NC STATE UNIVERSITY

Introduction to Data Science Using R Part II

Justin Post

August 7-8, 2017

What do we want to be able to do?

- **Read in data**
- **Manipulate data**
- Plot data
- Summarize data
- Analyze data

Schedule

Day 1

- Install R/R studio
- R Studio Interface
- Classes and Objects
- Attributes and Basic Data Object Manipulation
- **Reading in Data/Writing Out Data**
- **Logical Statements and Subsetting/Manipulating Data**

3/57

Reading in Data/Writing Out Data

Data comes in many formats

- 'Delimited' data: Character (such as ',' , '>', or [' ']) separated data
- Fixed field data
- Excel data
- SPSS formatted data
- SAS data sets
- Many ways to read in the data… How to choose?

Reading in Data/Writing Out Data

- Possible methods to read data
  - Base R (what comes installed)
  - Use an R 'package'
- R package
  - Collection of functions in one place
  - Packages exist to do almost anything
  - List of CRAN approved packages on R's website
  - Plenty of other packages on places like GitHub

5/57

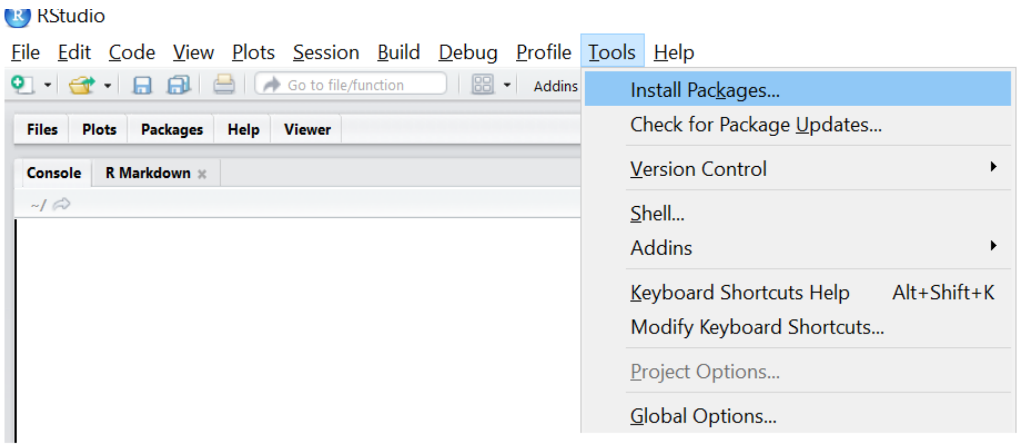
Reading in Data/Writing Out Data

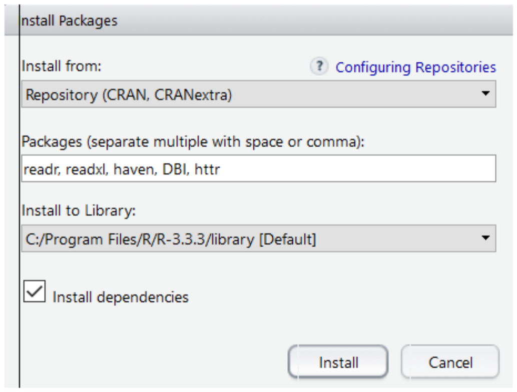
- First time using a package
  - Must install package (download)
  - Can use code or menus

```
install.packages("readr")
#can do multiple packages at once
install.packages(c("readr", "readxl", "haven", "DBI", "httr"))
```

Reading in Data/Writing Out Data

- First time using a package
    - Must install package (download)
    - Can use code or menus


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Files Plots Packages Help Viewer
Console R Markdown
Install Packages...
Check for Package Updates...
Version Control
Shell...
Addins
Keyboard Shortcuts Help Alt+Shift+K
Modify Keyboard Shortcuts...
Project Options...
Global Options...



