000  10.000   17378
```

```
nids%>%
  group_by(w1_hhgeo)%>%
  summarise(Mean=mean(satisfaction, na.rm=TRUE),
       Std.Dev=sd(satisfaction, na.rm=TRUE),
       Min=min(satisfaction, na.rm=TRUE),
       Max=max(satisfaction, na.rm=TRUE))
```

```
## # A tibble: 4 x 5
##   w1_hhgeo                Mean Std.Dev   Min   Max
##   <fct>                  <dbl>   <dbl> <dbl> <dbl>
## 1 Rural formal            5.46    2.37     1    10
## 2 Tribal authority areas  4.81    2.39     1    10
## 3 Urban formal            6.10    2.37     1    10
## 4 Urban informal          5.13    2.65     1    10
```

We see that the overall average level of satisfaction in the sample is 5.46 - just above halfway between "very dissatisfied" and "very satisfied." The overall median, which must be an integer since this variable takes only integer values, equals 5. It is unclear how far we can take the interpretation of these results. Some social scientists do not believe it is appropriate to take the mean of an ordinal variable, while others do see value in such an effort. Those who consider the mean of an ordinal variable as useless argue that means for these types of variables are based on arbitrary numerical values that have no real meaning. For instance do we even really know what an average value of 5 mean in terms of satisfaction? In addition, intervals between categories are not necessarily equal to one another as a numerical scale would suggest, for instance, taking 5 to hypothetically mean "moderately satisfied", does it take the same increase in perceived satisfaction to move from "very dissatisfied" to "moderately satisfied" as it does from "moderately satisfied" to "very satisfied"? Those who find value in the mean of an ordinal variable understand that in general, a mean of 5.46 indicates that most people in general feel neither "very dissatisfied" or "very satisfied" but rather lie in the middle portion of the distribution (if not leaning slightly more towards "the very satisfied" side of the distribution.

2. How would we compute the average food expenditure by province?

Question 2 Answer

5.12 Group summary

5.13 Worked example 2: Investigating BMI in South Africa

In Chapter 2 we created a body mass index (BMI) variable from the height and weight variables. Let's look at this variable.

A recap:

```
summary(nids$bmi)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   6.762  21.696  25.485  27.188  30.964 292.731   20296
```

Since BMI is a relationship between a person's weight and height it is not easy to interpret directly. However, the World Health Organisation (WHO) classifies a BMI under 18.5 as underweight, a BMI between 18.5 and 24.9 as normal, a BMI between 25 and 29.9 as overweight and a BMI of 30 or more as obese. The average BMI in the sample is 27.2, which indicates that the average adult in the sample is a little overweight.

We also see that the largest and smallest BMI values are 292.7 and 6.76 respectively. The vast majority (over 98%) of respondents, however, have a BMI between 15 and 50. How many people have values outside the range?

```
nids%>%
  filter(bmi < 15 | (bmi > 50 & !is.na(bmi)))%>%
  nrow
```

```
## [1] 240
```

```
#nrow(subset(nids, bmi < 15 | (bmi > 50 & !is.na(bmi))))
```

Let us inspect the height and weight variables for these extreme BMI values.

```
nids%>%
  select(weight, height, bmi)%>%
  filter(bmi < 15 | bmi > 50)%>%
  head(20)
```

```
## Adding missing grouping variables: `hhid`
```

```
## # A tibble: 20 x 4
## # Groups:   hhid [18]
##      hhid weight height   bmi
##     <int>  <dbl>  <dbl> <dbl>
##  1 101075  119.   1.51   51.9
##  2 101171   30.2  1.5    13.4
##  3 101196  111.   1.48   50.5
##  4 101244   34    1.56   14.0
##  5 101517   36    1.62   13.7
##  6 101517   35.5  1.61   13.6
##  7 101593  133    1.57   54.0
##  8 101681   66.7  1.1    55.1
##  9 101700   45.2  1.80   13.9
## 10 101700  128.   1.57   51.9
## 11 101749   44.6  1.81   13.6
## 12 102173   37.1  1.59   14.6
## 13 102191  111.   1.43   54.2
## 14 102254  125.   1.54   52.9
## 15 102485  132    1.57   53.8
## 16 102568   34.9  1.69   12.2
## 17 102637   36.5  1.58   14.6
## 18 102640   37.6  1.74   12.5
## 19 102663  121.   1.56   50.1
## 20 102681   66.6  0.585 195.
```

```
#head(subset(nids,subset=(bmi < 15 | (bmi > 50 & bmi<=max(bmi))), select=c(weight, height, bmi)) )
```

Do they seem realistic to you? Remember that when dealing with the BMI variable, we are only considering adults over the age of 20. It might be appropriate to restrict our sample to people who have BMI values between 15 and 50 and define anyone with a BMI value outside this range as an outlier. This will be important if we want to exclude outliers from our analysis.

Does the exclusion of the outliers change the mean significantly?

First summarise all values:

```
summary(nids$bmi)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   6.762  21.696  25.485  27.188  30.964 292.731   20296
```

Summarise a subset of the bmi variable:

```
summary(subset(nids, subset=(bmi >15 & bmi < 50), select=c(bmi)))
```

```
##       bmi
##  Min.   :15.01
##  1st Qu.:21.74
##  Median :25.45
##  Mean   :26.75
##  3rd Qu.:30.78
##  Max.   :49.98
```

```
#summary(nids[nids$bmi >15 & nids$bmi < 50,c("bmi")])
```

Removing outliers reduces the arithmetic mean by over 0.44 from 27.19 to 26.75. In doing a research project, one should always be extremely careful before removing outliers and make sure that the decision is justified. Here, it goes beyond the scope of the course to fully investigate whether the outliers are valid or not. Therefore, we will ignore outliers from now on even though we have not presented a strong case for doing so.

```
nids<-nids%>%
  mutate(bmi_valid = ifelse(bmi > 15 & bmi < 50,1,NA))
```

Since BMI is a continuous variable, if we want to use a pie graph to get a general picture of the data, we need to divide the respondents into BMI categories. Let's use the WHO categories described above.

You have learnt in the chapters how to recode a variable. Below, we give the base R code to generate BMI categories.

```
nids$bmi.bins.nolabel<-NA
nids$bmi.bins.nolabel[which(nids$bmi>=15 & nids$bmi<18.5)]<-1
nids$bmi.bins.nolabel[which(nids$bmi>=18.5 & nids$bmi<25)]<-2
nids$bmi.bins.nolabel[which(nids$bmi>=25 & nids$bmi<30)]<-3
nids$bmi.bins.nolabel[which(nids$bmi>=30 & nids$bmi<=50)]<-4
```

The code above only recoded those BMI values between 15 and 50. We saw that 240 adults have values outside this range and we want to classify these as outliers and change them to missing values in our recoded bmi.bins.nolabel variable.

To generate another new variable, but with labels or a factor, simply type;

```
nids$bmi.bins<-factor(nids$bmi.bins.nolabel, levels=1:4, labels = c("Underweight","Normal", "Overweight
```

Check that the variable looks correct.

```
nids%>%
  select(bmi, bmi.bins.nolabel, bmi.bins)%>%
  head(15)
```

```
## Adding missing grouping variables: `hhid`
```

```
## # A tibble: 15 x 4
## # Groups:   hhid [5]
##      hhid   bmi bmi.bins.nolabel bmi.bins
##     <int> <dbl>            <dbl> <fct>
##  1 101012  30.9                4 Obese
##  2 101013  19.3                2 Normal
##  3 101013  23.1                2 Normal
##  4 101013  NA                 NA <NA>
```

```
##  5 101013  NA                NA <NA>
##  6 101013  NA                NA <NA>
##  7 101014  NA                NA <NA>
##  8 101014  NA                NA <NA>
##  9 101014  29.4               3 Overweight
## 10 101014  25.7               3 Overweight
## 11 101014  22.9               2 Normal
## 12 101014  NA                NA <NA>
## 13 101015  19.8               2 Normal
## 14 101015  NA                NA <NA>
## 15 101016  NA                NA <NA>
```

Use the following questions to get better acquainted with the BMI variable before we move onto the second half of the worked example.

3. What proportion of the adult sample is obese? What proportion of adults in the sample has a missing BMI value? How many of these were outliers that we recoded to be missing due to having an extreme BMI value?

Question 3 Answer

4. First use a pie graph to graphically depict the BMI distribution in our sample. Now see if you can compare the BMI of individuals both by gender and by whether they reside in an urban or rural area. Include percentages to indicate the size of each of the slices of the pies. Are there bigger differences between the BMI distribution for men and women or between rural and urban areas?

Question 4 Answer

5. Bonus Question: See whether you can graphically examine the relationship between BMI and age and gender in a single graph.

Question 5 Answer

Continuation of Worked Example 2: Investigating BMI in South Africa

So far we have used some graphs to gain a better picture of our BMI variable and how it relates to gender, age and an urban/rural living environment. Graphs are often good first getting a first overview of a relationship. But we can also use descriptive statistics disaggregated by different demographic characteristics to give us an even better understanding of BMI.

Let us compare the mean BMI of different racial groups.

```
#by(nids[,c("bmi")], nids$race, summary)
nids%>%
  group_by(race)%>%
  summarise(Mean=mean(bmi, na.rm=TRUE),
        Std.Dev=sd(bmi, na.rm=TRUE),
        Min=min(bmi, na.rm=TRUE),
        Max=max(bmi, na.rm=TRUE))
```

```
## # A tibble: 5 x 5
##   race      Mean Std.Dev   Min   Max
##   <fct>    <dbl>   <dbl> <dbl> <dbl>
## 1 African   27.2    9.06  6.76 202.
## 2 Coloured  27.0   12.2  10.6  293.
## 3 Asian     26.6    5.69 16.1   44.1
## 4 White     28.2    5.83 14.5   63.2
## 5 <NA>      26.7    4.59 22.8   36.9
```

This shows that on average Whites have a higher BMI than the other racial categories. We can also see from the standard deviations that there is far more variation in the BMI values in the African and Coloured groups. The minimum value in the African group is 6.8; the maximum value is 202 and the standard deviation around 9. This compares with a minimum and maximum of 14.5 and 63 respectively, and a standard deviation of 5.8 in the White group. Why would Whites have the highest BMI on average? One reason could be to do with differences in the age distributions between racial groups. How does BMI vary with age?

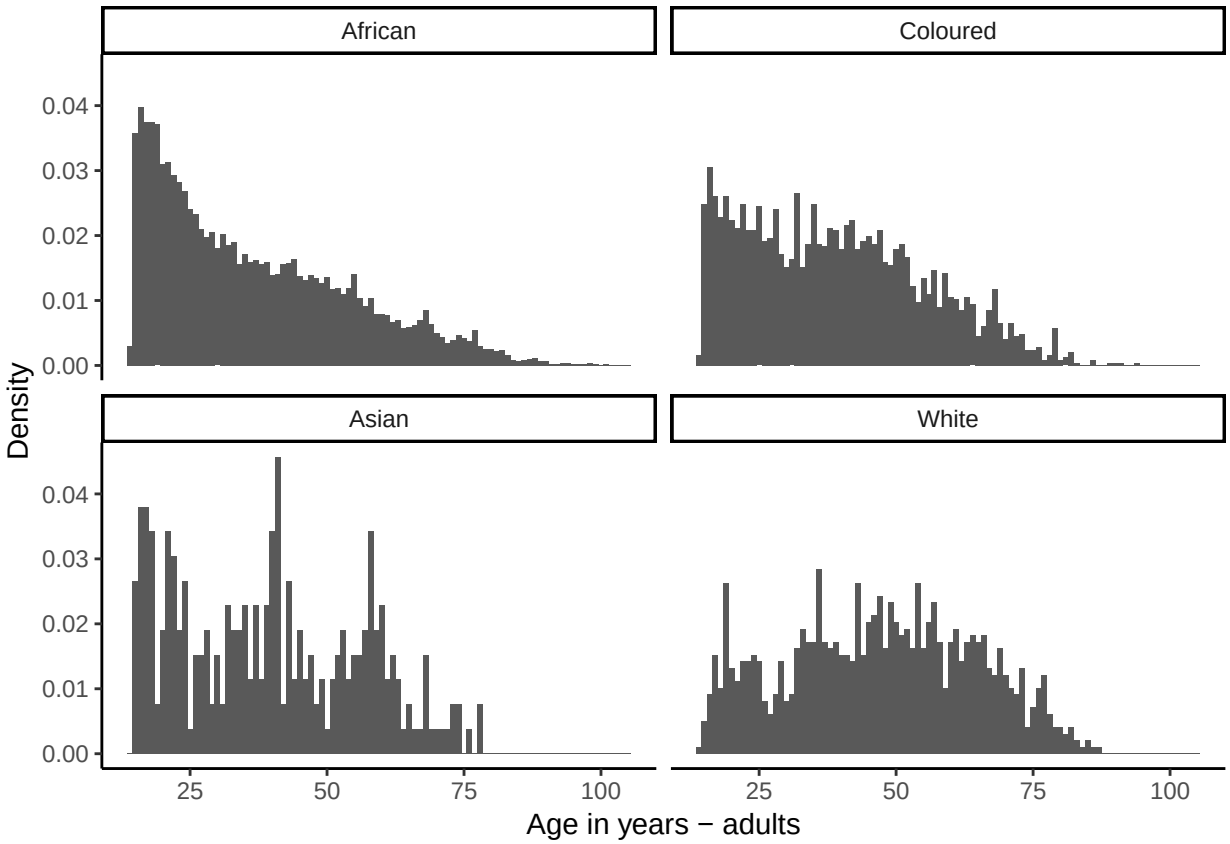
```
nids%>%
  group_by(age_bins)%>%
  summarise(Mean=mean(bmi, na.rm=TRUE),
       Std.Dev=sd(bmi, na.rm=TRUE),
       Min=min(bmi, na.rm=TRUE),
       Max=max(bmi, na.rm=TRUE)) %>%
  na.omit()
```

```
## # A tibble: 6 x 5
##   age_bins      Mean Std.Dev   Min   Max
##   <fct>        <dbl>   <dbl> <dbl> <dbl>
## 1 20 - 29 yrs   25.1    8.24  6.76  195.
## 2 30 - 39 yrs   27.2    9.57 11.6   211.
## 3 40 - 49 yrs   28.2    9.82  7.08  202.
## 4 50 - 59 yrs   28.4    8.34  9.90  109.
## 5 60 - 69 yrs   28.5    8.50 13.4   122.
## 6 70 - 120 yrs  27.7   13.3   8.62  293.
```

BMI increases with age in the first four categories and then stabilizes, before declining when individuals reach old age. Note also that for all age group categories the average BMI is in the overweight category. Notice also that BMI tends to be higher in older age categories. This means that if the White group has, on average, more people in the older age categories than other racial groups, this might contribute to the higher average White BMI value. Let's look at the age distributions across racial groups.

```
nids%>%
  filter(!is.na(race))%>%
  ggplot(., aes(x=age_adult)) +
  geom_histogram( aes(y = ..density..), binwidth = 1) +
  labs(x="Age in years - adults", y = "Density") +
  facet_wrap(~race) +
  theme_classic()
```

```
## Warning: Removed 11605 rows containing non-finite values (stat_bin).
```



There are differences in the age profiles across racial groups. The African and Coloured groups are weighted quite heavily towards younger adults, while the White group appears to have a more equal number of people across all ages.

Thus a better comparison of BMI across racial groups should take into consideration the differences in the age profile. We will look at this further in chapter 8.

6. Plot a histogram showing the BMI distribution in South Africa. What does this tell us about the BMI distribution in South Africa?

Question 6 Answer

7. You may have noticed in question 6 that the outliers stretch out the histogram (give it a long tail to the right) and make it difficult to read the graph. Repeat what you did in question 6 but restrict your histogram to exclude outliers. Also, add xlines which indicate which parts of the histogram reflect underweight, normal, overweight and obese individuals.

Question 7 Answer

5.14 Question answers

5.15 Exercises

Now it is our turn to explore the measures of central tendency and variability using the commands from Chapter 4. Using R and the NIDS data set, answer the following questions.

1. What is the average age of heads of households in South Africa?

Exercise 1 Answer

2. Do people with income levels at or above the median level report a higher level of satisfaction than those who report an income below the median level, and if so, by how much?

Exercise 2 Answer

3. Is income more widely distributed in rural or urban formal areas?

Exercise 3 Answer

4. What language is spoken by more households than any other? That is, what is the modal household language? Exercise 4 Answer

5. Is the percentage of respondents who self-identify as being a "professional" as their occupation higher for people who have recently moved or for people who have not moved recently?

Exercise 5 Answer

5.16 Exercise answers

5.17 Exercises

Now it is our turn to explore the measures of central tendency and variability using the commands from Module 4. Using R and the NIDS data set, answer the following questions.

1. What is the average age of heads of households in South Africa?

Exercise 1 Answer

2. Do people with income levels at or above the median level report a higher level of satisfaction than those who report an income below the median level, and if so, by how much?

Exercise 2 Answer

3. Is income more widely distributed in rural or urban formal areas?

Exercise 3 Answer

4. What language is spoken by more households than any other? That is, what is the modal household language?

Exercise 4 Answer

5. Is the percentage of respondents who self-identify as being a "professional" as their occupation higher for people who have recently moved or for people who have not moved recently?

Exercise 5 Answer

5.18 Exercise answers

5.19 Session information

```
print(sessionInfo(), locale = FALSE)
```

```
## R version 3.5.1 (2018-07-02)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 7 x64 (build 7601) Service Pack 1
##
## Matrix products: default
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] scales_0.5.0    bindrcpp_0.2.2  forcats_0.3.0   stringr_1.3.1
##  [5] dplyr_0.7.6     purrr_0.2.5     readr_1.1.1     tidyr_0.8.1
##  [9] tibble_1.4.2    ggplot2_3.0.0   tidyverse_1.2.1 foreign_0.8-70
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_0.2.4 xfun_0.2         reshape2_1.4.3   haven_1.1.2
##  [5] lattice_0.20-35  colorspace_1.3-2 htmltools_0.3.6  yaml_2.1.19
##  [9] utf8_1.1.4       rlang_0.2.1      pillar_1.2.3     withr_2.1.2
## [13] glue_1.2.0       modelr_0.1.2     readxl_1.1.0     bindr_0.1.1
## [17] plyr_1.8.4       munsell_0.5.0    gtable_0.2.0     cellranger_1.1.0
## [21] rvest_0.3.2      psych_1.8.4      evaluate_0.10.1  labeling_0.3
## [25] knitr_1.20       parallel_3.5.1   highr_0.7        broom_0.4.5
## [29] Rcpp_0.12.17     backports_1.1.2  jsonlite_1.5     mnormt_1.5-5
## [33] hms_0.4.2        digest_0.6.15    stringi_1.1.7    bookdown_0.7
## [37] grid_3.5.1       rprojroot_1.3-2  cli_1.0.0        tools_3.5.1
## [41] magrittr_1.5     lazyeval_0.2.1   crayon_1.3.4     pkgconfig_2.0.1
## [45] xml2_1.2.0       lubridate_1.7.4  assertthat_0.2.0 rmarkdown_1.10
## [49] httr_1.3.1       rstudioapi_0.7   R6_2.2.2         nlme_3.1-137
## [53] compiler_3.5.1
```

Chapter 6

Bivariate analysis (cross tabs)

6.1 Getting ready

In the previous chapters we generated some variables and ran some commands that will influence the results that we get in this chapter. If you are starting a new session of R, please run the following lines of code before you begin the chapter. Make sure that you remember what each line of code is doing.

```
library(foreign)
library(tidyverse)
```

```
## -- Attaching packages ------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

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

```
nids<-read.dta("./data/nids.dta", convert.factors=FALSE)

nids<-nids%>%
  arrange(hhid, pid)%>%
  group_by(hhid) %>%
  mutate(hhrestrict = 1:n()) %>%
  mutate(hhrestrict = ifelse(hhrestrict==1,1,0))


#Class
nids$class<-NA
nids$class[which(nids$w1_fwag<=1500)]<-1
nids$class[which(nids$w1_fwag>1500 & nids$w1_fwag<=4500)]<-2
nids$class[which(nids$w1_fwag>4500)]<-3

nids$class<-factor(nids$class, levels=1:3, labels = c("Lower Class","Middle Class","Upper Class"))


###Creating a BMI variable - from chapter 2 ***
```

```
#Height
nids<-nids %>%
  mutate(height = ifelse (w1_a_n1_1 >= 0 & w1_a_best_age_yrs >= 20, w1_a_n1_1/100, NA))

#Weight
nids<-nids %>%
  mutate(weight = ifelse (w1_a_n2_1 >= 0 & w1_a_best_age_yrs >= 20, w1_a_n2_1, NA))

#BMI
nids<-nids %>%
  mutate(bmi = weight/height^2)

#Valid BMI values
nids<-nids%>%
  mutate(bmi_valid = ifelse(bmi > 15 & bmi < 50,1,NA))

#BMI bins
nids$bmi.bins.nolabel<-NA
nids$bmi.bins.nolabel[which(nids$bmi>=15 & nids$bmi<18.5)]<-1
nids$bmi.bins.nolabel[which(nids$bmi>=18.5 & nids$bmi<25)]<-2
nids$bmi.bins.nolabel[which(nids$bmi>=25 & nids$bmi<30)]<-3
nids$bmi.bins.nolabel[which(nids$bmi>=30 & nids$bmi<=50)]<-4

nids$bmi.bins<-factor(nids$bmi.bins.nolabel, levels=1:4, labels = c("Underweight","Normal", "Overweight

#Age
nids<-nids%>%
  mutate(age_adult = ifelse(w1_a_best_age_yrs<0,NA, w1_a_best_age_yrs))

#Age bins
nids$age_bins<-NA
nids$age_bins[which(nids$w1_r_best_age_yrs>=20 & nids$w1_r_best_age_yrs<=29)]<-1
nids$age_bins[which(nids$w1_r_best_age_yrs>29 & nids$w1_r_best_age_yrs<=39)]<-2
nids$age_bins[which(nids$w1_r_best_age_yrs>39 & nids$w1_r_best_age_yrs<=49)]<-3
nids$age_bins[which(nids$w1_r_best_age_yrs>49 & nids$w1_r_best_age_yrs<=59)]<-4
nids$age_bins[which(nids$w1_r_best_age_yrs>59 & nids$w1_r_best_age_yrs<=69)]<-5
nids$age_bins[which(nids$w1_r_best_age_yrs>69 & nids$w1_r_best_age_yrs<=120)]<-6

nids$age_bins <- factor(nids$age_bins, levels = 1:6, labels = c("20 - 29 yrs","30 - 39 yrs", "40 - 49 y

#Rename
nids <- nids%>%
  mutate(race = w1_best_race,
         age = w1_r_best_age_yrs,
         gender = w1_r_b4,
         province = w1_hhprov,
         hhincome = w1_hhincome) %>%
  mutate(gender = factor(gender, levels = 1:2, labels = c("Male", "Female")),
          race = factor(race, levels = 1:4, labels = c("African", "Coloured","Asian", "White")),
          province = factor(province, levels=1:9, labels = c("Western Cape","Eastern Cape","Northern Cap
          w1_hhgeo = factor(w1_hhgeo, levels = 1:4, labels = c("Rural formal", "Tribal authority areas",
```

6.2 Introduction

Up to this point the majority of our analysis has looked at one variable at a time. This is certainly useful, but we will often want to look at how variables interact with one another. Doing this allows us to conduct more in depth analysis of a particular topic. For example, we have learned that to calculate the average monthly pay using R, we type:

```
summary(nids$w1_fwag)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    20.0   936.7  1600.0  3233.4  3550.0 90000.0   26671
```

This command gives us the mean for the entire sample. From the output we see that on average individuals earn R3233.36 per month (of those who report earning). When thinking about income, one can think of several factors that may affect what an individual earns.

Is it likely that an individual with a university degree will earn a higher wage than an individual with no formal education? Is it likely that an older people earn more on average? How do income levels differ by gender? By race?

Hence, while reporting the mean income for the entire sample is useful, examining how income varies by a second variable can be even more helpful in discovering trends in the data. This is an example of Bivariate Analysis. The term Bivariate Analysis sounds rather complicated but do not be intimidated by the name! Below we explain the basics of Bivariate Analysis:

What is bivariate analysis? Bivariate Analysis is the examination of two variables at the same time, hence the name bivariate. It is used frequently by social scientists and mathematicians to compare how two variables correspond with one another. While sophisticated equations can be written to model how one variable changes with respect other variables (regression, the subject of Chapters 7 and 8), we are concerned here with only two variables, whether they are mathematically related or not.

In this chapter, we will examine the relationship between these two variables (bivariate analysis) using crosstabs. A crosstab is a technique for analyzing the relationship between two variables that have been organized in a bivariate table. Using such a table, we can examine the presence and strength of the relationship between two variables.

When would we use bivariate analysis? Although it can be used whenever we have two variables that we want to examine at the same time, bivariate analysis is usually a good tool to apply when we have a hunch that two variables "go together." If this is the case then bivariate analysis allows us to compare them numerically.

For example, if we were interested in poverty across the country, it would be informative to know how much a household spends on food every month.

The variable we want for food expenditure is "`w1_h_expf` - food expenditure with full imputations". Now, we could use the `head()` command and generate a table with the household ID, individual's personal identification number (`pid`), and the amount his or her family spends on food every month.

```
nids%>%
  arrange(hhid)%>%
  select(hhid, pid, w1_h_expf)%>%
  head(20)
```

```
## # A tibble: 20 x 3
## # Groups:   hhid [7]
##      hhid    pid w1_h_expf
##     <int>  <int>     <dbl>
##  1 101012 314585       223
##  2 101013 314544       173
```

```
##  3 101013 314550       173
##  4 101013 406295       173
##  5 101013 406296       173
##  6 101013 406297       173
##  7 101014 301454       442
##  8 101014 314575       442
##  9 101014 314580       442
## 10 101014 314581       442
## 11 101014 314582       442
## 12 101014 406298       442
## 13 101015 314570       200
## 14 101015 406352       200
## 15 101016 314109       530
## 16 101016 314110       530
## 17 101017 314529       167
## 18 101017 314530       167
## 19 101017 314531       167
## 20 101018 314578       671
```

We can scan the chart to guess a family's well-being based upon monthly food expenditure, but the amount a family spends on food actually depends on many other things. For instance, we would expect that larger families would spend more on food every month than smaller families because they have more people to feed. In order to answer this question we will need to use the household size variable that we have worked with in previous chapters.

If you can recall, `w1_hhsizer` is the variable for household size (i.e. number of members). Try listing household size with the previous chart.

```
nids%>%
  arrange(hhid)%>%
  select(hhid, pid, w1_h_expf, w1_hhsizer)%>%
  head(20)
```

```
## # A tibble: 20 x 4
## # Groups:   hhid [7]
##      hhid    pid w1_h_expf w1_hhsizer
##     <int>  <int>     <dbl>      <int>
##  1 101012 314585       223          1
##  2 101013 314544       173          5
##  3 101013 314550       173          5
##  4 101013 406295       173          5
##  5 101013 406296       173          5
##  6 101013 406297       173          5
##  7 101014 301454       442          6
##  8 101014 314575       442          6
##  9 101014 314580       442          6
## 10 101014 314581       442          6
## 11 101014 314582       442          6
## 12 101014 406298       442          6
## 13 101015 314570       200          2
## 14 101015 406352       200          2
## 15 101016 314109       530          2
## 16 101016 314110       530          2
## 17 101017 314529       167          3
## 18 101017 314530       167          3
## 19 101017 314531       167          3
```

```
## 20 101018 314578       671          1
```

Now we have some more information that helps us get a better idea of why particular individuals might live in households that spend more money on food. Here, it makes sense that household 101014 would spend more than household 101015 on food because four more people live there. But notice that household size doesn't always explain the food expenditure we see. (For example compare household 101018 with household 101014). Let's consider another variable that may influence household food expenditure, such as its monthly income (`w1_hhincome`). Let us now try listing this variable with total household monthly food expenditure:

```
nids%>%
  arrange(hhid)%>%
  select(hhid, pid, w1_h_expf, w1_hhincome)%>%
  head(20)
```

```
## # A tibble: 20 x 4
## # Groups:   hhid [7]
##      hhid    pid w1_h_expf w1_hhincome
##     <int>  <int>     <dbl>       <dbl>
##  1 101012 314585       223       1045.
##  2 101013 314544       173        588.
##  3 101013 314550       173        588.
##  4 101013 406295       173        588.
##  5 101013 406296       173        588.
##  6 101013 406297       173        588.
##  7 101014 301454       442       1307.
##  8 101014 314575       442       1307.
##  9 101014 314580       442       1307.
## 10 101014 314581       442       1307.
## 11 101014 314582       442       1307.
## 12 101014 406298       442       1307.
## 13 101015 314570       200        291.
## 14 101015 406352       200        291.
## 15 101016 314109       530       1304.
## 16 101016 314110       530       1304.
## 17 101017 314529       167        213.
## 18 101017 314530       167        213.
## 19 101017 314531       167        213.
## 20 101018 314578       671       1785.
```

Here, it makes sense that household 101012 spends more money than 101013 on food because they have more money to begin with. Although we can't be sure that a relationship exists between the variables, we can get a general feel for the relationship between the variables by doing bivariate analysis. Later we will expand these techniques to include three (trivariate) and four (quadivariate) variables.

We have introduced the idea of bivariate analysis using the `head()`/`View()` commands. While these commands are simple in some respects, it is rather overwhelming when dealing with large numbers of observations and it is also not the most informative when we are trying to look at more than one variable at a time. In the above example, we know that household 101012 spends more on food every month than 101013 and, in addition, that household 101012 has a higher income. But to make any solid conclusions, we have to search up and down the large list of observations trying to find other examples that fit our hypotheses. This process is tedious and time consuming. We would never want to do it with all of the households in the sample.

6.3 Cross - tabulation

So how do we do bivariate analysis in R? Thank goodness there is another option - cross-tabulations. Suppose we want to look at how adult females perceive their health status compared to adult males. We should start by finding the variables. The two variables are: `w1_r_b4` (Gender of household member) and `w1_a_j1` (Respondent's perceived health status). We can tabulate each variable individually, but that isn't very helpful when trying to figure out how the two variables are related. There is only so much we can learn from tabulating each of these variables individually.

```
nids%>%
  group_by(w1_r_b4)%>%
  summarise(n=n())
```

```
## # A tibble: 2 x 2
##   w1_r_b4     n
##     <int> <int>
## 1       1 14643
## 2       2 16527
```

```
nids%>%
  group_by(w1_a_j1)%>%
  summarise(n=n())
```

```
## # A tibble: 9 x 2
##   w1_a_j1     n
##     <int> <int>
## 1      -9    21
## 2      -8     3
## 3      -3    71
## 4       1  4554
## 5       2  3927
## 6       3  3757
## 7       4  2058
## 8       5  1248
## 9      NA 15531
```

That's a good start, but we still haven't tabulated the two variables together. A bivariate table (or crosstab) is simply a table that displays the distribution of one variable "across" the categories of a second variable. There are several ways to create a bivariate table in R: `xtabs()` (`stats` package), `genTable()`(`memisc` package), `CrossTable()` (`descr`or `gmodels` package), `group_by` and `summarise()` (`dplyr` package) e.t.c.

We are going to use `CrossTable()` from the gmodels package. The first variable is treated as the row variable and the second is the column variable (see command description from the package documentation). Let us first tidy the perceived health status variable:

```
table(nids$w1_a_j1)
```

```
##
##   -9   -8   -3    1    2    3    4    5
##   21    3   71 4554 3927 3757 2058 1248
```

We see negative values for different types of missing and we can recode these to NA - missing and make it a factor.

```
nids$phealth<-ifelse(nids$w1_a_j1>0, nids$w1_a_j1, NA)
nids$phealth<-factor(nids$phealth, levels=1:5, labels=c("Excellent", "Very Good", "Good", "Fair", "Poor
table(nids$phealth)
```

J1 - Perceived Health Status
Variable: *w1_a_hldes*
Code list

-3	Missing
-8	Refused
-9	Don't Know
1	Excellent
2	Very good
3	Good
4	Fair
5	Poor

Figure 6.1: Labels for Perceived Health Status

```
##
## Excellent Very Good      Good      Fair      Poor
##      4554      3927      3757      2058      1248
```

```
library(gmodels)
CrossTable(nids$phealth, nids$gender)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  15544
##
##
##              | nids$gender
## nids$phealth |      Male |    Female | Row Total |
## -------------|-----------|-----------|-----------|
##    Excellent |      2136 |      2418 |      4554 |
##              |    52.050 |    34.890 |           |
##              |     0.469 |     0.531 |     0.293 |
##              |     0.342 |     0.260 |           |
##              |     0.137 |     0.156 |           |
## -------------|-----------|-----------|-----------|
##    Very Good |      1662 |      2265 |      3927 |
##              |     4.698 |     3.149 |           |
##              |     0.423 |     0.577 |     0.253 |
##              |     0.266 |     0.243 |           |
##              |     0.107 |     0.146 |           |
## -------------|-----------|-----------|-----------|
##         Good |      1362 |      2395 |      3757 |
##              |    14.086 |     9.442 |           |
##              |     0.363 |     0.637 |     0.242 |
```

```
##              |     0.218 |     0.257 |           |
##              |     0.088 |     0.154 |           |
## -------------|-----------|-----------|-----------|
##         Fair |       676 |      1382 |      2058 |
##              |    27.207 |    18.237 |           |
##              |     0.328 |     0.672 |     0.132 |
##              |     0.108 |     0.149 |           |
##              |     0.043 |     0.089 |           |
## -------------|-----------|-----------|-----------|
##         Poor |       402 |       846 |      1248 |
##              |    19.505 |    13.075 |           |
##              |     0.322 |     0.678 |     0.080 |
##              |     0.064 |     0.091 |           |
##              |     0.026 |     0.054 |           |
## -------------|-----------|-----------|-----------|
## Column Total |      6238 |      9306 |     15544 |
##              |     0.401 |     0.599 |           |
## -------------|-----------|-----------|-----------|
##
##
```

By looking at the table above we can see how perceived health status differs between the men and women in the survey. We see that out of the 6238 males who were meant to report their health status (i.e. not NA-missing), 2136 said their health status was excellent and 402 said it was poor. In general it seems that the results are similar for men and women in the sense that there seems to be a strong tendency for respondents to have a positive perception of their health. The majority of respondents, both male and female, perceived their health status to be above "Good". However, since we can see that the number of women who answered this question (9306 excluding missing) is much larger than the number of men (6238), it is difficult to compare them accurately by simply looking at the counts.

We can look at `N / Col Total`, that is the column proportions for each gender.We see that about 0.342 (34.2%) of men ({2136÷6238}×100) perceive their health to be "Excellent" compared to only 0.26 (26%) ({2418÷9306}×100) of women, suggesting that women may be more somewhat more critical of their health.

Let's try another example to answer one of the questions from the beginning of the chapter.

Is it likely that an individual with a university degree or higher will earn a higher wage than an individual with no form of university education?

Here it is important to first pay attention to the types of variables we are working with. Education is a categorical variable, while income tends to be a continuous variable.

We decided to divide the sample into those who have a university degree and those who do not. We can accomplish this by creating a dummy variable from `w1_r_b7`.

```
nids$uni<-1
nids$uni[nids$w1_r_b7 == 20 | nids$w1_r_b7 == 21 | nids$w1_r_b7 == 22 | nids$w1_r_b7 == 23]<-2
nids$uni[nids$w1_r_b7<0 | is.na(nids$w1_r_b7)]<-NA
nids$uni<-factor(nids$uni, levels=1:2, labels= c("no bachelor degree","bachelor degree"))
table(nids$uni)
```

```
##
## no bachelor degree    bachelor degree
##              30337                371
```

Method 1 - using `w1_fwag` (monthly take home pay)

Now we can summarise `w1_fwag` on the new `uni` education variable:

Table 5 - Education Codes

-3	Missing
-9	Don't Know
-8	Refused
-5	Not Applicable
0	Grade R/0
1	Grade 1/Sub A/Class 1
2	Grade 2/Sub B/Class 2
3	Grade 3/Std. 1
4	Grade 4/Std. 2
5	Grade 5/Std. 3
6	Grade 6/Std. 4
7	Grade 7/ Std. 5
8	Grade 8/ Std. 6/Form 1
9	Grade 9/Std. 7/Form 2
10	Grade 10/ Std. 8/Form 3
11	Grade 11/ Std. 9/Form 4
12	Grade 12/Std. 10/Form 5/Matric/Senior Certificate
13	NTC 1
14	NTC 2
15	NTC 3
16	Certificate with less than Grade 12/Std 10
17	Diploma with less than Grade 12/Std 10
18	Certificate with Grade 12/Std 10
19	Diploma with Grade 12/Std 10
20	Bachelors degree - Level 4
21	Bachelors degree and Diploma
22	Honours degree
23	Higher degree (Masters Doctorate)
24	Other
25	No Schooling
55	No Higher Education

Figure 6.2: (#fig:best_edu)Education codes

```
nids %>%
  filter(w1_fwag >0 & !is.na(w1_fwag)) %>%
  group_by(uni) %>%
  summarise(mean_inc=mean(w1_fwag, na.rm=TRUE),
            sd_inc=sd(w1_fwag, na.rm=TRUE),
            N=n())
```

```
## # A tibble: 3 x 4
##   uni                mean_inc sd_inc     N
##   <fct>                 <dbl>  <dbl> <int>
## 1 no bachelor degree    2844.  4373.  4249
## 2 bachelor degree      12194. 11588.   194
## 3 <NA>                  1764.  1711.    56
```

From the table, the average monthly income for University graduates is over 4 times that of people without a bachelor's degree.

