

Part 6: Putting your project on Github

In the previous chapters, we've set up a Github account, learned how to download the files from a Github repository, upload files into RStudio.Cloud, create and run R code, and commit changes using Git.

In this final chapter, we are going to create some figures and graphs for this project, then put these changes on Github in a way for people to find and share.

Moving R code into the .Rmd file

In the last chapter, we used the `README.Rmd` file to upload our data into RStudio. We'll continue using this file to add the contents from one additional .R scripts, `02-wrangle.R`. We will then create new section and script for visualizing the data (`03-visualize.R`).

We've provided a lot of code and comments in the `02-wrangle.R` script for you to explore, revise, and adapt to your liking. In the next few sections, we are going to move the code from the `02-wrangle.R` script into two new sections of the `README.Rmd` (you probably guessed it "Wrangle").

A quick lesson in compassionate programming

Your code will always be communicating to at least two audiences: your computer, and your future self. Be nice to both of them!*

Things like the pipe `%>%` in R can help with clarity. The pipe is part of the [magrittr package](#) and it takes code written like this:

```
outer_function(inner_function(Data_X), Data_Y)
```

And makes it look like this:

```
Data_X %>% # do this
  inner_function() %>% # then do this
  outer_function(Data_Y)
```

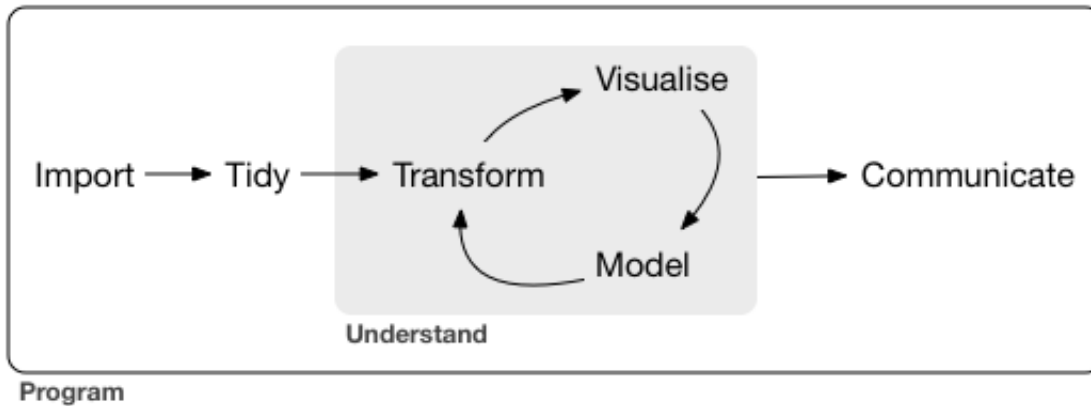
`%>%` is a form of [syntactic sugar](#), which is a fancy way of saying *"something that helps us communicate better."*

You'll see the pipe throughout the project's R code files, and you can always read it as, *"do this, then do this."*

Wrangling code

Data wrangling (or cleaning, or munging) is whatever steps need to be taken to take the raw data (which we always remember not to change) into something we can use to create a table, visualization, model, etc.

The `02-wrangle.R` script prepares the data from the `01-import.R` so that they can be used for the visualizations. If you think back to the process outlined in the figure in [R for Data Science](#), you will notice that wrangle isn't listed explicitly. This is because both **'Tidy'** and **'Transform'** would be considered wrangling steps (both of these need to happen before any visualizations or models can be properly run).



The simplest way to include this script in the README.Rmd file is to create a code chunk, insert the `base::source()` function, and enter the path to the 02-wrangle.R file.

However, we want to be nice to our future selves, so we will include some language that describes what the functions are doing above each code chunk.

Wrangling the data sets

The first data that need to be wrangled are the Wikipedia tables (seen in the 02-wrangle.R file on the section below).

```

21 # import -----
22 source("code/01-import.R")
23
24 # wrangle wikipedia data night 1 -----
25 WikiDemAirTime01 <- WikiDemAirTime01Raw %>%
26   magrittr::set_colnames(value = c("candidate", "airtime_night1"))
27
28 WikiDemAirTime01 <- WikiDemAirTime01 %>%
29   dplyr::filter(candidate %nin% c("Night one airtime", "Candidate") & |
30     airtime_night1 %nin% c("Night one airtime", "Airtime (min.)[58]")) %>%
31   dplyr::mutate(airtime_night1 = as.numeric(airtime_night1))
32
33 # wrangle wikipedia data night 2 -----
34 WikiDemAirTime02 <- WikiDemAirTime02Raw %>%
35   magrittr::set_colnames(value = c("candidate", "airtime_night2"))
36 WikiDemAirTime02 <- WikiDemAirTime02 %>%
37   dplyr::filter(candidate %nin% c("Night two airtime", "Candidate") &
38     airtime_night2 %nin% c("Night one airtime", "Airtime (min.)[58]")) %>%
39   dplyr::mutate(airtime_night2 = as.numeric(airtime_night2))