Install from: Repository (CRAN, CRANextra)

Packages (separate multiple with space or comma): readr, readxl, haven, DBI, httr

Install to Library: C:/Program Files/R/R-3.3.3/library [Default]

[✓] Install dependencies

7/57

Reading in Data/Writing Out Data

- Once 'installed' on computer, never need to install again (unless you update R)
- **Each session** read the package in using `library()` or `require()`

```
library("readr")
require("haven")
```

Reading in Data/Writing Out Data

- Difference - if no package
  - `library()` throws an error
  - `require()` returns FALSE

```
library("notAPackage")
```

```
## Error in library("notAPackage"): there is no package called 'notAPackage'
```

```
require("notAPackage")
```

```
## Loading required package: notAPackage
```

```
## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'notAPackage'
```

9/57

Reading in Data/Writing Out Data

- Many packages to read in data
- How to choose?
    - Want 'fast' code
    - Want 'easy' syntax
    - Good default settings on functions
- Base R has reasonable defaults and syntax but functions are slow
- "TidyVerse" - collection of R packages that share common philosophies and are designed to work together!
    - Very efficient code

Reading in Data/Writing Out Data

Reading in a comma separated value (.csv) file

- Let's install the `tidyverse` package

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

Reading in Data/Writing Out Data

Reading in a comma separated value (.csv) file

- Let's install the `tidyverse` package

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

- Load library

```
library(tidyverse)
```

- Once library loaded, check `help(read_csv)`
- Want to read in scores.csv file using `read_csv()`

Reading in Data/Writing Out Data

- How does R locate the file?

Reading in Data/Writing Out Data

- How does R locate the file?
  - Can give file *full path name*
    - ex: E:/Other/DataScienceR/datasets/data.txt

Reading in Data/Writing Out Data

- How does R locate the file?
  - Can give file *full path name*
    - ex: E:/Other/DataScienceR/datasets/data.txt
  - Can change *working directory*
    - Folder on computer usually
    - Where R 'looks' for files
    - Supply abbreviated path name

```
getwd()
```

```
## [1] "E:/Other/DataScienceR"
```

Reading in Data/Writing Out Data

- How does R locate the file?
    - Can change *working directory*

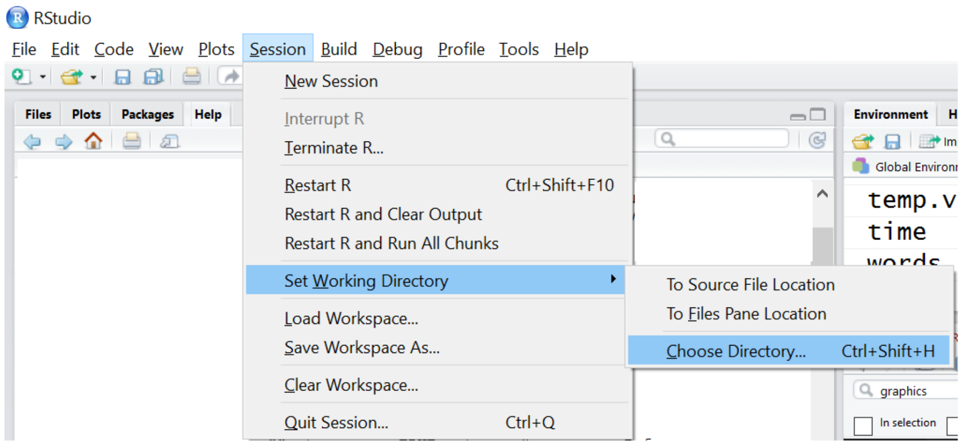
Reading in Data/Writing Out Data

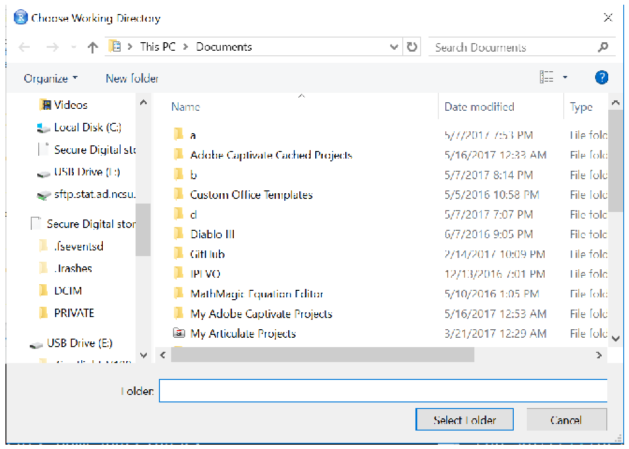
- How does R locate the file?
  - Can change *working directory*
  - Via code

```
setwd("E:\\Other\\DataScienceR")
#or
setwd("E:/Other/DataScienceR")
```

Reading in Data/Writing Out Data

- How does R locate the file?
  - Can change *working directory*
  - Via menus





Reading in Data/Writing Out Data

Reading in a comma separated value (.csv) file

- Often, create a folder with all files for your project
- Set working directory to that folder
- Read in data