Method 2 - using derived `class` variable

First, we want to reacquaint ourselves with the `class` variable that we created in the previous chapter (you should have it in your dataset from running the 'getting ready' commands). Can you remember whether a 1 is 'upper class' or 'lower class'? Luckily, we gave the variable a value label to remind us:

```
table(nids$class)
```

```
##
##  Lower Class Middle Class  Upper Class
##         2167         1503          829
```

Since, we already have our categorical/factor income variable, we are in a position to run a cross-tab of income and our university indicator variable:

```
CrossTable(nids$uni, nids$class)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  4443
##
##
##                     | nids$class
##           nids$uni |  Lower Class | Middle Class |  Upper Class |    Row Total |
## -------------------|--------------|--------------|--------------|--------------|
## no bachelor degree |         2131 |         1454 |          664 |         4249 |
##                    |        3.985 |        0.951 |       20.358 |              |
##                    |        0.502 |        0.342 |        0.156 |        0.956 |
##                    |        0.999 |        0.981 |        0.803 |              |
##                    |        0.480 |        0.327 |        0.149 |              |
```

```
## --------------------|--------------|--------------|--------------|--------------|
##    bachelor degree  |            3 |           28 |          163 |          194 |
##                     |       87.276 |       20.826 |      445.884 |              |
##                     |        0.015 |        0.144 |        0.840 |        0.044 |
##                     |        0.001 |        0.019 |        0.197 |              |
##                     |        0.001 |        0.006 |        0.037 |              |
## --------------------|--------------|--------------|--------------|--------------|
##       Column Total  |         2134 |         1482 |          827 |         4443 |
##                     |        0.480 |        0.334 |        0.186 |              |
## --------------------|--------------|--------------|--------------|--------------|
##
##
```

As we have mentioned before, dividing income into three arbitrary categories is rather crude. Nevertheless, it gives us a clear pattern regarding the relationship between having a university degree and the income level achieved in the sample. Looking along the first row of `N`, we see that the majority of people without a bachelors degree are in the 'lower class' income bracket. In contrast, looking along the second row we see that the vast majority of people with a bachelors degree are in the 'upper class' income bracket. It is useful to notice that you can interpret the table in a different way by looking at the columns. For instance, if we look at the third column we see that the majority of upper class people don't have a bachelor's degree (i.e. 664 out of the 827 in the 'upper class' income bracket don't have a bachelor's degree).

We can also look at this in terms of column percentage `N / Col Total` or row percentage `N / Row Total`, the third and fourth row in each cell. For example, 80.3 percent of "Upper Class" do not have a bachelor's degree or that 15.6% of those with no bachelor's degree are in upper class.

Now that you've learned these new methods, try answering the following questions:

1. How many people live in a house connected to electricity but still use wood as their main energy source for heating?

Question 1 Answer

2. In what province is English most widely spoken?

Question 2 Answer

6.4 Example 1: Where do households with an elderly member tend to live?

In order to answer this question, we will first want to create a household level variable that assigns the age of the oldest member of the household to every member of the household. It is not a simple task at all - in fact you are probably unable to do it with only the commands that you have learned so far. Luckily, the `dplyr` package provides us with a quick and simple way to do tasks of this nature. In this case, we use the **dplyr** command to calculate the oldest age within each household and return the result in a data frame:

```
nids<-nids%>%
  group_by(hhid)%>%
  mutate(maxage = max(w1_r_best_age_yrs, na.rm = T))
```

Let's examine more closely what this command tells R to do. We group nids data by the `hhid` variable and generating a new variable called `maxage` (`=max(w1_r_best_age_yrs)`) that is the maximum value of the `w1_r_best_age_yrs` variable in each unique `hhid`. Therefore R assigns every individual in the household a value that is equivalent to the oldest age of the person in that individual's household.

In summary, this line of commands tells R to generate a new variable called maxage and to assign it the value of the maximum age within the household for every member of the household. To verify this command,

type the command:

```
head(nids[,c("hhid", "w1_r_best_age_yrs", "maxage")], n = 25L)
```

```
## # A tibble: 25 x 3
## # Groups:   hhid [8]
##      hhid w1_r_best_age_yrs maxage
##     <int>             <int>  <int>
##  1 101012                51     51
##  2 101013                45     45
##  3 101013                32     45
##  4 101013                 9     45
##  5 101013                13     45
##  6 101013                11     45
##  7 101014                15     62
##  8 101014                25     62
##  9 101014                60     62
## 10 101014                22     62
## # ... with 15 more rows
```

We now have a variable that is the age of the oldest person in each household. It would be possible to run a simple cross tab now, but those results would be incorrect without controlling for the number bias (we only want to count each household once). Thus, we need to correct for that:

```
nids_max_age<-nids%>%
  filter(hhrestrict==1)

#CrossTable(nids_max_age$maxage, nids_max_age$w1_hhgeo, expected = FALSE, prop.r = FALSE, prop.c = FALS
```

For easy of printing to screen, we decided to shorten the level labels for `w1_hhgeo` to:

`RF` - Rural formal

`TAA` - Tribal authority areas

`UF` - Urban formal

`UI` - Urban informal

and create a new `w1_hhgeo1` variable.

```
nids_max_age$w1_hhgeo1<-nids_max_age$w1_hhgeo
levels(nids_max_age$w1_hhgeo1)<-c('RF','TAA','UF','UI')
CrossTable(nids_max_age$maxage, nids_max_age$w1_hhgeo1, expected = FALSE, prop.r = FALSE, prop.c = FALS
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |-------------------------|
##
##
## Total Observations in Table:  7305
##
##
##                      | nids_max_age$w1_hhgeo1
## nids_max_age$maxage |        RF |       TAA |        UF |        UI | Row Total |
## --------------------|-----------|-----------|-----------|-----------|-----------|
```

```
##                -9 |         6 |         3 |         6 |         0 |        15 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                -8 |         0 |         0 |         1 |         0 |         1 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                -3 |         1 |         0 |         0 |         0 |         1 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                 8 |         0 |         1 |         0 |         0 |         1 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                 9 |         0 |         1 |         0 |         0 |         1 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                12 |         0 |         1 |         0 |         0 |         1 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                14 |         0 |         1 |         0 |         0 |         1 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                15 |         0 |         2 |         4 |         0 |         6 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                16 |         1 |         1 |         1 |         0 |         3 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                17 |         0 |         0 |         1 |         0 |         1 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                18 |         1 |         8 |         5 |         2 |        16 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                19 |         2 |         9 |        23 |         3 |        37 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                20 |         4 |        13 |        13 |         3 |        33 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                21 |         5 |        12 |        17 |         5 |        39 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                22 |        10 |        12 |        27 |         4 |        53 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                23 |        11 |        13 |        34 |        10 |        68 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                24 |        12 |        15 |        30 |         7 |        64 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                25 |        15 |        16 |        36 |         7 |        74 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                26 |        17 |        21 |        33 |        10 |        81 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                27 |        14 |        15 |        38 |        12 |        79 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                28 |        18 |        23 |        48 |        11 |       100 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                29 |        14 |        29 |        51 |        12 |       106 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                30 |        25 |        25 |        40 |        13 |       103 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                31 |        19 |        22 |        52 |        11 |       104 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                32 |        17 |        29 |        59 |        16 |       121 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                33 |        18 |        39 |        66 |         9 |       132 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
##                34 |        19 |        33 |        65 |        12 |       129 |
## ------------------|-----------|-----------|-----------|-----------|-----------|
```

```
##                 35 |        17 |        43 |        68 |         9 |       137 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 36 |        25 |        29 |        64 |        12 |       130 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 37 |        19 |        43 |        87 |        12 |       161 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 38 |        24 |        36 |        68 |        13 |       141 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 39 |        19 |        39 |        80 |        16 |       154 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 40 |        20 |        43 |        79 |        10 |       152 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 41 |        22 |        35 |        87 |        13 |       157 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 42 |        21 |        50 |        82 |         7 |       160 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 43 |        17 |        49 |        95 |         9 |       170 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 44 |        22 |        55 |        97 |        15 |       189 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 45 |        23 |        50 |        63 |        11 |       147 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 46 |        18 |        61 |        74 |        15 |       168 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 47 |        17 |        59 |        80 |        10 |       166 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 48 |        12 |        61 |        90 |         9 |       172 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 49 |        20 |        51 |        75 |         8 |       154 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 50 |        17 |        61 |        77 |         9 |       164 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 51 |        27 |        44 |        92 |        13 |       176 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 52 |        12 |        66 |        68 |        11 |       157 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 53 |        15 |        58 |        68 |         7 |       148 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 54 |        12 |        65 |        74 |        17 |       168 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 55 |        20 |        78 |        78 |        12 |       188 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 56 |        21 |        56 |        57 |         7 |       141 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 57 |        13 |        53 |        64 |         5 |       135 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 58 |        10 |        69 |        55 |         4 |       138 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 59 |        14 |        58 |        39 |        10 |       121 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 60 |        11 |        46 |        52 |        11 |       120 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 61 |         8 |        48 |        48 |         7 |       111 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
```

```
##                 62 |         9 |        49 |        42 |         5 |       105 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 63 |        11 |        34 |        55 |        10 |       110 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 64 |         9 |        34 |        44 |         2 |        89 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 65 |         6 |        51 |        29 |         5 |        91 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 66 |         4 |        53 |        43 |         5 |       105 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 67 |        12 |        55 |        41 |         7 |       115 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 68 |         9 |        60 |        40 |         4 |       113 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 69 |        11 |        53 |        41 |         2 |       107 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 70 |         7 |        42 |        34 |         2 |        85 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 71 |         6 |        32 |        36 |         5 |        79 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 72 |         5 |        30 |        28 |         1 |        64 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 73 |         6 |        34 |        37 |         1 |        78 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 74 |         4 |        37 |        22 |         1 |        64 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 75 |        10 |        34 |        16 |         4 |        64 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 76 |         4 |        33 |        20 |         3 |        60 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 77 |         1 |        46 |        32 |         4 |        83 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 78 |         7 |        25 |        17 |         4 |        53 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 79 |         7 |        20 |        22 |         4 |        53 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 80 |         4 |        17 |        16 |         2 |        39 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 81 |         2 |        27 |         6 |         0 |        35 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 82 |         2 |        25 |        16 |         1 |        44 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 83 |         2 |        12 |        10 |         0 |        24 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 84 |         1 |         8 |         6 |         1 |        16 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 85 |         2 |         7 |         3 |         0 |        12 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 86 |         1 |        11 |         3 |         0 |        15 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 87 |         1 |        12 |         3 |         1 |        17 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
##                 88 |         1 |        15 |         4 |         1 |        21 |
## -------------------|-----------|-----------|-----------|-----------|-----------|
```

```
##                  89 |         0 |         3 |         5 |         2 |        10 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##                  90 |         3 |         6 |         3 |         0 |        12 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##                  91 |         0 |         2 |         4 |         0 |         6 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##                  92 |         0 |         3 |         2 |         0 |         5 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##                  93 |         0 |         4 |         2 |         0 |         6 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##                  94 |         1 |         5 |         2 |         0 |         8 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##                  95 |         1 |         1 |         2 |         0 |         4 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##                  96 |         0 |         2 |         0 |         0 |         2 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##                  97 |         1 |         1 |         1 |         1 |         4 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##                  98 |         0 |         2 |         2 |         1 |         5 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##                  99 |         0 |         2 |         0 |         0 |         2 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##                 100 |         1 |         0 |         1 |         0 |         2 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##                 101 |         0 |         1 |         1 |         0 |         2 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##                 105 |         0 |         1 |         0 |         0 |         1 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##        Column Total |       856 |      2639 |      3302 |       508 |      7305 |
## --------------------|-----------|-----------|-----------|-----------|-----------|
##
##
```

While there are some interesting households which report having very young members as the oldest in the household, if we examine the table closely it seems that most households with elderly people reside in Tribal Authority Areas and Urban Formal Areas. But if we look at the Total row, we can see that these areas also contain the majority of households. What do you think is the best way to address our question? One way would be to look at the proportion of total households in a given area that have a member over a certain age, say 80.

```
nids_max_age80<-nids_max_age%>%
  filter(maxage>80)

CrossTable(nids_max_age80$maxage, nids_max_age80$w1_hhgeo1, expected = FALSE, prop.r = FALSE, prop.c = F
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |-------------------------|
##
##
## Total Observations in Table:  253
##
```

```
##
##                        | nids_max_age80$w1_hhgeo1
## nids_max_age80$maxage |        RF |       TAA |        UF |        UI | Row Total |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    81 |         2 |        27 |         6 |         0 |        35 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    82 |         2 |        25 |        16 |         1 |        44 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    83 |         2 |        12 |        10 |         0 |        24 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    84 |         1 |         8 |         6 |         1 |        16 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    85 |         2 |         7 |         3 |         0 |        12 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    86 |         1 |        11 |         3 |         0 |        15 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    87 |         1 |        12 |         3 |         1 |        17 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    88 |         1 |        15 |         4 |         1 |        21 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    89 |         0 |         3 |         5 |         2 |        10 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    90 |         3 |         6 |         3 |         0 |        12 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    91 |         0 |         2 |         4 |         0 |         6 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    92 |         0 |         3 |         2 |         0 |         5 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    93 |         0 |         4 |         2 |         0 |         6 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    94 |         1 |         5 |         2 |         0 |         8 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    95 |         1 |         1 |         2 |         0 |         4 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    96 |         0 |         2 |         0 |         0 |         2 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    97 |         1 |         1 |         1 |         1 |         4 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    98 |         0 |         2 |         2 |         1 |         5 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                    99 |         0 |         2 |         0 |         0 |         2 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                   100 |         1 |         0 |         1 |         0 |         2 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                   101 |         0 |         1 |         1 |         0 |         2 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##                   105 |         0 |         1 |         0 |         0 |         1 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##          Column Total |        19 |       150 |        76 |         8 |       253 |
## ----------------------|-----------|-----------|-----------|-----------|-----------|
##
##
```

This gives us a slightly clearer picture as we can see that almost double the number of households with a

member over the age of 80 live in Tribal Authorities Areas in comparison to Urban Formal. Nevertheless, let's calculate the percentage of households with an elderly member in relation to the total number of households in a given area to get more precise results:

Total households

```
tothh<-nids_max_age %>%
  group_by(w1_hhgeo) %>%
  summarise(hh=n())
tothh
```

```
## # A tibble: 4 x 2
##   w1_hhgeo                  hh
##   <fct>                  <int>
## 1 Rural formal             856
## 2 Tribal authority areas  2639
## 3 Urban formal            3302
## 4 Urban informal           508
```

Over 80 households

```
tothh80<-nids_max_age80 %>%
  group_by(w1_hhgeo) %>%
  summarise(hh80=n())
tothh80
```

```
## # A tibble: 4 x 2
##   w1_hhgeo                hh80
##   <fct>                  <int>
## 1 Rural formal              19
## 2 Tribal authority areas   150
## 3 Urban formal              76
## 4 Urban informal             8
```

Bind together

```
hh<-cbind(tothh80,tothh[,2])
hh
```

```
##                 w1_hhgeo hh80   hh
## 1           Rural formal   19  856
## 2 Tribal authority areas  150 2639
## 3           Urban formal   76 3302
## 4         Urban informal    8  508
```

```
hh %>%
  mutate(percent=round(hh80/hh*100,2))
```

```
##                 w1_hhgeo hh80   hh percent
## 1           Rural formal   19  856    2.22
## 2 Tribal authority areas  150 2639    5.68
## 3           Urban formal   76 3302    2.30
## 4         Urban informal    8  508    1.57
```

It is clear from this table that the percentage of households containing a member over the age of 80 is far greater in Tribal Authorities Areas. It would be interesting to investigate the reasons for this (Is it simply because households are larger in Tribal Authority Areas? Are young families migrating to the city? Or is there some other reason?), but we will not do that here.

6.5 Example 2: Is there a relationship between the race of a household and the average age of the household?

In order to answer this question, the first thing we need to do is create a variable that assigns the average age of the household to every member of the household. The `group_by()` %>% `mutate()` command is necessary to do this. Type:

```
nids<-nids%>%
  mutate(age1 = ifelse(age>0, age, NA))%>%
  group_by(hhid)%>%
  mutate(avgage = mean(age1, na.rm=T))
```

Can you work out what each of the parts of the line of command is doing? We first create a variable `age1` that does not include no-response values or negative values. Then we `group_by(hhid)` telling R that whatever computation that is going to do will be grouped by `hhid`. The last `mutate` command is telling R to create a variable called `avgage` that gives each person in the sample a value equal to the average age of that individual's household (`hhid`).

It is instructive to think about what would happen if we specified the option **group_by(w1_hhgeo)** instead of **group_by(hhid)**? If that were the case, then the command would tell R to create a variable called `avgage`, which gave each person the value equal to the average age of all the people who live in the same geo-type. Try it and then check the new variable you create. Make sure you give the new variable you create a different name to `avgage`.

Moving on with our example, we are working at the household level and therefore we will need a household level race variable. Leaving aside the question of whether it is appropriate to assign a race to an entire household, for simplicity we will use the race of the resident head as the household race. Let's create a household level race variable:

```
nids<-nids%>%
  mutate(hrace = if_else(w1_r_b3==1, w1_best_race,NA_integer_)) %>%
  group_by(hhid)%>%
  mutate(hhrace = max(hrace, na.rm=TRUE))

table(nids$hhrace)
```

```
##
##  -Inf     1     2     3     4
##  2573 22494  4191   440  1472
```

What have we done here? In the first `mutate` we generated a variable `hrace` that takes on the race value of the head of the household, but it is only assigned this value for the household head (`w1_r_b3==1`). Every other member of the household is assigned an NA-missing value. The last `mutate()` code tells R to create a new variable `hhrace` and assign every member of the household the maximum value of `hrace` because we have grouped by `hhid`. Does this make sense? Surely, we don't want to take the maximum race in the household - what does maximum race even mean? The trick here is that the first `mutate` line of code assigns a race value to `hrace` for **only one** member of every household. Therefore, the maximum is just going to be that solitary value. You could equally well have used the minimum option and achieved the exact same result. We now have our household level race variable and can move on with the example. Before we move on, let's attach value/factor labels to our new `hhrace` variable:

```
nids<-nids%>%
  mutate(hhrace = factor(hhrace, levels = 1:4, labels=c("African","Coloured","Asian_Indian","White")))
```

Since we are working at the household level, we use hhrestrict to limit our cross tab to one observation per household. Type:

```
#nids_max_hhrace_age<-nids%>%
#  filter(hhrestrict==1)%>%
#  select(hhrace, avgage)

#CrossTable(nids_max_hhrace_age$avgage, nids_max_hhrace_age$hhrace, expected = FALSE, prop.r = FALSE, p
```

Now, we could look through this table and get some sort of indication of the household age distribution by race, but there are really too many entries in the table to draw any meaningful conclusions. In addition to this, we have the same challenge as we did previously where the African column has a far larger total which makes interpretation difficult if we don't use percentages.

The ideal situation would be to somehow split up the sample into racial categories and then get some sort of summary of the average age distribution in each racial group. While this may seem quite complicated, we can do it by combining tools that we have already learned. Let's try this:

```
nids%>%
  group_by(hhrace)%>%
  summarise(Mini = min(avgage, na.rm=TRUE),
            Mean = mean(avgage, na.rm=TRUE),
            Median = median(avgage, na.rm=TRUE),
            Max = max(avgage, na.rm=TRUE),
            N = n())
```

```
## # A tibble: 5 x 6
##   hhrace        Mini  Mean Median   Max     N
##   <fct>        <dbl> <dbl>  <dbl> <dbl> <int>
## 1 African       6.5   27.2   25    99   22494
## 2 Coloured     12     29.2   27    91    4191
## 3 Asian_Indian 15     32.1   28    83.5   440
## 4 White         8.33  39.7   35    83    1472
## 5 <NA>          9     25.1   23.8  72    2573
```

Note: Stata excludes those with missing age in in both mean and N whereas R includes them in N.

```
nids%>%
  group_by(hhrace)%>%
  summarise(Mini = min(avgage, na.rm=T),
            Mean = mean(avgage, na.rm=T),
            Median = median(avgage, na.rm=T),
            Max = max(avgage, na.rm=T),
            Std.dev = sd(avgage, na.rm=T),
            N = n())
```

```
## # A tibble: 5 x 7
##   hhrace        Mini  Mean Median   Max Std.dev     N
##   <fct>        <dbl> <dbl>  <dbl> <dbl>   <dbl> <int>
## 1 African       6.5   27.2   25    99     10.1  22494
## 2 Coloured     12     29.2   27    91     11.0   4191
## 3 Asian_Indian 15     32.1   28    83.5   12.4    440
## 4 White         8.33  39.7   35    83     17.0   1472
## 5 <NA>          9     25.1   23.8  72      7.75  2573
```

Now, let's return to the question at hand: is there a relationship between the average age of a household and the race of the household? Given the output above, we have a lot of information that can help us answer this question. One method would be to compare the median or mean of average household age in the different racial groups. For example, the median of average household age in White households is 35, while it is 25 in African households. Basically the median White household has an average age of around 35 - more than ten

years more than the median African household. This results are even stronger for the mean. These results suggest a strong relationship between the race of a household and the average age in the household.

Nevertheless, this process of acquiring these results regarding the relationship between household race and household average age has been rather ungainly and the precise relationship is still unclear. We don't know what is driving this relationship - are the elderly in White households simply living longer than the elderly in African households, or have African households been having more babies than White households in recent years? In order to address these questions and get to understand the determinants of the differing household age composition of the different racial groups, we will need to learn regression analysis. We will do this in chapter 7 and 8.

6.6 Example 3: Does the level of satisfaction differ in different race/gender groupings?

Here we introduce another useful command: `paste` (more like equivalent of STATA `egen`, `group()`) command to create three variable cross-tabulations. Suppose we are interested in examining the level of satisfaction of each person by race and gender.

We start by cleaning some of the variables

Using the paste command, we can type:

```
nids$race.gender<-paste(nids$race,nids$gender, sep = "-")
```

Let's make sure we understand what we just told R to do. The paste command tells R to concatenate and create a new variable, `race.gender`, that takes on the combinations African Male, African Female, for the race and gender combinations in our case. Let's look at the cross tabulation of our cleaned variables `race` and `gender` first:

```
CrossTable(nids$race, nids$gender, expected = FALSE, prop.r = FALSE, prop.c = FALSE, prop.t = FALSE,prop
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |-------------------------|
##
##
## Total Observations in Table:  28194
##
##
##                | nids$gender
##    nids$race |      Male |    Female | Row Total |
## -------------|-----------|-----------|-----------|
##      African |     10163 |     11994 |     22157 |
## -------------|-----------|-----------|-----------|
##     Coloured |      1948 |      2218 |      4166 |
## -------------|-----------|-----------|-----------|
##        Asian |       202 |       237 |       439 |
## -------------|-----------|-----------|-----------|
##        White |       683 |       749 |      1432 |
## -------------|-----------|-----------|-----------|
## Column Total |     12996 |     15198 |     28194 |
## -------------|-----------|-----------|-----------|
```

```
##
##
```

Or

```
mytable <- xtabs(~race+gender, data=nids)
ftable(mytable)
```

```
##          gender  Male Female
## race
## African         10163  11994
## Coloured         1948   2218
## Asian             202    237
## White             683    749
```

This simple crosstab tells us the frequency count of each combination of race and gender. For instance, we can see that there are 10163 African Males. Now, let's look at our newly created variable, `race.gender`:

```
nids%>%
  group_by(race.gender)%>%
  summarise(n=n())
```

```
## # A tibble: 10 x 2
##    race.gender         n
##    <chr>           <int>
##  1 African-Female  11994
##  2 African-Male    10163
##  3 Asian-Female      237
##  4 Asian-Male        202
##  5 Coloured-Female  2218
##  6 Coloured-Male    1948
##  7 NA-Female        1329
##  8 NA-Male          1647
##  9 White-Female      749
## 10 White-Male        683
```

Now we can examine the cross tab of level of satisfaction (`w1_a_m5`) and our new variable `race.gender`. This means we will be now examining the difference in the perceived satisfaction of respondents (adults only) by race and gender.

```
CrossTable(nids$w1_a_m5, nids$race.gender, expected = FALSE, prop.r = FALSE, prop.c = FALSE, prop.t = FA
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |-------------------------|
##
##
## Total Observations in Table:  15630
##
##
##              | nids$race.gender
## nids$w1_a_m5 |  African-Female |    African-Male |    Asian-Female |      Asian-Male | Coloured-Femal
## -------------|-----------------|-----------------|-----------------|-----------------|--------------
##           -9 |             786 |             487 |               4 |               2 |              9
## -------------|-----------------|-----------------|-----------------|-----------------|--------------
```

```
##           -8 |               80 |               63 |                0 |                0 |
## -------------|------------------|------------------|------------------|------------------|
##           -5 |                2 |                0 |                0 |                0 |
## -------------|------------------|------------------|------------------|------------------|
##           -3 |               87 |               62 |                0 |                0 |
## -------------|------------------|------------------|------------------|------------------|
##            1 |              568 |              370 |                3 |                4 |
## -------------|------------------|------------------|------------------|------------------|
##            2 |              407 |              247 |                1 |                1 |
## -------------|------------------|------------------|------------------|------------------|
##            3 |              736 |              432 |                2 |                3 |
## -------------|------------------|------------------|------------------|------------------|
##            4 |             1041 |              622 |                7 |                4 |
## -------------|------------------|------------------|------------------|------------------|
##            5 |             1220 |              781 |               28 |               14 |
## -------------|------------------|------------------|------------------|------------------|
##            6 |              805 |              599 |               13 |               12 |
## -------------|------------------|------------------|------------------|------------------|
##            7 |              647 |              473 |               30 |               18 |
## -------------|------------------|------------------|------------------|------------------|
##            8 |              387 |              315 |               33 |               26 |
## -------------|------------------|------------------|------------------|------------------|
##            9 |              142 |              110 |                1 |                4 |
## -------------|------------------|------------------|------------------|------------------|
##           10 |              441 |              337 |               15 |                7 |
## -------------|------------------|------------------|------------------|------------------|
## Column Total |             7349 |             4898 |              137 |               95 |
## -------------|------------------|------------------|------------------|------------------|
##
##
```

```
# Other options - (table(A,B))
# mycrosstab<-table(nids$w1_a_m5, nids$race.gender)
# mycrosstab #print table
#
# margin.table(mycrosstab, 1) # A frequencies (summed over B)
# margin.table(mycrosstab, 2) # B frequencies (summed over A)
#
# round(prop.table(mycrosstab),3) # cell percentages
# round(prop.table(mycrosstab, 1),3) # row percentages
# round(prop.table(mycrosstab, 2),3) # column percentages
```

There are many cells in this cross tab, so perhaps it is best if we try to `w1_a_m5>0` to help us with our interpretation:

```
nids_temp<-nids %>%
  filter(w1_a_m5>0 & !is.na(race))
CrossTable(nids_temp$w1_a_m5, nids_temp$race.gender, expected = FALSE,prop.r = FALSE, prop.c = TRUE, pr
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Col Total |
```

```
## |-------------------------|
##
##
## Total Observations in Table:  13781
##
##
##                     | nids_temp$race.gender
## nids_temp$w1_a_m5 |  African-Female |    African-Male |    Asian-Female |      Asian-Male | Coloured
## ------------------|-----------------|-----------------|-----------------|-----------------|---------
##                 1 |             568 |             370 |               3 |               4 |
##                   |           0.089 |           0.086 |           0.023 |           0.043 |
## ------------------|-----------------|-----------------|-----------------|-----------------|---------
##                 2 |             407 |             247 |               1 |               1 |
##                   |           0.064 |           0.058 |           0.008 |           0.011 |
## ------------------|-----------------|-----------------|-----------------|-----------------|---------
##                 3 |             736 |             432 |               2 |               3 |
##                   |           0.115 |           0.101 |           0.015 |           0.032 |
## ------------------|-----------------|-----------------|-----------------|-----------------|---------
##                 4 |            1041 |             622 |               7 |               4 |
##                   |           0.163 |           0.145 |           0.053 |           0.043 |
## ------------------|-----------------|-----------------|-----------------|-----------------|---------
##                 5 |            1220 |             781 |              28 |              14 |
##                   |           0.191 |           0.182 |           0.211 |           0.151 |
## ------------------|-----------------|-----------------|-----------------|-----------------|---------
##                 6 |             805 |             599 |              13 |              12 |
##                   |           0.126 |           0.140 |           0.098 |           0.129 |
## ------------------|-----------------|-----------------|-----------------|-----------------|---------
##                 7 |             647 |             473 |              30 |              18 |
##                   |           0.101 |           0.110 |           0.226 |           0.194 |
## ------------------|-----------------|-----------------|-----------------|-----------------|---------
##                 8 |             387 |             315 |              33 |              26 |
##                   |           0.061 |           0.073 |           0.248 |           0.280 |
## ------------------|-----------------|-----------------|-----------------|-----------------|---------
##                 9 |             142 |             110 |               1 |               4 |
##                   |           0.022 |           0.026 |           0.008 |           0.043 |
## ------------------|-----------------|-----------------|-----------------|-----------------|---------
##                10 |             441 |             337 |              15 |               7 |
##                   |           0.069 |           0.079 |           0.113 |           0.075 |
## ------------------|-----------------|-----------------|-----------------|-----------------|---------
##      Column Total |            6394 |            4286 |             133 |              93 |
##                   |           0.464 |           0.311 |           0.010 |           0.007 |
## ------------------|-----------------|-----------------|-----------------|-----------------|---------
##
##
```

This sets the non-response values of the satisfaction variable to missing and then generates the above table:

From this table we can see that there is not as much difference between the sexes within each race. However it appears that on average African adults have far lower satisfaction levels than the other racial groups, with White adults having the highest satisfaction levels.

6.7 Example 4: Comparing Household Monthly Income from the labour market to individual monthly takehome pay

In the NIDS data set, the total monthly income of a household from the labour market was computed using a number of different rules. According to the metadata: "This variable is calculated by aggregating income from formal employment, casual labour, self-employment and income from helping friends with their business". An interesting question then is: how different is the computed household labour-market monthly income (`w1_hhwag`) from the sum of the individual take-home pay (from primary employment) of every member in the household? Let's examine this using the `dplyr` commands. To compute a variable that is the sum of all individual incomes (`w1_fwag`) for each member of a household, we can type:

```
nids<-nids%>%
  group_by(hhid)%>%
  mutate(tot_fwag=sum(w1_fwag, na.rm=TRUE))
```

Now, to test to see how different these two measures of income are all one has to do is list `hhid tot_fwag` and `w1_hhwage` together.

```
#subset to match extract from the Stata course
head(subset(nids, subset=hhid>=101066, select=c(hhid,tot_fwag, w1_hhwage)), n=20L)
```

```
## # A tibble: 20 x 3
## # Groups:   hhid [6]
##      hhid tot_fwag w1_hhwage
##     <int>    <dbl>     <dbl>
##  1 101066     2500     2500
##  2 101066     2500     2500
##  3 101066     2500     2500
##  4 101066     2500     2500
##  5 101066     2500     2500
##  6 101066     2500     2500
##  7 101066     2500     2500
##  8 101066     2500     2500
##  9 101067     2400     2525
## 10 101067     2400     2525
## 11 101067     2400     2525
## 12 101067     2400     2525
## 13 101067     2400     2525
## 14 101069        0      180
## 15 101070        0      585.
## 16 101070        0      585.
## 17 101072        0       NA
## 18 101072        0       NA
## 19 101072        0       NA
## 20 101073        0       NA
```

In many cases, there are significant differences between the sum of a household's individual take-home pay and household monthly income from the labour-market. In addition, there are many instances where entire households have not given a single value for individual take-home pay but there is a value for the household variable. There are two possible reasons for these differences: firstly, the individual level variable only gives income from the a person's primary source of employment while the household variable uses aggregate income from various sources; secondly, in some cases income for the household income has been imputed based on various household characteristics.

One could also use the **dplyr** commands to compute other very useful variables. For example, one could compute the total number of children in a household for each household, or total number of births in the last

year by province.

There are many useful ways to use **dplyr** `group_by`, and hopefully, these practice questions can assist you in gaining speed.

6. Create a variable that is equal to one for exactly one member in every household.

Question 6 Answer

7. What is the total household income for the Kwa-Zulu Natal?

Question 7 Answer

6.8 Chi-Squared: testing for independence

By now, we have examined tables of variables. Perhaps you have noticed that in a few examples as one variable increased or decreased, the other variable in the cross tab decreased or increased. While the naked eye is fairly good at noticing these relationships, it is unclear how accurate the relationships are until we examine them statistically. It is a good idea before any further analysis of the variables occurs to test whether the variables in the crosstab are independent or not. By independent, we mean whether as X moves one way or another, Y's movements are completely random with respect to X. For instance, in children height is clearly not independent of age - older children are generally taller. In contrast, amongst 30 - 40 year olds, height is likely to be independent of age. As we shall see in the next chapter, testing for statistical independence is useful test to learn at this point. Note that this test for independence will test for any kind of functional relationship. However, in the next chapter, we will be working only with linear relationships.

Let's try this simple example with the variables, `w1_a_h35` - "Respondent has driver's license" and `w1_a_g4` -"Ownership of a motor vehicle (private) in running condition". Basically, we are going to ask whether there is a relationship between these two variables - are people who have a driver's license more likely to own a vehicle?

```
nids$dl<-nids$w1_a_h35
nids$dl[nids$w1_a_h35<0]<-NA

nids$mv<-nids$w1_a_g4
nids$mv[nids$w1_a_g4<0]<-NA

nids$mv<-factor(nids$mv, levels = 1:2, labels = c("Yes","No"))
nids$dl<-factor(nids$dl, levels = 1:2, labels = c("Yes","No"))

CrossTable(nids$dl, nids$mv, digits=3, max.width = 5, expected=FALSE, prop.r=TRUE, prop.c=TRUE,prop.t=FA
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |-------------------------|
##
##
## Total Observations in Table:  15522
##
##
```

```
##              | nids$mv
##      nids$dl |       Yes |        No | Row Total |
## -------------|-----------|-----------|-----------|
##          Yes |      1204 |      1082 |      2286 |
##              |  4657.895 |   474.850 |           |
##              |     0.527 |     0.473 |     0.147 |
##              |     0.838 |     0.077 |           |
## -------------|-----------|-----------|-----------|
##           No |       232 |     13004 |     13236 |
##              |   804.469 |    82.012 |           |
##              |     0.018 |     0.982 |     0.853 |
##              |     0.162 |     0.923 |           |
## -------------|-----------|-----------|-----------|
## Column Total |      1436 |     14086 |     15522 |
##              |     0.093 |     0.907 |           |
## -------------|-----------|-----------|-----------|
##
##
## Statistics for All Table Factors
##
##
## Pearson's Chi-squared test
## ------------------------------------------------------------
## Chi^2 =  6019.226     d.f. =  1     p =  0
##
## Pearson's Chi-squared test with Yates' continuity correction
## ------------------------------------------------------------
## Chi^2 =  6013.163     d.f. =  1     p =  0
##
##
```

First of all, notice that we have specified both the `row` (`prop.r=TRUE`) and `column` (`prop.c=TRUE`) options and therefore R gives percentages adding across rows (3rd entry in every cell) and percentages adding up down columns (4th entry in every cell). Secondly, since we also specified the `chisq = TRUE` option, we see that at the bottom of the table, R reports a chi2 value as well as a p-value. The p=0.000 tells us that the two variables are related (i.e. that the probability that they are not related is 0.000). Logically, this makes sense. We expect that people who own a private motor to have a license (i.e. we expect that a person's license status will be related to their car ownership status). Particularly, from the table we can see that 83.8% of those who own a car have a driver's license.