```

packages
import
wrangle wikipedia data night 1
wrangle wikipedia data night 2
wrangle polling criterion data
export wiki tables
wrangle Google trend data
bind
gender
join Gtrend with wikipedia data
poll_perc_cat
mapping data (by region)
create processed data folder
export GtrendDems2020Intere...
export GtrendWikiOTAirTime

These data were stored on the web in Wikipedia (.html) tables. We will create some new column names, remove some columns that used to be headers, and make the airtime variable numeric.

The polling criterion Wikipedia data starting at the section titled, wrangle polling criterion data. This section actually creates a list of candidates (in `cand_names_wiki`) and uses it to filter out the observations we want. Check out [this webinar](#) to get an understanding of how `dplyr`'s verbs work.

```

# wrangle polling criterion data -----
# create list from names using dput()
# dput(WikiPollCriterionRaw[ 1:11, 1])

```

```
cand_names_wiki <- c("Warren[note 2]",
                    "O'Rourke[note 2]",
                    "Booker[note 2]",
                    "Klobuchar[note 2]",
                    "Castro[note 2]",
                    "Gabbard",
                    "Ryan",
                    "Inslee",
                    "de Blasio",
                    "Delaney")

# subset WikiPollCriterionRaw with list from above
WikiPollCriterion <- WikiPollCriterionRaw %>%
  # this will remove all candidates not listed above
  dplyr::filter(`Candidates drawn for the June 26 debate` %in% cand_names_wiki)
```

After the Wikipedia tables have been wrangled, the script exports these files to a new processed/ folder. This helps ensure they won't be accidentally altered or mistaken for the data files in the raw/ data folder.

The export section also timestamps each file so we know the last time it was created. Read more about importing and exporting data in [this RStudio cheatsheet](#).

The Google trend data are a little more complicated because they come into RStudio.Cloud as a list, which is a data container in R that [doesn't have to be rectangular](#).

The image below outlines what each portion of code is doing. These are fairly common wrangling tasks, so we recommend going back or bookmarking these files as a reference.

The screenshot shows an RStudio window with two tabs: 'README.Rmd' and '02-wrangle.R'. The '02-wrangle.R' tab is active, displaying R code for wrangling data. The code is annotated with three orange callout boxes explaining specific tasks:

- Get data out of an R object into a rectangle we can manipulate easily:** This annotation points to lines 106-112, which convert Google trend data from a list to tibbles.


```
# wrangle Google trend data -----
# convert Dems2020Group1 to tibble
GTrendDems2020Group1IOT <- GTrendDems2020Night1G1$interest_over_time %>%
  as_tibble()
# convert Dems2020Group2 to tibble
GTrendDems2020Group2IOT <- GTrendDems2020Night1G2$interest_over_time %>%
  as_tibble()
```
- Stick two data sets (rectangles) together:** This annotation points to lines 120-123, which bind two tibbles into a single data frame.


```
# bind -----
GTrendDems2020Debate01IOT <- bind_rows(GTrendDems2020Group1IOT,
    GTrendDems2020Group2IOT,
    .id = "data")
```
- Create a new variable (column) on multiple conditions:** This annotation points to lines 125-131, which create a 'gender' variable based on keyword detection.


```
# gender -----
GTrendDems2020Debate01IOT <- GTrendDems2020Debate01IOT %>%
  dplyr::mutate(gender = case_when(
    stringr::str_detect(keyword, "Elizabeth Warren") ~ "Women",
    stringr::str_detect(keyword, "Amy Klobuchar") ~ "Women",
    stringr::str_detect(keyword, "Tulsi Gabbard") ~ "Women",
    TRUE ~ "Men"))
```

On the right side of the RStudio window, the 'Environment' pane shows the following objects:

```
packages
import
wrangle wikipedia data night 1
wrangle wikipedia data night 2
wrangle polling criterion data
export wiki tables
wrangle Google trend data
bind
gender
join Gtrend with wikipedia data
poll_perc_cat
mapping data (by region)
create processed data folder
export GTrendDems2020Intere...
export GTrendWikiIOTAirTime
```

We have different sources of data in RStudio right now (Wikipedia and Google trend data). They both have information on Candidates though. Often times we'll want to join two (or more) data sets on a common column (like candidates). We will perform an example of this in the section outlined below.