Reading in Data/Writing Out Data

Reading in a comma separated value (.csv) file

- To avoid dealing with downloading files, we'll pull straight from the web

```
scoreData <- read_csv(file = "https://raw.githubusercontent.com/
                    jbpost2/DataScienceR/master/datasets/scores.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_integer(),
##   week = col_character(),
##   date = col_character(),
##   day = col_character(),
##   awayTeam = col_character(),
##   homeTeam = col_character(),
##   stadium = col_character(),
##   startTime = col_time(format = ""),
##   toss = col_character(),
##   roof = col_character(),
```

Reading in Data/Writing Out Data

```
scoreData
```

```
## # A tibble: 3,471 × 30
##    week  date   day season            awayTeam   AQ1   AQ2   AQ3   AQ4
##   <chr> <chr> <chr>  <int>               <chr> <int> <int> <int> <int>
## 1     1 5-Sep   Thu   2002 San Francisco 49ers     3     0     7     6
## 2     1 8-Sep   Sun   2002   Minnesota Vikings     3    17     0     3
## 3     1 8-Sep   Sun   2002  New Orleans Saints     6     7     7     0
## 4     1 8-Sep   Sun   2002       New York Jets     0    17     3    11
## 5     1 8-Sep   Sun   2002   Arizona Cardinals    10     3     3     7
## # ... with 3,466 more rows, and 21 more variables: AOT <int>, AOT2 <int>,
## #   AFinal <int>, homeTeam <chr>, HQ1 <int>, HQ2 <int>, HQ3 <int>,
## #   HQ4 <int>, HOT <int>, HOT2 <int>, HFinal <int>, stadium <chr>,
## #   startTime <time>, toss <chr>, roof <chr>, surface <chr>,
## #   duration <int>, attendance <chr>, weather <chr>, vegasLine <chr>,
## #   OU <chr>
```

Reading in Data/Writing Out Data

Reading in a comma separated value (.csv) file

- Notice: fancy printing!
- tidyverse data frames are special class
- Printing method optimal

```
attributes(scoreData)$class
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

Reading in Data/Writing Out Data

Reading in a comma separated value (.csv) file

- Notice: fancy printing!
- tidyverse data frames are special class
- Printing method optimal

```
attributes(scoreData)$class
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

- How did R determine the column types?

Reading in Data/Writing Out Data

- Checking column type is a basic data validation step
- Check out scoresStub.csv
- Look at season column type!

```
scoreStub <- read_csv("https://raw.githubusercontent.com/
                    jbpost2/DataScienceR/master/datasets/scoresStub.csv")
```

```
## Parsed with column specification:
## cols(
##   week = col_integer(),
##   date = col_character(),
##   day = col_character(),
##   season = col_character(),
##   awayTeam = col_character(),
##   AQ1 = col_integer(),
##   AQ2 = col_integer()
## )
```