6.9 Cross-tabs and hypothesis testing (Chi^2) examples: The comparative level of happiness variable

6.9.1 Example 1: Comparative happiness and employment status

Now that we have learned to perform basic and more advanced cross-tabulations, we might find it beneficial to apply some of this new knowledge to one of the more interesting variables in the data set: the "level of happiness in comparison to 10 years ago" variable, `w1_a_m6`. The 2008 survey asked adults if they were "happier, the same or less happy with life than they were 10 years ago", with these responses being coded as 1, 2, and 3, respectively. Before tabulating the `w1_a_m6` variable with any of the other variables in our set, we might want to tabulate it on its own in order to get a better feel for it.

```
table(nids$w1_a_m6)
```

```
##
##   -8   -3    1    2    3    9
##    9  110 6358 4984 3806  368
```

What does the value of 9 represent? This is a typical example of why it is VERY important to look closely at the variables you are working with. Even though throughout almost all of the NIDS dataset, a -9 is used to represent a "don't know" response, here a 9 is used. This is likely to be due to an error made at some point in the data capturing or data cleaning process as the other non-responses have taken on negative values. Also note that there are not a huge number of non-respondents so that these categories have little influence on the total percentage. This may not always be the case. For example if a large number of people had refused to respond this would have affected the distribution of percentages across all categories. Also, we might be more worried about people who "refused" than those who said "don't know" as they might all have refused since they are all unhappy, but don't want to say so. But luckily there are only 9 refused and only about 3% non-responses in total.

Although this simple univariate tabulation is certainly educational, we can learn more by analyzing this variable against another, i.e., performing a bivariate analysis. Say, for instance, that we wanted to know how one's happiness compared to ten years ago varied with current employment status. In order to find this out, we would cross-tabulate `w1_a_m6` with `w1_a_e1` (coded as 1 for "currently employed" and 2 for "not currently employed"). Here we might wish to ignore the non-responses using qualifiers to obtain more readily interpretable results. We may start by generating new variables that cleans the types of non-responses.

```
nids$lh<-nids$w1_a_m6
nids$lh[nids$w1_a_m6<0 | nids$w1_a_m6>3]<-NA

nids$es<-nids$w1_a_e1
nids$es[nids$w1_a_e1<0]<-NA

nids$lh<-factor(nids$lh, levels = 1:3, labels = c("Happier","The same", "Less happy"))
nids$es<-factor(nids$es, levels = 1:2, labels = c("Yes","No"))

CrossTable(nids$lh, nids$es, digits=4, max.width = 5, expected=FALSE, prop.r=TRUE, prop.c=TRUE, prop.t='
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  15144
##
##
##              | nids$es
##      nids$lh |       Yes |        No | Row Total |
## -------------|-----------|-----------|-----------|
##      Happier |      1985 |      4371 |      6356 |
##              |  110.1326 |   36.1057 |           |
```

```
##              |    0.3123 |    0.6877 |    0.4197 |
##              |    0.5309 |    0.3833 |           |
##              |    0.1311 |    0.2886 |           |
## -------------|-----------|-----------|-----------|
##     The same |      1084 |      3898 |      4982 |
##              |   17.3386 |    5.6843 |           |
##              |    0.2176 |    0.7824 |    0.3290 |
##              |    0.2899 |    0.3418 |           |
##              |    0.0716 |    0.2574 |           |
## -------------|-----------|-----------|-----------|
##   Less happy |       670 |      3136 |      3806 |
##              |   77.3997 |   25.3746 |           |
##              |    0.1760 |    0.8240 |    0.2513 |
##              |    0.1792 |    0.2750 |           |
##              |    0.0442 |    0.2071 |           |
## -------------|-----------|-----------|-----------|
## Column Total |      3739 |     11405 |     15144 |
##              |    0.2469 |    0.7531 |           |
## -------------|-----------|-----------|-----------|
##
##
## Statistics for All Table Factors
##
##
## Pearson's Chi-squared test
## ------------------------------------------------------------
## Chi^2 =  272.0355     d.f. =  2     p =  8.476959e-60
##
##
##
```

Now we can say that of those who are employed 53% are happier than 10 years ago while only 38% of those who are unemployed are happier than 10 years ago. A less useful interpretation in this case would be to look across the first row and say that of those who are happier than they were ten years ago, 31% are employed and 69% are not employed. Does this make sense? Make sure you understand what is going on here. The clue for these seemingly strange results is the large difference between the number of unemployed (11 405) and employed (3 739) individuals.

6.9.2 Example 2: Comparative happiness by province

Let's try another example. For instance, a cross-tabulation of `w1_hhprov` on `w1_a_m6` (ignoring nonresponses) generates a table crammed with raw numbers. Using the `prop.r=TRUE` option with this same cross-tabulation yields a more decipherable result.

```
nids<-nids%>%
  mutate(w1_hhprov = factor(w1_hhprov, levels=1:9, labels = c("Western Cape","Eastern Cape","Northern C

CrossTable(nids$w1_hhprov, nids$lh, digits=4, max.width = 6, expected=FALSE, prop.r=TRUE, prop.c=FALSE,
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
```

```
## |           N / Row Total |
## |-------------------------|
##
##
## Total Observations in Table:  15148
##
##
##                | nids$lh
## nids$w1_hhprov |    Happier |   The same | Less happy |  Row Total |
## ---------------|------------|------------|------------|------------|
##   Western Cape |       1007 |        610 |        300 |       1917 |
##                |     0.5253 |     0.3182 |     0.1565 |     0.1266 |
## ---------------|------------|------------|------------|------------|
##   Eastern Cape |        729 |        623 |        603 |       1955 |
##                |     0.3729 |     0.3187 |     0.3084 |     0.1291 |
## ---------------|------------|------------|------------|------------|
##  Northern Cape |        696 |        207 |        154 |       1057 |
##                |     0.6585 |     0.1958 |     0.1457 |     0.0698 |
## ---------------|------------|------------|------------|------------|
##     Free State |        427 |        284 |        219 |        930 |
##                |     0.4591 |     0.3054 |     0.2355 |     0.0614 |
## ---------------|------------|------------|------------|------------|
##  KwaZulu-Natal |       1246 |       1311 |       1362 |       3919 |
##                |     0.3179 |     0.3345 |     0.3475 |     0.2587 |
## ---------------|------------|------------|------------|------------|
##     North West |        496 |        464 |        354 |       1314 |
##                |     0.3775 |     0.3531 |     0.2694 |     0.0867 |
## ---------------|------------|------------|------------|------------|
##        Gauteng |        801 |        466 |        286 |       1553 |
##                |     0.5158 |     0.3001 |     0.1842 |     0.1025 |
## ---------------|------------|------------|------------|------------|
##     Mpumalanga |        496 |        295 |        234 |       1025 |
##                |     0.4839 |     0.2878 |     0.2283 |     0.0677 |
## ---------------|------------|------------|------------|------------|
##        Limpopo |        460 |        724 |        294 |       1478 |
##                |     0.3112 |     0.4899 |     0.1989 |     0.0976 |
## ---------------|------------|------------|------------|------------|
##   Column Total |       6358 |       4984 |       3806 |      15148 |
## ---------------|------------|------------|------------|------------|
##
##
```

Without the benefit of row percentages, we would see that 1362 people in KwaZulu-Natal feel less happy than 10 years ago, for example. The row percentage puts this number in the context of the total number of respondents in KwaZulu-Natal, thus telling us that these 1362 people comprise about 34.75% of the respondents residing in KwaZulu-Natal. What percentage of the total sample is this?

```
CrossTable(nids$w1_hhprov, nids$lh)    #Will give you all stats
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
```

```
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  15148
##
##
##                | nids$lh
## nids$w1_hhprov |    Happier |   The same | Less happy |  Row Total |
## ---------------|------------|------------|------------|------------|
##   Western Cape |       1007 |        610 |        300 |       1917 |
##                |     50.907 |      0.681 |     68.510 |            |
##                |      0.525 |      0.318 |      0.156 |      0.127 |
##                |      0.158 |      0.122 |      0.079 |            |
##                |      0.066 |      0.040 |      0.020 |            |
## ---------------|------------|------------|------------|------------|
##   Eastern Cape |        729 |        623 |        603 |       1955 |
##                |     10.217 |      0.637 |     25.445 |            |
##                |      0.373 |      0.319 |      0.308 |      0.129 |
##                |      0.115 |      0.125 |      0.158 |            |
##                |      0.048 |      0.041 |      0.040 |            |
## ---------------|------------|------------|------------|------------|
##  Northern Cape |        696 |        207 |        154 |       1057 |
##                |    143.538 |     56.984 |     46.876 |            |
##                |      0.658 |      0.196 |      0.146 |      0.070 |
##                |      0.109 |      0.042 |      0.040 |            |
##                |      0.046 |      0.014 |      0.010 |            |
## ---------------|------------|------------|------------|------------|
##     Free State |        427 |        284 |        219 |        930 |
##                |      3.442 |      1.580 |      0.921 |            |
##                |      0.459 |      0.305 |      0.235 |      0.061 |
##                |      0.067 |      0.057 |      0.058 |            |
##                |      0.028 |      0.019 |      0.014 |            |
## ---------------|------------|------------|------------|------------|
##  KwaZulu-Natal |       1246 |       1311 |       1362 |       3919 |
##                |     96.738 |      0.361 |    144.599 |            |
##                |      0.318 |      0.335 |      0.348 |      0.259 |
##                |      0.196 |      0.263 |      0.358 |            |
##                |      0.082 |      0.087 |      0.090 |            |
## ---------------|------------|------------|------------|------------|
##     North West |        496 |        464 |        354 |       1314 |
##                |      5.589 |      2.320 |      1.723 |            |
##                |      0.377 |      0.353 |      0.269 |      0.087 |
##                |      0.078 |      0.093 |      0.093 |            |
##                |      0.033 |      0.031 |      0.023 |            |
## ---------------|------------|------------|------------|------------|
##        Gauteng |        801 |        466 |        286 |       1553 |
##                |     34.135 |      3.958 |     27.825 |            |
##                |      0.516 |      0.300 |      0.184 |      0.103 |
##                |      0.126 |      0.093 |      0.075 |            |
##                |      0.053 |      0.031 |      0.019 |            |
## ---------------|------------|------------|------------|------------|
```

```
##     Mpumalanga |        496 |        295 |        234 |       1025 |
##                |     10.058 |      5.292 |      2.151 |            |
##                |      0.484 |      0.288 |      0.228 |      0.068 |
##                |      0.078 |      0.059 |      0.061 |            |
##                |      0.033 |      0.019 |      0.015 |            |
## ---------------|------------|------------|------------|------------|
##        Limpopo |        460 |        724 |        294 |       1478 |
##                |     41.450 |    116.196 |     16.113 |            |
##                |      0.311 |      0.490 |      0.199 |      0.098 |
##                |      0.072 |      0.145 |      0.077 |            |
##                |      0.030 |      0.048 |      0.019 |            |
## ---------------|------------|------------|------------|------------|
##   Column Total |       6358 |       4984 |       3806 |      15148 |
##                |      0.420 |      0.329 |      0.251 |            |
## ---------------|------------|------------|------------|------------|
##
##
```

Thus we see a table revealing that the 1362 people living in KwaZulu-Natal and feeling less happy than 10 years ago account for 9% of all individuals responding to this item of the survey.

6.9.3 Example 3: Comparative happiness and household income

Thus far, we have only been tabulating the `w1_a_m6` variable with other categorical variables. What if we wanted to find out how people's comparative happiness varied with a continuous variable like total household income, for example? Notice that in this example, even though one of the variables we are using is a household level variable, we don't use `hhrestrict`. This is because we are asking: is there a relationship between an individual's happiness in comparison to ten years ago and that individual's household income? It is an individual level question.

```
nids%>%
  group_by(lh)%>%
  summarise(mean = mean(w1_hhincome, na.rm=T), sd = sd(w1_hhincome, na.rm=T), n=n()) %>%
  na.omit()
```

```
## # A tibble: 3 x 4
##   lh          mean    sd     n
##   <fct>      <dbl> <dbl> <int>
## 1 Happier    6074. 9495.  6358
## 2 The same   3942. 6840.  4984
## 3 Less happy 3447. 6301.  3806
```

Eliminating non-responses, we thus find that on average, adults in this survey who feel happier than ten years ago belong to a household with a lower average income than the respondents who felt the same or less happy. This analysis could also be done by province:

```
happiness<-nids%>%
  rename(level_happiness=lh) %>%
  group_by(level_happiness, province)%>%
  summarise(mean = mean(w1_hhincome, na.rm=T), sd = sd(w1_hhincome, na.rm=T), n=n()) %>%
  na.omit()
```

```
knitr::kable(
  head(happiness, 27), booktabs = TRUE,
  caption = 'Summary level of happiness by province and household income'
)
```

Table 6.1: Summary level of happiness by province and household income

level_happiness	province	mean	sd	n
Happier	Western Cape	8015.570	10469.686	1007
Happier	Eastern Cape	3972.763	6020.442	729
Happier	Northern Cape	5302.639	6034.517	696
Happier	Free State	4457.535	6128.849	427
Happier	KwaZulu-Natal	4204.751	6638.604	1246
Happier	North West	5667.018	10555.289	496
Happier	Gauteng	10785.584	14600.511	801
Happier	Mpumalanga	6607.049	9278.622	496
Happier	Limpopo	4550.641	9334.493	460
The same	Western Cape	6598.891	8651.250	610
The same	Eastern Cape	2680.517	4296.637	623
The same	Northern Cape	5183.479	7247.052	207
The same	Free State	3761.236	5190.223	284
The same	KwaZulu-Natal	2906.890	4177.492	1311
The same	North West	3470.998	6215.211	464
The same	Gauteng	5911.095	10162.332	466
The same	Mpumalanga	4976.882	10330.690	295
The same	Limpopo	2989.421	6172.425	724
Less happy	Western Cape	6028.609	7747.007	300
Less happy	Eastern Cape	1858.170	2178.584	603
Less happy	Northern Cape	6387.000	9644.741	154
Less happy	Free State	3251.774	7325.622	219
Less happy	KwaZulu-Natal	2903.290	4852.305	1362
Less happy	North West	3674.960	6183.364	354
Less happy	Gauteng	4871.022	9635.023	286
Less happy	Mpumalanga	4377.594	8708.411	234
Less happy	Limpopo	2790.755	5379.583	294

We see that difference in average household income according to relative happiness is more pronounced in Gauteng than in the country as a whole. The individuals who reported that they are happier than ten years ago have an average monthly income of over R10 000!

6.9.4 Example 4: Comparative happiness and geo-type

Finally, although our cross-tabulations might suggest that there is a relationship between two variables, the results from the survey might not be statistically significant (note that this section is optional for those unfamiliar with hypothesis testing). Say for instance that we run a bivariate analysis on the two variables: `w1_a_m6` (comparative happiness) and `w1_hhgeo` (geo-type).

```
CrossTable(nids$lh, nids$w1_hhgeo, digits=4, max.width = 5, expected=FALSE, prop.c=TRUE, prop.r=FALSE, p
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Col Total |
## |-------------------------|
##
##
## Total Observations in Table:  15148
##
##
##              | nids$w1_hhgeo
##      nids$lh |           Rural formal | Tribal authority areas |           Urban formal |         Ur
## -------------|------------------------|------------------------|------------------------|-----------
##      Happier |                    683 |                   1974 |                   3283 |
##              |                 0.4290 |                 0.3207 |                 0.5118 |
## -------------|------------------------|------------------------|------------------------|-----------
##     The same |                    553 |                   2350 |                   1790 |
##              |                 0.3474 |                 0.3818 |                 0.2790 |
## -------------|------------------------|------------------------|------------------------|-----------
##   Less happy |                    356 |                   1831 |                   1342 |
##              |                 0.2236 |                 0.2975 |                 0.2092 |
## -------------|------------------------|------------------------|------------------------|-----------
## Column Total |                   1592 |                   6155 |                   6415 |
##              |                 0.1051 |                 0.4063 |                 0.4235 |
## -------------|------------------------|------------------------|------------------------|-----------
##
##
## Statistics for All Table Factors
##
##
## Pearson's Chi-squared test
## ------------------------------------------------------------
## Chi^2 =  485.3229     d.f. =  6     p =  1.219035e-101
##
##
##
```

We can see in our table that effectively 51% of Urban Formal respondents felt happier than ten years ago while about 21% of Urban formal adult respondents felt less happy than 10 years ago. While the majority

of respondents in Urban Formal Areas seem happier than 10 years, this isn't the case in tribal authority areas. Here 32% of respondents felt happier than 10 years ago while 68% were the same or less happy. The question then arises: can we conclude that the difference in feelings of comparative happiness according to location that appears in our sample overstates the difference that exists in the population, i.e. are our results statistically significant? We would set up the null (H0) and alternative (H1) hypotheses like this:

H0: No significant relationship exists between `w1_a_m6` and `w1_hhgeo`.

H1: A significant relationship exists between `w1_a_m6` and `w1_hhgeo`.

R can test this for us while simultaneously performing our cross-tabulation requests. As we have seen above, this test is known as the chi square test and it determines whether or not a relationship exists between two categorical variables.

The chi2 option gives this line of output underneath the cross-tabulation results table. The most important element of the line is "p" The "p" value gives the probability that the null hypothesis (H0) is true and the two variables are statistically independent. The p-value is significant and we can thus conclude that comparative happiness did in fact differ according to metropolitan location. A possible interpretation of this relationship is that those in more rural areas have been disappointed by the lack of improvement in service delivery as promised by the government in 1994.

6.10 Worked Example: Assessing the impact of per capita income on BMI

In this worked example we aim to assess how BMI varies with income. Are rich people more likely to be obese than people with lower incomes? They have more money to spend on food, but can also afford a healthy diet and to buy gym memberships. By now we are reasonably well acquainted with the BMI variable, but we need to find a variable which gives us an indication of how rich (well-off) the person is.

With such a wide variety of income variables, we need to decide which one will be the most appropriate to investigate the relationship between BMI and income. Should we use the amount of income that an individual earns or the total income of the household?

Individual income does not take into consideration that families generally pool/share their income. Individual income would assign all people who do not work, receive grants or transfers, zero income. This is not what we want; we need a variable which reflects the level of resources available to the individual. Total household income (`w1_hhincome`) would give an indication of the wealth of the household, but we need to take into consideration the number of household members who will share this income.

We could just divide the total household income by the household size variable to create a per capita household income variable. But it may be more accurate to weight children less than adults since they generally consume a smaller proportion of household resources (and hence household income) than adults. For example, consider two households each with a total monthly income of R2 000, but one household consists of four adults and the other of two adults and two young children. If we assume for simplicity that all income is spent on food, we would expect that the adults in the household consisting of two adults and two children will have access to more food than the adults in the four adult household.

Let's assume that children under 15 use half the household resources used by those 15 years or older. We therefore need to calculate the number of household members under 15 and the number of members over 15 in the household. To do this, we create a variable that assigns to every member of the household, a value equal to the number of children under the age of 15 in the household.

```
nids$child.dummy<-0
nids$child.dummy[nids$w1_quest_typ == 2]<-1
```

The code above creates a variable that assigns to every member of the household, the number of individuals over the age of 15. However, it is worth noticing that we used the questionnaire type answered by the

individual as an indicator for their age as individuals under 15 were meant to answer the child questionnaire, while those over 15 were meant to have an adult or proxy questionnaire. However, there are several borderline cases (around 15 years old) which answered the wrong questionnaire. This isn't too worrying for us in this instance though.

Let's generate a variable that contains the number of adults in a household.

```
nids1<-nids%>%
  select(hhid, pid, w1_hhsizer, w1_hhincome, child.dummy, bmi, gender,bmi_valid)%>%
  mutate(hh.children=sum(child.dummy, na.rm=TRUE))%>%
  mutate(hh.adults = w1_hhsizer- hh.children)
```

Check to see that the variables give us the number of children and adults in the household.

```
head(nids1)
```

```
## # A tibble: 6 x 10
## # Groups:   hhid [2]
##     hhid    pid w1_hhsizer w1_hhincome child.dummy   bmi gender bmi_valid
##    <int>  <int>      <int>       <dbl>       <dbl> <dbl> <fct>      <dbl>
## 1 101012 314585          1       1045.           0  30.9 Female         1
## 2 101013 314544          5        588.           0  19.3 Male           1
## 3 101013 314550          5        588.           0  23.1 Female         1
## 4 101013 406295          5        588.           1  NA   Female        NA
## 5 101013 406296          5        588.           1  NA   Male          NA
## 6 101013 406297          5        588.           1  NA   Female        NA
## # ... with 2 more variables: hh.children <dbl>, hh.adults <dbl>
```

Now we can create our adult equivalent household size variable, where children count 0.5 of the amount that adults do. While this is crude, it is sufficient for our purposes.

```
nids1<-nids1%>%
  mutate(hhsize.adultequiv=hh.adults + 0.5*hh.children)
```

Finally create the household per capita adult equivalent income variable. While this sounds complicated, this is just a variable that gives us per capita income in every household, adjusting for the number of children in the household. It is a household level per capita income variable.

```
nids1<-nids1%>%
  mutate(hh.pcinc=w1_hhincome/hhsize.adultequiv)
```

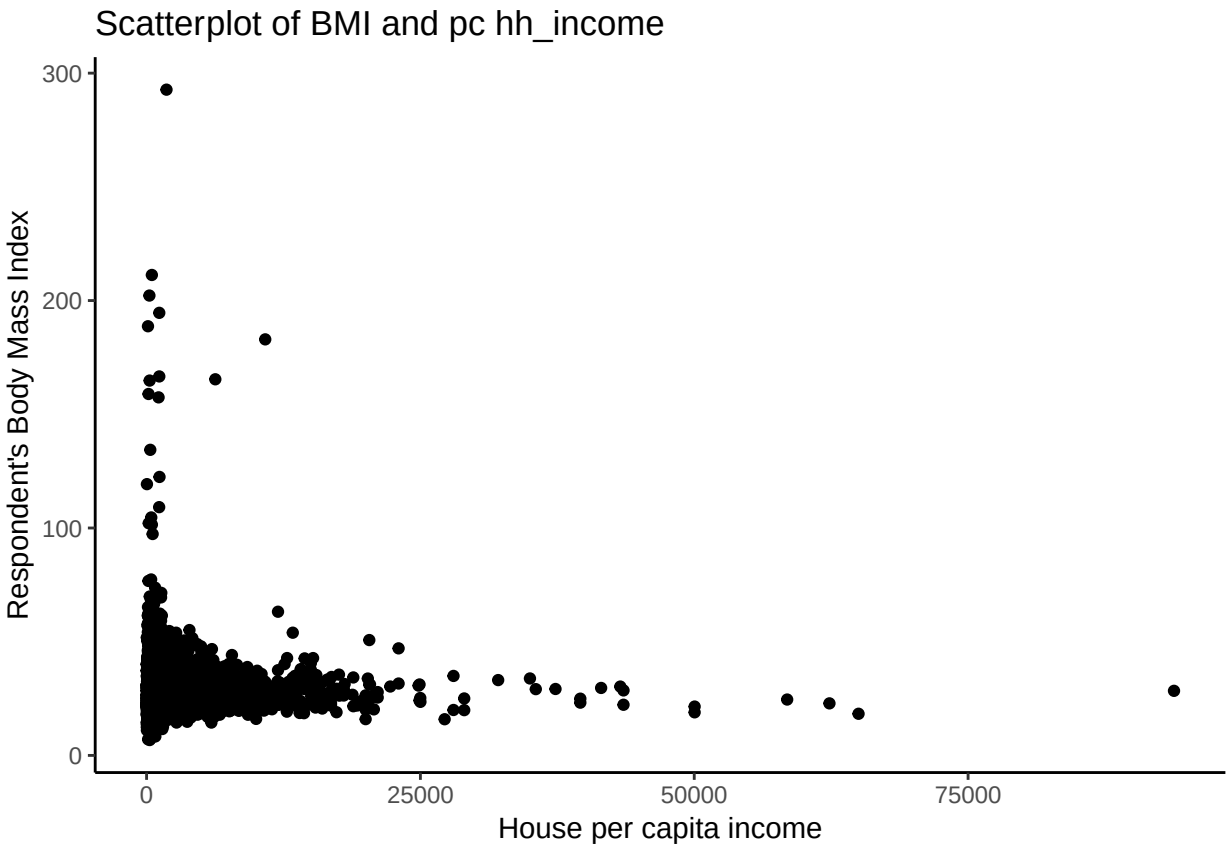
Notice that the `hh.pcinc` column is simply equal to the `w1_hhincome` column divided by the `hhsize.adultequiv` column. We now have an income variable which reflects the household income available to each individual and can proceed to assess the impact of financial wellbeing on BMI.

Recall that when we are dealing with BMI, we want to restrict our sample to adults over 20, but since we have calculated our BMI variable in such a way that it only has values for people over the age of 20, we will often not need to restrict our sample explicitly. We should however always be careful that we are indeed only working with a sample that includes individuals over the age of 20.

Since BMI and income are both continuous variables it is easiest to summarize the relationship between them using a scatter graph.

```
ggplot(nids1, aes(hh.pcinc, bmi)) +
  geom_point() +
  labs(x="House per capita income",y="Respondent's Body Mass Index", title="Scatterplot of BMI and pc hh
  theme_classic()
```

```
## Warning: Removed 19885 rows containing missing values (geom_point).
```

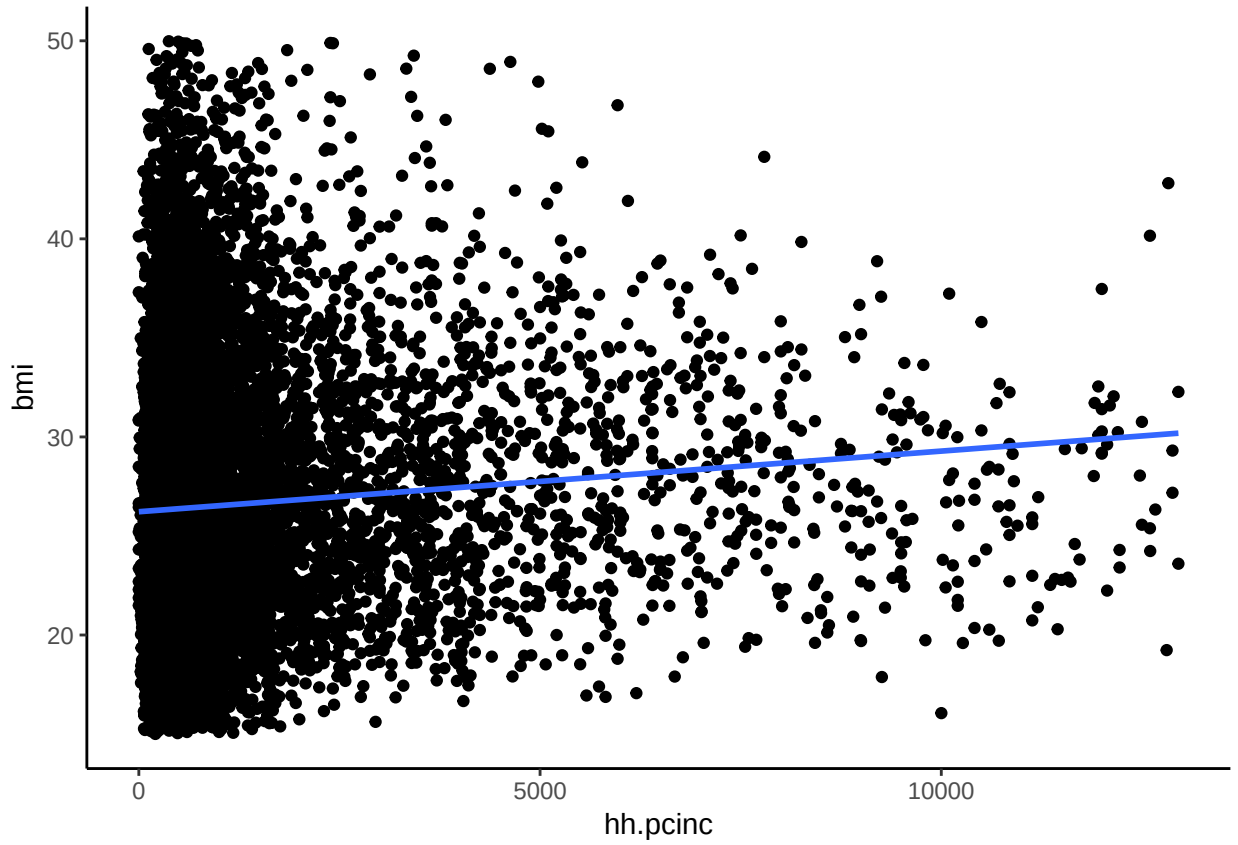


It is clear from the scatterplot that there are very large outliers on both dimensions (income and BMI). While the majority of the sample has BMI values below 100 and `hh.pcinc` values under 30000, there are a few extremely large BMI and income values. Do you think it is possible for someone to have a BMI of over 150? What height and weight combination would such an individual need to have? Check to see how tall and heavy the individuals with BMI greater than 150 in the sample are. While, these may be possible combinations, it seems more likely that at least some of them are errors.

In order to make the graph more useful, we should restrict the range of both BMI and income. Here, we will use the range [15, 50] for BMI as we did previously and [0, 14 000] for income, since 99% of the sample falls below this income level. Check this.

Redraw the scatterplot restricting the range of BMI and income as suggested and adding a trend line.

```
ggplot(nids1 %>% filter(bmi_valid == 1 & hh.pcinc < 13000), aes(hh.pcinc, bmi)) +
  geom_point() +
  stat_smooth(method = "lm", size=1, formula = y ~ x, se = FALSE) +
  theme_classic()
```

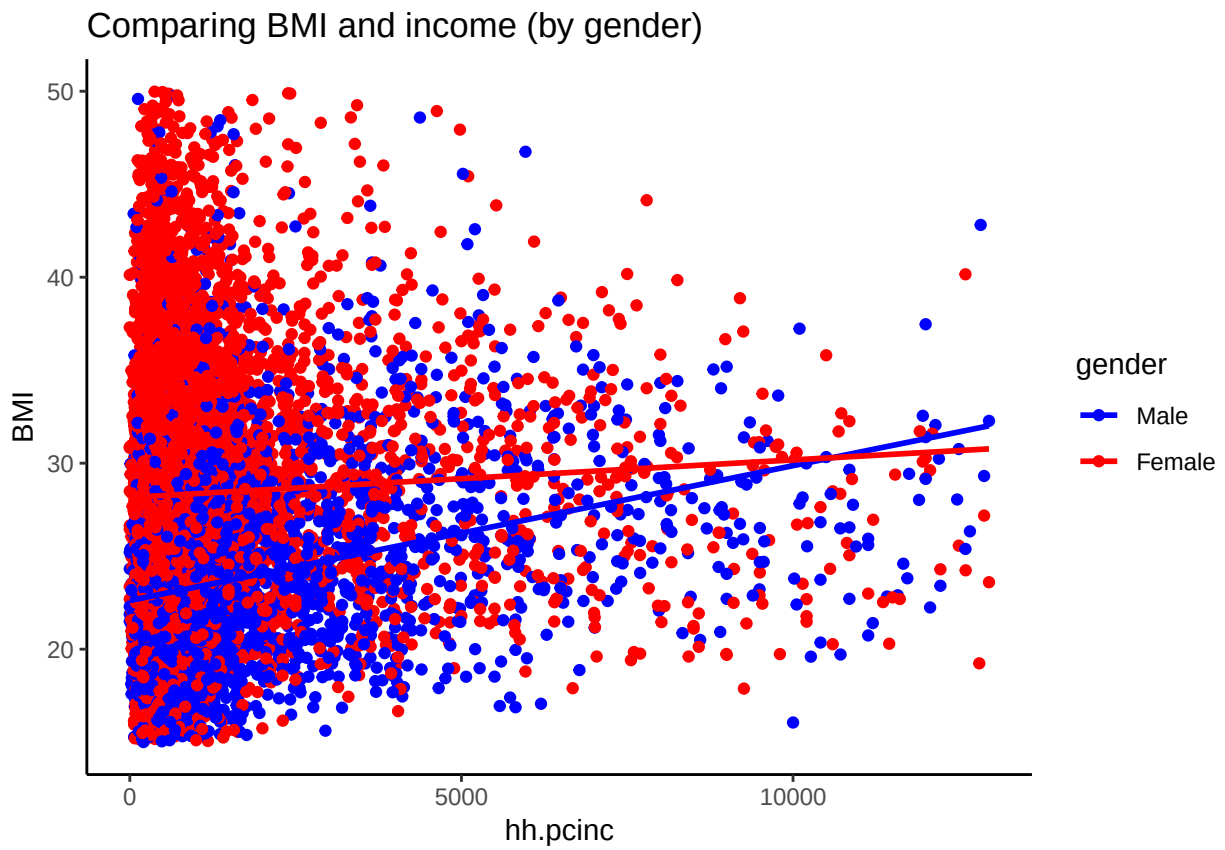


This graph is much clearer than the previous one. What can we say about the relationship between BMI and income from this graph?

The graph shows that the relationship between BMI and income is positive. As income increases, BMI increases. We see that at low income levels there are far more observations than at higher income levels and that the scatter is also more dispersed.

Recall that there are stark differences in mean BMI between men and women. In light of this, alter the graph to give a more complete picture of the relationship between BMI and income by taking gender into consideration.

```
ggplot(nids1 %>% filter(bmi_valid == 1 & hh.pcinc < 13000 & !is.na(gender)), aes(hh.pcinc, bmi, color =
  geom_point() +
  stat_smooth(method = "lm", size=1, formula = y ~ x, se = FALSE) +
  scale_color_manual(values = c("blue", "red")) +
  ggtitle("Comparing BMI and income (by gender)") +
  ylab("BMI") +
  theme_classic()
```



Does this reflect the expected gender differences in BMI? What exactly is this graph telling us?

The graph shows that the relationship between BMI and income is positive for both men and women. The trend line for men is, however, steeper than the trend line for females. This suggests that an increase in income is associated with a greater increase in BMI for men than for women. In fact, BMI is fairly constant across all income levels for females.

From the graph it appears that at low income levels, women tend to have a substantially higher BMI than men. However, at incomes of R10000 the male and female trend lines cross. Let's investigate this further.

```
nids1$inc.cat<-NA
nids1$inc.cat[which(nids1$hh.pcinc<2500)]<-1
nids1$inc.cat[which(nids1$hh.pcinc>=2500 & nids1$hh.pcinc<5000)]<-2
nids1$inc.cat[which(nids1$hh.pcinc>=5000 & nids1$hh.pcinc<10000)]<-3
nids1$inc.cat[which(nids1$hh.pcinc>=10000 & nids1$hh.pcinc <=max(nids1$hh.pcinc))]<-4

nids1$inc.cat<-factor(nids1$inc.cat,
                      levels = 1:4,
                      labels = c("<2500", "250-4999", "5000-9999", "10000+"))
```

```
nids1%>%
  group_by(inc.cat, gender)%>%
  summarise(mbmi = mean(bmi, na.rm=T)) %>%
  arrange(gender,inc.cat)
```

```
## # A tibble: 8 x 3
## # Groups:   inc.cat [4]
##   inc.cat   gender  mbmi
```

```
##   <fct>     <fct>  <dbl>
## 1 <2500     Male    23.3
## 2 250-4999  Male    25.9
## 3 5000-9999 Male    28.6
## 4 10000+    Male    27.5
## 5 <2500     Female  28.9
## 6 250-4999  Female  30.1
## 7 5000-9999 Female  28.9
## 8 10000+    Female  29.2
```

The table shows that the relationship between BMI and income is non-linear for females; BMI first increases with income and thereafter decreases. For males the trend is far more linear with only a marginal decline in BMI at very high income levels.

