# reticulate: R Interface to Python

author: Michael Espero date: 2020-1-28 autosize: true

## About Me

@michaelespero

**I'm very happy to take part in the R Community.**

- PhD Student at Claremont Graduate University
  - Biostatistics, Epidemiology
  - Neurocognitive Sciences
- I enjoy statistics and data science with R, Python, Julia, and SQL.

## Rewind

What did I learn last year from the R Community?

- R people are friendly and happy to share.
  - We're all still learning!

## Rewind

What did I learn last year from the R Community?

- R people are very diverse.
  - Academics
  - Entrepreneurs
  - Students
  - Analysts
  - Statisticians
  - Data Scientists
  - **Some of you are more than one!**

## Rewind - Some Takeaways from 2019

What did I learn last year from the R Community? (R-User Groups & satRday)

- Instead of using the x-axis, **time** may be represented as a color.
- `parApply()` helps make loops run faster.
- The `testthat` package can help us make unit testing more common.
- `Tensorflow` works well with R **and** Python

## Rewind - Some Takeaways from 2019

What did I learn last year from the R Community? (R-User Groups & satRday)

- Some of us like piping with `dplyr` and others prefer `data.table`.
  - We can take advantage of the strengths of both with `dtplyr`.
- `tidytext` allows us to pipe common NLP tasks.
- **Some** of the secret sauce is in hyperparameter tuning.

# R and Python Together

-Explicitly anticipate interoperability challenges - Partner has a python function I like and I like to work from RStudio. - My research collaborator needs me to use data aggregated in python to visualize with ggplot2 - A statistician has a .sav file for me to model in R and deploy in Python - I prefer zero indexing panda frames, but when I pass them for descriptives I prefer some of the cool functions in R - My friend's mental health stadmod could be translated in a simulation in R and partially explained with report()

# Around the web

- a few points of views on using R and Python together

# Use tools from Python in your R workflow.

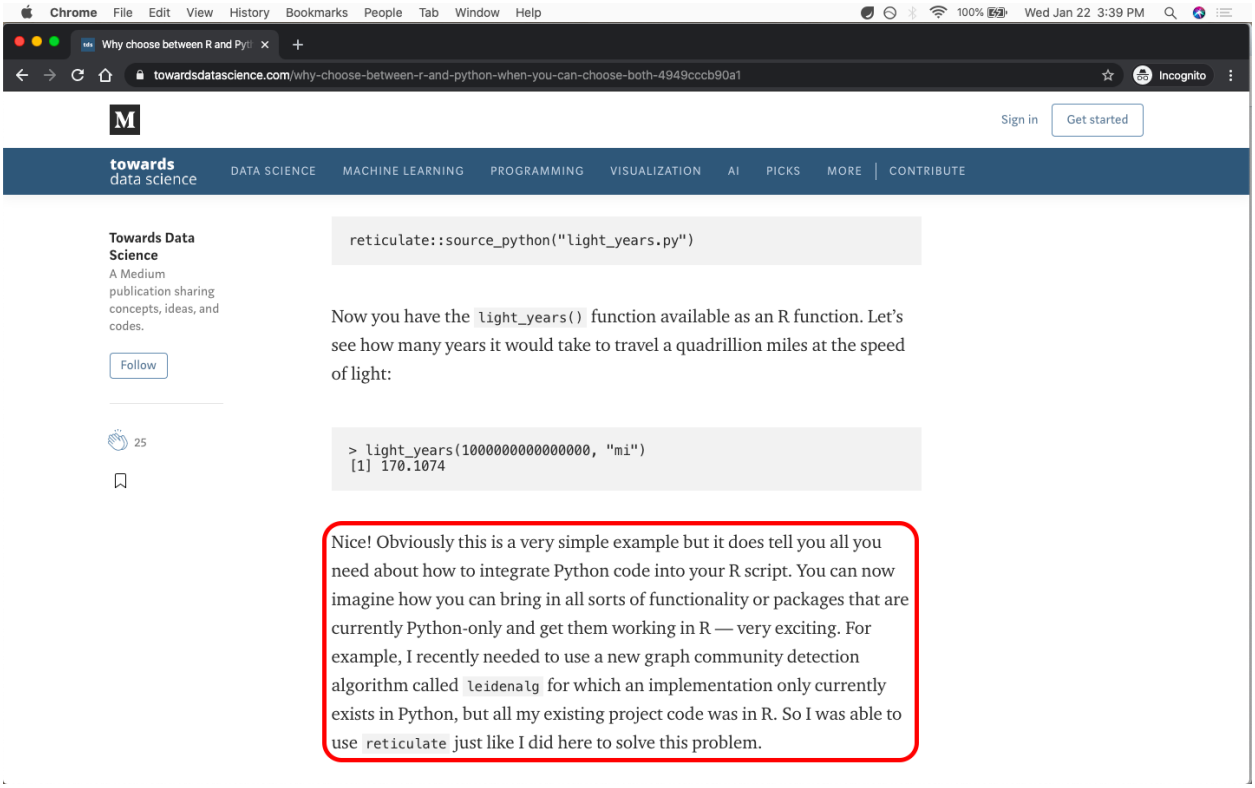



Figure 1: alt text

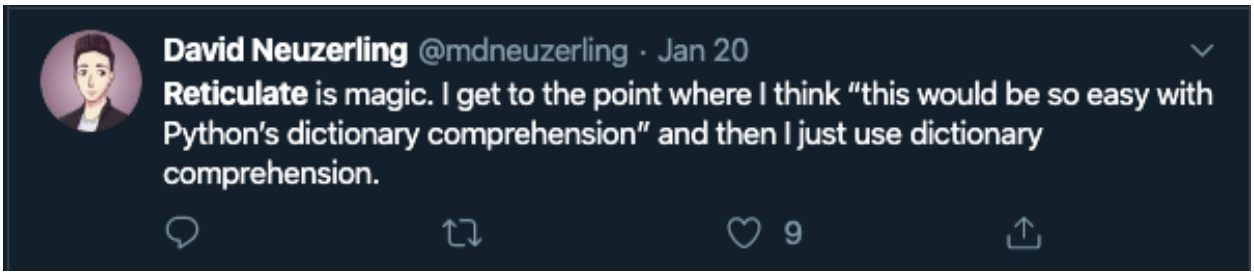



Figure 2: alt text

## Lists and Dictionaries

## Your workflows in your environments

## Statistics in R, statmod in Python, and distributions in Julia.

## Let's give it a try.

## That about sums it up.

## Let's start with something more familiar. How about installing packages in R?

How might our teams get tripped up in setup?