24/57

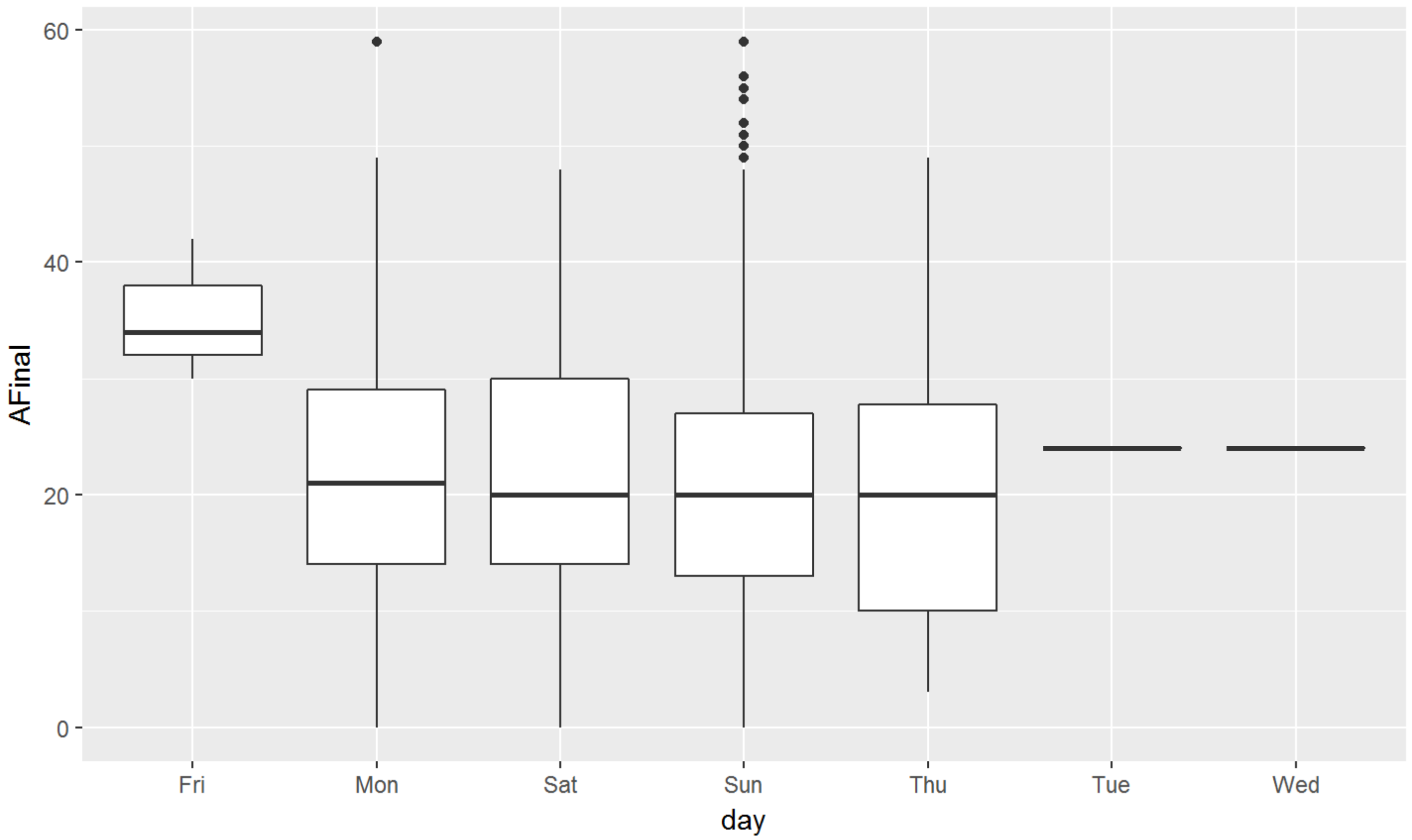
Reading in Data/Writing Out Data

```
scoreStub
```

```
## # A tibble: 9 × 7
##    week  date   day season            awayTeam   AQ1   AQ2
##   <int> <chr> <chr>  <chr>               <chr> <int> <int>
## 1     1 5-Sep   Thu   2002 San Francisco 49ers     3     0
## 2     1 8-Sep   Sun   2002   Minnesota Vikings     3    17
## 3     1 8-Sep   Sun   2002  New Orleans Saints     6     7
## 4     1 8-Sep   Sun   2002       New York Jets     0    17
## 5     1 8-Sep   Sun      a   Arizona Cardinals    10     3
## 6     1 8-Sep   Sun   2002 Philadelphia Eagles    14    10
## 7     1 8-Sep   Sun   2002  Indianapolis Colts     7     7
## 8     1 8-Sep   Sun   2002  Kansas City Chiefs     7     7
## 9     1 8-Sep   Sun   2002    Seattle Seahawks     7     0
```

- Can now make pretty plots (covered tomorrow)

```
ggplot(data = scoreData, aes(x = day, y = AFinal)) + geom_boxplot()
```



Reading in Data/Writing Out Data

- Base R `read.csv()`
  - Reads character variables as "factors" > - Factor - special class of vector
    - Great for variable with finite number of classes (**levels**)
    - Ex: day or week

Reading in Data/Writing Out Data

- Base R `read.csv()`
  - Reads character variables as "factors" > - Factor - special class of vector
    - Great for variable with finite number of classes (**levels**)
    - Ex: day or week

```
#overwrite day column with factor version
scoreData$day <- as.factor(scoreData$day)
levels(scoreData$day)
```

```
## [1] "Fri" "Mon" "Sat" "Sun" "Thu" "Tue" "Wed"
```