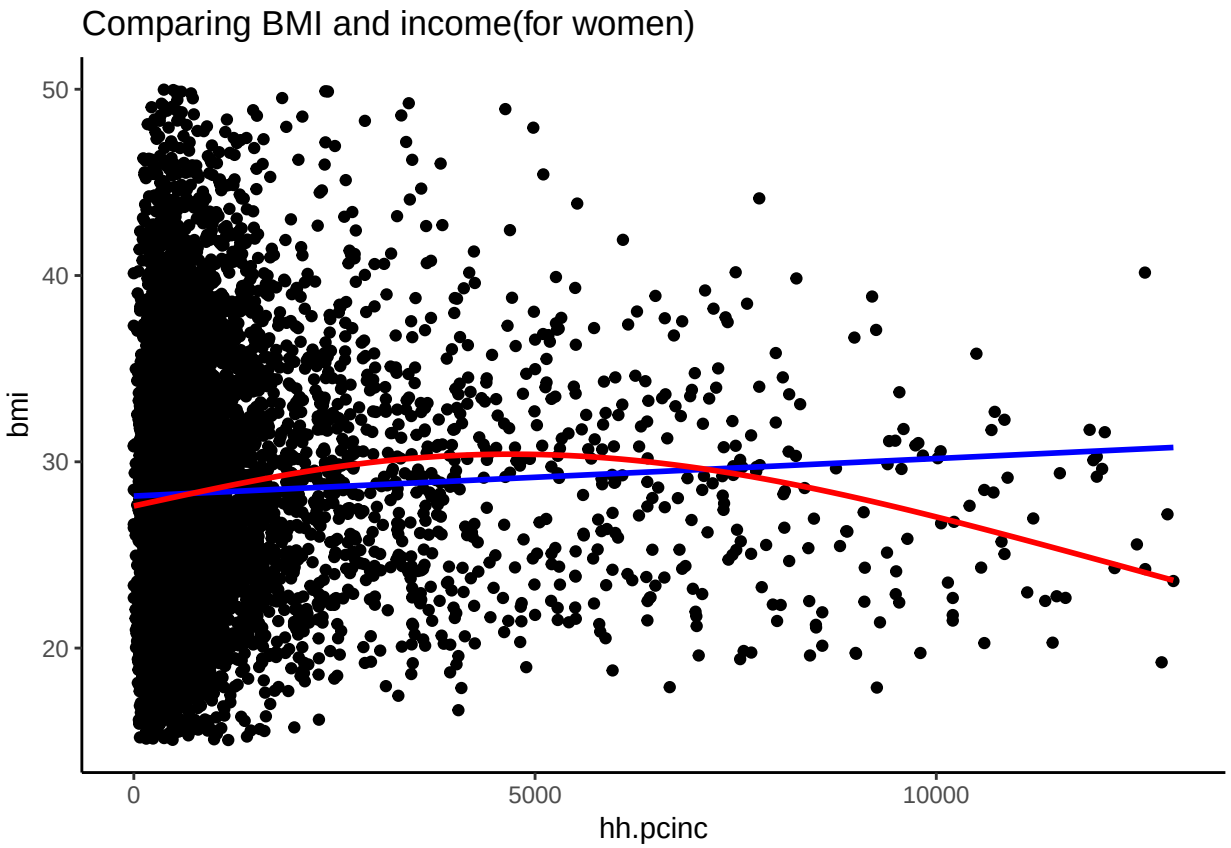
8. What are some of the reasons why the relationship would be non-linear?

Question 8 Answer

Let's add a non-linear trend line to our BMI-income graph and see whether it picks up the non-linear trend apparent in the table. We restrict our sample to females only.

In this case, we want to fit a generalised additive model that fit splines from the `mgcv` package.

```
library(mgcv)
nids1%>%
  filter(bmi_valid == 1 & hh.pcinc < 13000 & gender=="Female")%>%
  ggplot(., aes(x = hh.pcinc, y = bmi)) +
  geom_point() +
  stat_smooth(method = "lm", formula = y ~ x, se = FALSE, colour = "blue", aes(colour = "linear")) +
  stat_smooth(method = "gam", formula= y ~ s(x, k = 3), size = 1, se = FALSE, colour = "red", aes(colou
  ggtitle("Comparing BMI and income(for women)") +
  theme_classic()
```



The curved non-linear trend supports the findings from our table; female BMI initially increases with income up until an income level of approximately R 5000 per month and thereafter it declines.

So far we have investigated the relationships between several variables using graphical and simple descriptive statistical methods. In the next chapter, we will be introduced to using simple regression analysis in R which will allow us to conduct far more powerful analyses. But before moving on, use the exercises to revise what you have learnt.

6.11 Question Answers

6.12 Exercises

1. Which of the following two variables is more closely related to total monthly income - show graphically and statistically? Variables: Metropolitan area, number of biological children who cried but later died?

Exercise 1 Answer

2. Create a table and a pie graph showing occupation categories for the males in the survey.

Exercise 2 Answer

3. Compare the number of hours worked per day and racial status. First, generate a table without missing values, then generate one with missing values.

Exercise 3 Answer

4. What is the total number of African households for whom the number of children residing in the house is greater than four?

Exercise 4 Answer

5. Recall that we created an age group variable, age_bins, in an earlier chapter. Using this variable or otherwise answer the following questions:

a. How many overweight 30-40 year olds in the sample?
b. b.What percentage of 50-60 year olds are underweight?
c. What percentage of obese people are between 30 and 40 years old? How many obese people are there in the sample in total?

Exercise 5 Answer

6.
a. Create a table that contains the mean BMI values for every age and gender group. Comment on the differences between the cells. b. Create a variable that assigns the mean BMI for the age-gender group into which the individual falls. For example, if the individual is a 54 year old female then she will be assigned the mean BMI of 50-60 year old females. Discuss how the results you obtain here relate to your answer to question 5.2.a.

Exercise 6 Answer

6.13 Exercise answers

6.14 Session information

```
print(sessionInfo(), locale = FALSE)
```

```
## R version 3.5.1 (2018-07-02)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 7 x64 (build 7601) Service Pack 1
##
## Matrix products: default
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] mgcv_1.8-24     nlme_3.1-137    gmodels_2.18.1  bindrcpp_0.2.2
##  [5] forcats_0.3.0   stringr_1.3.1   dplyr_0.7.6     purrr_0.2.5
##  [9] readr_1.1.1     tidyr_0.8.1     tibble_1.4.2    ggplot2_3.0.0
## [13] tidyverse_1.2.1 foreign_0.8-70
##
## loaded via a namespace (and not attached):
##  [1] gtools_3.8.1     tidyselect_0.2.4 xfun_0.2         reshape2_1.4.3
##  [5] haven_1.1.2      lattice_0.20-35  colorspace_1.3-2 htmltools_0.3.6
##  [9] yaml_2.1.19      utf8_1.1.4       rlang_0.2.1      pillar_1.2.3
## [13] withr_2.1.2      glue_1.2.0       modelr_0.1.2     readxl_1.1.0
## [17] bindr_0.1.1      plyr_1.8.4       munsell_0.5.0    gtable_0.2.0
## [21] cellranger_1.1.0 rvest_0.3.2      psych_1.8.4      evaluate_0.10.1
## [25] labeling_0.3     knitr_1.20       parallel_3.5.1   highr_0.7
## [29] broom_0.4.5      Rcpp_0.12.17     backports_1.1.2  scales_0.5.0
```

```
## [33] gdata_2.18.0     jsonlite_1.5     mnormt_1.5-5     hms_0.4.2
## [37] digest_0.6.15    stringi_1.1.7    bookdown_0.7     grid_3.5.1
## [41] rprojroot_1.3-2  cli_1.0.0        tools_3.5.1      magrittr_1.5
## [45] lazyeval_0.2.1   crayon_1.3.4     pkgconfig_2.0.1  Matrix_1.2-14
## [49] MASS_7.3-50      xml2_1.2.0       lubridate_1.7.4  assertthat_0.2.0
## [53] rmarkdown_1.10   httr_1.3.1       rstudioapi_0.7   R6_2.2.2
## [57] compiler_3.5.1
```

Chapter 7

Simple regression analysis

7.1 Getting ready

In the previous chapters we generated some variables and ran some commands that will influence the results that we get in this chapter. If you are starting a new session of R, please copy the following lines of code into the R script editor and then run it before you begin the chapter. Make sure that you remember what each line of code is doing.

```
library(foreign)
library(tidyverse)
```

```
## -- Attaching packages ------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

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

```
nids<-read.dta("./data/nids.dta", convert.factors=FALSE)

nids<-nids%>%
  arrange(hhid, pid)%>%
  group_by(hhid) %>%
  mutate(hhrestrict = 1:n()) %>%
  mutate(hhrestrict = ifelse(hhrestrict==1,1,0))


#Class
nids$class<-NA
nids$class[which(nids$w1_fwag<=1500)]<-1
nids$class[which(nids$w1_fwag>1500 & nids$w1_fwag<=4500)]<-2
nids$class[which(nids$w1_fwag>4500)]<-3

nids$class<-factor(nids$class, levels=1:3, labels = c("Lower Class","Middle Class","Upper Class"))
```

```
###Creating a BMI variable - from chapter 2 ***

#Height
nids<-nids %>%
  mutate(height = ifelse (w1_a_n1_1 >= 0 & w1_a_best_age_yrs >= 20, w1_a_n1_1/100, NA))

#Weight
nids<-nids %>%
  mutate(weight = ifelse (w1_a_n2_1 >= 0 & w1_a_best_age_yrs > 20, w1_a_n2_1, NA))

#BMI
nids<-nids %>%
  mutate(bmi = weight/height^2)

#Valid BMI values
nids<-nids%>%
  mutate(bmi_valid = ifelse(bmi > 15 & bmi < 50,1,NA))

#BMI bins
nids$bmi.bins.nolabel<-NA
nids$bmi.bins.nolabel[which(nids$bmi>=15 & nids$bmi<18.5)]<-1
nids$bmi.bins.nolabel[which(nids$bmi>=18.5 & nids$bmi<25)]<-2
nids$bmi.bins.nolabel[which(nids$bmi>=25 & nids$bmi<30)]<-3
nids$bmi.bins.nolabel[which(nids$bmi>=30 & nids$bmi<=50)]<-4

nids$bmi.bins<-factor(nids$bmi.bins.nolabel, levels=1:4, labels = c("Underweight","Normal", "Overweight

#Age
nids<-nids%>%
  mutate(age_adult = ifelse(w1_a_best_age_yrs<0,NA, w1_a_best_age_yrs))

#Age bins
nids$age_bins<-NA
nids$age_bins[which(nids$w1_r_best_age_yrs>=20 & nids$w1_r_best_age_yrs<=29)]<-1
nids$age_bins[which(nids$w1_r_best_age_yrs>29 & nids$w1_r_best_age_yrs<=39)]<-2
nids$age_bins[which(nids$w1_r_best_age_yrs>39 & nids$w1_r_best_age_yrs<=49)]<-3
nids$age_bins[which(nids$w1_r_best_age_yrs>49 & nids$w1_r_best_age_yrs<=59)]<-4
nids$age_bins[which(nids$w1_r_best_age_yrs>59 & nids$w1_r_best_age_yrs<=69)]<-5
nids$age_bins[which(nids$w1_r_best_age_yrs>69 & nids$w1_r_best_age_yrs<=120)]<-6

nids$age_bins <- factor(nids$age_bins, levels = 1:6, labels = c("20 - 29 yrs","30 - 39 yrs", "40 - 49 y

#Rename
nids <- nids%>%
  mutate(race = w1_best_race,
         age = w1_r_best_age_yrs,
         gender = w1_r_b4,
         province = w1_hhprov,
         hhincome = w1_hhincome) %>%
  mutate(gender = factor(gender, levels = 1:2, labels = c("Male", "Female")),
          race = factor(race, levels = 1:4, labels = c("African", "Coloured","Asian", "White")),
          province = factor(province, levels=1:9, labels = c("Western Cape","Eastern Cape","Northern Cap
          w1_hhgeo = factor(w1_hhgeo, levels = 1:4, labels = c("Rural formal", "Tribal authority areas",
```

7.2 Introduction

In Chapter 6, you learned methods in R that allowed you to determine whether two variables were statistically related or independent of one another. While this is indeed important, it is often necessary to take your analysis a few steps further to determine the actual relationship between variables.

In this *Simple Regression Chapter*, we will cover the first two methods commonly used to determine the relationship between two variables. The first is correlation analysis, which simply measures the strength or degree of association between two continuous variables. The second is *simple regression analysis*, which allows us to determine how one variable changes in relation to a change in another variable.

In regression analysis, we are often interested in causal relationships. For example, we could be interested in whether political participation depends on individual income? Or we could be interested in whether product advertisement on the Internet leads to higher sales? In general, we are interested in whether variable X has an effect on variable Y. As such, it is often useful to think of variable X as the "independent" or "explanatory" variable and to think of variable Y as the "dependent" variable or as the "effect". However, it is important to note that a regression tests only for an association between the movements of two variables and does not tell us whether one of the variables causes the other.

In this chapter, we will concentrate on the relationship between total monthly household expenditures (`w1_h_expenditure`) and total monthly household income (`w1_hhincome`). Do you think that one of these two variables exerts a causal influence on the other? We might hypothesize that there is a strong causal link between a household's income and the amount they spend. It makes sense that the more money you receive, the more you will spend. It is your hypothesis which tells you which variable should be your independent variable and which should be your dependent variable.

According to our hypothesis that income has a causal effect on expenditure, expenditure will be the **dependent** variable and income will be the **independent** variable. Once we have used theory and logical reasoning to decide on our hypothesis, we use a regression to test whether there is indeed an association between the relevant variables. And further, to determine the magnitude of the relationship. Below, we will test whether a higher income will be associated with a higher level of expenditure. If it does, we will try to determine how much more a household spends when their income increases.

Since the two variables that we are interested in (`w1_h_expenditure` and `w1_hhincome`) are household-level variables, the first thing that we want to do is to limit the observations to one per household. As we have done in the previous chapters, we need to be careful to use only one observation per household; otherwise we will have the problem of multiple counting. This would lead to larger households having a greater influence on the results than smaller households.

For now, let's concentrate on the first method we mentioned, correlation analysis. Then we will proceed on to simple regression.

7.3 Correlation of variables

Suppose someone made the statement, "Households that earn 6,000 Rand spend more than households that earn 3,000." This sounds like a reasonable statement to make, however, being the researchers that we are; we want to confirm our intuition with empirical facts. Since we are dealing with two continuous variables and we want to test whether there is a linear relationship (i.e. that a R1 increase in income causes a constant increase in expenditure, regardless of the person's income level), the appropriate measure of association is a Pearson correlation which in R performs with the `cor` command from the `stats` package.

Note, even if we don't find a linear relationship, this does not mean the variables are not related. For instance, in this scenario, you might contend that someone who moves from an income of R100 to R150 will probably spend the entire R50 increase, while someone who moves from an income of R10 000 to R10 050 is more likely to save a portion of the R50. If this is the case, then the relationship may be non-linear (the effect of

an increase in income on expenditure is different at different income levels). Nevertheless, testing for a linear relationship is generally a very good starting point for analysis.

The Pearson correlation measures the degree to which variables are related or in other words, the degree to which they co-vary. As mentioned above, when using correlation in our analysis, we must make the assumption that the relationship between our two variables is linear. If we suspect otherwise, we should make the proper adjustments to the variable that does not meet the assumption (we will cover this in more detail in Chapter 8). Overall, the initial use of the correlation coefficient is a good way to start investigating whether your intuition about a relationship is remotely correct.

(Notice how we use our `hhrestrict` dummy variable to ensure only one observation per household)

R produces the following results:

```
cor(nids[nids$hhrestrict==1,c("w1_hhincome","w1_h_expenditure")], use="complete.obs")
```

```
##                  w1_hhincome w1_h_expenditure
## w1_hhincome         1.000000         0.689326
## w1_h_expenditure    0.689326         1.000000
```

Or pipe using `dplyr`

```
nids %>%
  filter(hhrestrict==1) %>%
  ungroup() %>%
  select(w1_hhincome,w1_h_expenditure) %>%
  cor() #check cor arguments if you want to change the default
```

```
##                  w1_hhincome w1_h_expenditure
## w1_hhincome         1.000000         0.689326
## w1_h_expenditure    0.689326         1.000000
```

R produces the above table, but what does it mean? A correlation value can range from -1 to +1, with 0 indicating that there is no linear association and ± 1 being a perfect linear association. If the correlation value is low (near 0), there may still be a non-linear association between the variables of interest.

In our example, you will notice that the coefficient 0.6893 is relatively large and positive. This means that the linear association between our two variables is relatively strong. Thus, as the values of `w1_h_expenditure` increase, so do the `w1_hhincome` values. More clearly, households with a higher total monthly income tend to also have higher total monthly expenditures.

Let's get back to the original statement, "Households that earn 6,000 Rand spend more than households that earn 3,000." Our initial study of the matter suggests that this is likely to be true, according to our R correlation calculations. This initial approach, however, is too simplistic. R can do much more. We can go further and figure out by exactly how much total monthly household income influences total monthly household expenditure. To do this, we can fit a linear model using the `lm()` command. Before we move on, however, try the following questions:

1. What the correlation between the average household's food expenditure and household size?

Question 1 Answer

7.4 Outliers

Before we continue on to simple regression analysis, it is a good idea to spend a few minutes reviewing the issue of outliers again. We must be extremely mindful of possible outliers and their adverse effects during any attempt to measure the relationship between two continuous variables. This is particularly true when using

methods that rely on the mean of a given variable, as is the case in both correlation and regression analysis. You should recall from an earlier chapter that **means** are extremely sensitive to outliers, whether positively or negatively skewed. Therefore, let's take a quick look at how our two variables `w1_h_expenditure` and `w1_hhincome` are distributed. Note, it is always a good idea to get a feel for the variables you are using, before jumping into statistical analyses. One simple way to accomplish this is to graph these variables in one scatter plot. Let's try it:

```
library(scales)
```

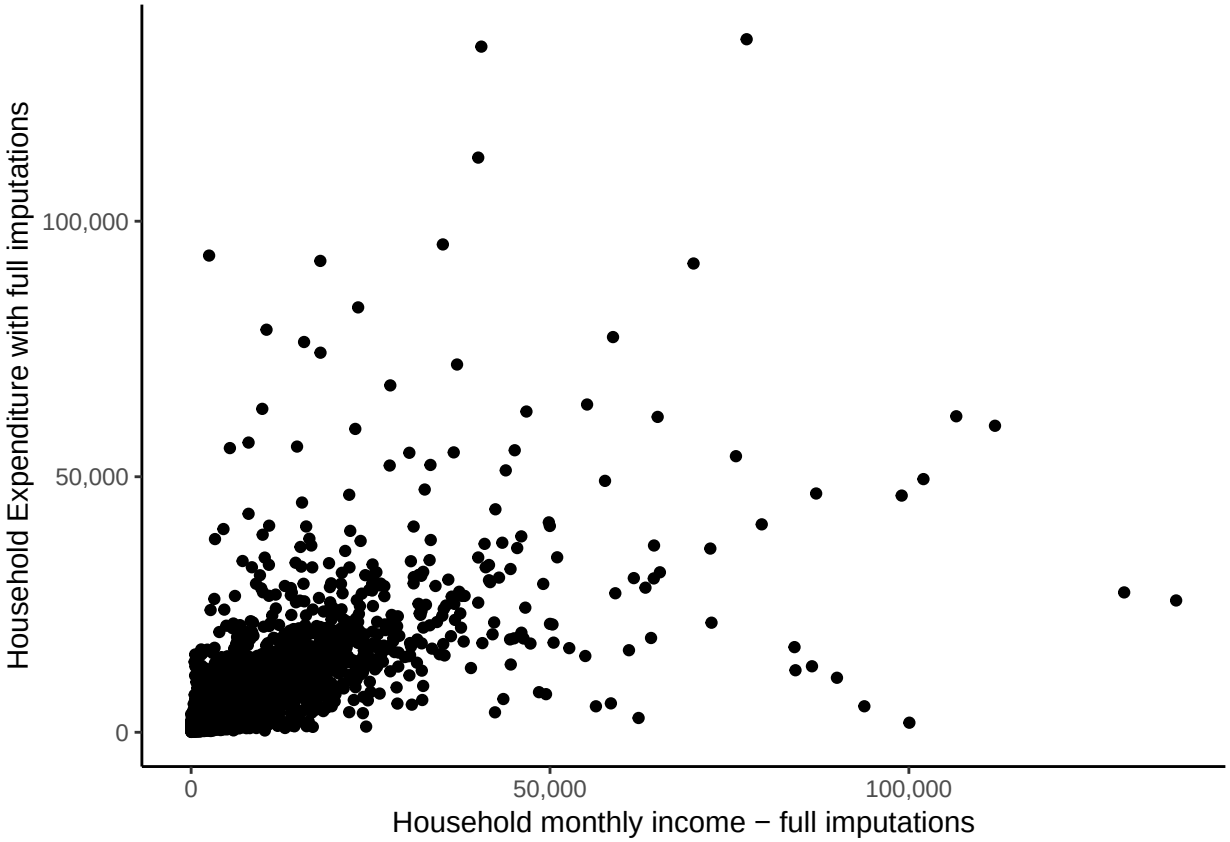
```
##
## Attaching package: 'scales'
```

```
## The following object is masked from 'package:purrr':
##
##     discard
```

```
## The following object is masked from 'package:readr':
##
##     col_factor
```

```
nids%>%
  filter(hhrestrict==1)%>%
  select(w1_h_expenditure, w1_hhincome)%>%
  ggplot(., aes(x = w1_hhincome, y = w1_h_expenditure)) +
  geom_point() +
  scale_x_continuous(breaks=seq(0,150000,50000), labels = comma) +
  scale_y_continuous(breaks=seq(0,150000,50000), labels = comma) +
  xlab("Household monthly income - full imputations") + ylab("Household Expenditure with full imputatio
  theme_classic()
```

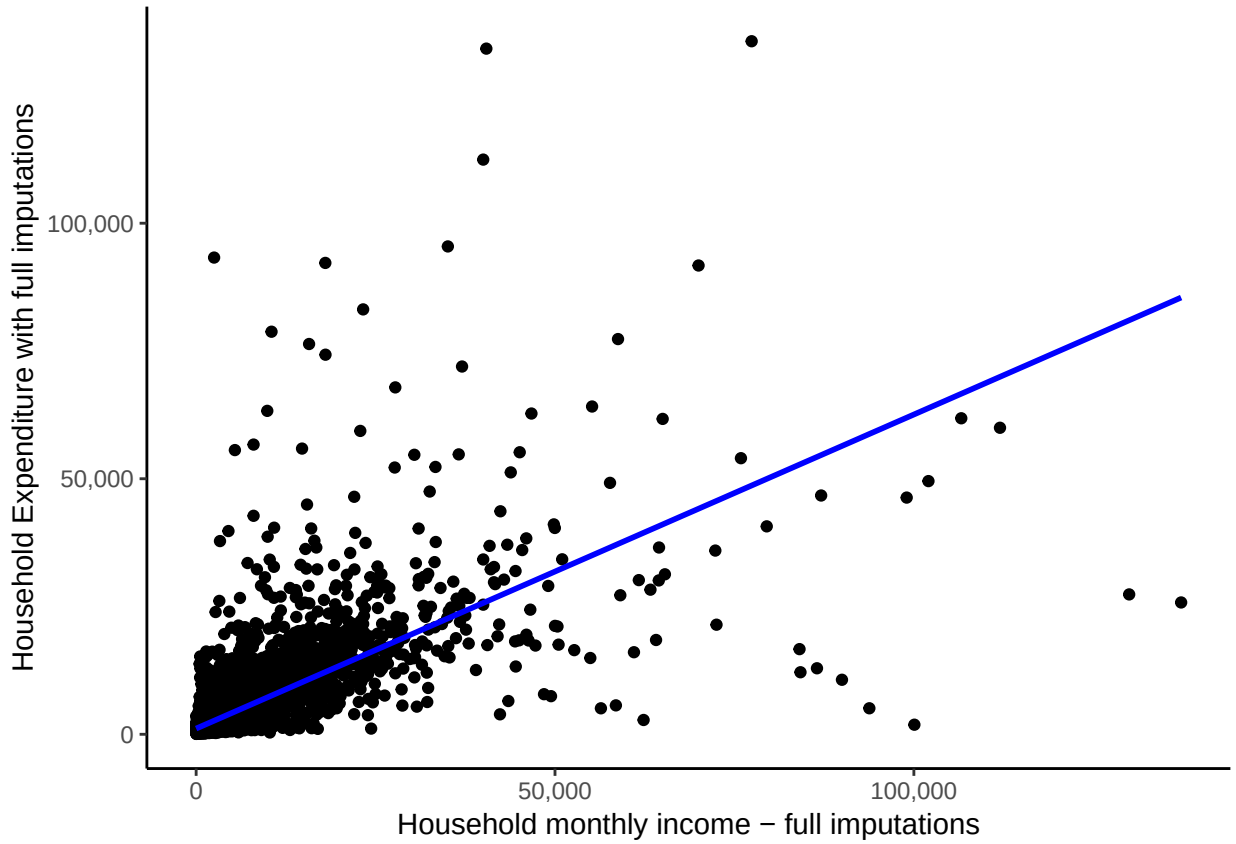
```
## Adding missing grouping variables: `hhid`
```



Now, it appears that there is a positive relationship between the two variables as the points with higher income levels tend to also have higher expenditure values. However, a large proportion of the points are concentrated in the lower left-hand corner of the graph and it is very difficult to see what is going on there. It may be the case that the positive relationship is only exhibited by households with high income and expenditure values. In order to get a better idea, we can ask R to include a fitted line which might provide us with a better indication of the relationship between the two variables. The `stat_smooth` function helps us to fit the line as follows:

```
nids%>%
  filter(hhrestrict==1)%>%
  select(w1_h_expenditure, w1_hhincome)%>%
  ggplot(., aes(x = w1_hhincome, y = w1_h_expenditure)) +
  geom_point() +
  scale_x_continuous(breaks=seq(0,150000,50000), labels = comma) +
  scale_y_continuous(breaks=seq(0,150000,50000), labels = comma) +
  xlab("Household monthly income - full imputations") + ylab("Household Expenditure with full imputatio
  stat_smooth(method = "lm", se = FALSE, colour = "blue") +
  theme_classic()
```

```
## Adding missing grouping variables: `hhid`
```

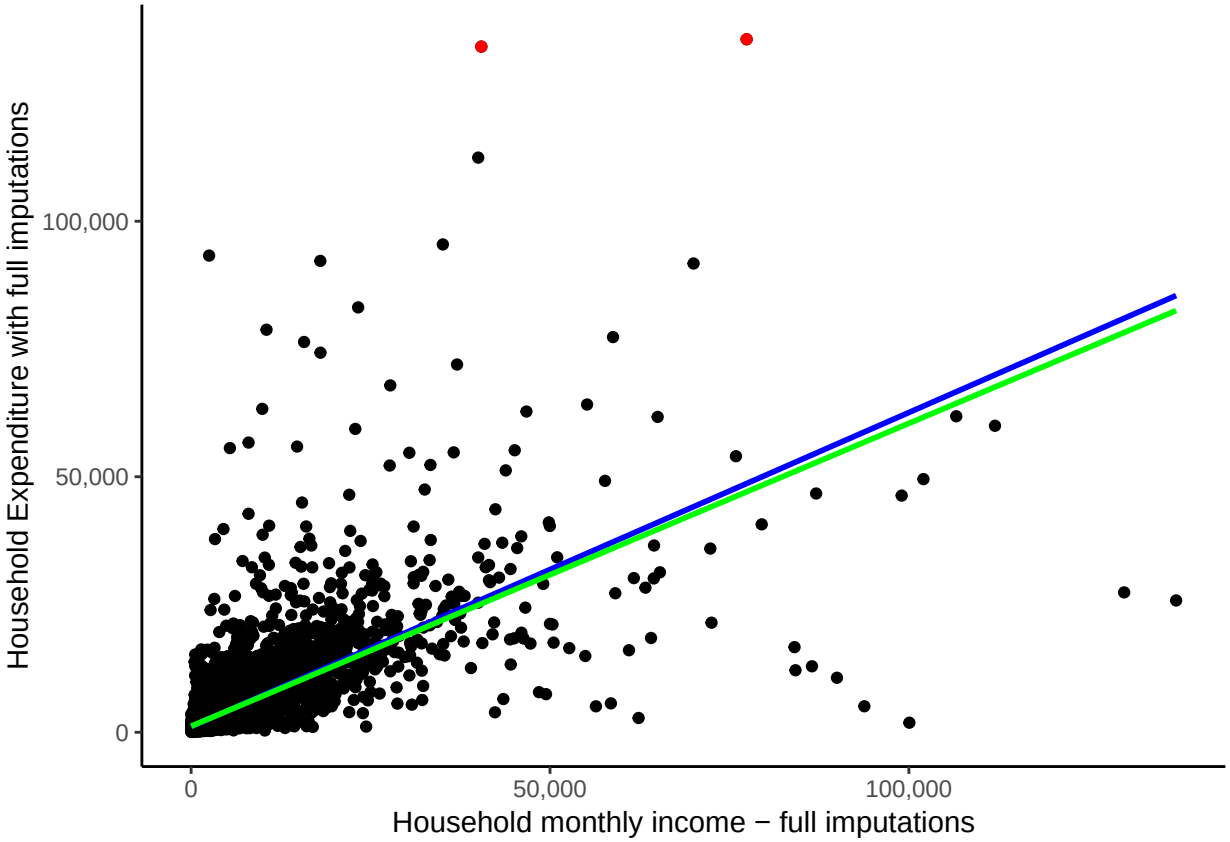


We see here that the fitted line is upward sloping which supports our initial conclusion of a positive relationship between expenditure and income. Further, it is informative to think about exactly how R draws a fitted line. Basically it tries to find the straight line that minimizes the 'distance' between the all points and the fitted line. Exactly how it does this is beyond the scope of this course. However, the point is that this is essentially a pictorial representation of what a simple linear regression does. It finds the best fit straight line to reflect the relationship between two variables. However, regression analysis is much richer and can give us far more information than just the parameters of the straight line. More on that below, but first let us complete our brief look at outliers by seeing what happens to the straight line when we remove the two outliers circled in red below:

```
scatter<-nids%>%
  filter(hhrestrict==1)%>%
  select(w1_h_expenditure, w1_hhincome)
```

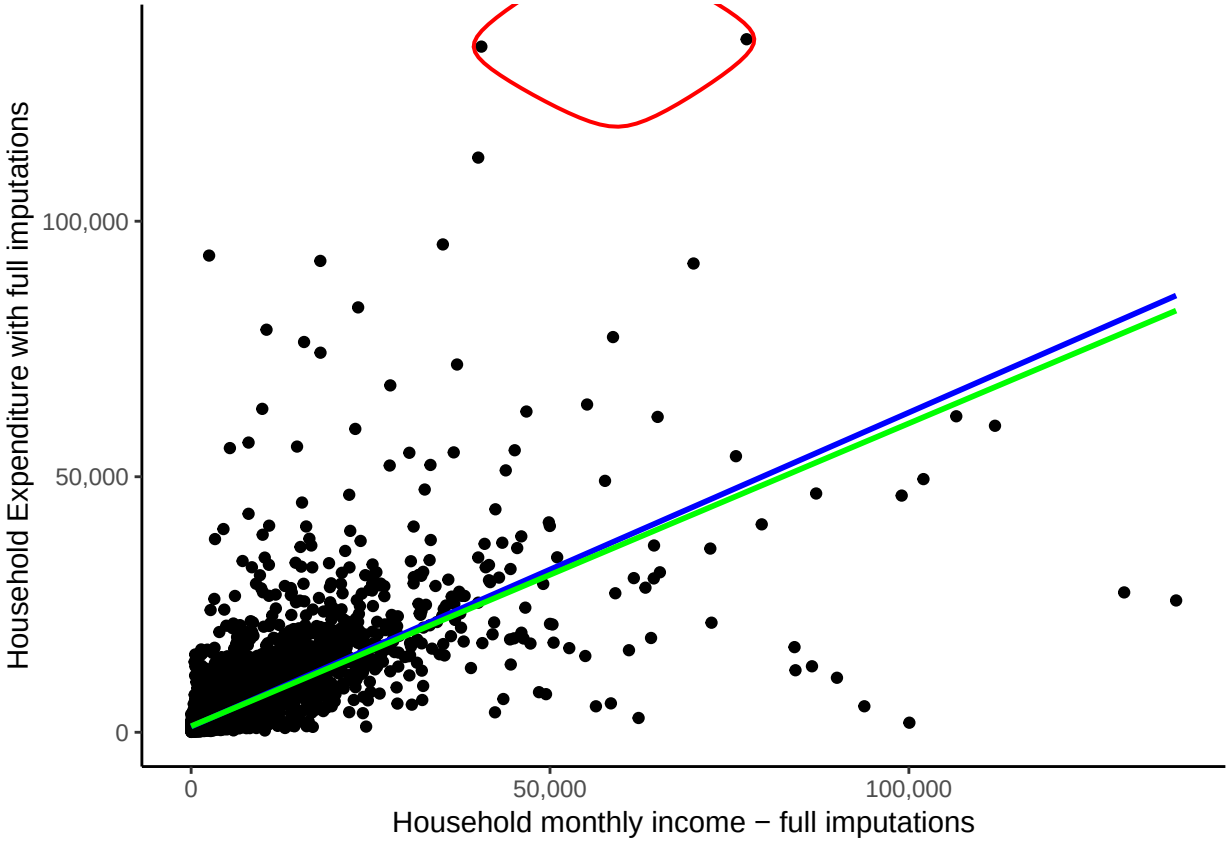
```
## Adding missing grouping variables: `hhid`
```

```
ggplot(scatter, aes(x = w1_hhincome, y = w1_h_expenditure)) +
  geom_point() +
  scale_x_continuous(breaks=seq(0,150000,50000), labels = comma) +
  scale_y_continuous(breaks=seq(0,150000,50000), labels = comma) +
  xlab("Household monthly income - full imputations") + ylab("Household Expenditure with full imputatio
  stat_smooth(method = "lm", se = FALSE, colour = "blue") +
  geom_point(data = subset(scatter, w1_h_expenditure > 130000), aes(x = w1_hhincome, y = w1_h_expenditu
  stat_smooth(data = subset(scatter, w1_h_expenditure < 130000), method = "lm", se = FALSE, colour = "g
  theme_classic()
```



A more elegant way of highlighting points using `ggalt geom_encircle` function:

```
library(ggalt)
ggplot(scatter, aes(x = w1_hhincome, y = w1_h_expenditure)) +
  geom_point() +
  scale_x_continuous(breaks=seq(0,150000,50000), labels = comma) +
  scale_y_continuous(breaks=seq(0,150000,50000), labels = comma) +
  xlab("Household monthly income - full imputations") + ylab("Household Expenditure with full imputatio
  stat_smooth(method = "lm", se = FALSE, colour = "blue") +
  stat_smooth(data = scatter %>% filter(w1_h_expenditure < 130000), method = "lm", se = FALSE, colour =
  geom_encircle(data = scatter %>% filter(w1_h_expenditure > 130000),
                aes(x = w1_hhincome, y = w1_h_expenditure),
                color="red",
                size=2,
                expand=0.01) +
  theme_classic()
```



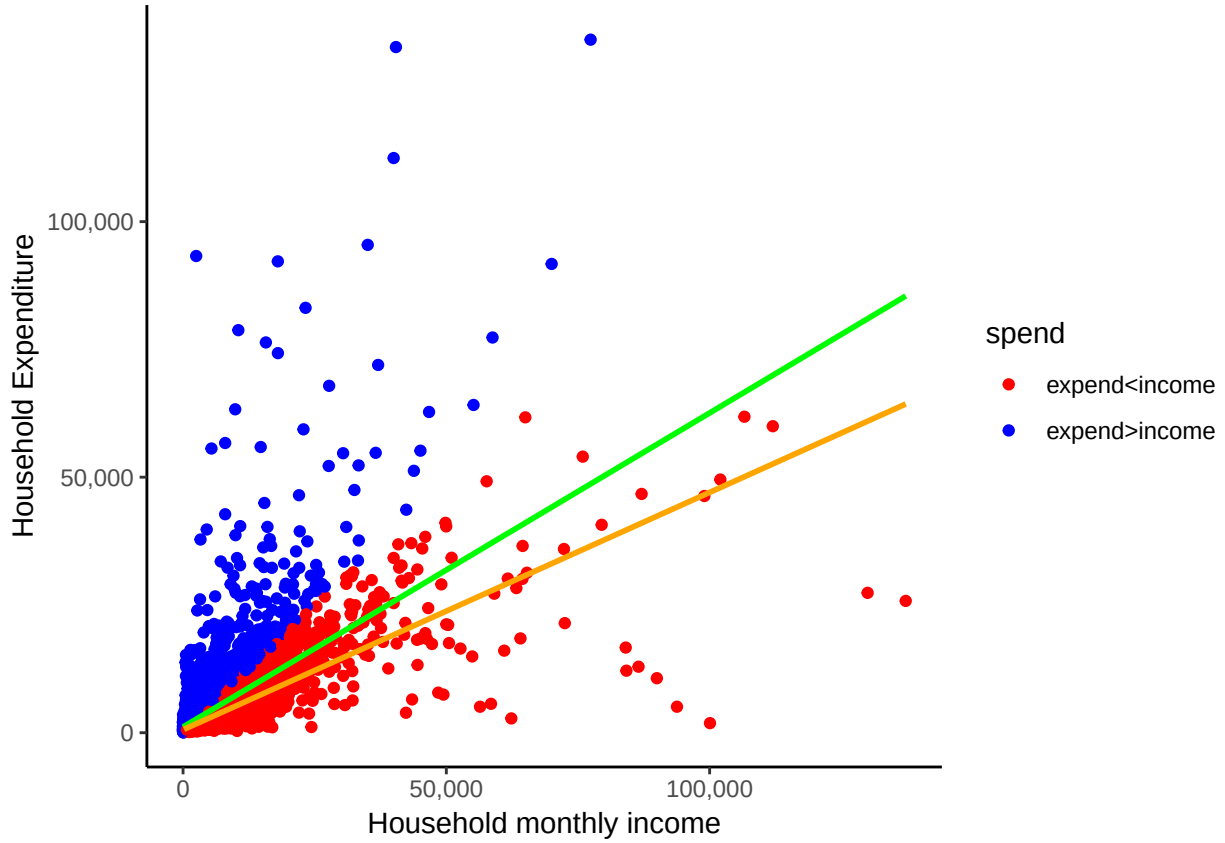
These two circled households are spending FAR more than their income. This in itself is not a contradiction as it is possible that they are spending wealth that was accumulated in the past or through inheritance. Here, it is simply informative to see how much of an influence they exert on our fitted line, regardless of whether they are valid expenditure values. The green line indicates the fitted line without their influence and while it may seem close to the red line, there is a marked difference considering that we have only excluded 2 out of 7305 observations. We will see below that slope of the red line is 0.615, while the slope of the green line is 0.593. If we interpret these results, we would say in the first case that for every extra rand of income the average household will spend 61.5 cents more. In the second case, they will spend 59.3 cents more. This becomes a sizable difference if we talk about increasing a household's income by R5000: The first case will predict a R3075 increase in expenditure, while the second case will predict only a R2965 increase in expenditure. This again illustrates the large effect that outliers can have.

A point of caution: You should be extremely wary of deleting data (e.g. in the case of outliers). You must always make sure you have strong reasons for doing so and are not simply doing so because it will give you 'nice' results! For instance, here deleting these two circled observations because the households' expenditures exceed their incomes would be unjustified as many of the other households also spend more than their income (see the graph below).