- Potential Scenarios
  - A new R user may be used to important statistical and machine learning routines being installed as default in their commercial programming environment.
  - Collaborator attempts to run R functions contained in packages that are not installed or loaded in their current environment.
  - Conflicts with package versions
- Ideally, we may make our own R packages with our teams to standardize workflows.

## Here's one way I like to ready R packages with my teams.

```
# We have the people who made the pacman package to thank for this one.
library(pacman)

# This is a wrapper that trys to load what you ask of it and attempts to install what's not available.

p_load(tidyverse, janitor, report, ggthemes)
```

## Setting up Python in RStudio with reticulate

```
# Load the reticulate package like any other package.
library(reticulate)

# Specify the location of the version of Python you'd like to use.
```

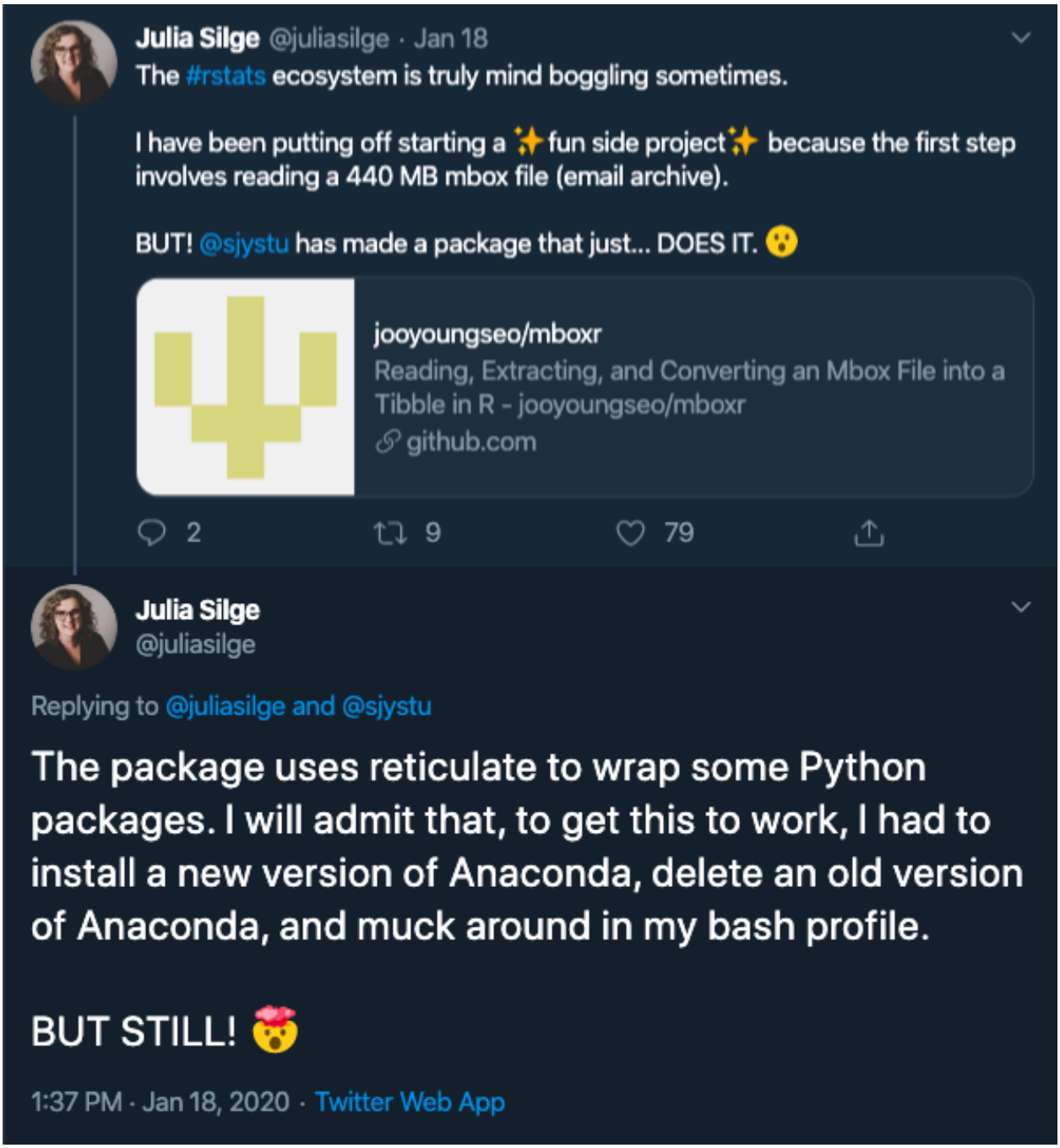



Figure 3: alt text

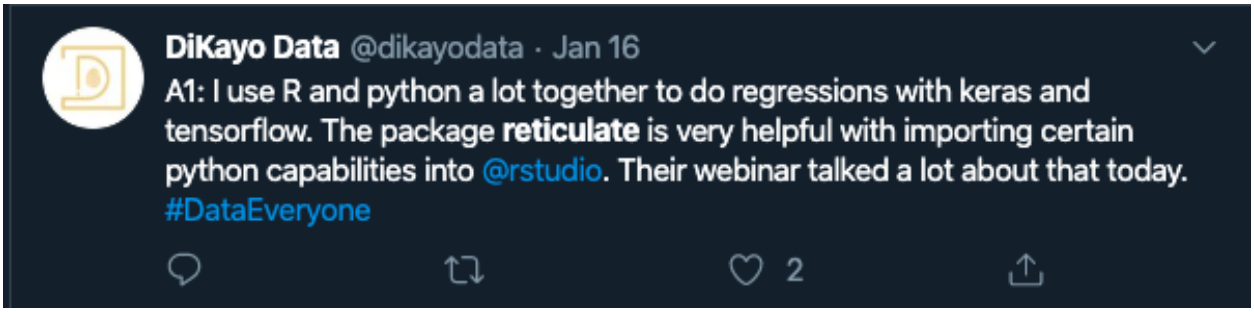



Figure 4: alt text

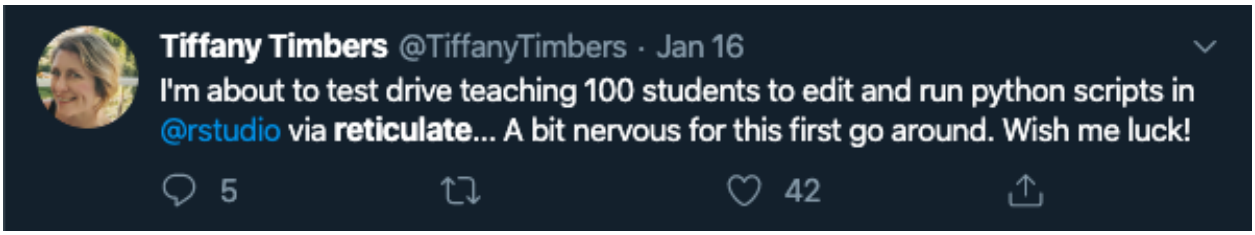



Figure 5: alt text


Figure 6: alt text

```
use_python("/Users/michaelespero/opt/anaconda3/bin/python")
```