<pre> # join Gtrend with wikipedia data ----- # sort alphabetically, join on id WikiDemAirTime01 <- WikiDemAirTime01 %>% dplyr::arrange(desc(candidate)) # add id WikiDemAirTime01 <- WikiDemAirTime01 %>% mutate(candidate_id = row_number()) # create candidate_id for GTrendDems2020Debate01IOT GTrendDems2020Debate01IOT <- GTrendDems2020Debate01IOT %>% dplyr::mutate(candidate_id = case_when(stringr::str_detect(string = keyword, pattern = "Warren") ~ 1, stringr::str_detect(string = keyword, pattern = "Ryan") ~ 2, stringr::str_detect(string = keyword, pattern = "Beto") ~ 3, stringr::str_detect(string = keyword, pattern = "Klobuchar") ~ 4, stringr::str_detect(string = keyword, pattern = "Inslee") ~ 5, stringr::str_detect(string = keyword, pattern = "Gabbard") ~ 6, stringr::str_detect(string = keyword, pattern = "Delaney") ~ 7, stringr::str_detect(string = keyword, pattern = "de Blasio") ~ 8, stringr::str_detect(string = keyword, pattern = "Castro") ~ 9, stringr::str_detect(string = keyword, pattern = "Booker") ~ 10)) %>% dplyr::arrange(desc(candidate_id)) # Join WikiDemAirTime01 to GTrendDems2020Debate01IOT on candidate_id GtrendWikiIOTAirTime <- GTrendDems2020Debate01IOT %>% dplyr::left_join(x = ., y = WikiDemAirTime01, by = "candidate_id") </pre>	<pre> gender join Gtrend with wikipedia data poll_perc_cat mapping data (by region) create processed data folder export GTrendDems2020Intere... export GTrendWikiIOTAirTime </pre> <p><i>Prep work to get a common id in each data set</i></p> <p><i>Joining two data sets 'candidate_id'</i></p>
---	---

The process usually isn't so involved, but we included extra to give more explicit instructions. Be sure to check out the [relational data chapter](#) the R for Data Science book.

We'll also be creating a map with the Google (or Twitter) data. Doing this requires another common task, which is loading a dataset from a package in R. The code below loads a state-level map into RStudio.Cloud and joins it to the Google trend data.

<pre> # mapping data (by region) ----- # convert to tibble (another data structure in R) GtrendDems2020IBRGroup1 <- tibble::as_tibble(GtrendDems2020Night1G1\$interest_by_region) GtrendDems2020IBRGroup2 <- tibble::as_tibble(GtrendDems2020Night1G2\$interest_by_region) # bind Dems2020IBRGroup1 Dems2020IBRGroup2 together GtrendDems2020IBR <- bind_rows(GtrendDems2020IBRGroup1, GtrendDems2020IBRGroup2, .id = "data") # convert the region to lowercase GtrendDems2020InterestByRegion <- GTrendDems2020IBR %>% dplyr::mutate(region = stringr::str_to_lower(location)) # create a data set for the states in the US statesMap = ggplot2::map_data("state") # now merge the two data sources together GtrendDems2020InterestByRegion <- GTrendDems2020InterestByRegion %>% dplyr::inner_join(x = ., y = statesMap, by = "region") </pre>	<pre> wrangle Google trend data bind gender join Gtrend with wikipedia data poll_perc_cat mapping data (by region) create processed data folder export GTrendDems2020Intere... export GTrendWikiIOTAirTime </pre> <p><i>Load the data from the ggplot2 package and join it to the Google trend data</i></p>
---	---

The Google trend data are also exported with time-stamp into the processed/ folder. We should continue adding the code into the README.Rmd file until we're confident all the functions will run and we don't see any errors.