```
scatter<-scatter %>%
  mutate(spend=ifelse(w1_h_expenditure < w1_hhincome,1,2)) %>%
  mutate(spend=factor(spend, levels=1:2,labels=c("expend<income", "expend>income")))

  ggplot(scatter, aes(x = w1_hhincome, y = w1_h_expenditure)) +
    geom_point(aes(x = w1_hhincome, y = w1_h_expenditure, color=spend)) +
    scale_color_manual(values=c("red","blue")) +
    stat_smooth(method = "lm", se = FALSE, colour = "green") +
    stat_smooth(data = subset(scatter, w1_h_expenditure < w1_hhincome), method = "lm", se = FALSE, colou
```

```
    scale_x_continuous(breaks=seq(0,150000,50000), labels = comma) +
    scale_y_continuous(breaks=seq(0,150000,50000), labels = comma) +
    xlab("Household monthly income") + ylab("Household Expenditure") +
    theme_classic()
```



All the blue dots represent households that have a higher expenditure than income. The orange line is the fitted line that we would get if we excluded all of these values. While it may be the case that some of the blue dots are errors, it is highly unlikely that all of them are. Therefore, we would have been unjustified in deleting the two outliers we initially identified simply because their expenditure exceeded their income. The general message is that while outliers can influence your results substantially, you need to be VERY careful in how you deal with them! Also, take the time to get well acquainted with your data!

Formally Testing for Outliers

There is some disagreement in the field regarding the handling of outliers. Some fields in social research suggest and embrace an active approach to the handling of outliers, whereas others take a more hands-off approach. Neither one approach is superior to the other; after all, both are efforts to minimize the effects of extreme values. On one hand, the aggressive approach chooses to control for the ill effects by eliminating cases from the models. Whereas the handsoff approach, often chooses to use more robust estimation procedures which can handle extreme values in the data.

For our purposes, we do not eliminate any observations from our data for 3 reasons: 1) we do not have a serious outlier problem, 2) an in depth study of how to formally handle outliers is beyond the scope of this course, and 3) we advocate the use of more robust procedures to handle possible outliers, however, those procedures are also beyond the scope of this course.

7.5 Simple regression

Now that we have reviewed the issue of outliers, we can proceed with our study of regression using R.

Simple OLS regression (Ordinary Least Square regression) is a procedure that determines the best fitting regression line between two variables. In essence, the OLS regression line reduces the sum of squared errors to a minimum between two variables. We saw a graphical representation of this when we used R to construct fitted straight lines. It is beyond the scope of this course to teach you the finer points and intricacies of regression analysis; however, we will provide useful examples to give you a feel for what it is in general. Our main purpose here will be to show you how to use R to calculate the regression line between two variables and how to interpret the results. If you are not clear on what exactly regression is or would like to have a deeper understanding of it, we suggest that you take a course in statistics relating to your field of interest.

In general, the simplest relationship between an independent and dependent variable can be expressed in the linear formula,

$Y = \alpha + \beta X$

where **Y** is the dependent variable and **X** is the independent variable. This is just the same formula that you used in school for defining a straight line. The coefficient "β" is referred to as the slope and tells us how a 1 unit change in **X** will change the value of **Y**. The coefficient "α"" tells us the value of **Y** when the independent variable **X** is zero. On an **X**-by-**Y** graph, the coefficient "α" is where the regression line intercepts with the y-axis.

In the case of `w1_hhincome` and `w1_h_expenditure`, the equation can be written as follows:

$w1\_h\_expenditure = \alpha + \beta * w1\_hhincome$

This equation states that if you increase total monthly household income by one unit there will be a corresponding change (β) in total monthly household expenditure.

Remember that we can type `help()` and the command of interest to learn more about that command R. If we type `help(lm)` we will get a full description of the regression command, its options and its syntax. For our purposes we can type:

```
lm0 <- lm(w1_h_expenditure ~ w1_hhincome, data = nids %>% filter(hhrestrict==1))
```

To see full results, we can use the summary function as follows:

```
#anova(lm0)
summary(lm0)
```

```
##
## Call:
## lm(formula = w1_h_expenditure ~ w1_hhincome, data = nids %>%
##     filter(hhrestrict == 1))
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -60739  -1231   -758     53 108202
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.115e+03  6.954e+01   16.03   <2e-16 ***
## w1_hhincome 6.146e-01  7.558e-03   81.31   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5194 on 7303 degrees of freedom
```

```
## Multiple R-squared:  0.4752, Adjusted R-squared:  0.4751
## F-statistic:  6612 on 1 and 7303 DF,  p-value: < 2.2e-16
```

To get the coefficients:

```
summary(lm0)$coefficients
```

```
##                 Estimate   Std. Error  t value     Pr(>|t|)
## (Intercept) 1115.0143048 69.544789721 16.03304 6.969086e-57
## w1_hhincome    0.6145542  0.007557776 81.31416 0.000000e+00
```

You might also want to check the following functions and many more regression related output:

```
attributes(lm0)
```

```
attributes(summary(lm0))
```

7.6 Understanding regression output tables

What do all these numbers mean? By way of comparison, Stata produces the analysis of variance (ANOVA) table by default and this can be obtained in R by typing `anova(lm0)`. Although we are not particularly interested in this portion of the results, you can learn more about it, if interested click here.

For the purposes of understanding the basic relationship between `w1_h_expenditure` and `w1_hhincome`, we will focus on three pieces of information provided by the output above. First, let's recall the basic linear regression equation:

$Y = a + bX$ or in our case: $w1\_h\_expenditure = \alpha + \beta * w1\_hhincome$

If we look at the regression output coefficients above, we can find the relevant estimates of the parameters 'a' and 'b' in the 'Estiamte' column of the `summary(lm0)$coefficients` output table. The `(Intercept)` is our estimated value for 'a' and the coefficient corresponding to `w1_hhincome` is the estimate for 'b'. If we plug these results into their appropriate spot in the equation, we get:

(predicted w1_h_expenditure) = 1115.014 + 0.6145542 x (w1_hhincome)

In words, this equation is telling us that for every one rand increase in total monthly household income (`w1_hhincome`), total monthly household expenditure (`w1_h_expenditure`) will increase by almost 0.615 Rand (in other words, on average 61.5% of household income goes into household expenditure). The regression output also gives us a t-value and associated probability value for each coefficient. Here, the t-value of 81.31 and associated `Pr(>|t|)`-value of 0 the income coefficient is statistically significant. Or in other words, that the probability that it is equal to zero is less than 0.1 percent.

If we turn our attention to our estimated value of 'a', the constant (`(Intercept)`) tells us that when our independent variable w1_hhincome equals zero, w1_h_expenditure is R1 115.01. This would suggest that when a household has no income its expenditure will be R1 1150.01. We might view this as a crude measure of the minimum necessary expenditure of a household.

Another important piece of information is the R-squared (`Multiple R-squared`) at the bottom of the `summary` output which equals 0.475. In essence, this value tells us that by knowing the value of our independent variable (`w1_hhincome`) we can estimate the value of the dependent variable (`w1_h_expenditure`) approximately 47.5% better than by simply guessing the mean. Or as more commonly talked about, we can account for about 47.5% of the variation around the mean of `w1_h_expenditure` with the knowledge of `w1_hhincome`. If you are interested in knowing what some of the other output means, click here

The Case of Simple Regression

Now that we have a formula which relates a household's income to its expenditure, if someone we are told what a household's total income is, we can use our formula to predict what its expenditure might be. I'm sure you can see that there will be cases in which this prediction is extremely far off. Nevertheless, it is better

than a pure guess. Our regression output also told us that income only explains 47.5% of the variation in expenditure, so we are not claiming to make perfect predictions. In the next chapter, we will look at how we can improve our regression predictions by adding extra relevant variables.

But for now, let's see how well we can predict expenditure of households, simply by using our equation and the level of income that each household has. Let's say that we come across a South African household that has a total monthly income of R5000 and we want to know roughly how much they are likely to spend each month. Using our formula, we plug in 5000 in place of X and then calculate Y:

Y = a + b∗X gives Y = 1115.014 + 0.6145542$*$5000

Now, you can simply type the following in R's console to evaluate the right-hand side of this equation.

```
1115.014 + 0.6145542*5000
```

```
## [1] 4187.785
```

This tells us that the average household that has an income of R5 000 will have a total household expenditure of R4 187.79. It is important to realize that a regression equation will almost never fit the actual observed values perfectly. Therefore, the estimated value of monthly expenditure that our calculation predicts (4 187.79) is just that, a prediction.

We can look at how well our equation predicts the expenditure values for the real households in our sample. We do this by creating a new variable that takes each household's income and uses it to predict how that household's expenditure. We can then compare the predicted expenditure values with the actual expenditure values of the households in the sample. Let's do this.

R calculates predicted values (`fitted.values`) and residuals (`residuals`) automatically (compared to Stata). You can check other attributes of an `lm` by typing (for the previous model):

```
attributes(lm0)
```

```
## $names
##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
##
## $class
## [1] "lm"
```

We just need to bring in these fitted values to the data frame, they are in the same order of the data that generated the model if we have not rearranged the data.

We are going to create a cleaner subset of the data (`nids2`) and run the model:

```
nids2<-subset(nids,subset=hhrestrict==1, select=c(hhid, w1_hhincome,w1_h_expenditure))
lm0 <- lm(w1_h_expenditure ~ w1_hhincome, data = nids2)
nids2$yhat<-lm0$fitted.values
```

Let's compare our new estimated variable with the actual household expenditure values:

```
head(nids2, n=20L)
```

```
## # A tibble: 20 x 4
## # Groups:   hhid [20]
##      hhid w1_hhincome w1_h_expenditure  yhat
##     <int>       <dbl>            <dbl> <dbl>
##  1 101012       1045.             518. 1757.
##  2 101013        588.             861. 1477.
##  3 101014       1307.             614. 1918.
##  4 101015        291.             369. 1294.
```

```
##  5 101016       1304.           1259. 1916.
##  6 101017        213.            483. 1246.
##  7 101018       1785.           1464. 2212.
##  8 101020       1831.           1023. 2241.
##  9 101021       1235.           1426. 1874.
## 10 101022       2489            2581  2645.
## 11 101023       2745.           1280. 2802.
## 12 101024        309.           1494. 1305.
## 13 101025       1087.           1456. 1783.
## 14 101027       1268.           2080. 1894.
## 15 101028       1128.           1351. 1808.
## 16 101029       1096.           1717. 1789.
## 17 101030       1608.           1487. 2103.
## 18 101033        815.           1495. 1616.
## 19 101034       2040            1903. 2369.
## 20 101035        996.            826. 1727.
```

Here is a partial view of what the resulting table should look like:

The third column gives the actual expenditure values, the fourth gives the predicted expenditure values and the second gives the actual income levels used in our prediction. Comparing columns 3 and 4, it is immediately clear that the predicted values are quite far off the actual values. You may also notice that the predicted values tend to be substantially larger than the actual values and often the predicted values are higher than the income level of the household. If we look back at our prediction equation, we see that this is due to the large 'a' value of R1 115.01. This means that anyone earning less than R3145.30 will have a predicted expenditure greater than their income. As we saw above, a large proportion of the sample households do have higher reported expenditures than income. Nevertheless, it appears that the predicted values are too high. This is due to the fact that by running a regression, we are forcing a straight line relationship between income and expenditure and therefore households with extremely high income levels might be pulling the estimated 'a' value far higher than it should be.

This serves to illustrate that it is always good to examine your results critically. For instance, after running the regression above, you should ask yourself whether it makes sense that a household with an income level of R0 will on average spend R1 115.01. And if not, you should try to figure out what is causing the result to be biased. We will not go further in investigating this, but you might want to check whether you can get better estimates if you exclude individuals with high income levels. It might be the case that the relationship between income and expenditure is different for rich and poor people.

While our expenditure predictions are not great, it is worth realizing that they are still better than just guessing R3777.12 for each household, the sample mean for `w1_h_expenditure`.

BUT, is this the best we can do? In the next chapter, we will see how by adding more information to our regression, we may be able to improve our predictions. What other variables do you think might influence a household's expenditure?

For now, let's practice some simple regressions to make sure we are comfortable with what we have learnt.

2. By how much would you expect a household to increase its total monthly food expenditure for every additional household member?

Question 2 Answer

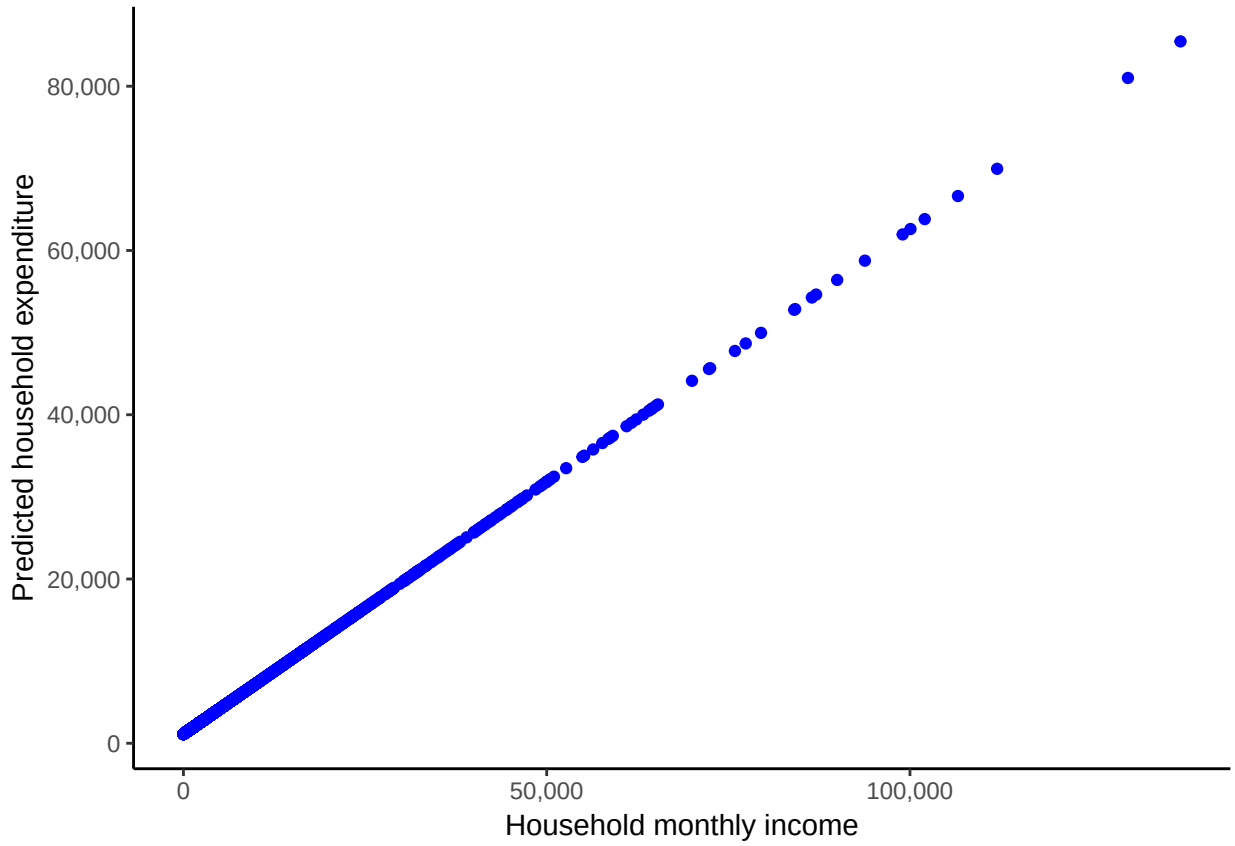
3. Income also affects households' total monthly food expenditure. Try a simple regression of w1_h_expf on total monthly income. Write an equation for this relationship and interpret the result.

Question 3 Answer

7.7 Graphing the regression equation

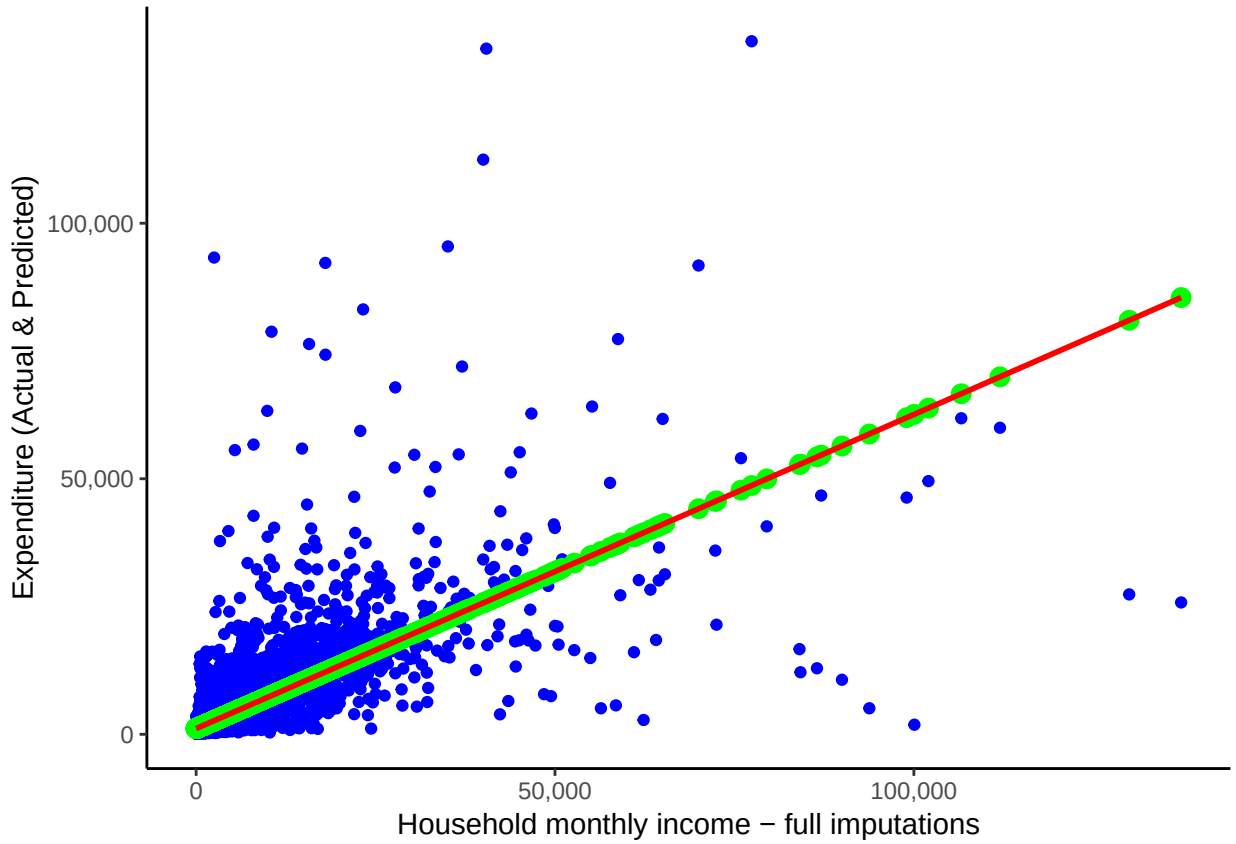
Having obtained the predicted values of the dependent variable `w1_h_expenditure`, we can plot them against income using a scatter plot.

```
ggplot(nids2, aes(x = w1_hhincome, y = yhat)) +
  geom_point(aes(x = w1_hhincome, y = yhat), colour="blue") +
  scale_x_continuous(breaks=seq(0,150000,50000), labels = comma) +
  scale_y_continuous(breaks=seq(0,80000,20000), labels = comma) +
  xlab("Household monthly income") + ylab("Predicted household expenditure") +
  theme_classic()
```



This shows us how R is essentially forcing a straight line relationship between the two variables when we run an ordinary linear regression as we did above. We can also see how the observations are much denser at lower income levels. It may be informative for us to add the scatterplot of the actual income and actual expenditure values, and also to add in the fitted line to cement the basic idea of what a regression is doing.

```
ggplot(nids2, aes(x = w1_hhincome, y = w1_h_expenditure)) +
  geom_point(aes(x = w1_hhincome, y = w1_h_expenditure), colour="blue") +
  geom_point(aes(x = w1_hhincome, yhat), colour="green", size = 3) +
  stat_smooth(method = "lm", se = FALSE, colour = "red") +
  scale_x_continuous(breaks=seq(0,150000,50000), labels = comma) +
  scale_y_continuous(breaks=seq(0,150000,50000), labels = comma) +
  xlab("Household monthly income - full imputations") + ylab("Expenditure (Actual & Predicted)") +
  theme_classic()
```



Here, green dots are the scatterpoints from the graph directly above - i.e. the scatter of `yhat` against household monthly income. And we saw the blue scatterplot and red line towards the beginning of this chapter when we discussed outliers. We can see that all the scatterpoints fall directly on the fitted line. Notice, the scatterpoints come from the predicted values that we obtained from our regression, while the fitted line is constructed by R simply using the original data and finding the best fitted line. This graph clearly illustrates the idea that what a regression is doing is finding the best-fit straight line and then estimating the slope and intercept of this line. As we saw above, it also provides us with a range of further useful information.

We asked earlier whether this is the best we can do in terms of predicting household expenditure or whether you could think of any other variables that might influence a household's expenditure level. In answer to the first question, it is not! How about the number of household members? Surely, a household with 10 members will spend more than a household with 1 - even if they have the same level of income. How about whether the household is in a rural versus a metropolitan setting? If we could incorporate the information contained in these extra variables then it is likely that we will be able to better understand the determinants of household spending. While, to a large extent we have focused in this chapter on using the parameters that we calculate using regression to obtain predicted values of expenditure, prediction is more of a byproduct of regression analysis, not its primary aim. Regression analysis is more generally used to understand and quantify the relationship between a set of variables. With regards to our expenditure example, this will take the form of trying to understand what factors influence household spending and by how much does each factor effect household spending. This information can then be used in various different ways, depending on your research question - one of these is to predict what would happen to household expenditure if we varied one or more of the dependent variables.

This type of analysis, however, requires more than a simple regression between two variables, it requires what is known as multiple regression. We cover multiple regression in the next chapter. But first, let's work through an interesting worked example which looks at the relationship between a person's body index and her age. What are your expectations about this relationship?

7.8 Worked example: Is age a strong determinant of BMI?

In chapter 5, we began investigating the relationship between BMI and age. Recall that BMI is derived from weight and height. Since height does not change substantially in adulthood, adult BMI changes as a result of changes in weight. Think about how a person's weight changes over their adult life cycle. As a result of this, what would you expect the relationship between BMI and age to be?

You are probably beginning to realize that when analyzing a relationship between variables it is useful to start with a hypothesis or set of hypotheses and then interrogate them. During this process we reject the hypotheses found to be false and accept or update the others.

What is your hypothesis about the relationship between BMI and age? Do you think it will be a linear or non-linear relationship? Would you expect this relationship to be different for men and women?

We restrict the BMI value to exclude people with extreme BMI values and check the correlation between BMI and age. Note, that we also created a new variable called `'adult_age'` in the "getting ready" section at the beginning of the chapter. This variable is essentially exactly the same as the `w1_a_best_age_yrs` variable except that it has no negative values and has a slightly less cumbersome name.

```
corr2.df<-nids[nids$bmi_valid==1,c("bmi","age_adult")]
cor(corr2.df,use="complete.obs")
```

```
##                  bmi age_adult
## bmi       1.0000000 0.1465733
## age_adult 0.1465733 1.0000000
```

BMI and age are positively correlated. The correlation between BMI and age is much lower than the correlation found between expenditure and income seen previously in this chapter. Why do you think this is? Does this mean that BMI is not related to age?

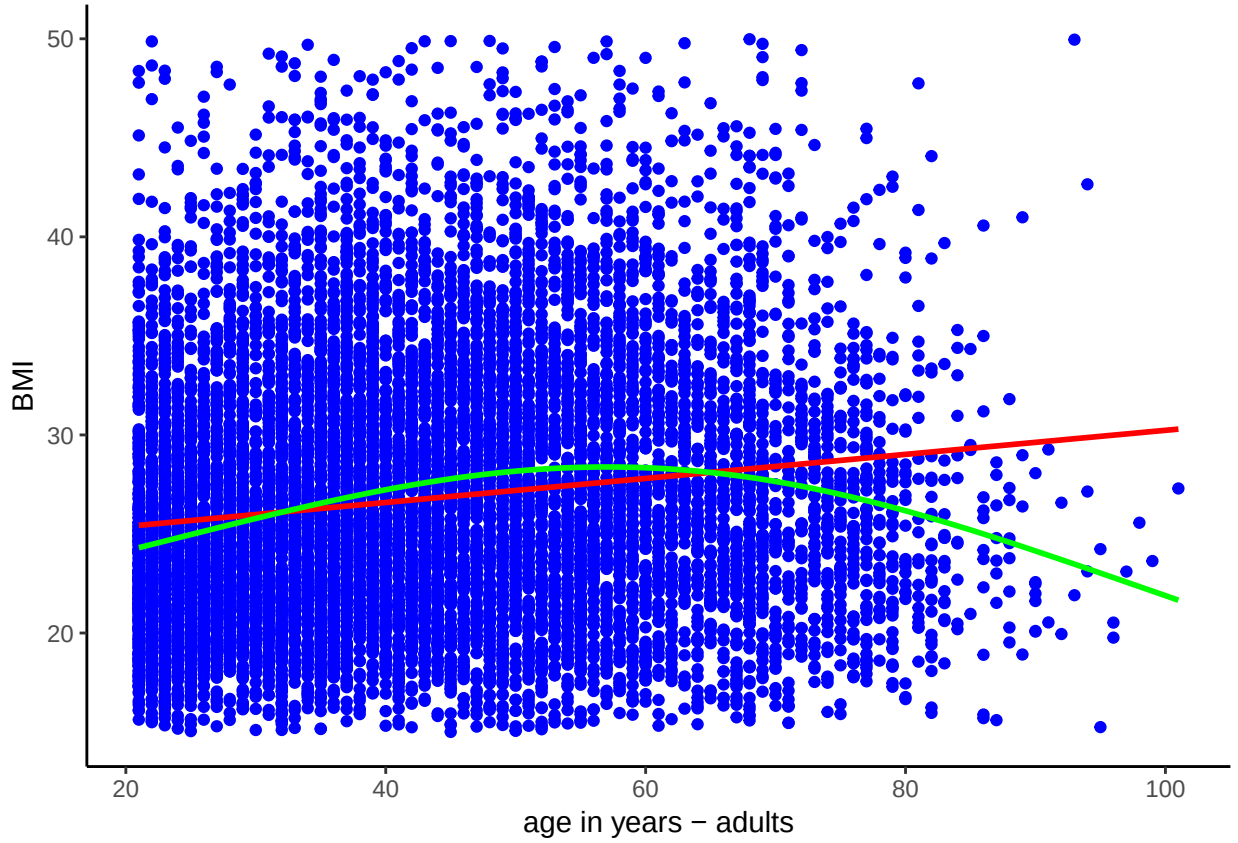
A simple scatterplot gives us the relationship between BMI and age over different values of BMI and age.

```
ggplot(corr2.df, aes(x = age_adult, y = bmi)) +
  geom_point(aes(x = age_adult, y = bmi), colour="blue") +
  stat_smooth(method = "lm", formula = y ~ x, se = FALSE, colour = "red") +
  stat_smooth(method = "gam", formula= y ~ s(x, k = 3), se = FALSE, colour = "green", aes(colour = "poly
  scale_x_continuous(breaks=seq(20,100,20), labels = comma) +
  scale_y_continuous(breaks=seq(10,50,10), labels = comma) +
  xlab("age in years - adults") + ylab("BMI") +
  theme_classic()
```

```
## Warning: Removed 20536 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 20536 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 20536 rows containing missing values (geom_point).
```



Observing the scatter, the relationship between BMI and age looks tenuous. However, the trend lines suggest that there is a nonlinear relationship between BMI and age; BMI increases with age until around age 50 and decreases thereafter. This means that adult's weight tends to increase until they are around 50 and thereafter starts to decrease. Does this make sense? Let's investigate this nonlinear trend.

```
bmi_age_df<-corr2.df%>%
  group_by(age_adult)%>%
  summarise(bmi_age = mean(bmi, na.rm = T)) #use mutate if you want to generate a new var as opposed to
```

This code assigns the mean BMI for individuals of a given age to each individual of that age. For example, if there are only 3 people of age 30 in the sample with BMI values of 20, 22 and 27, then they will all get assigned a bmi_age value of 23 (the mean of the three BMI values). To see the mean bmi by age, browse the following.

```
head(bmi_age_df, n=20L)
```

```
## # A tibble: 20 x 2
##    age_adult bmi_age
##        <int>   <dbl>
##  1        21    24.3
##  2        22    24.0
##  3        23    24.5
##  4        24    24.7
##  5        25    24.5
##  6        26    24.9
##  7        27    25.4
##  8        28    25.5
##  9        29    26.0
```

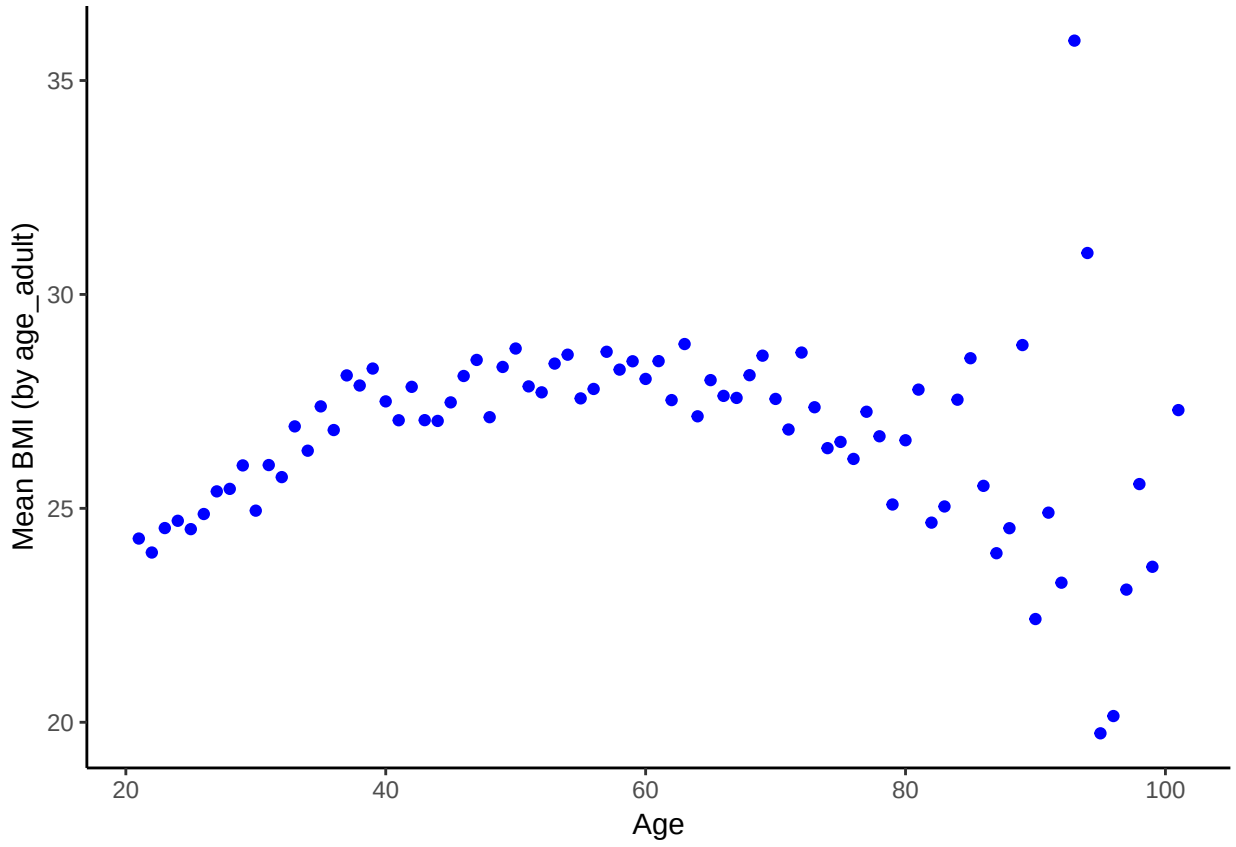
```
## 10        30    24.9
## 11        31    26.0
## 12        32    25.7
## 13        33    26.9
## 14        34    26.3
## 15        35    27.4
## 16        36    26.8
## 17        37    28.1
## 18        38    27.9
## 19        39    28.3
## 20        40    27.5
```

You can then see that all the 21 year olds in the sample were assigned a `bmi_age` value of 24.3. On average, 21 year olds in our sample have a BMI of 24.3. It will be much more interesting to see how this average `bmi_age` variable changes with age.

We can investigate this by plotting the mean bmi by age variable against age.

```
ggplot(bmi_age_df, aes(x = age_adult, y = bmi_age)) +
  geom_point(aes(x = age_adult, y = bmi_age), colour="blue") +
  xlab("Age") + ylab("Mean BMI (by age_adult)") +
  theme_classic()
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```



The nonlinear relationship between BMI and ages emerges clearly. BMI increases sharply with age between 20 and 40 years of age, stays relatively stable through the middle ages and then becomes more erratic at older ages, with a general downward trend. If your hypothesis about the relationship between BMI and age

was that BMI increases with age up until 60 and thereafter declines, the scatterplot supports your hypothesis. Part of the reason the relationship is less stable at older ages is that there are fewer individuals in these older groups.

The relationship between BMI and age might be different for men and women. How would you go about investigating this?

```
nids%>%
  group_by(age_bins, gender)%>%
  summarise(Mean = mean(bmi, na.rm=TRUE)) %>%
  na.omit() %>%
  arrange(gender, age_bins)
```

```
## # A tibble: 12 x 3
## # Groups:   age_bins [6]
##    age_bins     gender  Mean
##    <fct>        <fct>  <dbl>
##  1 20 - 29 yrs  Male    22.7
##  2 30 - 39 yrs  Male    23.7
##  3 40 - 49 yrs  Male    24.2
##  4 50 - 59 yrs  Male    25.5
##  5 60 - 69 yrs  Male    25.0
##  6 70 - 120 yrs Male    25.0
##  7 20 - 29 yrs  Female  26.8
##  8 30 - 39 yrs  Female  29.4
##  9 40 - 49 yrs  Female  30.5
## 10 50 - 59 yrs  Female  30.1
## 11 60 - 69 yrs  Female  30.4
## 12 70 - 120 yrs Female  29.0
```

The table suggests that the quadratic relationship between BMI and age is slightly stronger for females than males. Female mean BMI increases substantially between the 20s and 30s, increases a little further between the 30s and 40s, then remains fairly stable between the 40s and 60s and decreases thereafter. Male BMI increases more steadily until the 50s and decreases marginally thereafter. Is this supported when we use the correlation command?