## Is Python ready to use on your system?

```
# The initialize argument attempts to start Python bindings if they aren't already available.
py_available(initialize = T)
```

```
[1] TRUE
```

## Are Python modules available?

```
# Specify the name of the module you wish to check on in quotes.
py_module_available(module = "pandas")
```

```
[1] TRUE
```

## Another look at the configuration

```
# Detect some information about your setup
py_config()
```

```
python:         /Users/michaelespero/opt/anaconda3/bin/python
libpython:      /Users/michaelespero/opt/anaconda3/lib/libpython3.7m.dylib
pythonhome:     /Users/michaelespero/opt/anaconda3:/Users/michaelespero/opt/anaconda3
version:        3.7.4 (default, Aug 13 2019, 15:17:50)  [Clang 4.0.1 (tags/RELEASE_401/final)]
numpy:          /Users/michaelespero/opt/anaconda3/lib/python3.7/site-packages/numpy
numpy_version:  1.17.2

python versions found:
 /Users/michaelespero/opt/anaconda3/bin/python
 /Users/michaelespero/.virtualenvs/r-reticulate/bin/python
 /usr/bin/python
 /usr/bin/python3
 /usr/local/bin/python3
 /Users/michaelespero/anaconda3/bin/python
 /Users/michaelespero/.virtualenvs/py3-virtualenv/bin/python
 /Users/michaelespero/.virtualenvs/test-v37/bin/python
 /Users/michaelespero/anaconda3/envs/r-tensorflow/bin/python
 /Users/michaelespero/anaconda3/envs/spacy_condaenv/bin/python
 /Users/michaelespero/anaconda3/envs/two/bin/python
```

## How are the contents of Python objects accessed?

- The contents of R objects are accessed with $ following the object name (ex. df_weather).
  - ex) df_weather$date
- Similary, the contents of Python objects can be accessed with . following the object name.
  - ex) df_weather.date

## How are Python modules loaded?

- You have a few choices from the RStudio IDE
  - For python chunks
    - import model_name
      - ex) import pandas
    - Using an alias
      - ex) import pandas as pd
  - For R chunks
    - `import()`
      - pd <- `import("pandas")`

## Access Python functions from an R code chunk.

```
# The import function allows you to name an object with access to its functions using $.

pd <- import("pandas")

# pd$
```

Python packages may be *installed* using the console.
- pip install package name

## Let's get set up.

## You can source Python scripts in R.

- Scenario: Your collaborator has a function for you to use in your R workflow.

## Make a simple Python function.

```
# Let's make a simple function in Python that reads a csv file, selects only rows with organic produce,

import pandas as pd

def make_guac(file):
  guac2 = pd.read_csv(file)
  guac2 = guac2[guac2["type"] == "organic"]
  guac2 = guac2[["Date", "year", "type", "AveragePrice", "region"]]
  return guac2
```

## Source a Python function in R.

```
source_python("guac.py")

guac2 <- make_guac("avocado.csv")

head(guac2)
```

```
           Date year    type AveragePrice region
```

Figure 7: alt text

```
9126 2015-12-27 2015 organic         1.83 Albany
9127 2015-12-20 2015 organic         1.89 Albany
9128 2015-12-13 2015 organic         1.85 Albany
9129 2015-12-06 2015 organic         1.84 Albany
9130 2015-11-29 2015 organic         1.94 Albany
9131 2015-11-22 2015 organic         1.94 Albany
```

## Now let's confirm translation from R to Python.

## Data dictionary of tweets on wisdom from Kaggle.

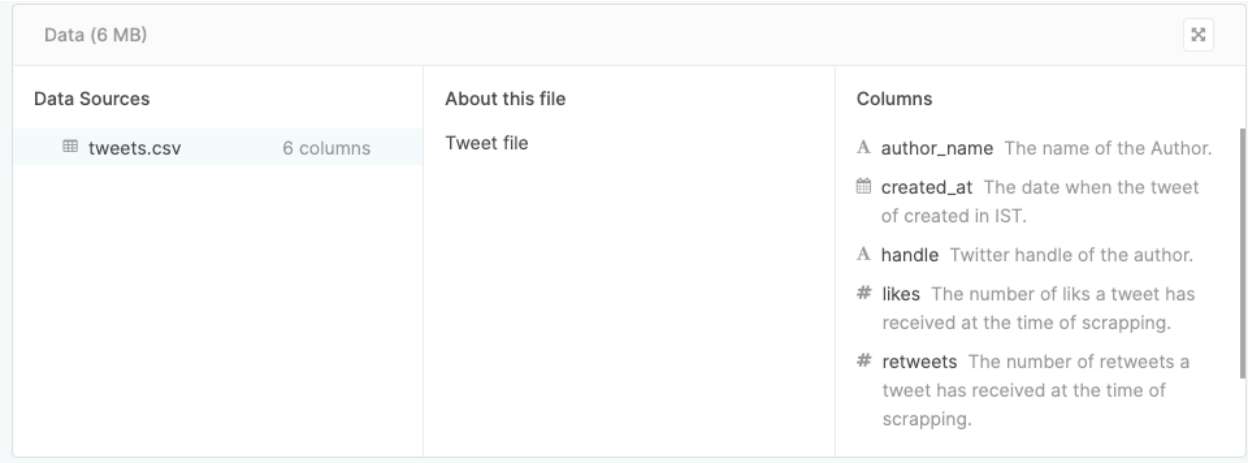



Figure 8: alt text

## Load the data in R.

```
library(readr)

# Let's use read_csv() to read in "tweets.csv", a data file containing a selection of tweets on the sub

df_r <- read_csv("tweets.csv")
```