README.Rmd x

O2-wrangle.R x

ABC

Knit

Insert

Run

```

130
131 ## Wrangle Google trend data
132
133 ```{r wrangle-google}
134 # wrangle Google trend data -----
135 # convert Dems2020Group1 to tibble
136 GTrendDems2020Group1IOT <- GTrendDems2020Night1G1$interest_over_time %>%
137   as_tibble()
138 # convert Dems2020Group2 to tibble
139 GTrendDems2020Group2IOT <- GTrendDems2020Night1G2$interest_over_time %>%
140   as_tibble()
141
142 # create numeric hits
143 GTrendDems2020Group1IOT <- GTrendDems2020Group1IOT %>%
144   dplyr::mutate(hits = as.numeric(hits))
145 GTrendDems2020Group2IOT <- GTrendDems2020Group2IOT %>%
146   dplyr::mutate(hits = as.numeric(hits))
147
148 # bind -----
149 GTrendDems2020Debate01IOT <- bind_rows(GTrendDems2020Group1IOT,
150   GTrendDems2020Group2IOT,
151   .id = "data")
152
153 # gender -----
154 GTrendDems2020Debate01IOT <- GTrendDems2020Debate01IOT %>%
155   dplyr::mutate(gender = case_when(
156     stringr::str_detect(keyword, "Elizabeth Warren") ~ "Women",
157     stringr::str_detect(keyword, "Amy Klobuchar") ~ "Women",
158     stringr::str_detect(keyword, "Tulsi Gabbard") ~ "Women",
159     TRUE ~ "Men"))
160
161 # get distinct
162 GTrendDems2020Debate01IOT <- GTrendDems2020Debate01IOT %>% distinct()
163 # GTrendDems2020Debate01IOT %>% glimpse(78)
164 ```
165

```

Motivation

Data files

Importing data

Wrangle wikipedia data

Wrangle Google trend data

Join Google trend and Wikipedia data

Add polling percentage categories variable

Mapping data

Export wrangled data

↑

These sections correspond to similar sections in the O2-wrangle.R script.

Each section can get a new code block and whatever description you want to include.

Note: The O2-wrangle.R file is in the code/ folder, but you won't have to alter the file paths because you're using an RStudio project file. Read more about how these are so helpful to your workflow [here](#).

Visualizations

OK, we've completed our section for the wrangling the data. We are going to insert a divider (***) and start a new visualize section (## Visualize) in the README.Rmd file.

We've created a O3-visualize.Rmd file for you to download from Github. You can do this by typing the following code into your README.Rmd file:


```
README.Rmd x
306
307
308 ## Visualize copy this code into your README.Rmd file
309
310 Download this file for the visualization script: http://bit.ly/viz-2019-data
311
312 ```{r download-vis-rmd}
313 download.file(url = "http://bit.ly/viz-2019-data",
314               destfile = "03-visualize.Rmd")
315 ```
```

- Motivation
- Data files
- Importing data
- Wrangle Wikipedia data
- Wrangle Google trend data
- Join Google trend and Wikipedia data
- Add polling percentage categories variable
- Mapping data
- Export wrangled data
- Visualize

The hyperlink is here: <http://bit.ly/viz-2019-data>

After we've downloaded our 03-visualize.Rmd file, we can open this file and copy the code starting from the line just below the # Visualize data heading (it should be on about line 65) and extending all the way to the end of the file.

```
README.Rmd x 03-visualize.Rmd x
59
60 ```{r wrangle, eval=TRUE, message=FALSE, warning=FALSE}
61 source("code/02-wrangle.R")
62 ```
63
64 # Visualize data
65
66 Start with visualizing as much of the data as possible. These two graphs
67 answer the following two questions about the Twitter data: 1) "what kind of
68 variables are in the data set?" and 2) "how much are missing?"
69
70 ```{r visdat-inspectdf}
71 inspectdf::inspect_types(TwitterData) %>%
72 inspectdf::show_plot()
73 visdat::vis_miss(TwitterUsersData) +
74 ggplot2::coord_flip()
75 ```
76
77 ## Table summaries
78
79 We will use tables and graphs to explore the data we imported and wrangled
80 and see if it looks like the `fivethirtyeight` data. We will start with a
81 table of the data on voter preferences on candidates before and after the
82 night of the election.
83
84 ```{r GSheetCand538Fav}
85 dplyr::filter(GSheetCand538Fav, candidate %in% c("Elizabeth Warren",
86 "Tim Ryan", "Beto O'Rourke", "Amy Klobuchar", "Jay Inslee",
87 "Tulsi Gabbard", "John Delaney", "Bill de Blasio", "Julian Castro",
88 "Cory Booker")) %>%
89 DT::datatable(data = ., colnames = c("Democratic Candidate",
90 "Before the election", "After the first election",
91 "After the second election"),
92               caption = 'source: https://53eig.ht/2Yg0Smp')
93 ```
```

Visualization Code

Copy all the code highlighted in blue into the README.Rmd file.