- **Females: Below 60 years old**

```
fb60<-nids[nids$bmi_valid==1 & nids$age_adult < 60 & nids$w1_a_b2 == 2,c("bmi","age_adult")]
cor(fb60, use="complete.obs")
```

```
##                 bmi age_adult
## bmi       1.0000000 0.2135916
## age_adult 0.2135916 1.0000000
```

- **Females: Over 60 years old**

```
fo60<-nids[nids$bmi_valid==1 & nids$age_adult > 60 & nids$w1_a_b2 == 2,c("bmi","age_adult")]
cor(fo60, use="complete.obs")
```

```
##                  bmi  age_adult
## bmi        1.0000000 -0.1449253
## age_adult -0.1449253  1.0000000
```

- **Males: Below 60 years old**

```
mb60<-nids[nids$bmi_valid==1 & nids$age_adult < 60 & nids$w1_a_b2 == 1,c("bmi","age_adult")]
cor(mb60, use="complete.obs")
```

```
##                  bmi age_adult
## bmi       1.0000000 0.1897421
## age_adult 0.1897421 1.0000000
```

- **Males: Over 60 years old**

```
mo60<-nids[nids$bmi_valid==1 & nids$age_adult > 60 & nids$w1_a_b2 == 1,c("bmi","age_adult")]
cor(mo60, use="complete.obs")
```

```
##                   bmi   age_adult
## bmi        1.00000000 -0.08561469
## age_adult -0.08561469  1.00000000
```

Alternatively, in one pipe:

```
nids %>%
  filter(bmi_valid==1) %>%
  select(bmi,age_adult, gender) %>%
  ungroup() %>%
  mutate(age_adult_2=ifelse(age_adult < 60,1,2)) %>%
  mutate(age_adult_2=factor(age_adult_2, labels=c("Under 60", "Over 60"))) %>%
  group_by(age_adult_2,gender) %>%
  summarize(corr=cor(bmi, age_adult))
```

```
## Adding missing grouping variables: `hhid`
```

```
## # A tibble: 4 x 3
## # Groups:   age_adult_2 [?]
##   age_adult_2 gender    corr
##   <fct>       <fct>    <dbl>
## 1 Under 60    Male     0.190
## 2 Under 60    Female   0.214
## 3 Over 60     Male    -0.0697
## 4 Over 60     Female -0.145
```

What do these results tell us? They seem to further support the idea that the relationship between BMI and age is positive for younger adults and negative over the age of 60. They also suggest that relationship between age and BMI is stronger for women than men on both the positive and negative sections of the parabolic relationship.

We now have a fairly good idea of the general relationship between BMI and age. To gain a more precise understanding of this relationship we use regression. Regression allows us to quantify the relationship between the two variables. How large is the influence of age on BMI?

```
lm1 <- lm(bmi~age_adult, data = subset(nids,subset=bmi_valid==1, select=c(bmi,age_adult)))
summary(lm1)
```

```
##
## Call:
## lm(formula = bmi ~ age_adult, data = subset(nids, subset = bmi_valid ==
##     1, select = c(bmi, age_adult)))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.680  -4.791  -1.172   3.899  24.372
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 24.160418   0.180727  133.69   <2e-16 ***
```

```
## age_adult    0.060722   0.003974   15.28   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.554 on 10632 degrees of freedom
## Multiple R-squared:  0.02148,    Adjusted R-squared:  0.02139
## F-statistic: 233.4 on 1 and 10632 DF,  p-value: < 2.2e-16
```

What does this tell us? On average, a one year increase in age is associated with a 0.06 unit increase in BMI.

You will notice that the R-squared value is far lower than the one from the regression of expenditure on income. Why do you think this is? The relationship between expenditure and income is well defined - the more you have, the more you can spend. In general, relationships between variables are more complex. A low R-squared value just says that age only explains a small part of the variation in BMI values. Alternatively, it might be reflecting the fact that by running a regression we are assuming a linear relationship. As we have seen above, it is likely that the relationship is in fact non-linear.

Even if we only look at 21 - 40 year olds where we might expect a linear relationship between BMI and age, it is to be expected that age would only explain a relatively small part of the variation in BMI, as we showed that BMI varied between genders, as well as between racial and income groups and we have not accounted for these characteristics. Besides these factors, can you think of other factors that would explain variation in BMI? How about genetics? Education? Exercise? Quality of diet? We will only be able to account for these other factors when we move onto including multiple explanatory variables in our regressions in chapter 8.

Given what we have learnt from the summary measures about the relationship between BMI and age, is there any way we can improve on this regression? As we mentioned above, it is more appropriate to use regression where there is a linear relationship between the variables.

Therefore, we break our sample into three groups: between 20 and 40 years of age; between 40 and 60 years; and over 60 years. We then examine the relationship between BMI and age for each group.

```
lm2 <- lm(bmi~age_adult, data = subset(nids,subset=age_adult < 40 & bmi_valid==1, select=c(bmi,age_adul
lm3 <- lm(bmi~age_adult, data = subset(nids,subset=age_adult >=40 & age_adult <60 & bmi_valid == 1, sel
lm4 <- lm(bmi~age_adult, data = subset(nids,subset=age_adult >= 60 & bmi_valid == 1, select=c(bmi,age_a
```

Using the R package,`stargazer` (Hlavac 2018), to present the regression models:

```
library(stargazer)
stargazer(lm2,lm3,lm4, type="text", model.numbers = FALSE, column.labels = c("<Under 40","40-60","Over
```

```
##
## ===============================================================================================
##                                              Dependent variable:
##                          ---------------------------------------------------------------------
##                                                      bmi
##                                 <Under 40                40-60                  Over 60
## -----------------------------------------------------------------------------------------------
## age_adult                       0.225***               0.058***               -0.101***
##                                  (0.015)                (0.020)                (0.022)
##
## Constant                        19.179***              25.003***              34.485***
##                                  (0.444)                (0.999)                (1.541)
##
## -----------------------------------------------------------------------------------------------
## Observations                      5,199                  3,664                  1,771
## R2                                0.042                  0.002                  0.012
## Adjusted R2                       0.041                  0.002                  0.011
## Residual Std. Error     6.029 (df = 5197)      6.953 (df = 3662)      6.742 (df = 1769)
```

```
## F Statistic         225.338*** (df = 1; 5197) 8.205*** (df = 1; 3662) 20.916*** (df = 1; 1769)
## ================================================================================================
## Note:                                                                *p<0.1; **p<0.05; ***p<0.01
```

Does restricting the age range of our sample affect the coefficient on age? Does it affect the constant? What is the interpretation of these two values and do they support the pattern that we saw in the graphs above?

The tabulation showed that the relationship between BMI and age is different for men and women. We saw that for each age group, the average female BMI was above the average male BMI. However, it is unclear from the table whether BMI increases and then decreases with age at the same rate for men and women. In other words, if you were to plot the relationship between BMI and age for men and women on the same graph, would they have the same slope?

4. As a challenge, plot the relationship between BMI and age for men and women under 40 on the same graph. [Hint: Use a similar graph to one you plotted earlier in this chapter.]

Question 4 Answer

The graph showed that the female trendline had a steeper slope than the male trendline. BMI increases at a faster rate with age for women than for men. Let's use regression analysis to add number to the picture. How much faster does female BMI increase than male BMI between age 20 and 40.

```
lm5 <- lm(bmi~age_adult, data = subset(nids,subset=bmi_valid == 1 & age_adult < 40 & w1_a_b2 == 1, sele
summary(lm5)
```

```
##
## Call:
## lm(formula = bmi ~ age_adult, data = subset(nids, subset = bmi_valid ==
##     1 & age_adult < 40 & w1_a_b2 == 1, select = c(bmi, age_adult)))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.7623 -3.0057 -0.7909  1.8930 25.7992
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 19.09161    0.52813  36.149  < 2e-16 ***
## age_adult    0.13784    0.01788   7.709 1.96e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.461 on 2062 degrees of freedom
## Multiple R-squared:  0.02801,    Adjusted R-squared:  0.02754
## F-statistic: 59.42 on 1 and 2062 DF,  p-value: 1.965e-14
```

```
lm6 <- lm(bmi~age_adult, data = subset(nids,subset=bmi_valid == 1 & age_adult < 40 & w1_a_b2 == 2, sele
summary(lm6)
```

```
##
## Call:
## lm(formula = bmi ~ age_adult, data = subset(nids, subset = bmi_valid ==
##     1 & age_adult < 40 & w1_a_b2 == 2, select = c(bmi, age_adult)))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.0241  -4.4192  -0.8955   3.7965  24.3588
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 19.55102    0.59101   33.08   <2e-16 ***
## age_adult    0.27082    0.01989   13.62   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.288 on 3133 degrees of freedom
## Multiple R-squared:  0.05586,    Adjusted R-squared:  0.05556
## F-statistic: 185.4 on 1 and 3133 DF,  p-value: < 2.2e-16
```

Does our regression output agree with the graph in question 4? What are the 'slope' coefficients of the two regressions? How do they compare to one another?

The age coefficients of the regressions clearly show that BMI increases with age substantially faster for women than for men between the ages of 20 and 40.

Now that you have had a decent introduction to running simple linear regressions in R, try your hand at the exercises to make sure that you have fully grasped the important concepts as you will need to be very familiar with the material in this chapter before you move on to chapter 8.

7.9 Question answers:

QUESTION 1 Answer

What is the correlation between the average household's food expenditure and household size?

Both household size, w1_hhsizer, and total monthly household food expenditure w1_h_expf, are household-level variables. Since we do not want to allow larger households to be weighted more heavily, let's again use only one observation per household.

```
q1<-data.frame(subset(nids, subset=hhrestrict==1, select=c(w1_hhsizer,w1_h_expf)))
cor(q1[,c("w1_hhsizer","w1_h_expf")],use="complete.obs")
```

```
##            w1_hhsizer w1_h_expf
## w1_hhsizer  1.0000000 0.1452078
## w1_h_expf   0.1452078 1.0000000
```

From these results, we can conclude that there is a weak positive relationship between household size and expenditure on food. That is, larger households are somewhat more likely to spend more on food than smaller households. However, we cannot determine how much more without further analysis.

QUESTION 2 ANSWER

By how much would you expect a household to increase its total monthly food expenditure for every additional household member?

To figure this out, we need to use the following regress:

```
q2 <- lm(w1_h_expf~w1_hhsizer, data = subset(nids,subset=hhrestrict==1, select=c(w1_hhsizer,w1_h_expf)))
summary(q2)
```

```
##
## Call:
## lm(formula = w1_h_expf ~ w1_hhsizer, data = subset(nids, subset = hhrestrict ==
##     1, select = c(w1_hhsizer, w1_h_expf)))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -1209.1  -458.3  -238.3   179.6 13797.1
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  713.600     16.914   42.19   <2e-16 ***
## w1_hhsizer    45.711      3.645   12.54   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 798.9 on 7303 degrees of freedom
## Multiple R-squared:  0.02109,    Adjusted R-squared:  0.02095
## F-statistic: 157.3 on 1 and 7303 DF,  p-value: < 2.2e-16
```

This gives you a regression equation of:

$(expfhat) = 713.600 + 45.711 * (w1\_hhsizer)$

Therefore, for every additional member, the average household can be expected to increase food expenditure by about R45.71 per month. If we tried to interpret the constant term here, we would want to say that if the household had no members expenditure on food would be 713.6 rand! This of course does not make much sense. We generally only interpret the constant when it makes sense to do so. Notice also that our model has significant F and T - values (Prob>F = 0.0000 & P>|t|= 0.0000) although our R-squared value is very low; only 2% of the variation in monthly household food expenditure around its mean is explained by household size.

QUESTION 3 ANSWER

Income also affects households' total monthly food expenditure. Try a simple regression of w1_h_expf on total monthly household income. Write an equation for this relationship and interpret the result.

Try the regression as follows:

```
q3 <- lm(w1_h_expf~w1_hhincome, data = subset(nids,subset=hhrestrict==1, select=c(w1_hhincome,w1_h_expf)
summary(q3)
```

```
##
## Call:
## lm(formula = w1_h_expf ~ w1_hhincome, data = subset(nids, subset = hhrestrict ==
##     1, select = c(w1_hhincome, w1_h_expf)))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6088.0  -354.3  -140.7   184.6  9687.9
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.323e+02  8.848e+00   71.46   <2e-16 ***
## w1_hhincome 5.769e-02  9.616e-04   59.99   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 660.9 on 7303 degrees of freedom
## Multiple R-squared:  0.3301, Adjusted R-squared:   0.33
## F-statistic:  3599 on 1 and 7303 DF,  p-value: < 2.2e-16
```

```
print(q3)
```

```
##
## Call:
## lm(formula = w1_h_expf ~ w1_hhincome, data = subset(nids, subset = hhrestrict ==
##     1, select = c(w1_hhincome, w1_h_expf)))
##
## Coefficients:
## (Intercept)  w1_hhincome
##   632.32020      0.05769
```

The equation made by these coefficients equals:

$$(predicted w1\_h\_expf) = 632.3202 + .0576867 * (w1\_hhincome)$$

For every R1000 increase in total monthly household income, monthly household food expenditure increases by approximately R58. Here we can interpret the coefficient of R632.32 as the mean monthly expenditure on food when a household has no income (i.e. the minimum food expenditure needed to survive). The R-squared value implies that the amount of variance of monthly household food expenditure explained by total monthly household income is 33%.

QUESTION 4 - ANSWER

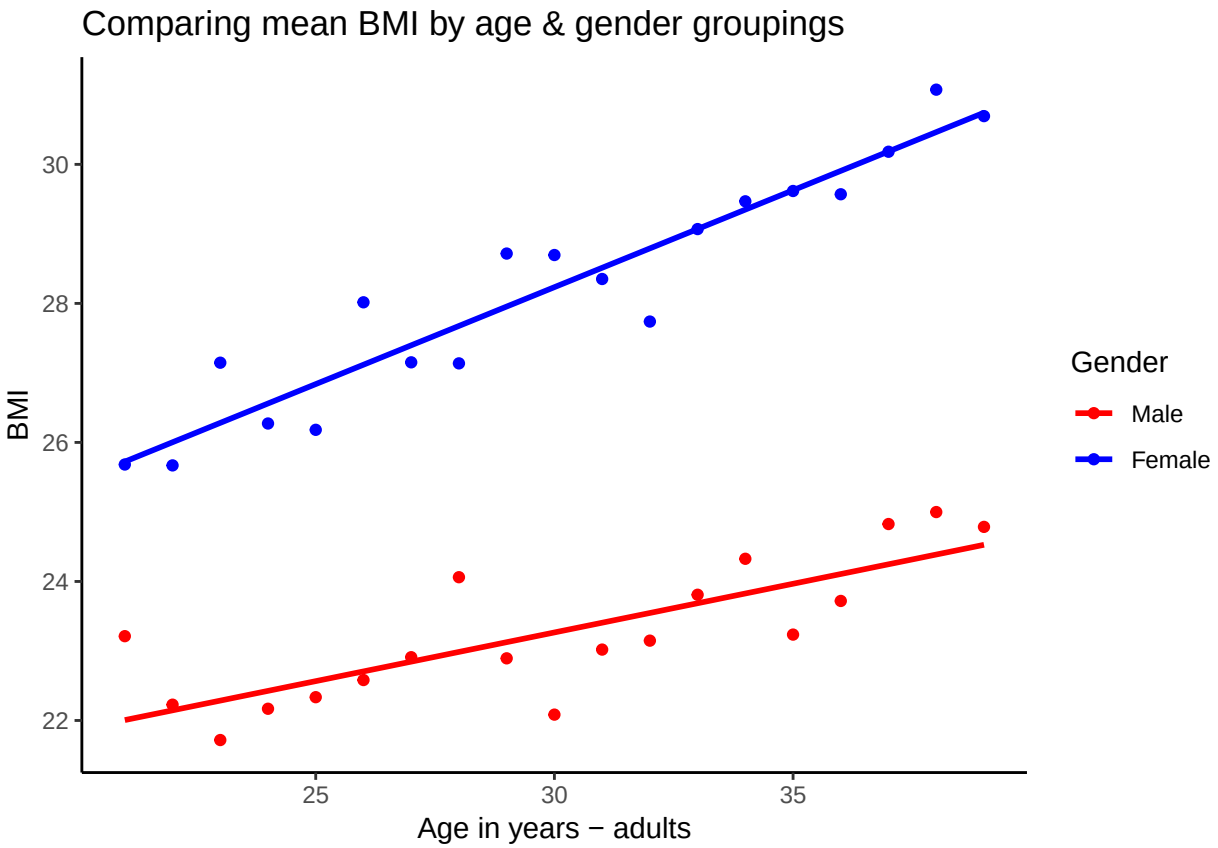
As a challenge plot the relationship between BMI and age for men and women under 40 on the same graph. [Hint: Use a similar graph to one you plotted earlier in this chapter.]

You can copy the following (including the comments) into a do file to run:

```
bmiage<-subset(nids, subset=(age_adult > 20 & age_adult < 40), select=c(bmi, age_adult, gender))

#Generating a BMI variable giving mean BMI by age and gender to individuals of a common age-gender grou
bmiage<-bmiage%>%
  group_by(age_adult, gender)%>%
  summarise(bmi_age_gen=mean(bmi, na.rm=TRUE))

ggplot(bmiage, aes(x = age_adult, y = bmi_age_gen, group = gender, color = gender)) +
  geom_point() +
  stat_smooth(method = "lm", se = FALSE) +
  scale_color_manual(values = c("red","blue")) +
  labs(x ="Age in years - adults", y  ="BMI", title = "Comparing mean BMI by age & gender groupings", c
  theme_classic()
```



From this graph it appears that the fitted line for men has a lower intercept and also a more gradual slope. This suggests that on average women have a higher BMI at age 21 and their BMI increases at a faster rate between age 21 and age 40.

7.10 Exercises

So now that we have learned quite a bit about regression analysis, it's time to put our knowledge to the test!

1. What is the correlation between where a person lives, their total monthly take home pay, and their associated total monthly household food expenditure?

Exercise 1 Answer

2. Generally speaking (with a normal labour supply curve) people are willing to work more when they get paid more (after tax). Investigate this relationship by regressing "hours worked last week" on "individual take home pay" Have you estimated a labour supply function?

Exercise 2 Answer

3. It is a reasonable expectation that the bigger a residence is, the more that it costs to live in it. Is this true in the NIDS dataset? If so, how much more will an additional room cost? Interpret the data with a regression and then show a graph. What would happen if we didn't condition on hhrestrict when graphing?

Exercise 3 Answer

4. Now, let's consider whether the number of people in a family will determine the size of the house that it lives in. If so, an additional person is likely to make a family acquire how many

more rooms (on average)? Please show the graph for this relationship.

Exercise 4 Answer

5. Eating out is considered to be a luxury. Is it reasonable to assume that as income goes up, money spent on eating out would go up too? See if this is the case in the NIDS

So now that we have learned quite a bit about regression analysis, it's time to put our knowledge to the test! 1. What is the correlation between where a person lives, their total monthly take home pay, and their associated total monthly household food expenditure?

Exercise 1 Answer

7.11 Exercise

So now that we have learned quite a bit about regression analysis, it's time to put our knowledge to the test!

1. What is the correlation between where a person lives, their total monthly take home pay, and their associated total monthly household food expenditure?

Exercise 1 Answer

2. Generally speaking (with a normal labour supply curve) people are willing to work more when they get paid more (after tax). Investigate this relationship by regressing "hours worked last week" on "individual take home pay" Have you estimated a labour supply function?

Exercise 2 Answer

3. It is a reasonable expectation that the bigger a residence is, the more that it costs to live in it. Is this true in the NIDS dataset? If so, how much more will an additional room cost? Interpret the data with a regression and then show a graph. What would happen if we didn't condition on hhrestrict when graphing?

Exercise 3 Answer

4. Now, let's consider whether the number of people in a family will determine the size of the house that it lives in. If so, an additional person is likely to make a family acquire how many more rooms (on average)? Please show the graph for this relationship.

Exercise 4 Answer

5. Eating out is considered to be a luxury. Is it reasonable to assume that as income goes up, money spent on eating out would go up too? See if this is the case in the NIDS dataset. If so, for every rand a family earns, how much is likely to be spent on eating out?

Exercise 5 Answer

7.12 Exercise answers

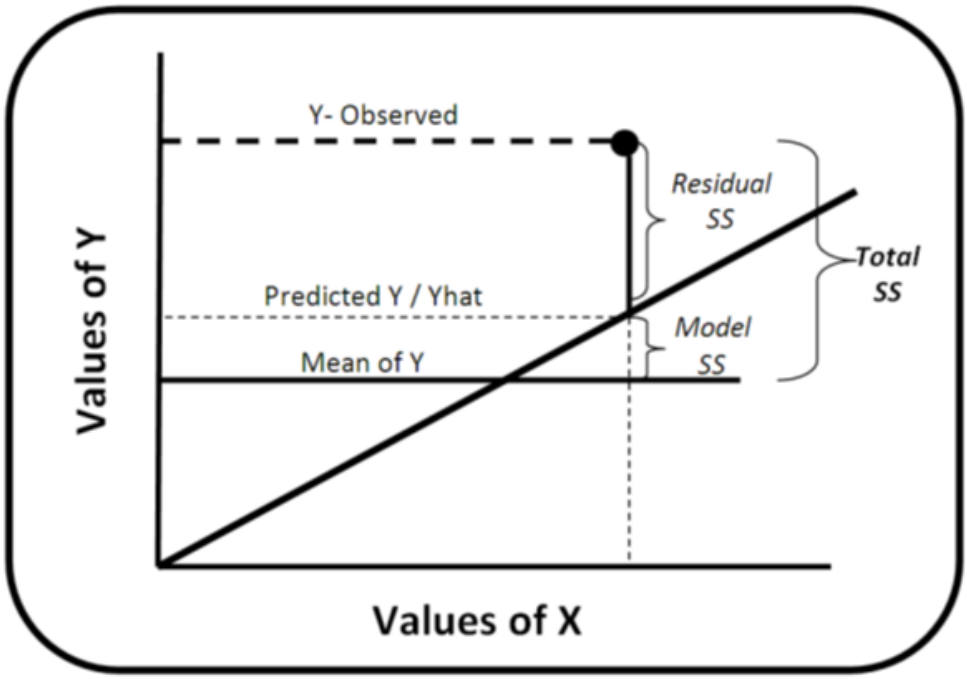
7.13 Appendix A: Analysis of variance (anova) table

```
anova(lm0)
```

```
## Analysis of Variance Table
##
## Response: w1_h_expenditure
##               Df     Sum Sq    Mean Sq F value    Pr(>F)
```

```
## w1_hhincome    1 1.7838e+11 1.7838e+11    6612 < 2.2e-16 ***
## Residuals   7303 1.9702e+11 2.6978e+07
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Although we are not particularly concerned with ANOVA tables in this chapter, it is nice to know the following definitions. SS (`Sum Sq`) stands for the sum-of-squares. The "Total" sum-of-squares (TSS) is the sum of the squared distances between each observation point (Y-observed) and the mean value of Y (remember in this case our Y is `w1_h_expenditure`). The "Model" (`w1_hhincome`) sum-of-squares (also sometimes called "Explained" sum-of-squares) is the sum of the squared distances between the mean value of Y and the predicted values of Y generated by the "best-fit" line (also known as Yhat). The `Residuals` sum-of-squares is the sum of the squared distances between each Y- observation point and our Yhat's. Thus "Model"SS + "Resdiduals" SS = "Total" SS (or alternatively MSS + RSS = TSS).



Notice also that `Df` stands for degrees of freedom, which is number of observations in the sample minus the number of independent constraints. The number of degrees of freedom serves as a flexibility measure in the t distribution. `Mean Sq` is the mean of squares or simply the relevant SS divided by the corresponding df. The `F value` and `Pr(>F)` will be explained in Appendix B

Back to Understanding Regression Output Tables

7.14 Appendix B: Further understanding the regression results table

```
summary(lm0)
```

```
##
## Call:
## lm(formula = w1_h_expenditure ~ w1_hhincome, data = nids2)
##
```

```
## Residuals:
##    Min     1Q Median     3Q    Max
## -60739  -1231   -758     53 108202
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.115e+03  6.954e+01   16.03   <2e-16 ***
## w1_hhincome 6.146e-01  7.558e-03   81.31   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5194 on 7303 degrees of freedom
## Multiple R-squared:  0.4752, Adjusted R-squared:  0.4751
## F-statistic:  6612 on 1 and 7303 DF,  p-value: < 2.2e-16
```

Meaning behind each section of `summary()`

`Call`: This is an R feature that shows what function and parameters were used to create the model.

`Residuals`: Difference between what the model predicted and the actual value of y. This section provides a summary of the residuals - `Min`, `1Q`, `Median`,`3Q` and `Max`

Coefficients: These are the weights that minimize the sum of the square of the errors.

Under `coefficients`, we have the following:

`Estimate`: these are the estimated weights or coeficient for each covariate. In our case, for `w1_hhincome` and the constant (`(Intercept)`)

`Std. Error` is residual standard error divided by the square root of the sum of the square of that particular x variable.

`t value`: is the value on the t distribution associated with our coefficients. That is `Estimate` divided by `Std. Error`.

`Pr(>|t|)`: is the probability that our statistic is greater than the critical value of 1.96. Look up your t value in a T distribution table with the given degrees of freedom. For both cases (`w1_hhincome` and `(Intercept)`), the probability that these coefficients are not different from the null hypothesis of zero, is basically none.

`Residual Standard Error`: standard error of the residuals. It is a measure of the quality of a linear regression fit.

`Multiple R-Squared`: is a measure of how much of the variance around Y does X account for. Because the lowest possible R-squared is 0.0 and the highest is 1.0, our R-squared of 0.4751 means that we are explaining just over 47% of the variance in total monthly household expenditures by using total monthly household income.

`Adjusted R-Squared`: is just the R-squared adjusted by the degrees of freedom(We use this measure instead of `Multiple R-squared` when we have more than one independent variable)

`F-Statistic`: stands for F distribution and it used test the overall statistical significance of the regression.

Back to Understanding Regression Output Tables

7.15 Session information

```
print(sessionInfo(), locale = FALSE)
```

```
## R version 3.5.1 (2018-07-02)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 7 x64 (build 7601) Service Pack 1
##
## Matrix products: default
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] stargazer_5.2.2 ggalt_0.4.0     scales_0.5.0    bindrcpp_0.2.2
##  [5] forcats_0.3.0   stringr_1.3.1   dplyr_0.7.6     purrr_0.2.5
##  [9] readr_1.1.1     tidyr_0.8.1     tibble_1.4.2    ggplot2_3.0.0
## [13] tidyverse_1.2.1 foreign_0.8-70
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.17       lubridate_1.7.4    lattice_0.20-35
##  [4] assertthat_0.2.0   rprojroot_1.3-2    digest_0.6.15
##  [7] proj4_1.0-8        psych_1.8.4        utf8_1.1.4
## [10] R6_2.2.2           cellranger_1.1.0   plyr_1.8.4
## [13] backports_1.1.2    evaluate_0.10.1    httr_1.3.1
## [16] pillar_1.2.3       rlang_0.2.1        lazyeval_0.2.1
## [19] readxl_1.1.0       rstudioapi_0.7     extrafontdb_1.0
## [22] Matrix_1.2-14      rmarkdown_1.10     labeling_0.3
## [25] extrafont_0.17     munsell_0.5.0      broom_0.4.5
## [28] compiler_3.5.1     modelr_0.1.2       xfun_0.2
## [31] pkgconfig_2.0.1    mnormt_1.5-5       mgcv_1.8-24
## [34] htmltools_0.3.6    tidyselect_0.2.4   bookdown_0.7
## [37] crayon_1.3.4       withr_2.1.2        MASS_7.3-50
## [40] grid_3.5.1         nlme_3.1-137       jsonlite_1.5
## [43] Rttf2pt1_1.3.7     gtable_0.2.0       magrittr_1.5
## [46] KernSmooth_2.23-15 cli_1.0.0          stringi_1.1.7
## [49] reshape2_1.4.3     xml2_1.2.0         ash_1.0-15
## [52] RColorBrewer_1.1-2 tools_3.5.1        glue_1.2.0
## [55] maps_3.3.0         hms_0.4.2          parallel_3.5.1
## [58] yaml_2.1.19        colorspace_1.3-2   rvest_0.3.2
## [61] knitr_1.20         bindr_0.1.1        haven_1.1.2
```

Chapter 8

Multiple regression analysis

8.1 Getting ready

In the previous chapters we generated some variables and ran a few commands that will influence the results that we get in this chapter. If you are starting a new session of R, please copy the following lines of code into an R-script and then run it before you begin the chapter. Make sure that you remember what each line of code is doing.

```
library(foreign)
library(tidyverse)
```

```
## -- Attaching packages ------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

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

```
nids<-read.dta("./data/nids.dta", convert.factors=FALSE)

nids<-nids%>%
  arrange(hhid, pid)%>%
  group_by(hhid) %>%
  mutate(hhrestrict = 1:n()) %>%
  mutate(hhrestrict = ifelse(hhrestrict==1,1,0))

#Education
nids$educ.new<-nids$w1_best_edu
nids$educ.new[which(nids$w1_best_edu == 25)]<-0
nids$educ.new[which(nids$w1_best_edu <0)]<-NA
nids$educ.new[which(nids$w1_best_edu == 24)]<-NA
nids$educ.new[which(nids$w1_best_edu == 13 | nids$w1_best_edu == 16)]<-10
nids$educ.new[which(nids$w1_best_edu == 14 | nids$w1_best_edu == 17)]<-11
nids$educ.new[which(nids$w1_best_edu == 15)]<-12
nids$educ.new[which(nids$w1_best_edu == 18)]<-13
nids$educ.new[which(nids$w1_best_edu == 18)]<-13
```

```
nids$educ.new[which(nids$w1_best_edu == 19)]<-14
nids$educ.new[which(nids$w1_best_edu == 20)]<-15
nids$educ.new[which(nids$w1_best_edu == 21 | nids$w1_best_edu == 22)]<-16
nids$educ.new[which(nids$w1_best_edu == 23)]<-17

#Age
nids$age<-nids$w1_r_best_age_yrs[!is.na(nids$w1_r_best_age_yrs)]

#Rename
nids <- nids%>%
  mutate(race = w1_best_race,
         age = w1_r_best_age_yrs,
         gender = w1_r_b4,
         province = w1_hhprov,
         hhincome = w1_hhincome) %>%
  mutate(gender = factor(gender, levels = 1:2, labels = c("Male", "Female")),
          race = factor(race, levels = 1:4, labels = c("African", "Coloured","Asian", "White")),
          province = factor(province, levels=1:9, labels = c("Western Cape","Eastern Cape","Northern Cap
          w1_hhgeo = factor(w1_hhgeo, levels = 1:4, labels = c("Rural formal", "Tribal authority areas",
```

8.2 Introduction

In *Chapter 7*, we learned about simple bivariate regression. Now, it is time to move on to the more complex, but also more exciting multiple regression.

Let's quickly review what we know about simple regression analysis. In its general form, the simple linear regression model has one independent variable (X) and one dependent variable (Y). In multiple regression, the dependent variable Y is assumed to be a function of a set of K independent variables - X_1, X_2, X_3, ..., X_k. This yields a new regression equation - an extension of the one in *Chapter 7*:

$Y = a + b_1X_1 + b_2X_2 + ... + b_kX_k$

As with the simple regression equation, the interpretation of each of these coefficients is straight forward. Each "b" is a partial slope coefficient. Put differently, each "b" coefficient is the slope between that particular independent variable X and the dependent variable Y when all other independent variables in the model are equal to zero, or "held constant." For example, the b1 coefficient refers to the slope between X1 and the dependent variable Y when all other variables in the equation, X2, X3, etc., equal zero. Similarly, the value for b2 is the slope for the relationship between X2 and the dependent variable Y, when all other variables, X1, X3, etc., are equal to zero. As in Chapter 7, the "a" refers to the intercept, also known as the constant. This value is the value of predicted Y (i.e. yhat) when all of the independent variables, X1, X2, X3, etc., are equal to zero. Thus, multiple-regression allows us to state relationships between two main variables while controlling for other factors - also known as partial effects.

It should be obvious how useful this approach can be for quantitative social researchers, since we are often interested in social phenomena that go beyond a basic bivariate relationship. As mentioned in the previous chapter, we might be interested in whether the relationship between total monthly household income and total monthly household expenditures vary by rural setting. Or perhaps the relationship is not a matter of household income, but rather of how many household members are present in the home. All of these types of interests require multiple regression. This new approach will allow us to investigate the initial relationship while controlling for a 3rd, a 4th, or even many more factors.

Therefore, for this chapter we will investigate in depth, the relationship between income (`w1_fwag`) and education (`w1_best_edu`). In particular, we are hypothesizing that the amount of income earned by any individual is dependent upon their individual level of education. First, we will need to recode the education

variable into linear form. In Chapter 5 we had to recode the original `w1_best_edu` variable into the linear variable `educ_new`. We have therefore including the code in the 'getting ready' of this chapter. If you need to go back to review the explanation behind this coding click here.

We should always check that our recoding worked. So let's type:

```
nids%>%
  group_by(educ.new)%>%
  summarise(n=n())
```

```
## # A tibble: 19 x 2
##    educ.new     n
##       <dbl> <int>
##  1        0  6655
##  2        1   883
##  3        2   905
##  4        3  1175
##  5        4  1340
##  6        5  1381
##  7        6  1531
##  8        7  1766
##  9        8  1782
## 10        9  1701
## 11       10  2270
## 12       11  1821
## 13       12  2806
## 14       13   407
## 15       14   582
## 16       15   158
## 17       16   145
## 18       17    67
## 19       NA  3795
```

When running a regression we will often only want to consider a certain subset of observations. For instance in our regression we want to look at returns to education but not everyone in the population is of working age, thus the returns we are interested in only apply to a certain group in the population. In our context we choose to limit our sample to people between the ages of 25 to 60. A lower bound of 25 allows us to assume that individuals have completed their formal education while an upper bound of 60 was slightly below the official retirement age in 2008.

There are a few ways to enforce this restriction in our data. One option would be to subset: `subset= (age >= 25 & age <= 60)` or `[age >= 25 & age <= 60]`, but this of course can be rather tedious! A much more useful way, is to create a dummy variable that marks a subsample from the dataset and then subset on that variable.

```
nids<-nids %>%
  mutate(sample1=ifelse(age>=25 & age<=60,1,0))
```

This command creates a dummy variable called sample1 which has a value of one wherever age is between 25 and 60 and a value of 0 otherwise. Thus, to restrict our attention to the subsample of interest we simply have to subset/filter on `sample1==1` (we shall use this third and final option for the remainder of this chapter).

Everything looks good so far. Now, as we have done in the past, we want to do a cursory check of the relationship between our variables using the `cor` command. A check like this also helps us to pick up if we made a mistake in our code. For instance, if we find a negative relationship between education and income (meaning that more education was associated with lower wages), one of the first explanations we might investigate is whether we made a mistake in our coding.

Let's type:

```
cor(subset(nids, subset=sample1==1, select=c(w1_fwag, educ.new)), use="complete.obs")
```

```
##            w1_fwag  educ.new
## w1_fwag  1.0000000 0.3953829
## educ.new 0.3953829 1.0000000
```

The correlation between `educ_new` and `w1_fwag` is 0.3954, so education level is moderately associated with income earned. With a correlation, however, we do not know to what extent education makes a difference; we just know that it is positively associated with income. To further understand this relationship, we need to estimate the regression of income on education.

We accomplish this by typing:

```
lm <- lm(w1_fwag~educ.new, data = nids %>% filter(sample1==1))
summary(lm)
```

```
##
## Call:
## lm(formula = w1_fwag ~ educ.new, data = nids %>% filter(sample1 ==
##     1))
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -6446  -2399  -1029   1077  84077
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1241.14     198.00  -6.268 4.06e-10 ***
## educ.new      511.71      19.46  26.290  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5066 on 3730 degrees of freedom
##   (7993 observations deleted due to missingness)
## Multiple R-squared:  0.1563, Adjusted R-squared:  0.1561
## F-statistic: 691.1 on 1 and 3730 DF,  p-value: < 2.2e-16
```

$Y = a + bX$

In our case, this equation becomes:

$(predictedw1\_fwag) = -1241.1 + 511.7 * (educ.new)$

We can immediately interpret the slope coefficient for `educ.new` as the number of Rands (511.7) that average monthly take home pay would increase by for every additional year of education (`educ.new`). Judging from the t-value (26.29), we can tell that the coefficient is significant.

The constant, as discussed before, reflects the value of the dependent variable Y when the independent variables are equal to zero. While this property is technically useful in the calculation of the regression coefficients and calculation of predicted Y values, its actual value is not always of use. Obviously we do not want to ignore it, but we also do not need to dwell on it since it is often not easily interpretable. In our current case, it literally says that when education level is zero, predicted income is -1241.135 Rand. If, however, we had centered our education variable on the sample's mean education, then the "zero" value would actually be the average level of education. Interpreting the constant in that case would be more useful.