## Let's take a peek.

```
library(tidyverse)
# glimpse() gives us a nice peek of our dataframe complete with its dimensions (number of rows and colu
glimpse(df_r)
```

```
Observations: 31,115
Variables: 6
$ author_name   <chr> "Naval", "Naval", "Naval", "Naval", "Naval", "Naval", "...
$ created_at    <dttm> 2019-08-07 22:36:56, 2019-08-07 05:00:38, 2019-08-07 0...
$ handle        <chr> "naval", "naval", "naval", "naval", "naval", "naval", "...
$ likes         <dbl> 7566, 21886, 6462, 466, 3971, 6141, 15681, 1633, 510, 2...
$ retweets      <dbl> 1498, 5984, 1266, 61, 906, 1114, 4805, 153, 126, 690, 7...
$ tweet_content <chr> "Unresolved thoughts, prematurely pushed out of the min...
```

## What's in the first row of the R dataframe.

```
# The "tweet_content" column contains character vectors with the text of each tweet. R begins row index
df_r$tweet_content[1]
```

```
[1] "Unresolved thoughts, prematurely pushed out of the mind, pile up in an internal landfill - which e
```

## Let's use the pipe to see who wrote the tweet and what's in it again.

```
# Let's get the author name along with the tweet to show who posted this opinion.
df_r[1,] %>%
  select(c("author_name", "tweet_content"))
```

```
# A tibble: 1 x 2
  author_name tweet_content
  <chr>       <chr>
1 Naval       Unresolved thoughts, prematurely pushed out of the mind, pile up ...
```

## `reticulate` allows us to turn an R dataframe into a Pandas dataframe for Python.

```
# Now that we have an idea of the first row of the tweet data in R, let's convert it to Pandas datafram
df_py <- r_to_py(df_r)
```

## Loading modules (packages) in Python.

```
# While the previous code chunk was an R chunk, we can specify that we'll write in python in this one b

# In python, instead of using library() to load packages, we can use "import" to load modules.

import numpy as np

import pandas as pd

import  nltk
```

## Use the dot to access the dataframe we made in R for Python.

```
# In python, we can get the dimensions of a dataframe with ".shape" added after the data object. Notice

r.df_py.shape

# We can get the column names with ".columns" following our r.df_py Pandas dataframe from the above R c
```

```
(31115, 6)
```

```
r.df_py.columns
```

```
Index(['author_name', 'created_at', 'handle', 'likes', 'retweets',
       'tweet_content'],
```

```
      dtype='object')
```

## In Python we start with zero.

```
# In the R version of the tweet data we checked the content of the first row's tweet content. Recall, i

r.df_py.author_name[0]
```

```
'Naval'
```

```
r.df_py.tweet_content[0]
```

```
'Unresolved thoughts, prematurely pushed out of the mind, pile up in an internal landfill - which eventu
```

## Back in an R chunk we can print a bit of the Python version of the tweet data by calling its name.

```
# We're back in an R chunk and let's check on our tweets about wisdom dataframes.
# First the pandas version.

df_py
```

```
            author_name  ...                                      tweet_content
0                 Naval  ...  Unresolved thoughts, prematurely pushed out of...
1                 Naval  ...  The modern mind is overstimulated and the mode...
2                 Naval  ...  The Lindy Effect for startups:\n\nThe longer y...
3                 Naval  ...                @orangebook_ This was a good tweet.
4                 Naval  ...  Social media lowers the cost of raising &amp; ...
...                 ...  ...                                                ...
31110  Uncanny Insights  ...  Our general behavior is the reflection of our ...
31111  Uncanny Insights  ...  In every matter being an unorthodox is a sure ...
31112  Uncanny Insights  ...  You can change your way of thinking, by changi...
31113  Uncanny Insights  ...  People fear that they will be dispossessed if ...
31114  Uncanny Insights  ...  To diagnose your thoughts solitude is the firs...

[31115 rows x 6 columns]
```

## How many authors are in this collection of tweets?.

```
# Let's see how many unique authors (of tweets) are in the dataframe.

unique(df_r$author_name) %>% length
```

```
[1] 2782
```

## Let's check the top 5 authors with regards to likes.

```
# The second argument is where you indicate the top n in terms of the third argument.

df_r_top5 <- top_n(df_r, 5, likes)
```

## Practice makes perfect.


Figure 9: alt text

## Let's check the top 5 authors with regards to likes.

```
library(ggthemes)

# Aside from the standard aesthetic arguments we set our labels, rotate our x-axis labels, and let the

p1 <- df_r_top5 %>%
  ggplot(aes(x = author_name, y = likes)) +
  geom_col() +
  labs(x = "Author", y = "Likes") +
  theme_tufte() +
  theme(axis.text.x = element_text(face="bold", color="black", size=8, angle=45)) +
  scale_color_viridis_c()
```