- Motivation
- Packages
- Import data
- Wrangle data
- Visualize data
- Table summaries
- Visualize Google trend data
- Candidates with high polling criterions
- Candidates with low polling criterions
- Candidates with low polling criterions
- Women candidates
- Men candidates
- Twitter data
- Summarize Interest Data
- Map the data by state
- Tulsi Gabbard 2020 searches
- Cory Booker 2020 searches
- Sharing your work
- Ask more questions

106:24 Chunk 8: top-3-candidates R Markdown

After selecting the code from 03-visualize.Rmd, we should click on the line directly under the previous code chunk we used to download the .Rmd file.

After pasting the code from 03-visualize.Rmd into the README.Rmd file, we can click on the *Run > Run All Chunks Below* (this will run all the code starting at line 319 until the end of the document).

The screenshot shows the RStudio Cloud interface. The top pane displays R code chunks. Chunk 11 (lines 308-314) contains code to download a file from a URL. The console output shows the download was successful, with a message: "trying URL 'http://bit.ly/viz-2019-data' Content type 'text/plain; charset=utf-8' length 10650 bytes (10 KB) downloaded 10 KB". A green arrow points from the text "Run all the code chunks below the download.file() function!" to the "Run All Chunks Below" option in the run menu. The bottom pane shows a markdown chunk (lines 316-329) with text about visualizing data. A red box highlights the text "Copied Visualization Code" and the subsequent markdown text. The right sidebar shows a list of search results related to the data visualization.

```
308
309 Download this file for the visualization script: http://bit.ly/viz-2019-data
310
311 ```{r download-vis-rmd}
312 download.file(url = "http://bit.ly/viz-2019-data",
313               destfile = "03-visualize.Rmd")
314 ```
```

trying URL 'http://bit.ly/viz-2019-data'
Content type 'text/plain; charset=utf-8' length 10650 bytes (10 KB)
downloaded 10 KB

Run all the code chunks below the download.file() function!

```
315
316 Start with visualizing as much of the data as possible. These two questions answer the following two questions about the Twitter data: 1) "what kind of variables are in the data set?" and 2) "how much are missing?"
317
318
319 ```{r visdat-inspectdf}
320 inspectdf::inspect_types(TwitterData) %>%
321   inspectdf::show_plot()
322 visdat::vis_miss(TwitterUsersData) +
323   ggplot2::coord_flip()
324 ```
```

Table summaries

We will use tables and graphs to explore the data we imported and wrangled and see if it looks like the 'fivethirtyeight' data. We will start with a table of the data on voter preferences on candidates before and after the night of the election.

```
329
330 ```{r GSheetCand538Fav}
331 dplyr::filter(GSheetCand538Fav, candidate %in% c("Elizabeth Warren"))
332 ```
```