Reading in Data/Writing Out Data

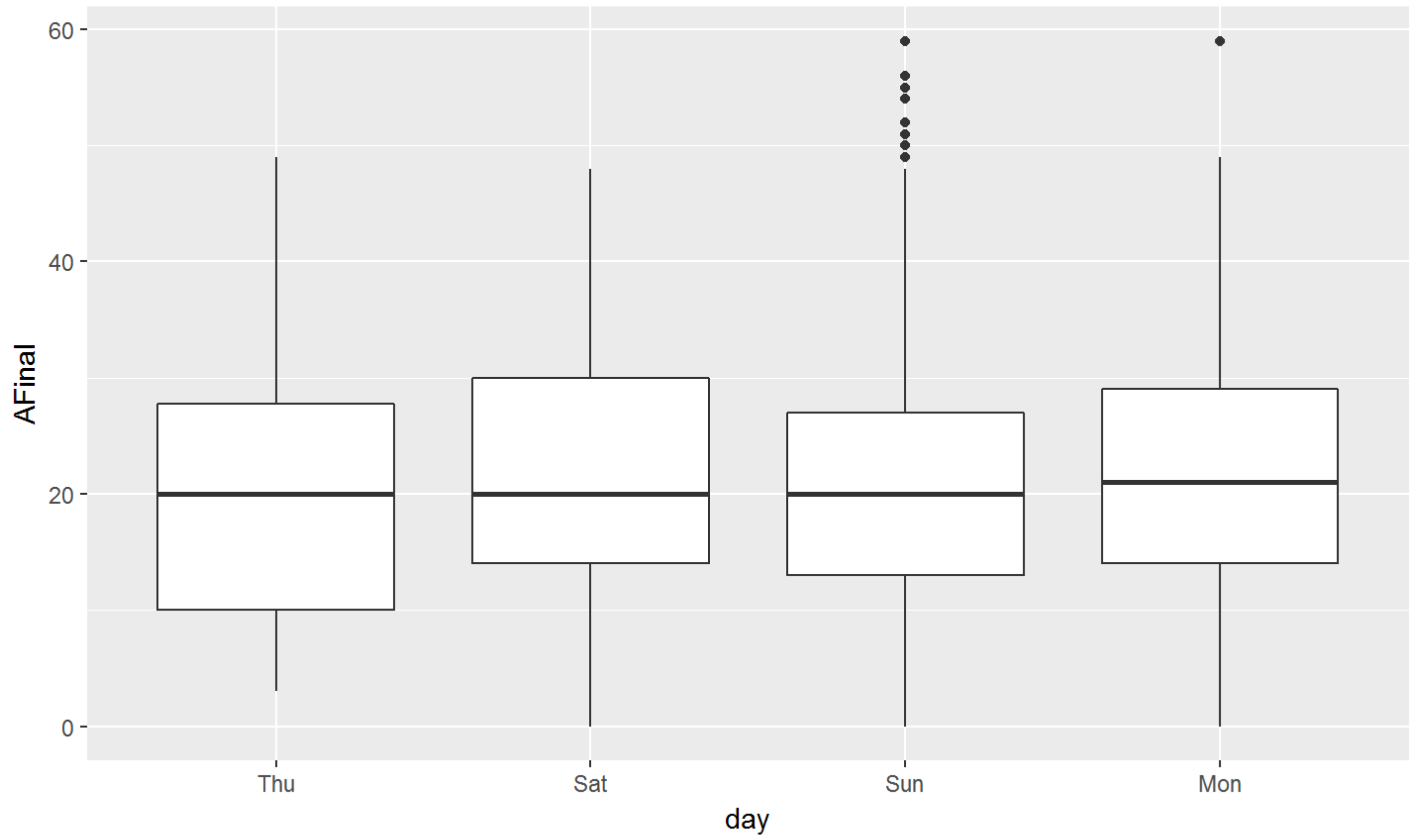
- Factor - special class of vector
  - Great for variable with finite number of classes
  - Can now reorder (useful when plotting)
  - Use `ordered` funtion on a *factor* to order the levels

```
scoreData$day <- ordered(scoreData$day,
                   levels = levels(scoreData$day)[c(7, 5, 1, 3, 4, 2, 6)])

levels(scoreData$day)
```

```
## [1] "Wed" "Thu" "Fri" "Sat" "Sun" "Mon" "Tue"
```

- Plot with reordered levels (remove F, T, W)



Reading in Data/Writing Out Data

Reading in any delimited file

- Read in umps.txt file (a '>' delimited file)
- Notice no column names provided
    - Year Month Day Home Away HPUmpire
- Use `read_delim()` (check help!)

```
umpData <- read_delim("https://raw.githubusercontent.com/
                    jbpost2/DataScienceR/master/datasets/umps2012.txt",
                    delim = ">",
     col_names = c("Year", "Month", "Day", "Home", "Away", "HPUmpire")
)
```

```
## Parsed with column specification:
## cols(
##   Year = col_integer(),
##   Month = col_integer(),
##   Day = col_integer(),
##   Home = col_character(),
##   Away = col_character(),
##   HPUmpire = col_character()
## )
```