## Let's check the top 5 authors with regards to likes.

```
p1
```

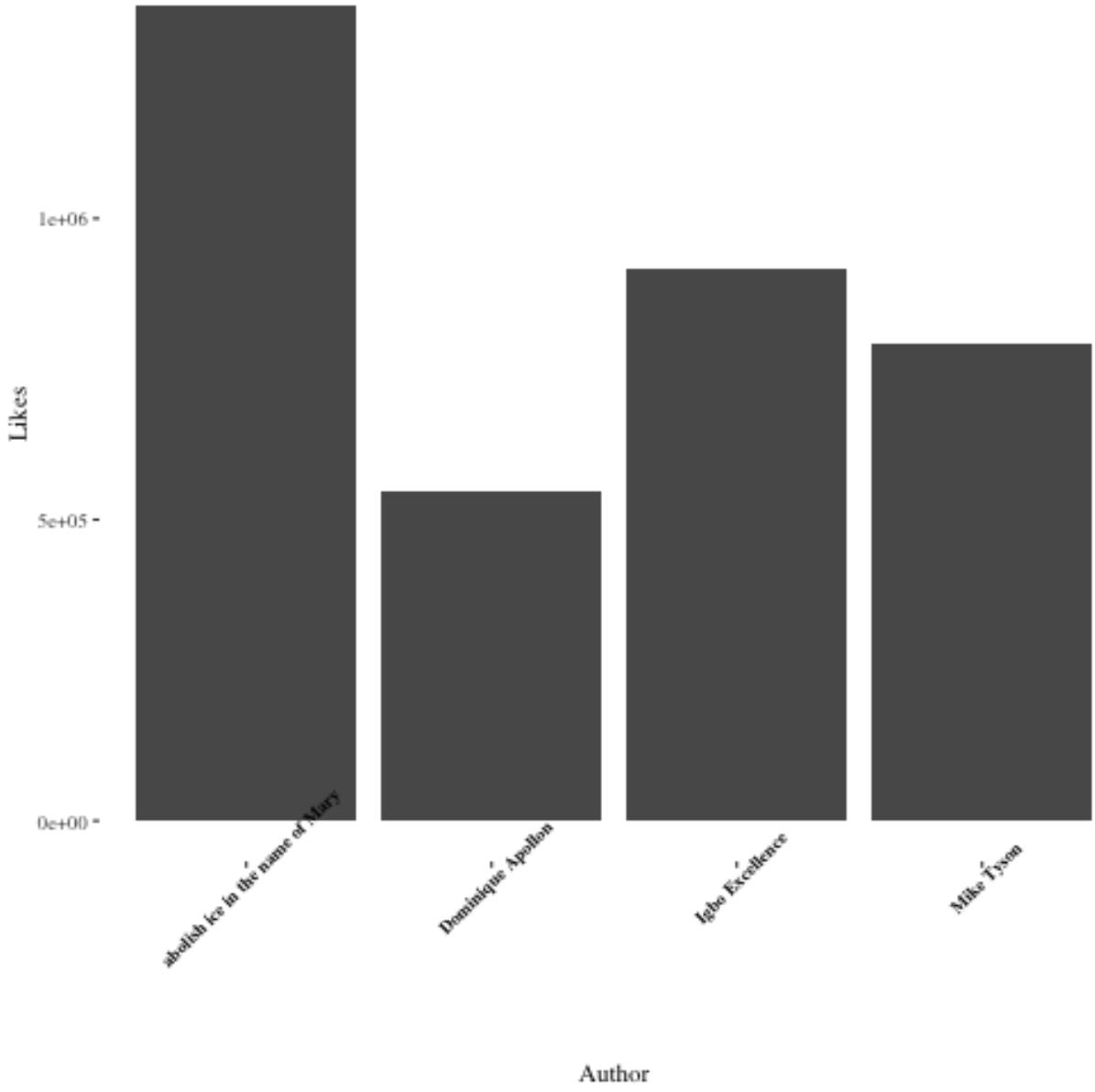



Figure 10: plot of chunk unnamed-chunk-21

## How about sorting data in the tidyverse?

```
# desc refers to descending order by the variable you pass to it.

df_r %>%
  arrange(desc(likes))
```

```
# A tibble: 31,115 x 6
   author_name   created_at          handle   likes retweets tweet_content
   <chr>         <dttm>              <chr>    <dbl>    <dbl> <chr>
 1 abolish ice ... 2019-08-10 20:16:00 hashta... 1.35e6   424270 "oh my god, i need...
 2 Mike Tyson    2019-01-16 22:19:41 MikeTy... 7.92e5   172538 "Stop sending me t...
 3 Dominique Ap... 2019-04-19 16:27:58 Apollo... 5.48e5   104471 "It's taken me 45 ...
 4 Igbo Excelle... 2019-08-31 10:40:27 1ncogn... 4.57e5   146526 "I noticed African...
 5 Igbo Excelle... 2019-08-31 10:40:27 1ncogn... 4.57e5   146526 "I noticed African...
 6 Mark Phillips 2019-08-20 21:39:16 Suprem... 3.22e5   128305 "How Chick Fila Wo...
 7 Bill Dixon    2018-12-09 18:54:56 BillDi... 3.17e5    43219 "About 5 years ago...
 8 Fernando De ... 2019-09-11 18:50:11 DeJesu... 3.11e5    65422 "Bro we're in fuck...
 9 Mike Drucker  2019-06-07 18:46:26 MikeDr... 3.06e5    69278 "Twitter is fun be...
10 Koopington F... 2018-12-29 04:50:40 koopa_... 2.70e5    90209 "This the most Flo...
# ... with 31,105 more rows
```

## Let's move from tweets to avocados.

```
# You might like to use = instead of <-- in R for assignment because it works in python too.

guac <- read_csv("avocado.csv")

glimpse(guac)
```

```
Observations: 18,249
Variables: 14
$ X1             <dbl> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, ...
$ Date           <date> 2015-12-27, 2015-12-20, 2015-12-13, 2015-12-06, 2015-...
$ AveragePrice   <dbl> 1.33, 1.35, 0.93, 1.08, 1.28, 1.26, 0.99, 0.98, 1.02, ...
$ `Total Volume` <dbl> 64236.62, 54876.98, 118220.22, 78992.15, 51039.60, 559...
$ `4046`         <dbl> 1036.74, 674.28, 794.70, 1132.00, 941.48, 1184.27, 136...
$ `4225`         <dbl> 54454.85, 44638.81, 109149.67, 71976.41, 43838.39, 480...
$ `4770`         <dbl> 48.16, 58.33, 130.50, 72.58, 75.78, 43.61, 93.26, 80.0...
$ `Total Bags`   <dbl> 8696.87, 9505.56, 8145.35, 5811.16, 6183.95, 6683.91, ...
$ `Small Bags`   <dbl> 8603.62, 9408.07, 8042.21, 5677.40, 5986.26, 6556.47, ...
$ `Large Bags`   <dbl> 93.25, 97.49, 103.14, 133.76, 197.69, 127.44, 122.05, ...
$ `XLarge Bags`  <dbl> 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, ...
$ type           <chr> "conventional", "conventional", "conventional", "conve...
$ year           <dbl> 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015, ...
$ region         <chr> "Albany", "Albany", "Albany", "Albany", "Albany", "Alb...
```

## Let's move from tweets to avocados.

```
library(janitor)

# clean_names() gives us standard lowercase feature names
```

```
guac = guac %>%
  clean_names() %>%
  select(c("date", "year", "type", "average_price", "total_bags"))

names(guac)
```

```
[1] "date"          "year"          "type"          "average_price"
[5] "total_bags"
```

## Plot the price by week.

```
# The type of avocado was encoded as a character vector. Let's make it a factor.
guac$type <- as_factor(guac$type)
library(lubridate)
# Notice that we're manipulating the our x-axis, date, to be shown by week.
p2 <- ggplot(guac, aes(x = week(date), y = average_price), color = type) +
  geom_jitter() +
  geom_smooth(method = "lm", se = FALSE) +
  theme_tufte() +
  labs(
    title = "Is the Price of Guac Rising?",
    x = "Week",
    y = "Average Price",
    caption = "Data Source: https://www.kaggle.com/neuromusic/avocado-prices")
```