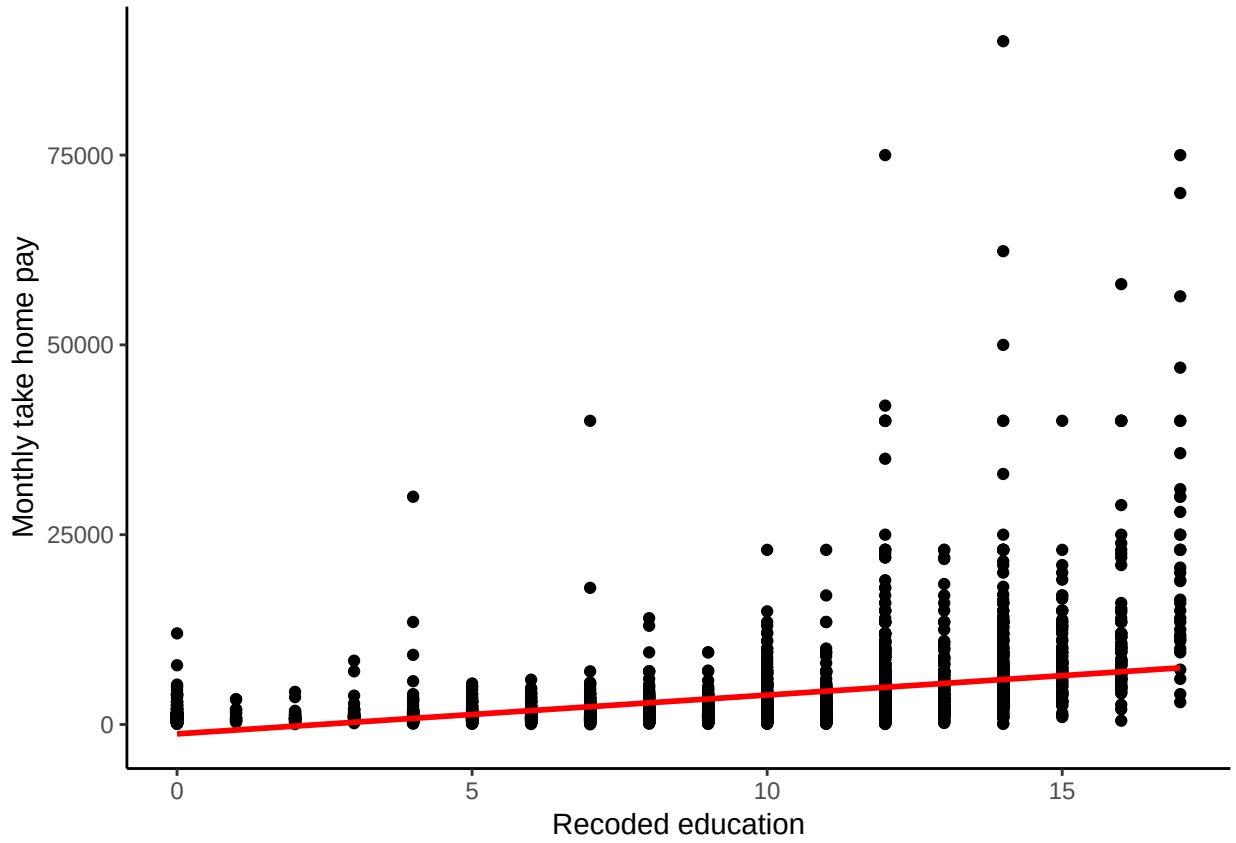
Moving along, the R-squared for this regression tells us that education accounts for almost 16% of the variation around the mean of income. Another way to think about it is if we were asked to guess at random

the income for an individual in a population sample, our guess would improve by approximately 16% if we knew the education level of the individual instead of just knowing the mean income of the sample.

Let's now try graphing the regression equation.

```
scatter<-data.frame(subset(nids, subset=sample1==1, select=c(educ.new, w1_fwag)))
scatter<-na.omit(scatter)

ggplot(scatter, aes(x = educ.new, y = w1_fwag)) +
  geom_point() +
  stat_smooth(method = "lm", se = FALSE, colour = "red") +
  xlab("Recoded education") +
  ylab("Monthly take home pay") +
  theme_classic()
```

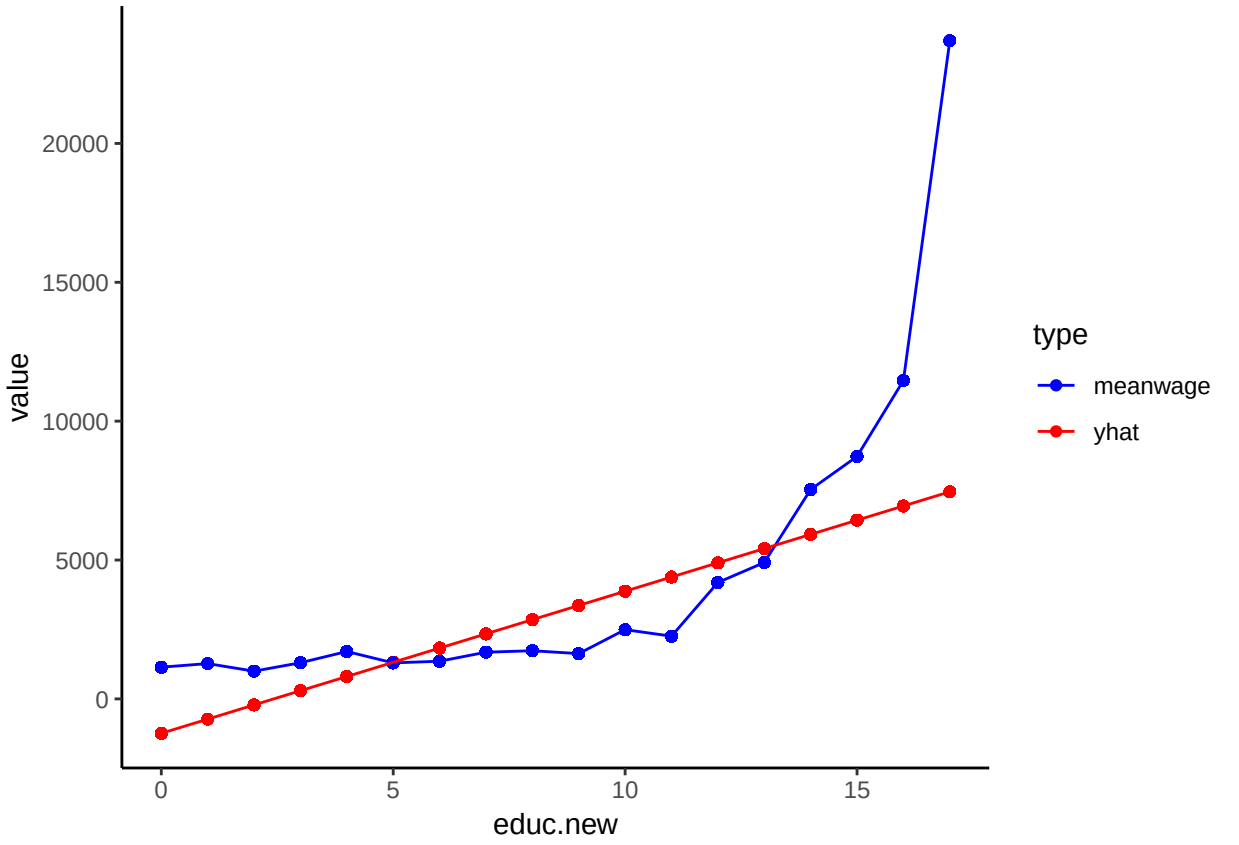


Is our regression line a good fit? It is rather hard to tell from the graph above, we have too many data points at each education level to be able to tell how well the line fits these observations! What if we compared our line to the mean wage at each education level? This should give us a better indication of the fit.

First we generate a `meanwage` variable and then we can plot the graph::

```
scatter$yhat<-lm$fitted.values
scatter%>%
  group_by(educ.new)%>%
  mutate(meanwage=mean(w1_fwag))%>%
  gather(key = type,value=value, -educ.new, -w1_fwag)%>% #reshaping the data using tidyr functions
  ggplot(., aes(educ.new,value, color = type)) +
  geom_point() +
```

```
  geom_line() +
  scale_color_manual(values = c("blue","red"))+
  theme_classic()
```



This graph is much easier to evaluate: the blue line is the mean wage across education while the red line plots our fitted/estimated values. The fundamental problem that we see in this graph is that we have fitted a straight line while the relationship between education and wage seems non-linear! We will correct for this non-linear relationship later in this chapter, but for now, we will stick with the assumption that we have a linear relationship between education and wage.

Issues of Parsimony vs. Saturation

When thinking about introducing variables into a model, it is important to keep the notions of parsimony and saturation in mind. That is, we should always strive to include only the variables that make sense and that are efficient at capturing the desired social phenomenon. Model building is often a balancing act between parsimony and saturation. When we say that a model is "saturated," we mean that the model has too many variables - it is over specified. Overspecification of a model can have an adverse effect on the results. Therefore, when selecting variables for a model, it is prudent to only include the most necessary variables or risk over specifying the model. With that in mind, let's proceed.

Introducing a Third Variable

At this point, we can consider including our first control variable. It is likely that the amount of income earned by any one person, is not only dependent on their years of education, but also on their age. By including age in our model, we acknowledge that income is also a function of age. It is important to include this factor because most people accumulate not only life experience as they age, but also work experience and skills, thus making them more likely to earn a higher wage.

If you remember our earlier discussion on how to interpret coefficients, each coefficient in a regression model is a partial effect, meaning that the coefficient reflects the effect of a given variable while controlling for the others. In this case it means that when we include `age`, our coefficient for `educ.new` will be the effect of education while controlling for `age`. Let's try running the multiple regression model now:

```
lm2 <- lm(w1_fwag~age+educ.new, data = nids %>% filter(sample1==1))
summary(lm2)
```

```
##
## Call:
## lm(formula = w1_fwag ~ age + educ.new, data = nids %>% filter(sample1 ==
##     1))
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -7908  -2304   -968    915  83784
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6804.60     447.31  -15.21   <2e-16 ***
## age           123.42       8.95   13.79   <2e-16 ***
## educ.new      586.22      19.74   29.69   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4943 on 3729 degrees of freedom
##   (7993 observations deleted due to missingness)
## Multiple R-squared:  0.1973, Adjusted R-squared:  0.1968
## F-statistic: 458.2 on 2 and 3729 DF,  p-value: < 2.2e-16
```

Compare our old equation (from above):

$(predictedw1\_fwag) = -1241.1 + 511.7(educ\_new)$

$\rightarrow R - squared = 0.1563$

To our new multiple regression equation:

$(predictedw1\_fwag) = -6804.60 + 586.22 * (educ\_new) + 123.4208 * (age)$

$\rightarrow R - squared = 0.1968$

Right away you should notice the effect that age has on our model. With age included, there is an increase in the coefficient of education (586.22), up from 511.71. Therefore, after controlling for age, on average every additional year of education produces an additional R586.22 per month of income. In addition to this, people tend to earn R123.42 more for every year that they age. Another way of thinking about these new results is that in the initial model, the "true" effect of `educ.new` was being masked by the effect of age.

The addition of a single regressor to the bivariate model probably does not seem that difficult, but as we progress in this chapter, you will realize that this is merely the tip of the iceberg. Now that you have been introduced to multiple regression, try the following two exercises:

1. It is possible that income as well as the number of household members predict the amount of money a household spends on clothing. Run a regression to see if this hypothesis finds support in our data.

Question 1 Answer

2. Among people who reported working, to what extent does the number of hours worked in a week predict a person's monthly income? Without running any further regressions, what

other variables might also help predict a person's monthly income?

Question 2 Answer

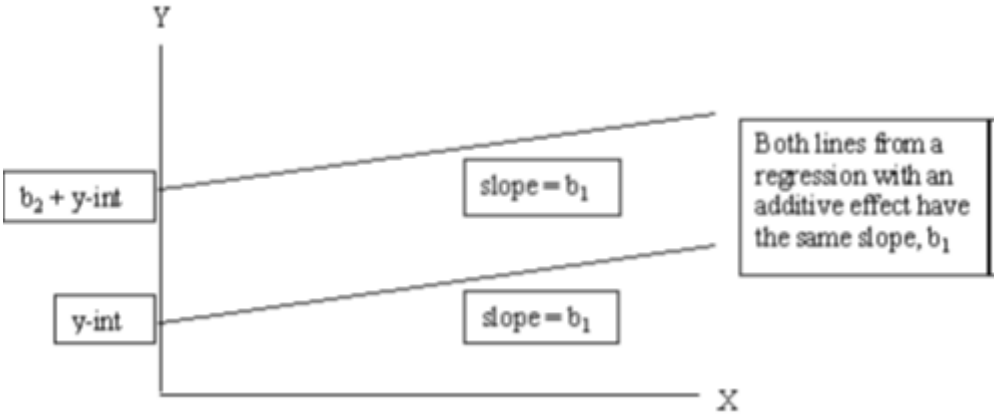
8.3 Dummy variables

8.4 Interactions with dummy Variables

Thus far, we have only dealt with the additive effects of dummy variables. That is, the assumption has been that for each independent variable Xi, the amount of change in our dependent variable Y is the same regardless of the values of the other independent variables in the equation. That is to say, if the mean income is higher for males than for females, this is so regardless of race. What if you have a situation where White males earn more than White females, but African women and men tend to earn the same amount? This assumption that there is no interaction between our variables allows us to interpret the partial coefficients as the effect of a variable while controlling for the other independent variables in the model.

The additive assumption, however, does not always hold. As we mentioned above, you may have an interaction between gender and race, where gender differences in income are different for different racial groups. In other words there may be interaction between the categorical variables `gender` and `w1_best_race`. More generally, when results indicate a statistically significant interaction effect between gender and race, the data suggests that being an "African Female" or a "White Male", or any other combination of race and gender, is qualitatively different from being in both race and gender categories independently.

We can illustrate what we mean by the additive effect of dummy variables in regression with the graphs below. Let's assume we have a regression consisting of three variables: income is the dependent variable (Y) and age (X1) and gender(X2) are the dependent variables. Now, the two lines in the graph below give the relationship between income and age for men (top line) and women (bottom line). That is, because men on average earn more than women we would expect to find predicted regression lines like the ones below. However, notice that these lines are parallel with slope b1 (coefficient of age) and that we assumed in the regression that holding age constant men earn b2 Rand more than women.



This graph is not merely theoretical. We can easily reproduce it using our data:

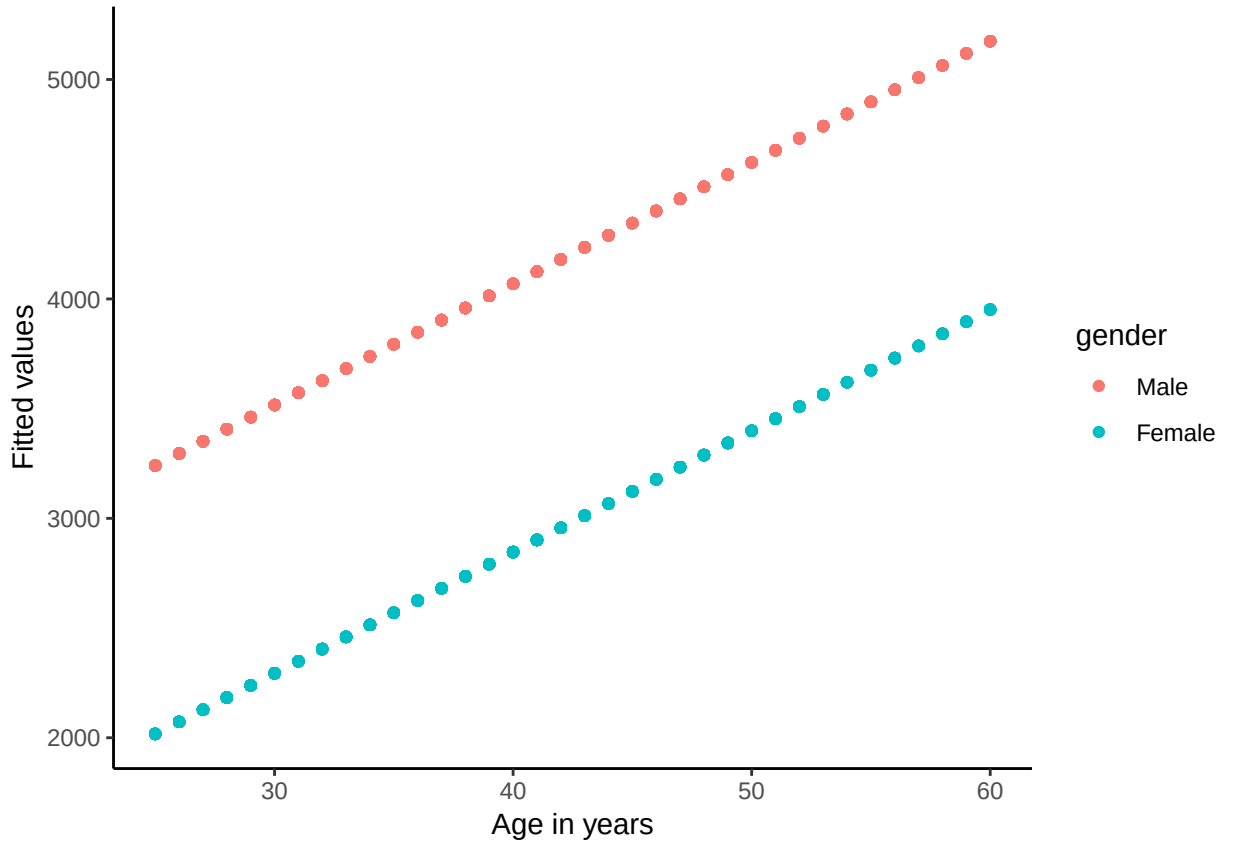
```
dum<-nids %>%
  filter(sample1==1) %>%
  select(hhid, pid, hhrestrict, w1_best_race, age, educ.new, w1_fwag, w1_r_b4, gender, race) %>%
  na.omit()

lmx <- lm(w1_fwag~age+gender, data = dum)
summary(lmx)
```

```
##
## Call:
## lm(formula = w1_fwag ~ age + gender, data = dum)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -4812  -2454  -1517    373  85986
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1858.95     390.89   4.756 2.05e-06 ***
## age             55.26       9.55   5.786 7.79e-09 ***
## genderFemale -1222.89     180.21  -6.786 1.34e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5465 on 3725 degrees of freedom
## Multiple R-squared:  0.01963,    Adjusted R-squared:  0.01911
## F-statistic:  37.3 on 2 and 3725 DF,  p-value: < 2.2e-16
```

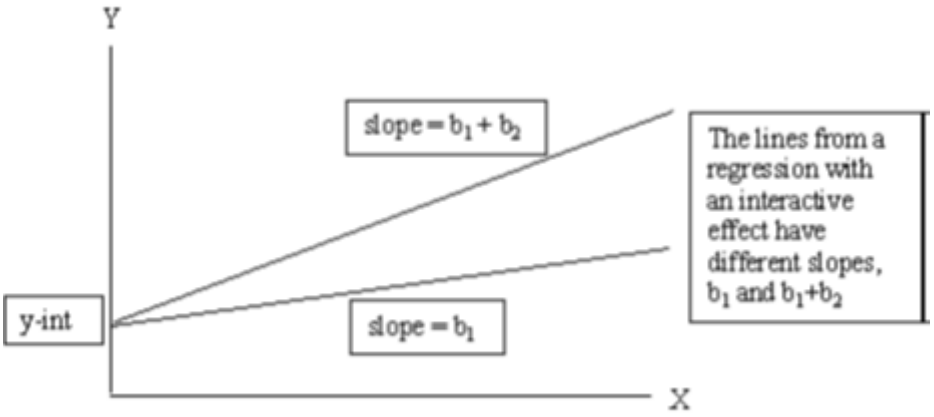
```
dum$prediction<-lmx$fitted.values

ggplot(dum, aes(x = age, y = prediction, color=gender))+
geom_point() +
xlab("Age in years") +
  ylab("Fitted values") +
  theme_classic()
```



It is important to realize that these two lines are parallel by construction. No matter what data we have, these lines are forced to be parallel by the way in which we include gender and age separately in our regression. At the risk of sounding repetitive, we force the difference in income between men and women to be exactly the same, after we have controlled for the other variables included in the regression (here only age). If we want to allow for this relationship to vary, we must use interaction terms.

In the second graph below, we find a hypothetical interaction effect between age and gender. That is, the effect of age (slope of the line) on income depends on the gender of the individual. In this case, we find that the upper-most line on the graph has a steeper slope than the line below it, thus the effect of gender depends on the value of Xi – in this case, the gender of the individual. This is saying that we might have a situation where young men and women tend to earn similar amounts, but as people get older men's income tend to increase by a greater amount every year than women's income.



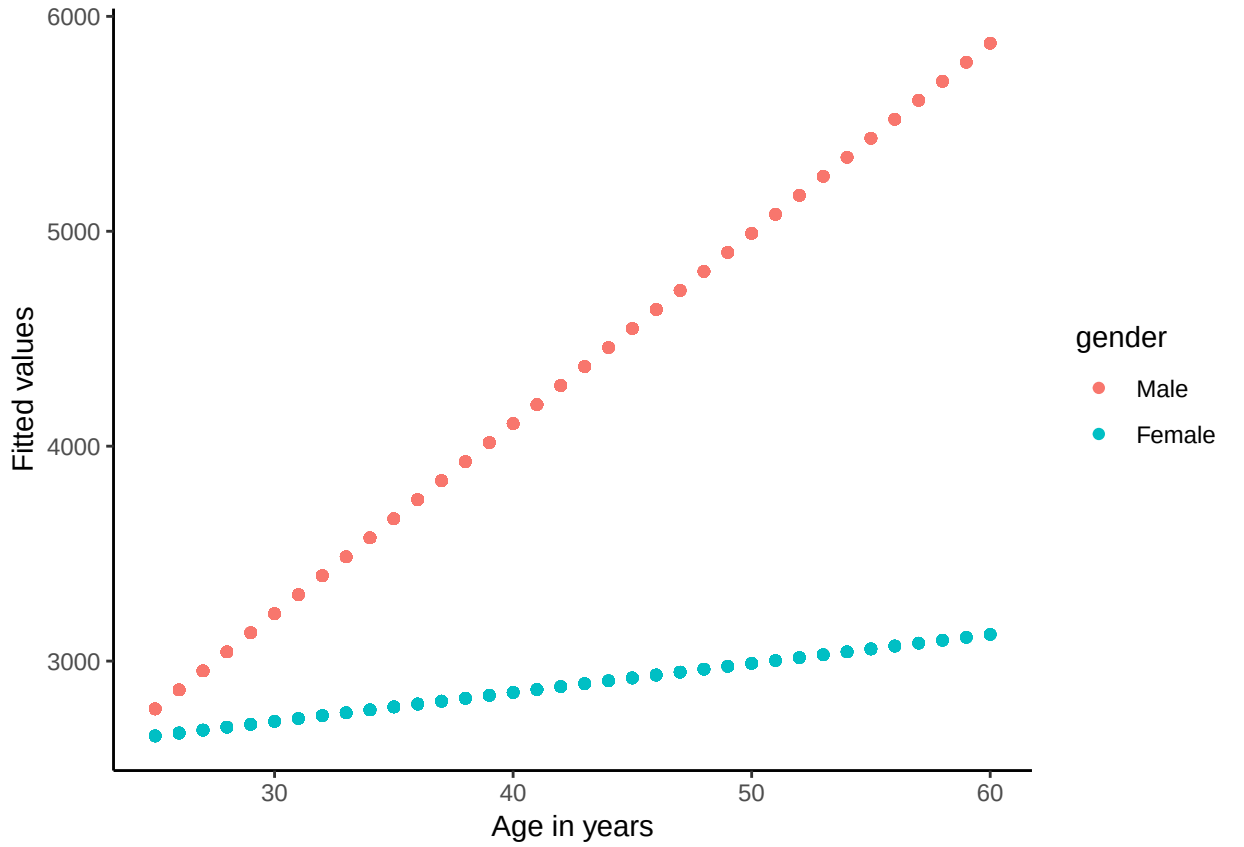
The above graph is simply a hypothetical example. Do you think that this relationship will be reflected in the data? Let's see:

```
lmz <- lm(w1_fwag~age*gender, data = dum)
summary(lmz)
```

```
##
## Call:
## lm(formula = w1_fwag ~ age * gender, data = dum)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -5479  -2303  -1548    351  85983
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         565.20     511.62   1.105   0.2693
## age                  88.50      12.77   6.928 5.00e-12 ***
## genderFemale       1749.25     781.31   2.239   0.0252 *
## age:genderFemale    -75.00      19.19  -3.909 9.43e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5454 on 3724 degrees of freedom
## Multiple R-squared:  0.02364,    Adjusted R-squared:  0.02285
## F-statistic: 30.06 on 3 and 3724 DF,  p-value: < 2.2e-16
```

```
dum$prediction<-lmz$fitted.values

ggplot(dum, aes(x = age, y = prediction, color=gender)) +
  geom_point() +
  xlab("Age in years") +
  ylab("Fitted values") +
  theme_classic()
```



It appears as if there is a strong interaction between age and gender in this simple regression. The regression output and graph suggest that men and women start with a similar income level at a young age (25), but thereafter the average man's income increases by R87 every year, while the average woman's income increases by only R88.5-R75=R13.5 per year.

Let's return to our slightly more complete model and see whether we can use interaction between variables to improve our model. It is important however to always remember that just as you should not arbitrarily include new variables in your model without a theoretical justification; similarly you should never arbitrarily interact two variables without a good reason for doing so.

In practice, creating an interaction term in R between two categorical variables is as easy as inserting an asterisk " $*$ " between the two variables you wish to interact. Understanding fully what you are doing is a little trickier. For instance, we might wish to interact gender (2 categories) with race (4 categories).

```
lm5 <- lm(w1_fwag~educ.new+age+relevel(race, "White")*gender, data = dum)
```

Using `stargazer` package (Hlavac 2018)

```
library(stargazer)
#summary(lm5)
stargazer(lm5, type = "text")
```

```
##
## ===============================================================
##                                         Dependent variable:
##                                     ---------------------------
##                                               w1_fwag
## ---------------------------------------------------------------
## educ.new                                    461.210***
```

```
##                                                    (19.498)
##
## age                                                99.577***
##                                                     (8.452)
##
## relevel(race, "White")African                     -7,536.520***
##                                                    (366.344)
##
## relevel(race, "White")Coloured                    -7,584.998***
##                                                    (407.904)
##
## relevel(race, "White")Asian                       -3,416.403***
##                                                    (778.996)
##
## genderFemale                                      -5,522.407***
##                                                    (474.680)
##
## relevel(race, "White")African:genderFemale         4,266.182***
##                                                    (509.259)
##
## relevel(race, "White")Coloured:genderFemale        4,486.902***
##                                                    (572.918)
##
## relevel(race, "White")Asian:genderFemale           4,646.987***
##                                                   (1,180.159)
##
## Constant                                          2,763.515***
##                                                    (591.960)
##
## ------------------------------------------------------------------------
## Observations                                            3,728
## R2                                                      0.318
## Adjusted R2                                             0.316
## Residual Std. Error                           4,562.474 (df = 3718)
## F Statistic                                 192.537*** (df = 9; 3718)
## ========================================================================
## Note:                                      *p<0.1; **p<0.05; ***p<0.01
```

When interpreting regressions with interaction effects, we need to be careful to not just interpret the partial effects and ignore the interaction terms. The partial coefficients no longer "stand alone" and the interaction terms need to be incorporated into our explanations. For instance, we can say that White women earn R5522 less than White men, on average holding education and age constant, since White is the reference racial group. But if we compare African men to African women, we can say that African women earn R5522 - R4266 = R1256 less than African men. This gender difference in earnings is **much smaller** for Africans than for Whites. Previously, we were forcing it to be equal across racial categories. This is why it is so important to include these interactions in our regression!

Provided we update our formula, we still calculate any estimated yhat value as normal. For example, if we were interested in calculating the income for an African female aged 35 with a 12 year level of education, we compute the following:

$predictedincome = 2763.515 + 461.2096(12) + 99.57732(35) - 7536.52(1) - 5522.407(1) + 4266.182(1)$

$predictedincome = 1731.95$

5. In figuring out what predicts someone's net pay, is there an interaction effect between

education and gender? Compute a regression where education, age, gender and race are the independent variables explaining someone's net pay. (Restrict your analysis to sample1).

Question 5 Answer

8.5 Linear transformations of non-linear relationships

Thus far, we have assumed linear relationships for all of our regression models. In fact, a linear relationship is a basic requirement for regression analysis. Empirically, however, variables are often not associated in a linear fashion. Yet this reality hardly precludes regression analyses from accurately predicting and describing real world phenomenon. In this section we will show you two basic approaches to dealing with non-linear relationships. The first method will be to include a quadratic term; and the second will use natural logarithms to improve our model. It will often be the case that we can use the natural logarithm of a term to transform a non-linear relationship into an approximately linear one and vastly improve the fit of a regression line. Note: Logarithmic and Quadratic transformations are not restricted to multiple regression, however, we have placed them in the multiple regression chapter because they are rather advanced topics and should only be addressed after one has a clear understanding of all of the material in chapters 7 and 8 prior to this section.

Transformations Using the Natural Logarithm

Often it is desirable to run a regression using the natural logarithm (a log to the base of the mathematical constant e) of a variable instead of the variable itself. The advantages of using logarithms include the following: 1. Firstly, logarithms transform a variable in such a way that the logged variable contains exactly the same information as the original variable but the influence of outliers is greatly diminished. This can sometimes drastically affect the slope of the regression line because the natural logarithm of a variable compresses the data and is much less sensitive to extreme observations than is the variable itself. 2. Secondly, taking logarithms of a variable allows us to interpret the relevant coefficients in our regression as percentage changes. 3. Thirdly, taking the logarithm of a variable can often make the distribution closer to a 'normal' distribution. 4. And lastly, if the relationship between a dependent variable and the independent variable is non-linear, making one or both of the variables logarithmic can sometimes produce a linear relationship. Therefore, although a linear relationship might not exist between two variables, a linear relationship might exist between the natural logarithms of the two variables.

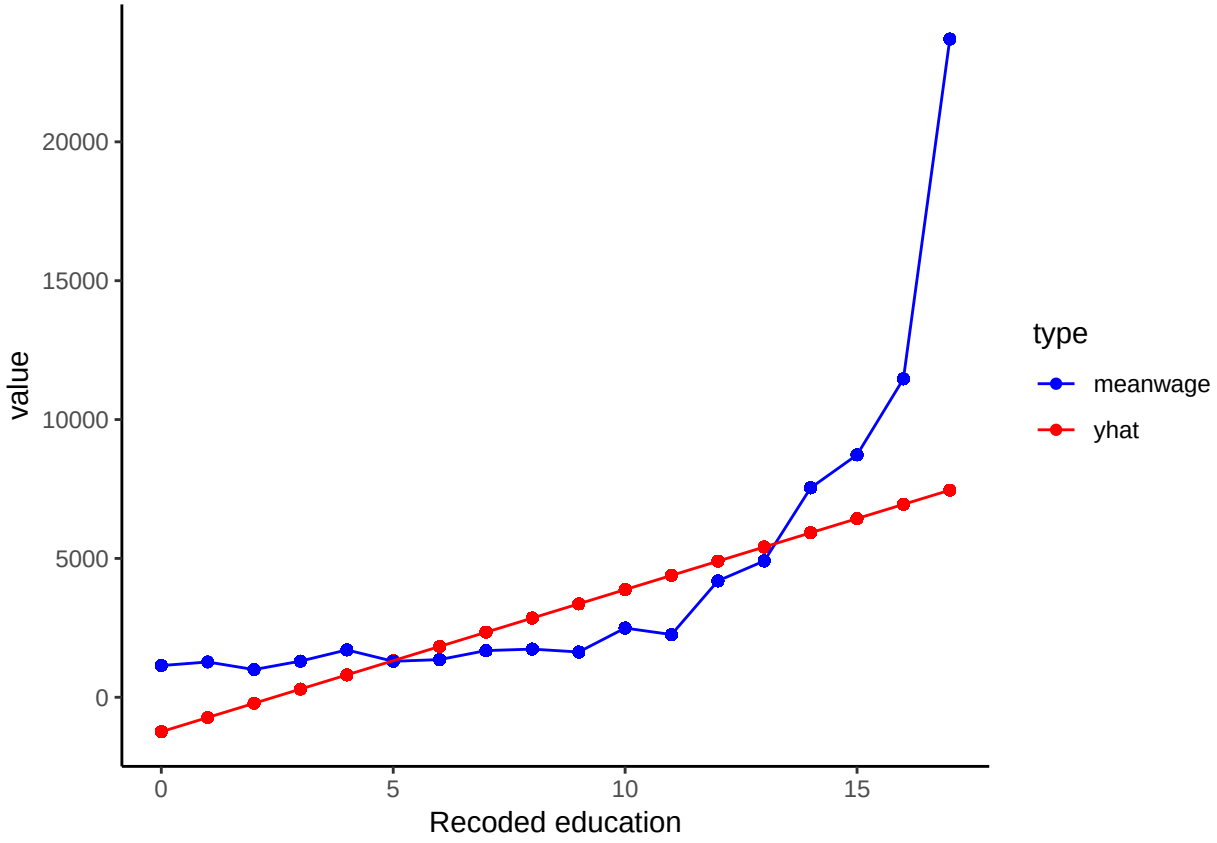
At the beginning of this chapter we hinted that the relationship between wage and education in our regression was non-linear. Let us examine the relationship between these two variables in more depth. If we type:

```
nids%>%
  group_by(educ.new)%>%
  summarise(Mean = mean(w1_fwag, na.rm=TRUE), sd=sd(w1_fwag, na.rm=TRUE))
```

```
## # A tibble: 19 x 3
##    educ.new   Mean     sd
##       <dbl>  <dbl>  <dbl>
##  1        0  1162.  1053.
##  2        1  1292.  1032.
##  3        2   992.   767.
##  4        3  1282.  1222.
##  5        4  1668.  3272.
##  6        5  1228.  1003.
##  7        6  1329.   988.
##  8        7  1584.  2645.
##  9        8  1698.  1668.
## 10        9  1528.  1285.
## 11       10  2396.  2438.
## 12       11  2114.  2302.
```

```
## 13       12  3862.  5485.
## 14       13  4572.  4275.
## 15       14  7417.  7834.
## 16       15  8414.  5942.
## 17       16 11163.  9193.
## 18       17 22572. 17303.
## 19       NA  1774.  2979.
```

The above data implies reveals that the returns to education are non-linear. In fact, earnings are fairly flat in early grades, but we see big increases from 9 to 10, 12 to 13 and for every additional year of tertiary education. Below we also reproduce the graph we generated in the beginning of this chapter, which corroborates our conclusions above.