Reading in Data/Writing Out Data

```
umpData
```

```
## # A tibble: 2,359 × 6
##    Year Month   Day  Home  Away         HPUmpire
##   <int> <int> <int> <chr> <chr>            <chr>
## 1  2012     4    12   MIN   LAA     D.J. Reyburn
## 2  2012     4    12    SD   ARI     Marty Foster
## 3  2012     4    12   WSH   CIN     Mike Everitt
## 4  2012     4    12   PHI   MIA      Jeff Nelson
## 5  2012     4    12   CHC   MIL Fieldin Culbreth
## # ... with 2,354 more rows
```

Reading in Data/Writing Out Data

Reading in any delimited file

- Functions from *readr* and their purpose

Delimiter	Function
comma ','	read_csv()
tab	read_tsv()
space ' '	read_table()
semi-colon ';'	read_csv2()
other	read_delim(…,delim = ,…)

Reading in Data/Writing Out Data

Fixed field data

- Open the cigarettes.txt file: Read using `read_fwf()`
- Can specify columns in many ways

```
#a guess based on reading a few columns
cigData <- read_fwf("https://raw.githubusercontent.com/jbpost2/
                  DataScienceR/master/datasets/cigarettes.txt",
                  col_positions =
                      fwf_empty("https://raw.githubusercontent.com/jbpost2/
                            DataScienceR/master/datasets/cigarettes.txt",
                 col_names = c("brand", "tar", "nicotine", "weight", "co"))
)
```

Reading in Data/Writing Out Data

```
cigData
```

```
## # A tibble: 24 × 5
##         brand   tar nicotine weight    co
##         <chr> <chr>    <chr>  <chr> <chr>
## 1       brand   tar nicotine weight   co\t
## 2      Alpine  14.1     0.86 0.9853  13.6
## 3      Benson  16.0     1.06 1.0938  16.6
## 4 CamelLights   8.0     0.67 0.9280  10.2
## 5     Carlton   4.1     0.40 0.9462   5.4
## # ... with 19 more rows
```

Reading in Data/Writing Out Data

Fixed field data

- Must skip first line!

```
#need to skip first line!
cigData<-read_fwf("https://raw.githubusercontent.com/
                  jbpost2/DataScienceR/master/datasets/cigarettes.txt",
                  col_positions = fwf_empty("https://raw.githubusercontent.com/jbpost2/
                       DataScienceR/master/datasets/cigarettes.txt",
                 col_names = c("brand", "tar", "nicotine", "weight", "co")),
       skip = 1
)
```

Reading in Data/Writing Out Data

```
cigData
```

```
## # A tibble: 23 × 5
##          brand   tar nicotine weight    co
##          <chr> <dbl>    <dbl>  <dbl> <dbl>
## 1       Alpine  14.1     0.86 0.9853  13.6
## 2       Benson  16.0     1.06 1.0938  16.6
## 3  CamelLights   8.0     0.67 0.9280  10.2
## 4      Carlton   4.1     0.40 0.9462   5.4
## 5 Chesterfield  15.0     1.04 0.8885  15.0
## # ... with 18 more rows
```

Reading in Data/Writing Out Data

Fixed field data

- Can specify columns in many ways

```
#another option
cigData<-read_fwf("https://raw.githubusercontent.com/jbpost2/
                   DataScienceR/master/datasets/cigarettes.txt",
                   col_positions =
                                  fwf_widths(c(17, 4, 5, 11, 4),
                col_names = c("brand", "tar", "nicotine", "weight", "co")),
                    skip = 1
  )
```

Reading in Data/Writing Out Data

Other useful functions for tricky data

- read_file()
  - reads an entire file into a single string
- read_lines()
  - reads a file into a character vector with one element per line

Reading in Data/Writing Out Data

Excel Data

- Read in censusEd.xls
- Unfortunately can't xls from gitHub easily
- Download censusEd.xls
- Place in folder called 'datasets' in working directory

Reading in Data/Writing Out Data

Excel Data

- Read in censusEd.xls
- Using `read_excel()` from `readxl` package
  - Reads both xls and xlsx files
  - Detects format from extension given
  - Specify sheet with name or integers (or NULL for 1st)

```
library(readxl)
#just first sheet
edData <- read_excel("datasets/censusEd.xls", sheet = "EDU01A")
```