## Plot the price by week.

```
p2
```

## Plotting avocados in Python.

```
import seaborn as sns

# Plot r.guac using Seaborn

c_py_plot = sns.regplot(x = "year", y = "average_price", data = r.guac)
c_py_plot.set(xlabel = "Year", ylabel = "Average Price", title = "Is Guac Getting Richer?")
```

```
[Text(0, 0.5, 'Average Price'), Text(0.5, 0, 'Year'), Text(0.5, 1.0, 'Is Guac Getting Richer?')]
```

```
c_py_plot
```

## Plotting avocados in Python.

```
c_py_plot
```

```
<matplotlib.axes._subplots.AxesSubplot object at 0x11e588290>
```

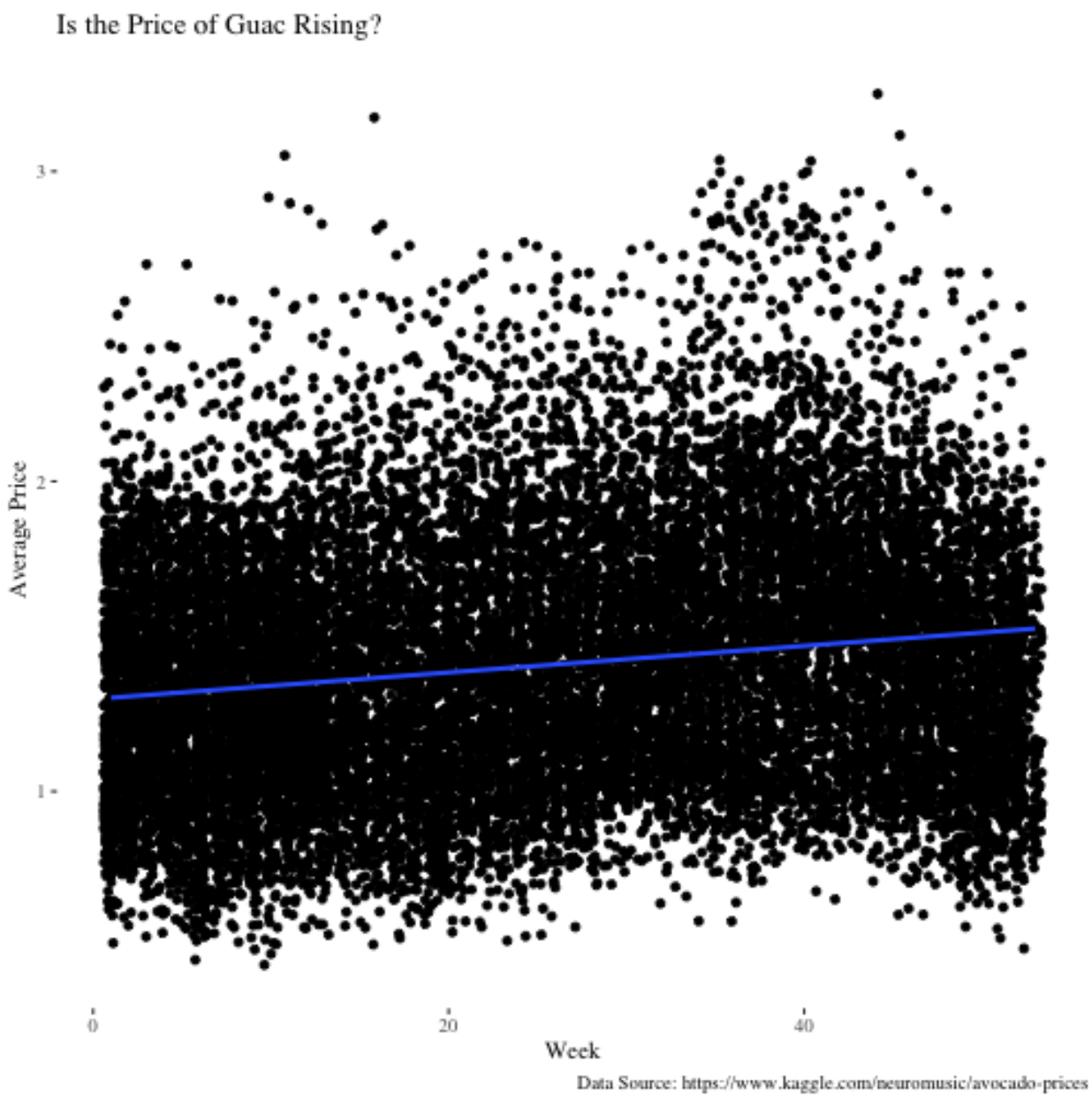



Figure 11: plot of chunk unnamed-chunk-26

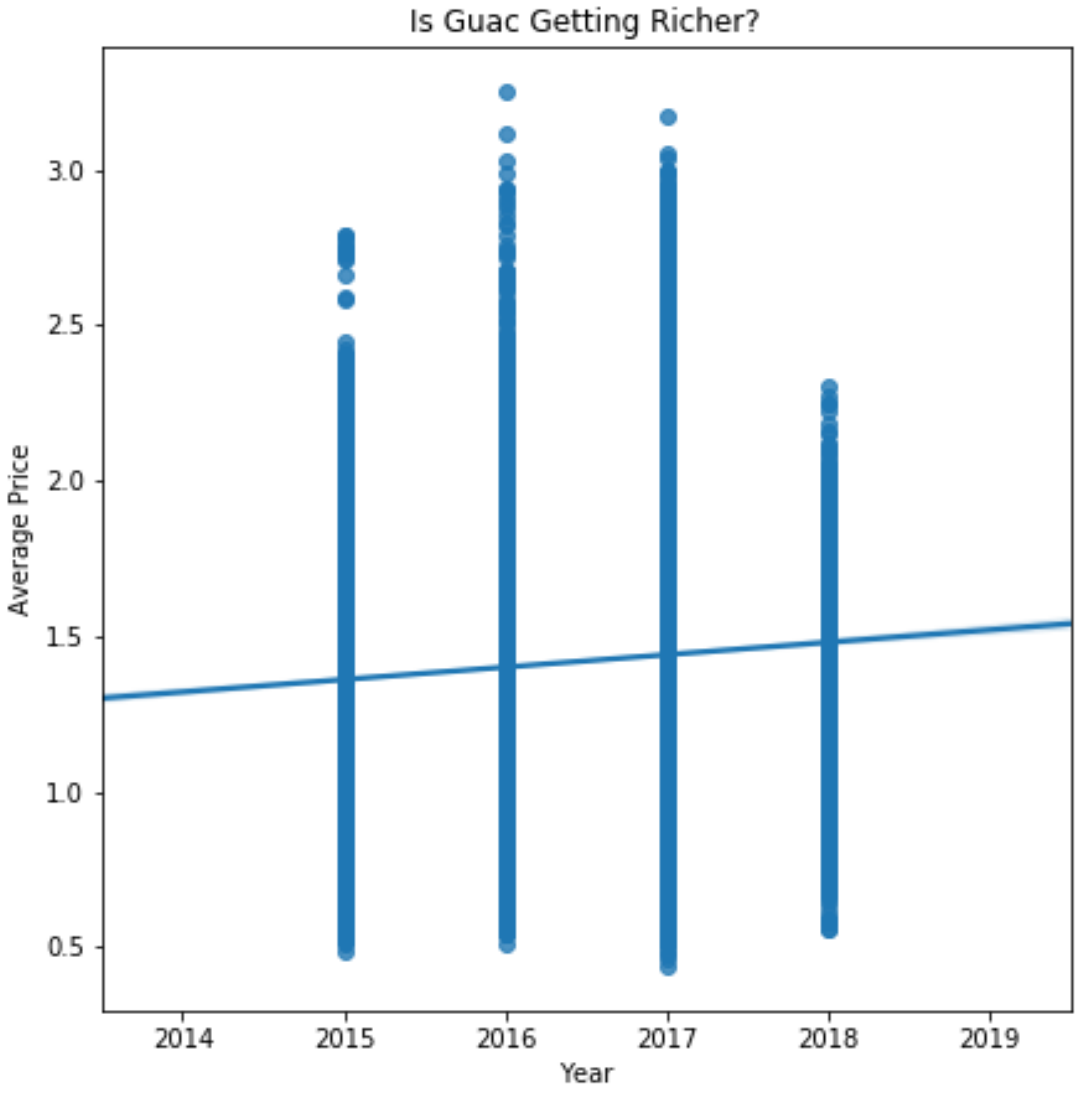



Figure 12: plot of chunk unnamed-chunk-27

## Let's give it a go.

Figure 13: alt text

## What about the R and Python love story?

```
# Let's bring pre-processed corpi into the environment for easy access. See https://www.nltk.org/book/c

from nltk.book import *

# If we call the object "text8" we can see it's title. It appears to be a collection personal ads.
```

```
*** Introductory Examples for the NLTK Book ***
Loading text1, ..., text9 and sent1, ..., sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.
text1: Moby Dick by Herman Melville 1851
text2: Sense and Sensibility by Jane Austen 1811
text3: The Book of Genesis
text4: Inaugural Address Corpus
text5: Chat Corpus
text6: Monty Python and the Holy Grail
text7: Wall Street Journal
text8: Personals Corpus