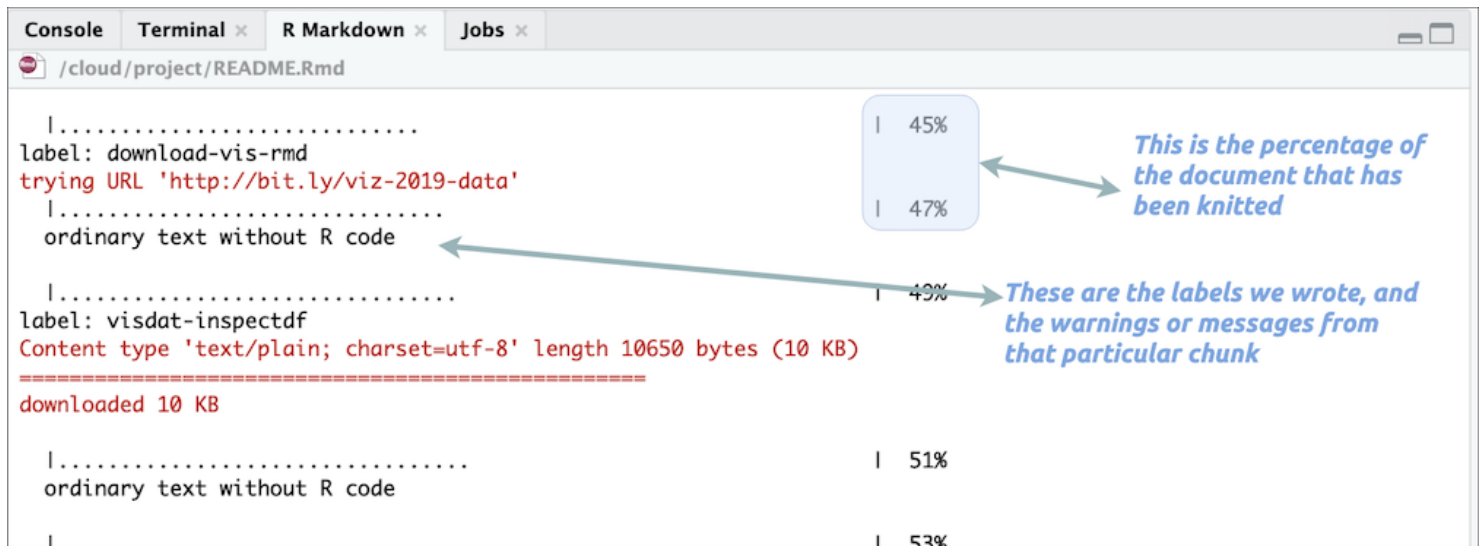
Run Selected Line(s)
Run Current Chunk
Run Next Chunk
Run Setup Chunk
Run Setup Chunk Automatically
Run All Chunks Above
Run All Chunks Below
Restart R and Run All Chunks
Restart R and Clear Output
Run All

Map the data by state
Tulsi Gabbard 2020 searches
Cory Booker 2020 searches
Sharing your work
Ask more questions

Running the code will create multiple tables and figures in the README.Rmd file. We'll go over these in more depth below. For now, we'll follow the directions at the bottom of the pasted code and "Click knit to get the markdown file to share."