Reading in Data/Writing Out Data

```
edData
```

```
## # A tibble: 3,198 × 42
##       Area_name STCOU EDU010187F EDU010187D EDU010187N1 EDU010187N2
##           <chr> <chr>      <dbl>      <dbl>       <chr>       <chr>
## 1 UNITED STATES 00000          0   40024299        0000        0000
## 2       ALABAMA 01000          0     733735        0000        0000
## 3   Autauga, AL 01001          0       6829        0000        0000
## 4   Baldwin, AL 01003          0      16417        0000        0000
## 5   Barbour, AL 01005          0       5071        0000        0000
## # ... with 3,193 more rows, and 36 more variables: EDU010188F <dbl>,
## #   EDU010188D <dbl>, EDU010188N1 <chr>, EDU010188N2 <chr>,
## #   EDU010189F <dbl>, EDU010189D <dbl>, EDU010189N1 <chr>,
## #   EDU010189N2 <chr>, EDU010190F <dbl>, EDU010190D <dbl>,
## #   EDU010190N1 <chr>, EDU010190N2 <chr>, EDU010191F <dbl>,
## #   EDU010191D <dbl>, EDU010191N1 <chr>, EDU010191N2 <chr>,
## #   EDU010192F <dbl>, EDU010192D <dbl>, EDU010192N1 <chr>,
## #   EDU010192N2 <chr>, EDU010193F <dbl>, EDU010193D <dbl>,
## #   EDU010193N1 <chr>, EDU010193N2 <chr>, EDU010194F <dbl>,
## #   EDU010194D <dbl>, EDU010194N1 <chr>, EDU010194N2 <chr>,
```

Reading in Data/Writing Out Data

Excel Data

- Using `read_excel()` from `readxl` package
  - Specify sheet with name or integers (or `NULL` for 1st)
  - Look at sheets available

```
excel_sheets("datasets/censusEd.xls")
```

```
##  [1] "EDU01A" "EDU01B" "EDU01C" "EDU01D" "EDU01E" "EDU01F" "EDU01G"
##  [8] "EDU01H" "EDU01I" "EDU01J"
```

Reading in Data/Writing Out Data

Excel Data

- Using `read_excel()` from readxl package
  - Specify cells with contiguous range

```
library(readxl)
#just first sheet
edData <- read_excel("datasets/censusEd.xls", sheet = "EDU01A",
                     range = cell_cols("A:D")
                     )
```

Reading in Data/Writing Out Data

```
edData
```

```
## # A tibble: 3,198 × 4
##       Area_name STCOU EDU010187F EDU010187D
##           <chr> <chr>      <dbl>      <dbl>
## 1 UNITED STATES 00000          0   40024299
## 2       ALABAMA 01000          0     733735
## 3   Autauga, AL 01001          0       6829
## 4   Baldwin, AL 01003          0      16417
## 5   Barbour, AL 01005          0       5071
## # ... with 3,193 more rows
```

Excel Data Recap

Using `read_excel()` from `readxl` package

- Reads both xls and xlsx files
- Specify sheet with name or integers (or `NULL` for 1st)
  - Use `sheet = "name"` or `sheet = #`
- Look at sheets available
  - Use `excel_sheets`
- Specify cells with continguous range
  - `range = cell_cols("...")`
  - `range = cell_rows("...")`
- Specify cells
  - `range = "R1C2:R2C5"`

47/57

Reading in Data/Writing Out Data

SPSS Data

- SPSS data has extension ".sav"
- Read in bodyFat.sav
- Use `read_spss()` from `haven` package
- Not many options!

```
library(haven)
bodyFatData <- read_spss("https://github.com/jbpost2/
                      DataScienceR/blob/master/datasets/
                      bodyFat.sav?raw=true.sav")
```

Reading in Data/Writing Out Data

```
bodyFatData
```

```
## # A tibble: 20 × 4
##       y    x1    x2    x3
##   <dbl> <dbl> <dbl> <dbl>
## 1  19.5  43.1  29.1  11.9
## 2  24.7  49.8  28.2  22.8
## 3  30.7  51.9  37.0  18.7
## 4  29.8  54.3  31.1  20.1
## 5  19.1  42.2  30.9  12.9
## # ... with 15 more rows
```

Reading in Data/Writing Out Data

SAS Data

- SAS data has extension '.sas7bdat'
- Read in smoke2003.sas7bdat
- Use `read_sas()` from haven package
- Not many options!

```
smokeData <- read_sas("https://github.com/jbpost2/
                    DataScienceR/blob/master/datasets/
                    smoke2003.sas7bdat?raw=true")
```