text9: The Man Who Was Thursday by G . K . Chesterton 1908
```

```
text8
```

```
<Text: Personals Corpus>
```

## The context of love in this corpus

```
# Since this data must be about people seeking a mate, let's check concordance for the word "love". It'

text8.concordance("love")
```

```
Displaying 10 of 10 matches:
od sense of humour , am romantic and love drives , fishing , camping and music
ives , fishing , camping and music . Love my 2 kids . Am looking for a lady wi
hip ASIAN LADY sought . Kids OK as I love family . Nice & honest guy . ASIAN L
ie mid 40s b / man f / ship r / ship LOVE to meet widowed lady over 50 , no ch
late , intelligent & very flexible . Love a good laugh , love life & enjoy con
 very flexible . Love a good laugh , love life & enjoy contrasts and the finer
er callers welcome to reply . DO YOU LOVE TO DANCE ? Fun loving and employed ,
0s , working full time . Looking for love & laughter . Are you at least 59 , n
t not fanatical ? A bonus would be a love of dancing . GARDEN LOVER Hi ! I am
dinners , fine wine , romance & true love . YORKE PENINSULA LADY Late 70s , 53
```

## Word similarity

```
# We can search the personals corpus for words thought to be similar to the word "meet", for instance.

text8.similar("meet")
```

```
relationship australian share beginning meeting find very
```

## Plot the dispersion of words from the personal ads corpus.

```
text8.dispersion_plot(["love", "looking", "fun", "true"])
```