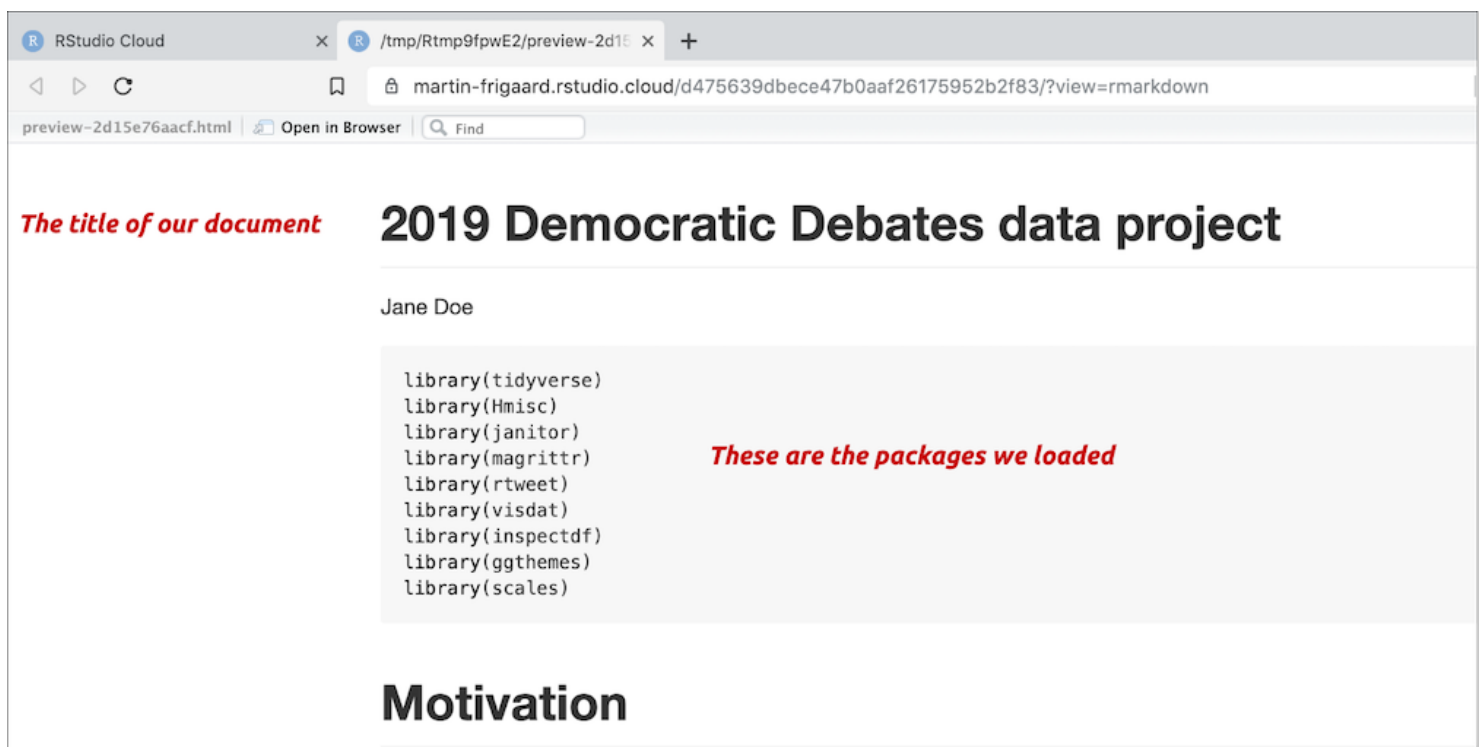
Knitting RMarkdown files

Clicking *Knit* (or clicking `shift+cmd+k`) activates the **Markdown** pane in RStudio.Cloud, and we see the code chunks being run for the entire document.



When the knitting process completes, a new browser window will pop up with our README.md document. The README.md will have sections of formatted text (from the Markdown), R code, and the various outputs.

The top of the file should list the title and the packages:



Lets scroll down to the visualize section and look at the section titled, **Candidates with high polling criterions**). We can see the different parts of the Rmarkdown file in the image below:

Header & text for this section

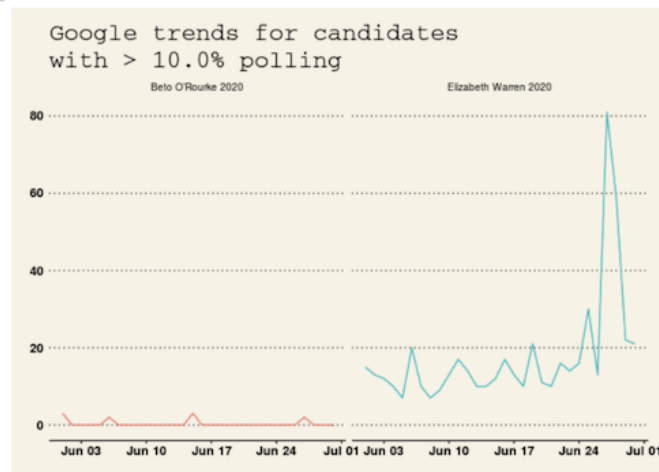
Candidates with high polling criterions

When we look at the candidates with the highest percent of polling criterion on the 26th, we see the following:

Code and comments for this graph

```
GtrendWikiIOTAirTime %>%
# limit to only the candidates with more than 10% of voters
dplyr::filter(poll_perc_fct == "> 10.0% of voters") %>%
# put data on the x,
ggplot2::ggplot(aes(x = date,
# hits on the y
y = hits,
# color it by keyword
color = keyword)) +
# use a line
ggplot2::geom_line(aes(group = keyword), show.legend = FALSE) +
# labels
ggplot2::labs(
x = "Date",
y = "Google search hits",
subtitle = "Google trends for candidates \nwith > 10.0% polling") +
# add themes from ggthemes
ggthemes::theme_wsj(base_size = 9.5,
title_family = "mono") +
# use facets for both candidates on the same graph
ggplot2::facet_wrap(~ keyword, ncol = 2)
```