As you have probably guessed by now, we can most likely improve the fit of our model by logging one or both of the variables. The distribution of the wage variable has been well researched and it has become common-practice to use a log transformation to make it 'behave better'. Apart from the non-linear relationship between wage and education, we can see the benefit of logging the wage variable from the two graphs below. The first shows a density plot of the wage variable and the second shows a density plot of the transformed log-wage variable.

```
kernel<-nids %>%
  filter(sample1==1) %>%
  select(hhid, w1_fwag, educ.new, sample1, age, race, gender) %>%
  na.omit()

library(scales)
```

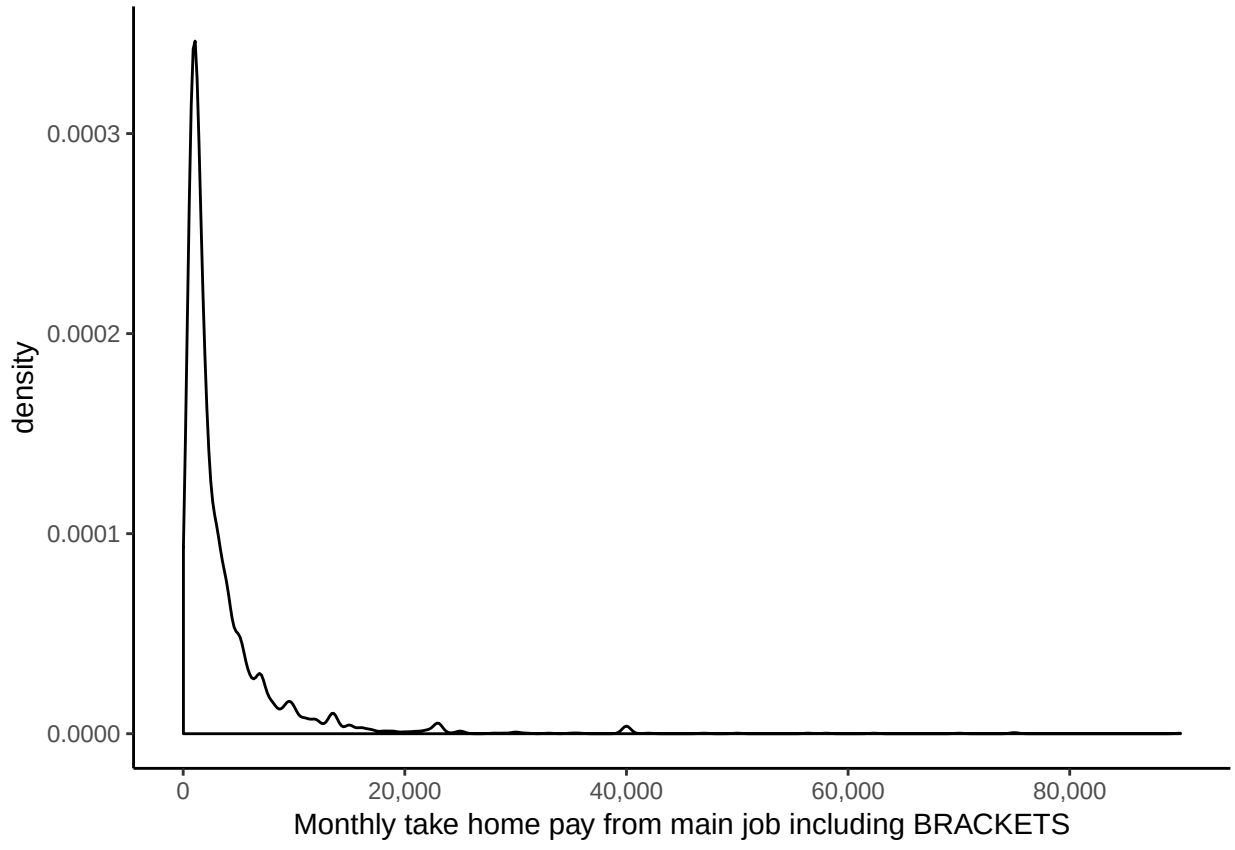
```
##
```

```
## Attaching package: 'scales'
```

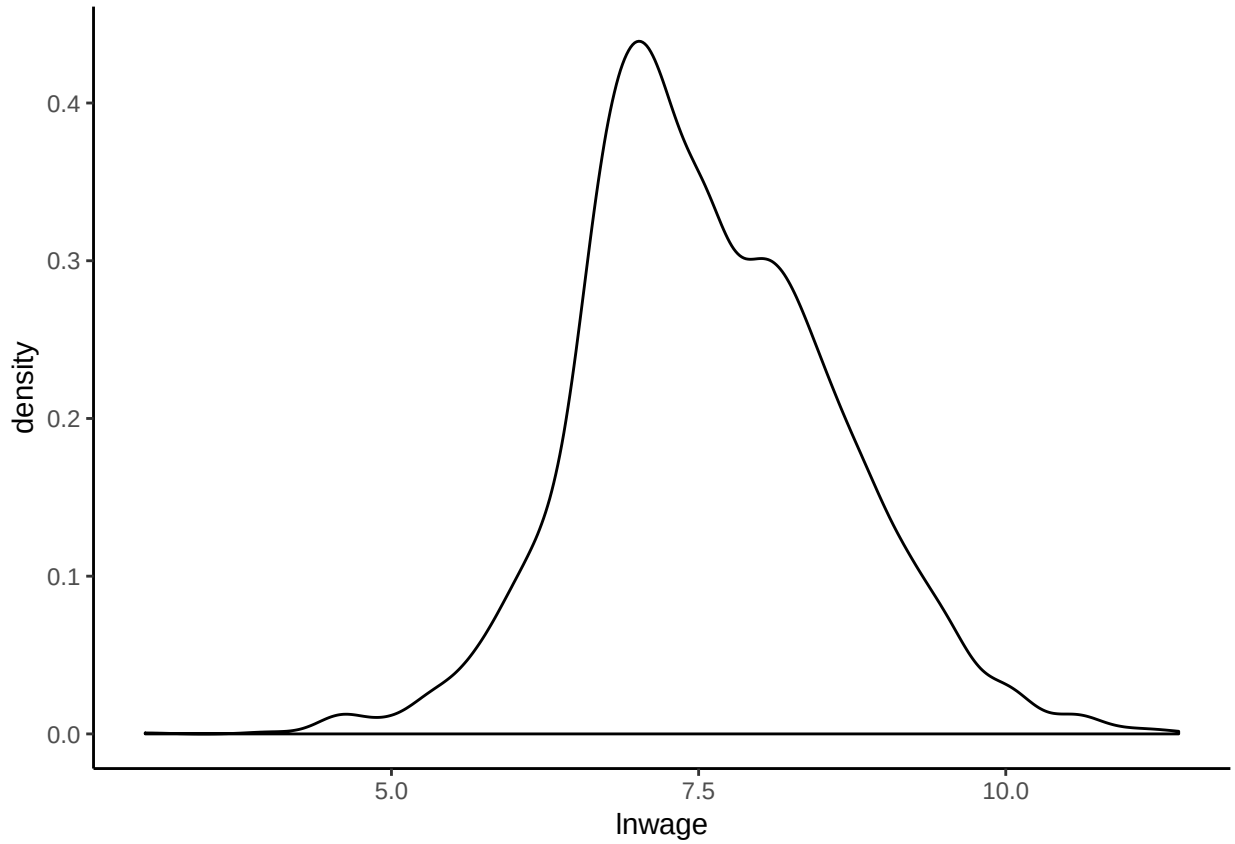
```
## The following object is masked from 'package:purrr':
##
##     discard
```

```
## The following object is masked from 'package:readr':
##
##     col_factor
```

```
ggplot(kernel, aes(x = w1_fwag)) +
  geom_density() +
  scale_y_continuous(breaks=seq(0,0.0003,0.0001), labels = comma) +
  scale_x_continuous(breaks=seq(0,100000,20000), labels = comma) +
  labs(x="Monthly take home pay from main job including BRACKETS") +
  theme_classic()
```




```
kernel$lnwage<-log(kernel$w1_fwag)
ggplot(kernel, aes(x = lnwage)) +
  geom_density() +
  theme_classic()
```



From these two graphs, it is immediately clear that the second has a far more 'normal' distribution and that since all the values are squashed closer together, outliers are unlikely to exert as great an influence on our regression. These two reasons, in addition to the nonlinear relationship between wage and education, as well as the precedent set in the literature, provide ample rationale for taking the log of our wage variable.

Thus we are now going to investigate what happens if we include the logged wage variable in our simple linear regression with education as our independent variable:

```
lmy <- lm(lnwage~educ.new, data = subset(kernel, subset=sample1 ==1))
summary(lmy)
```

```
##
## Call:
## lm(formula = lnwage ~ educ.new, data = subset(kernel, subset = sample1 ==
##     1))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2731 -0.5126 -0.0075  0.5421  3.4487
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.315605   0.033577  188.10   <2e-16 ***
## educ.new    0.136170   0.003301   41.25   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.8588 on 3726 degrees of freedom
## Multiple R-squared:  0.3135, Adjusted R-squared:  0.3134
## F-statistic:  1702 on 1 and 3726 DF,  p-value: < 2.2e-16
```

The R-squared value suggests that we have a much better fitting regression line when compared to a regression of `w1_fwag` on `educ_new` (see the first regression run in this chapter). In fact, a simple log transformation of the wage variable has increased the predictive power of our model from 15.63 to 31.34%! Let's generate a new graph for this regression:

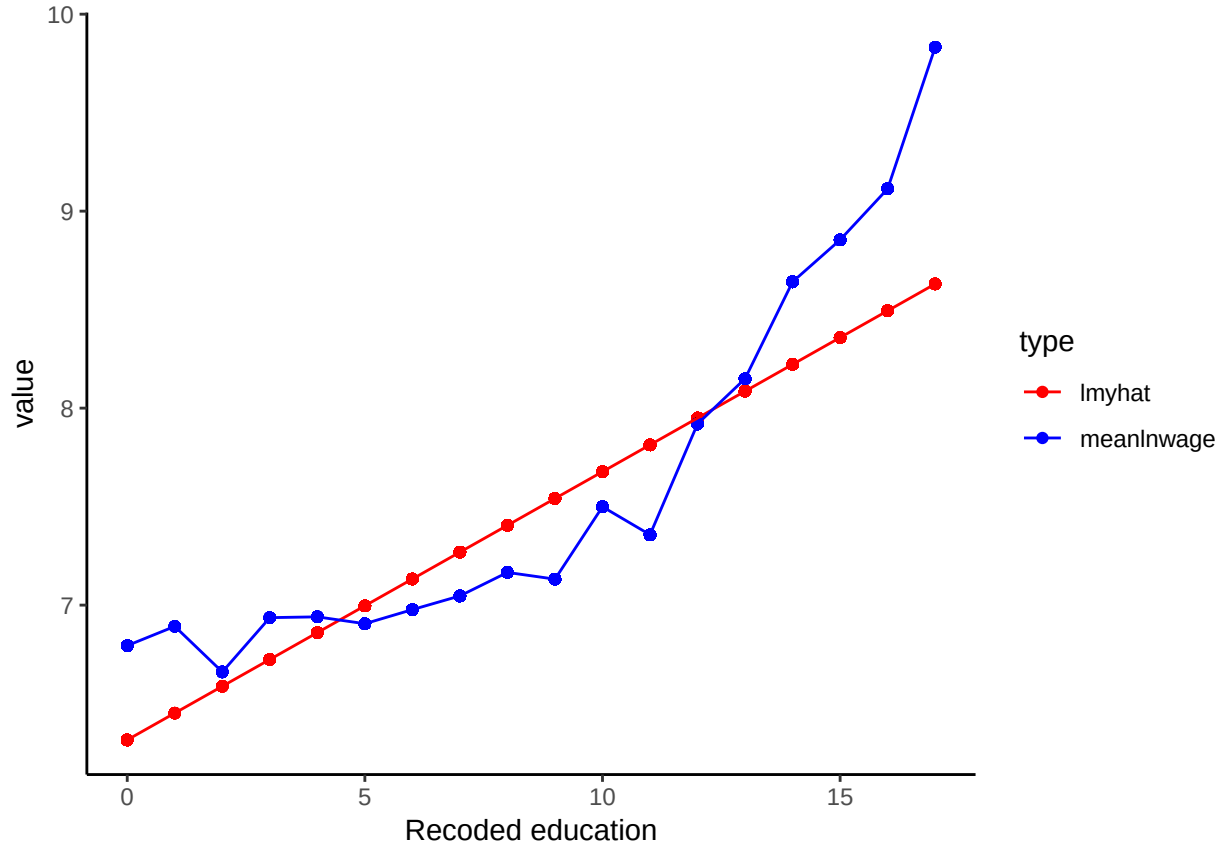
```
kernel$lnwage<-log(kernel$w1_fwag)

lmy <- lm(lnwage~educ.new, data = subset(kernel, subset=sample1==1))
summary(lmy)
```

```
##
## Call:
## lm(formula = lnwage ~ educ.new, data = subset(kernel, subset = sample1 ==
##     1))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2731 -0.5126 -0.0075  0.5421  3.4487
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.315605   0.033577  188.10   <2e-16 ***
## educ.new    0.136170   0.003301   41.25   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8588 on 3726 degrees of freedom
## Multiple R-squared:  0.3135, Adjusted R-squared:  0.3134
## F-statistic:  1702 on 1 and 3726 DF,  p-value: < 2.2e-16
```

```
kernel$lmyhat<-lmy$fitted.values   #other predict options

kernel%>%
  filter(sample1==1)%>%
  group_by(educ.new)%>%
  mutate(meanlnwage=mean(lnwage), lmyhat=lmyhat)%>%
  select(educ.new, lmyhat, meanlnwage)%>%
  gather(key = type,value=value, -educ.new)%>%
  ggplot(., aes(educ.new,value, color = type)) +
  geom_point() +
  geom_line() +
  scale_color_manual(values=c("red","blue")) +
  xlab("Recoded education") +
  theme_classic()
```



Transforming the wage variable has helped to linearise the data (the blue line) and so our linear model (red line) can produce a better fit!

Now we regress `lnwage` on `educ_new` as well as our set of control variables.

```
lmy <- lm(lnwage~educ.new+age+relevel(race, "White")*gender, data = subset(kernel, subset=sample1==1))
#summary(lmy)
stargazer(lmy, type="text")
```

```
##
## ================================================================================
##                                                  Dependent variable:
##                                              ---------------------------
##                                                         lnwage
## --------------------------------------------------------------------------------
## educ.new                                                0.133***
##                                                          (0.003)
##
## age                                                     0.022***
##                                                          (0.001)
##
## relevel(race, "White")African                           -0.909***
##                                                          (0.061)
##
## relevel(race, "White")Coloured                          -0.905***
##                                                          (0.067)
##
```

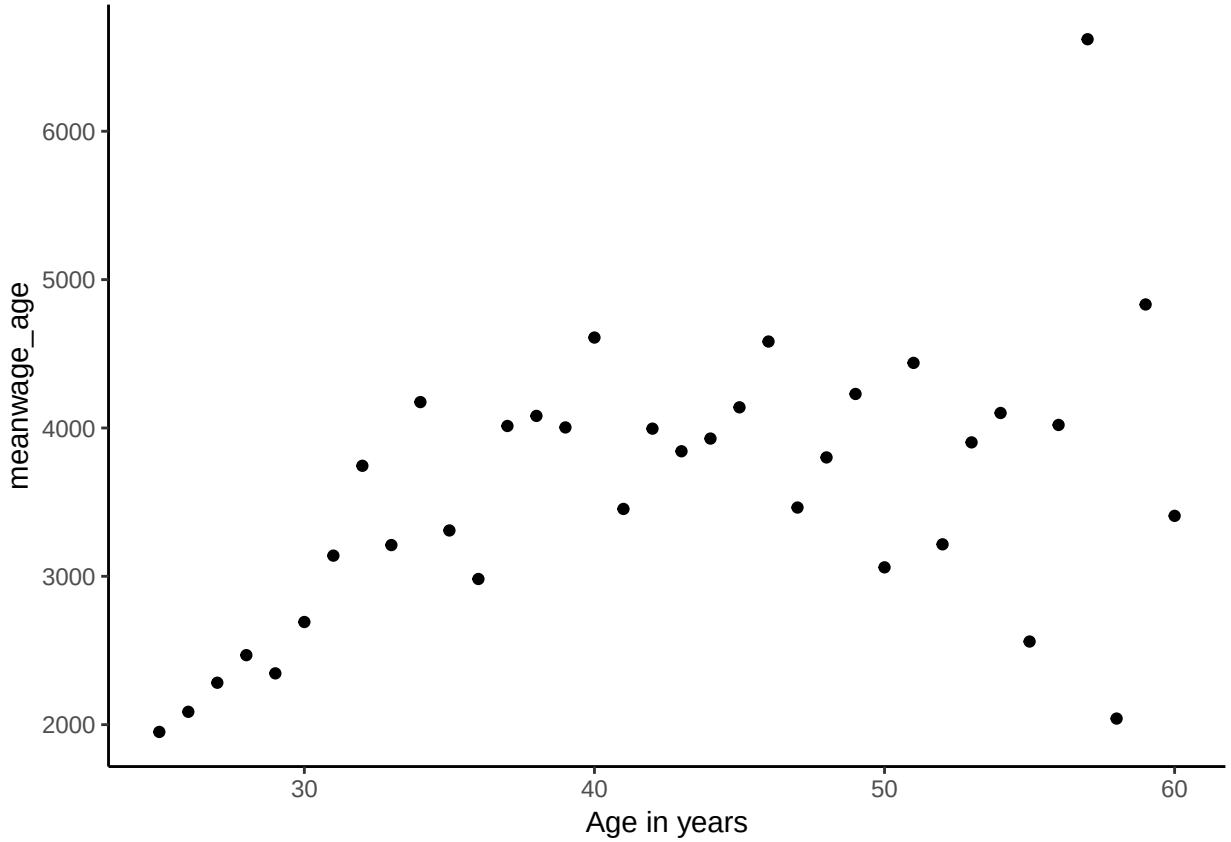
```
## relevel(race, "White")Asian                                -0.164
##                                                            (0.129)
##
## genderFemale                                              -0.462***
##                                                            (0.079)
##
## relevel(race, "White")African:genderFemale                  0.025
##                                                            (0.084)
##
## relevel(race, "White")Coloured:genderFemale                 0.084
##                                                            (0.095)
##
## relevel(race, "White")Asian:genderFemale                    0.236
##                                                            (0.195)
##
## Constant                                                  6.474***
##                                                            (0.098)
##
## ------------------------------------------------------------------------
## Observations                                                3,728
## R2                                                          0.471
## Adjusted R2                                                 0.469
## Residual Std. Error                                  0.755 (df = 3718)
## F Statistic                                     367.382*** (df = 9; 3718)
## ========================================================================
## Note:                                        *p<0.1; **p<0.05; ***p<0.01
```

The Adjusted R-squared value is 0.4694, thus logging the wage variable appears to be advantageous. We need to be careful when we interpret the coefficients in this regression since our dependant variable is now in a different form. The coefficients should, in fact, be interpreted as rates of return. For example, the rate of return to education is 0.1326 - that is, an extra year of education will, on average and controlling for other variables, increase average wages by 13.26%.

Transformations using Squared Terms

You may remember that we previously noted that it may not make sense that as we get older we continue to earn more money. Researchers often include both age and age2 in regression models because it allows the effect of one-year increase in age to change as a person gets older. By including age squared, we are controlling for the idea that there comes a point in time after which age no longer provides an advantage in the workforce. That is, the effect of age is not likely to remain the same as we get older; after all, a 90 year old employee is likely to earn less, rather than more, than say a 60 year old. Therefore, by including age2, we allow for age to have a parabolic effect on income (think back to the simple parabolas you used to draw in school - first up and then down). We can see this relationship by plotting mean wage levels for every age group against age:

```
kernel%>%
  group_by(age)%>%
  summarise(mean.wage.age=mean(w1_fwag, na.rm=TRUE))%>%
  ggplot(., aes(x = age, y = mean.wage.age)) +
  geom_point() +
  xlab("Age in years") +
  ylab("meanwage_age") +
  theme_classic()
```



It is clear from this graph that the relationship between mean wages at every age and age is parabolic. So, let's create a new age-squared variable in order to try to improve the fit of our regression.

```
kernel<-kernel%>%
  mutate(age2 = age*age)

lmy <- lm(lnwage~educ.new+age+age2+relevel(race, "White")+gender, data = kernel)
#summary(lmy)
stargazer(lmy, type="text")
```

```
##
## ============================================================
##                                      Dependent variable:
##                                  ---------------------------
##                                             lnwage
## ------------------------------------------------------------
## educ.new                                   0.132***
##                                             (0.003)
##
## age                                        0.075***
##                                             (0.012)
##
## age2                                       -0.001***
##                                            (0.0001)
##
## relevel(race, "White")African              -0.899***
##                                             (0.045)
```

```
##
## relevel(race, "White")Coloured          -0.872***
##                                          (0.050)
##
## relevel(race, "White")Asian              -0.077
##                                          (0.097)
##
## genderFemale                            -0.426***
##                                          (0.025)
##
## Constant                                5.441***
##                                          (0.236)
##
## ------------------------------------------------------------
## Observations                               3,728
## R2                                         0.473
## Adjusted R2                                0.472
## Residual Std. Error                  0.753 (df = 3720)
## F Statistic                      477.787*** (df = 7; 3720)
## ============================================================
## Note:                           *p<0.1; **p<0.05; ***p<0.01
```

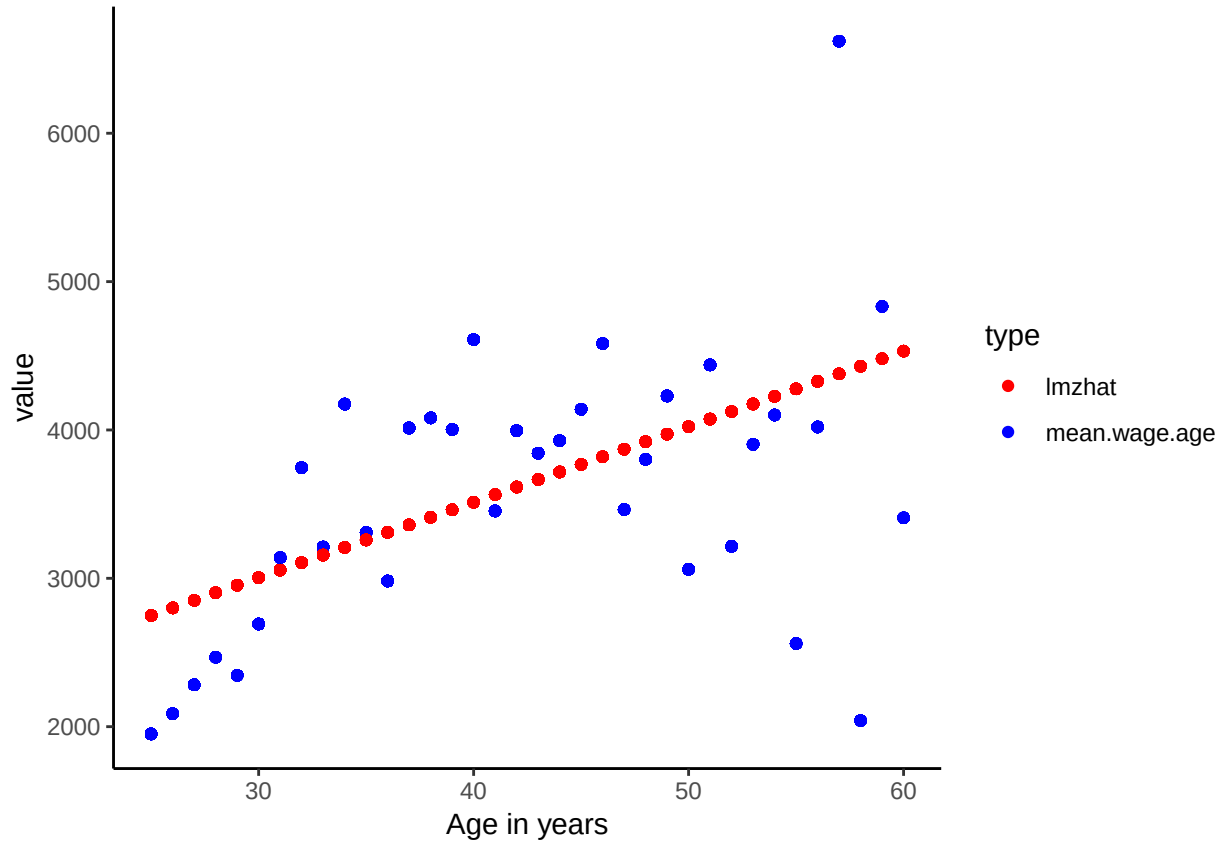
Let's interpret our coefficients. We find that each year of education increases income by 13.18% on average. Looking at age, we see that coefficient on `age` is positive, but the coefficient on `age2` is negative. If you think back to the quadratic formula ($ax^2 + bx + c$), this means that **a** is negative and **b** is positive. This suggests a parabolic influence of `age` on `lnwage` with a maximum at the turning point $\frac{-b}{2a}$. In other words, every additional year of age increases wages up to the age of 58 and thereafter every additional year decreases wages. We calculate this turning point using the turning point formula ($\frac{-b}{2a}$), which for our `age` and `age2` coefficients is:

$$\frac{-0.0751816}{2*(-0.0006515)}$$

We can also get a graphical idea of what is going on if we look at a simple regression of wage on age:

```
lmz <- lm(w1_fwag~age, data = kernel)
kernel$lmzhat<-lmz$fitted.values

kernel%>%
  group_by(age)%>%
  mutate(mean.wage.age=mean(w1_fwag, na.rm=TRUE))%>%
  select(age, mean.wage.age, lmzhat)%>%
  gather(key = type,value=value, -age)%>%
  ggplot(., aes(age,value, color = type)) +
  geom_point() +
  scale_color_manual(values=c("red","blue")) +
  xlab("Age in years") +
  theme_classic()
```

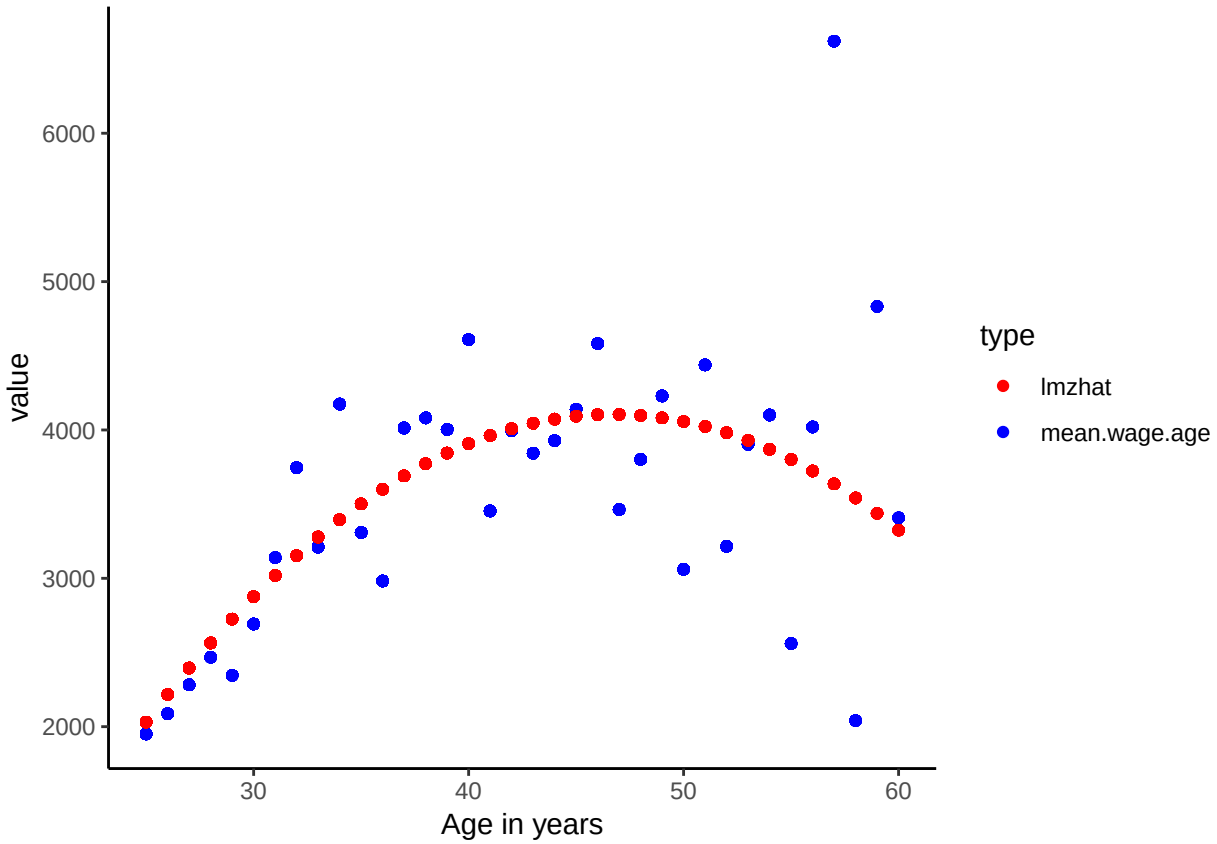


As expected the linear predicted values are poor estimates for the actual income values. Let's see whether included the age-squared variable in the regression improves the fit of the predicted values.

```
kernel<-kernel%>%
  mutate(age2 = age*age)

lmz <- lm(w1_fwag~age + age2, data = kernel)
kernel$lmzhat<-lmz$fitted.values

kernel%>%
  group_by(age)%>%
  mutate(mean.wage.age=mean(w1_fwag, na.rm=TRUE))%>%
  select(age, mean.wage.age, lmzhat)%>%
  gather(key = type,value=value, -age)%>%
  ggplot(., aes(age,value, color = type)) +
  geom_point() +
  scale_color_manual(values=c("red","blue")) +
  xlab("Age in years") +
  theme_classic()
```



Much better! Also, you may have found that many of the ideas learnt in these chapters simpler to understand when they were expressed as graphs. There is a lesson here. As they say, a picture says a thousand words. Even though you are now starting to become more familiar with more complicated statistical analysis, such as multiple regression, you should never overlook the power of simple descriptive statistics and graphical representations of the data. In fact, it is a common mistake that people jump into complicated statistical analysis without getting properly acquainted with their data. This can lead to rubbish results which could have been easily avoided.

Nevertheless, since you have now completed this course, we are sure that you are well on your way to doing some fantastic statistical research and will be able to avoid many of the potential pitfalls! But before you do, quickly try the exercises to make sure you are comfortable with these new methods and to explore some slightly more complicated ideas. Good luck!

8.6 Question Answers

8.7 Exercises

1. What effect does being Indian/Asian have on household size, in comparison to being Coloured?

Exercise 1 Answer

2. Regress household expenditure on household income and number of rooms in the house. What exactly are the F and t statistics testing in this regression? (Difficult)

Exercise 2 Answer

3. Households speaking which language have the lowest total monthly expenditure (controlling for total monthly income)? What about the lowest?

Exercise 3 Answer

4. Graph total monthly nonfood expenditure on total monthly household income. Now logarithmically transform both variables and graph them. Which do you think will produce a stronger regression model? Now run the regression with the original variables and the log transformed ones. Were you right? Why or why not?

Exercise 4 Answer

8.8 Exercise answers

8.9 Session information

```
print(sessionInfo(), locale = FALSE)
```

```
## R version 3.5.1 (2018-07-02)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 7 x64 (build 7601) Service Pack 1
##
## Matrix products: default
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] scales_0.5.0    stargazer_5.2.2 bindrcpp_0.2.2  forcats_0.3.0
##  [5] stringr_1.3.1   dplyr_0.7.6     purrr_0.2.5     readr_1.1.1
##  [9] tidyr_0.8.1     tibble_1.4.2    ggplot2_3.0.0   tidyverse_1.2.1
## [13] foreign_0.8-70
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_0.2.4 xfun_0.2         reshape2_1.4.3   haven_1.1.2
##  [5] lattice_0.20-35  colorspace_1.3-2 htmltools_0.3.6  yaml_2.1.19
##  [9] utf8_1.1.4       rlang_0.2.1      pillar_1.2.3     withr_2.1.2
## [13] glue_1.2.0       modelr_0.1.2     readxl_1.1.0     bindr_0.1.1
## [17] plyr_1.8.4       munsell_0.5.0    gtable_0.2.0     cellranger_1.1.0
## [21] rvest_0.3.2      psych_1.8.4      evaluate_0.10.1  labeling_0.3
## [25] knitr_1.20       parallel_3.5.1   broom_0.4.5      Rcpp_0.12.17
## [29] backports_1.1.2  jsonlite_1.5     mnormt_1.5-5     hms_0.4.2
## [33] digest_0.6.15    stringi_1.1.7    bookdown_0.7     grid_3.5.1
## [37] rprojroot_1.3-2  cli_1.0.0        tools_3.5.1      magrittr_1.5
## [41] lazyeval_0.2.1   crayon_1.3.4     pkgconfig_2.0.1  xml2_1.2.0
## [45] lubridate_1.7.4  assertthat_0.2.0 rmarkdown_1.10   httr_1.3.1
## [49] rstudioapi_0.7   R6_2.2.2         nlme_3.1-137     compiler_3.5.1
```

Chapter 9

Further analysis and useful online resources

9.1 Further analyses

The preceding sections covers most of the content in the original Stata course. Here we provide further information for users who might be interested in carrying out advanced statistical analyses. The list below is non-exhaustive but we hope it can provide a good starting point for those interested in the respective types of analyses.

Regression analysis:

- In the guide we introduced `lm()` to fit linear models. There is another function, `glm()`, for fitting simple and multiple linear and non linear regressions including logistic regression and more generally models falling under the Generalized Linear Model (GLM) framework. This function is implemented in the base R `stats` package.
- `lme4` (Bates et al. 2018) and `nlme` (Pinheiro, Bates, and R-core 2018) - packages for fitting linear multilevel (i.e. mixed effects) models. The `nlme` package also allows to fit non linear multilevel models.

Complex sample survey design and data analysis

- Main suite of functions used to do complex survey weighting in R is in the `survey` package (Lumley 2018). It includes commands to specify a complex survey design (stratified sampling design, cluster sampling, multi-stage sampling and pps sampling with or without replacement), calibration (post-stratification, generalized raking/calibration, GREG estimation and trimming of weights), e.t.c. You can also check supporting vignettes here.

9.2 Useful online resources

There are hundreds of web sites and online resources dedicated to R that users can consult. It is difficult to do justice to all of them and a few of the most common ones are listed below to help you get started:

Introductory

- The R manuals.
- An Introduction to R.
- Quick-R homepage - statmethods.net - a good place to start learning R.

- DataCamp's free interactive introduction to R programming.
- Cookbook for R.
- UCLA website https://stats.idre.ucla.edu/r/ - provides a good starting point for the R beginner and other statistical packages.
- R Reference Card.

Expert

- The R for Data Science book. The book provides readers with a good grounding in basic aspects of data analysis, from import and cleaning to visualizing and modeling. The book was authored by Hadley Wickham and Garrett Grolemund who both work at RStudio. Wickham is the main developer of the `tidyverse` packages that are used in this guide. *R for Data Science* is also available for free online.
- The R Inferno. The abstract sums up everything: "If you are using R and you think you're in hell, this is a map for you." - Patrick Burns.

Data visualization (ggplot2 graphics).

Useful resources for starting to learn `ggplot2` include:

- gglot2 Documentation (particularly the function reference).
- Data Visualization Chapter of R for Data Science Book.
- RStudio's ggplot2 cheat sheet.
- R Graphics Cookbook.
- UCLA's introduction to `ggplot2`.

Blogs

- https://www.r-bloggers.com/ - a blog aggregator that posts or repost R related articles contributed by bloggers. It helps you keep up to date with changes in packages, new techniques and better applications. R bloggers is a good place to find R tutorials, announcements, and other random happenings.
- http://blog.revolutionanalytics.com/ - now part of Microsoft and is a blog that is dedicated to a wide variety of R technical updates.

Ask questions

- StackOverflow - a searchable forum of questions and answers about computer programming. It is a great resource with many questions for many specific packages in R and most developers of the packages are also active on StackOverflow to answer questions related to their packages. There is also a rating system for answers.
- stats.stackexchange.com - not specific to R but contains statistics/machine learning/data analysis/data mining/data visualization related questions and answers raised by R users.

Other

- rseek.org - the search engine just for R.

Bates, Douglas, Martin Maechler, Ben Bolker, and Steven Walker. 2018. *Lme4: Linear Mixed-Effects Models Using 'Eigen' and S4*. https://CRAN.R-project.org/package=lme4.

Henry, Lionel, and Hadley Wickham. 2018. *Purrr: Functional Programming Tools*. https://CRAN.R-project.org/package=purrr.

Hlavac, Marek. 2018. *Stargazer: Well-Formatted Regression and Summary Statistics Tables*. https://CRAN.R-project.org/package=stargazer.

Lumley, Thomas. 2018. *Survey: Analysis of Complex Survey Samples*. https://CRAN.R-project.org/

package=survey.

Müller, Kirill, and Hadley Wickham. 2018. *Tibble: Simple Data Frames*. https://CRAN.R-project.org/package=tibble.

Pinheiro, José, Douglas Bates, and R-core. 2018. *Nlme: Linear and Nonlinear Mixed Effects Models*. https://CRAN.R-project.org/package=nlme.

R Core Team. 2017. *Foreign: Read Data Stored by 'Minitab', 'S', 'Sas', 'Spss', 'Stata', 'Systat', 'Weka', 'dBase', ...* https://CRAN.R-project.org/package=foreign.

Wickham, Hadley. 2017. *Tidyverse: Easily Install and Load the 'Tidyverse'*. https://CRAN.R-project.org/package=tidyverse.

Wickham, Hadley, and Lionel Henry. 2018. *Tidyr: Easily Tidy Data with 'Spread()' and 'Gather()' Functions*. https://CRAN.R-project.org/package=tidyr.

Wickham, Hadley, and Evan Miller. 2018. *Haven: Import and Export 'Spss', 'Stata' and 'Sas' Files*. https://CRAN.R-project.org/package=haven.

Wickham, Hadley, Winston Chang, Lionel Henry, Thomas Lin Pedersen, Kohske Takahashi, Claus Wilke, and Kara Woo. 2018. *Ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics*. https://CRAN.R-project.org/package=ggplot2.

Wickham, Hadley, Romain François, Lionel Henry, and Kirill Müller. 2018. *Dplyr: A Grammar of Data Manipulation*. https://CRAN.R-project.org/package=dplyr.

Wickham, Hadley, Jim Hester, and Romain Francois. 2017. *Readr: Read Rectangular Text Data*. https://CRAN.R-project.org/package=readr.