Reading in Data/Writing Out Data

```
smokeData
```

```
## # A tibble: 443 × 54
##    SEQN SDDSRVYR RIDSTATR RIDEXMON RIAGENDR RIDAGEYR RIDAGEMN RIDAGEEX
##   <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 21010        3        2        2        2       52      633      634
## 2 21012        3        2        2        1       63      765      766
## 3 21048        3        2        1        2       42      504      504
## 4 21084        3        2        1        2       57      692      693
## 5 21093        3        2        1        2       64      778      778
## # ... with 438 more rows, and 46 more variables: RIDRETH1 <dbl>,
## #   RIDRETH2 <dbl>, DMQMILIT <dbl>, DMDBORN <dbl>, DMDCITZN <dbl>,
## #   DMDYRSUS <dbl>, DMDEDUC3 <dbl>, DMDEDUC2 <dbl>, DMDEDUC <dbl>,
## #   DMDSCHOL <dbl>, DMDMARTL <dbl>, DMDHHSIZ <dbl>, INDHHINC <dbl>,
## #   INDFMINC <dbl>, INDFMPIR <dbl>, RIDEXPRG <dbl>, DMDHRGND <dbl>,
## #   DMDHRAGE <dbl>, DMDHRBRN <dbl>, DMDHREDU <dbl>, DMDHRMAR <dbl>,
## #   DMDHSEDU <dbl>, SIALANG <dbl>, SIAPROXY <dbl>, SIAINTRP <dbl>,
## #   FIALANG <dbl>, FIAPROXY <dbl>, FIAINTRP <dbl>, MIALANG <dbl>,
## #   MIAPROXY <dbl>, MIAINTRP <dbl>, AIALANG <dbl>, WTINT2YR <dbl>,
## #   WTMEC2YR <dbl>, SDMVPSU <dbl>, SDMVSTRA <dbl>, Gender <dbl>,
```

Reading in Data/Writing Out Data

SAS Data

- Note: Variables had SAS labels. Don't show on print!
  - Will show on View(smokeData) (or click on data from environment)

```
str(smokeData)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    443 obs. of  54 variables:
##  $ SEQN      : atomic  21010 21012 21048 21084 21093 ...
##   ..- attr(*, "label")= chr "Patient ID"
##  $ SDDSRVYR  : atomic  3 3 3 3 3 3 3 3 3 3 ...
##   ..- attr(*, "label")= chr "Data Release Number"
##  $ RIDSTATR  : atomic  2 2 2 2 2 2 2 2 2 2 ...
##   ..- attr(*, "label")= chr "Interview/Examination Status"
##  $ RIDEXMON  : atomic  2 2 1 1 1 2 1 2 1 1 ...
##   ..- attr(*, "label")= chr "Six month time period"
##  $ RIAGENDR  : atomic  2 1 2 2 2 2 1 2 1 2 ...
##   ..- attr(*, "label")= chr "Gender 1=M 2=F"
```

Reading in Data/Writing Out Data

SAS Data

- Note: Variables had SAS labels. Don't show on print!
  - Will show on View(smokeData) (or click on data from environment)
  - Can access via

```
attr(smokeData$SDDSRVYR, "label")
```

```
## [1] "Data Release Number"
```

Reading in Data/Writing Out Data

Writing Data

- Usually write to .csv (or other delimiter)
- Use `write_csv()` from `readr` package
- Check help!
  - Will write to path or working directory

```
write_csv(x = smokeData,
          path = "E:/Other/DataScienceR/datasets/output/smokeData.csv")
```

54/57

Reading in Data/Writing Out Data

Writing Data

- Usually write to .csv (or other delimiter)
- Use `write_csv()` from `readr` package
- Check help!
    - Will write to path or working directory
    - `append` option won't overwrite but structures must match…

```
write_csv(x = bodyFatData,
          path = "E:/Other/DataScienceR/datasets/smokeData.csv",
          append = TRUE)
```

Recap

- Reading Data

Type of file	Package	Function
Delimited	`readr`	`read_csv()`, `read_tsv()`,`read_table()`, `read_delim(...,delim = ,...)`
Excel (.xls,.xlsx)	`readxl`	`read_excel`
SPSS (.sav)	`haven`	`read_spss`
SAS (.sas7bdat)	`haven`	`read_sas`

- Write data with `write_csv()` from readr

Activity

- **Reading/Writing Data Activity** instructions available on web
- Work in small groups
- Ask questions! TAs and I will float about the room
- Feel free to ask questions about anything you didn't understand as well!