Graph output

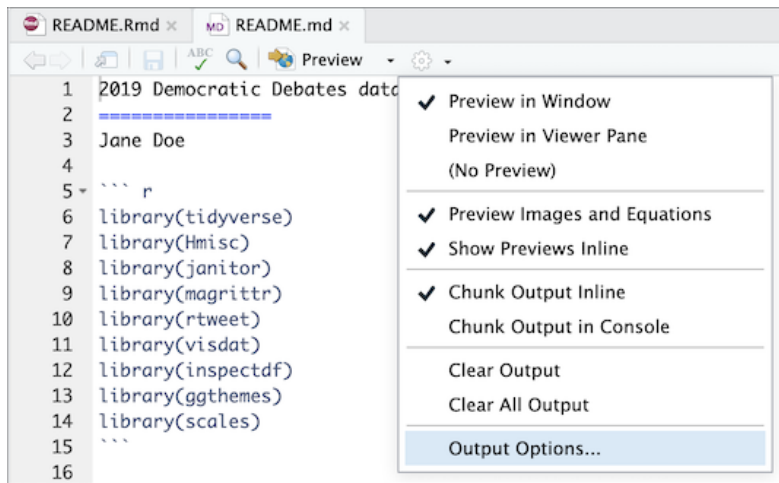


More text interpreting the output

This shows Warren doing well after the first night. If we narrow this down to the week of the debates and widen the list of candidates, we see the following:

The file output is actually a *Preview* of our markdown file (README.md). Our browser renders the markdown as a webpage (README.html).

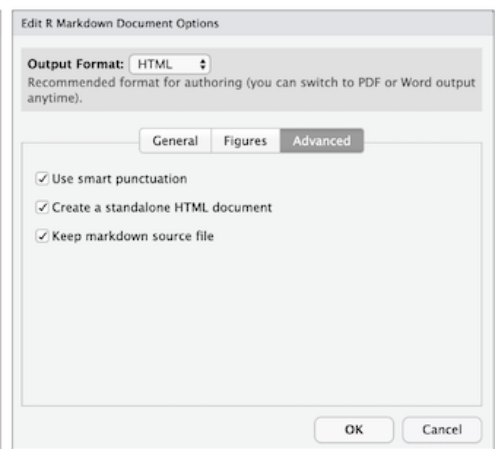
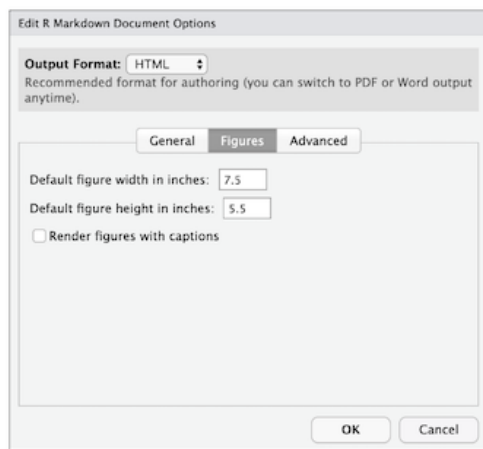
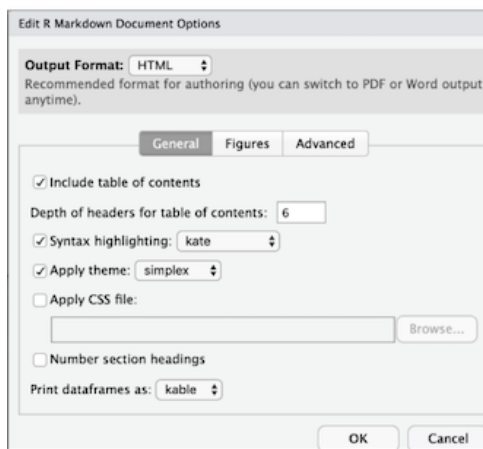
Let's keep a version of this file in .html. We can do this by opening the README.md file in the **Source** pane, and clicking on the small gear next to the *Preview* button. Follow the directions in the figure below to setup the .html output file.



Markdown document options

Click on the small gear icon to get the list of settings for the markdown options.

*From this list, select the **Output Options***



Change the settings to match these options

When we click *Ok* and look at the top of the README.md file we see another YAML header.

```
---
output:
  html_document:
    df_print: kable
    fig_height: 5.5
    fig_width: 7.5
    highlight: kate
    keep_md: yes
    theme: simplex
    toc: yes
    toc_depth: 6
---
```

The settings displayed above are some of the ways we can customize our .Rmd files. Read more about the html documents [here in the Rmarkdown guide](#).

To see what these settings do, we will click *Preview* on the README.md file.

This conversion is incredibly handy for weaving formatted text, code, and media (tables, images, and graphs).

Extracting the .R from the .Rmd

But now we have all our visualize code in the `03-visualize.Rmd` file—what if we wanted this code in an `.R` script?

We can run the following code in the **Console** pane.

```
knitr::purl("03-visualize.Rmd")
```

We'll see the following script file is generated.

Push the changes to Github