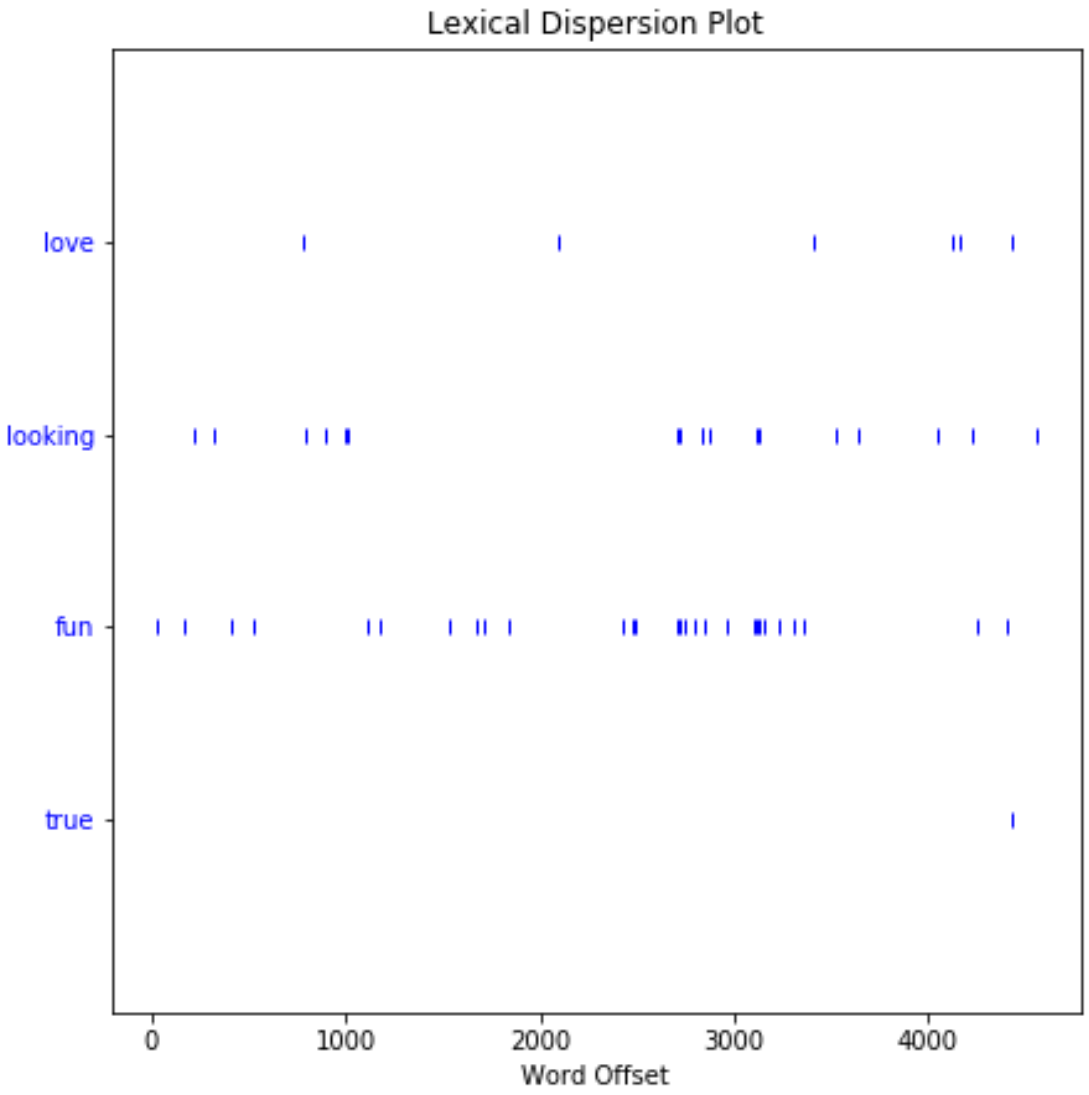



Figure 14: plot of chunk unnamed-chunk-32

Figure 15: alt text


Figure 16: alt text

Figure 17: alt text

Figure 18: alt text

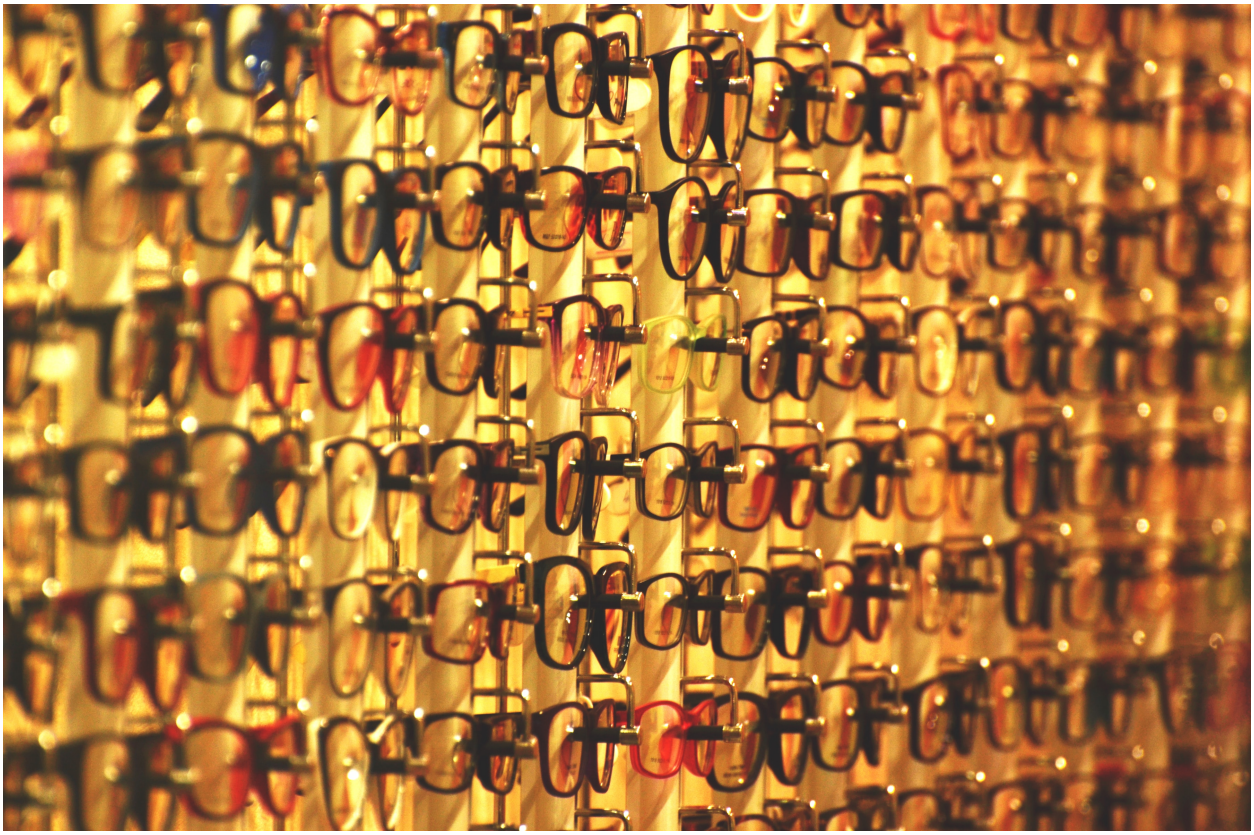
**Once we get set up the bridge is already there.**

**It helps to know what to expect on the other side.**

**It's about making connections.**

**Once we get set up you have R and Python at your fingertips.**

**If you haven't learned anything...**



Hopefully I've convinced you there are many resources available.

## More resources

- Documentation:
  - Reticulate
    - https://rstudio.github.io/reticulate/
  - R & Python: A Love Story
    - https://rstudio.com/solutions/r-and-python/

## More resources

- Videos:
  - PyCon.DE 2018: Reticulate: R Interface To Python - Jens Bruno Wittek
    - https://youtu.be/EJxQSa9lwfM

  - Using Python in the RStudio IDE - Matt Dancho
    * https://youtu.be/YI_hEtbpz-s

## More resources

- Blogs:
  - R or Python? Why not both? Using Anaconda Python within R with {reticulate}
    * Bruno Rodrigues
    * https://www.brodrigues.co/blog/2018-12-30-reticulate/
  - The best of both worlds: R meets Python via reticulate
    * Martin Henze
    * https://heads0rtai1s.github.io/2019/10/03/reticulate-intro/

## However you use R and Python, I hope it brings you access to more tools and greater collaboration.