Beginner Tutorial

Livia Popa & Nick Gawron

Tutorial Learning Goals

We will be tackling these objectives:

- *Define* open data and reproducible science
- *Describe* how to navigate important aspects of the R/RStudio user-interface
- *Recall* how to extract public data from a web portal (Cardinal) and import it into a data software (R/RStudio)
- *Demonstrate* understanding of dataset and statistical software through exploratory data analysis plots and numerical summaries

Meet Mr. Wuf!

Mr. Wuf works for Mount Mitchell State Park in Burnsville, NC and was recently asked by his boss to write a report summarizing rainfall and temperature data for 2021. This report will be used to help optimize 2022 event planning (e.g., fall color viewing) for park visitors and maintenance scheduling for park staff. Mr. Wuf's wife, Mrs. Wuf, recently told him about the State Climate Office of North Carolina's new Cardinal and Station Scout data portals. He agrees with her that it would be a great opportunity to check out these new, free tools. After some preliminary sleuthing around Station Scout, he discovered there was a National Weather Service Cooperative Observer Program (COOP; https://www.weather.gov/rah/coop) station on park property (station # 315923). How did he miss this? Once he downloads these data from Cardinal, Mr. Wuf plans to put the skills he learned in an online R programming course to the test for this real-world, work-related project.

Open Data Science

What is Open Data and Reproducible Science?

Open data documents and shares research data openly for re-use.

Open data research aims to transform research by pushing change in the way that research is carried out and disseminated by digital tools. Open data should be:

- Publicly available: Open data is freely available on the internet.
- Reusable: Proper licensing is essential for research outputs so that users know any limitations on re-use
- Transparent: With appropriate metadata to explain how research output was produced and what it contains

For more information go to this website: https://the-turing-way.netlify.app/reproducible-research/reproducible-research.html

Importing Data From Cardinals

- Cardinal is a high-powered, user-oriented, one-stop-shop for North Carolina weather and climate data housed at the North Carolina State Climate Office.
- Cardinal makes weather and climate data more accessible to users, with features and prompts that take the guesswork out of station and parameter identification and selection.

Using data inspection functions

- `str` is a powerful function that allows you to determine what kind of variable we are working with

Data Types

- Date
- Str
- Char
- Double
- Num

```r
library(readr)
cardinal <- read_csv("cardinal_data.csv")
#cardinal%>%filter(colnames(cardinal)!="MV")
```

Cleaning Data

- usually a lot more messy
- R can handle a lot of small details

```r
# drop rows of missing values
cardinal<-drop_na(cardinal)

#determine data types of all cols
str(cardinal)
```

```
## tibble [729 x 11] (S3: tbl_df/tbl/data.frame)
##  $ Date                                 : chr [1:729] "1/1/20" "1/2/20" "1/3/20" "1/4/20" ...
##  $ Average Air Temperature (F)          : num [1:729] 43.1 44.9 52.8 57.2 42.1 44.1 41.4 42.5 40.4 5
##  $ Maximum Air Temperature (F)          : num [1:729] 53.6 55.4 64.9 65.1 50.5 58.5 52 57.6 50.5 65
##  $ Minimum Air Temperature (F)          : num [1:729] 35.1 35.2 45.7 42.6 34.9 32 31.3 29.7 31.3 38
##  $ Average Experimental Leaf Wetness (mV): num [1:729] 266 274 362 373 265 ...
##  $ Total Precipitation (in)             : num [1:729] 0 0.05 0.95 0.52 0 0 0.07 0 0 0 ...
##  $ Average Relative Humidity (%)        : num [1:729] 63.8 72 92.1 83.5 57 ...
##  $ Average Soil Moisture (m3/m3)        : num [1:729] 0.28 0.28 0.29 0.35 0.33 0.31 0.3 0.3 0.3 0.2
##  $ Average Soil Temperature (F)         : num [1:729] 48.6 47.6 51 54.6 48.3 46.1 44.6 43.3 43.3 46
##  $ Average Solar Radiation (W/m2)       : num [1:729] 134.8 66 31.1 44.9 135.4 ...
##  $ Average Station Pressure (mb)        : num [1:729] 999 1003 998 993 1005 ...
```

```
#create a date
cardinal$Date<-as.Date(cardinal$Date, tryFormats= c("%m/%d/%y"))
view(cardinal)

#changes col names
colnames(cardinal)=c("date","AvgT","MaxT","MinT","AvgLw","Tprep","AvgHum","AvgSm","AvgSt","AvgSr","AvgS
```

Making new Data

When does it Rain ?

```
#me making new data

cardinal$IfRain<- (cardinal$Tprep>0)
cardinal$IfRain<-as.factor(as.integer(cardinal$IfRain))
cardinal
```

```
## # A tibble: 729 x 12
##    date        AvgT  MaxT  MinT AvgLw Tprep AvgHum AvgSm AvgSt AvgSr AvgStp
##    <date>     <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl>  <dbl>
##  1 2020-01-01  43.1  53.6  35.1  266.  0       63.8  0.28  48.6 135.    999.
##  2 2020-01-02  44.9  55.4  35.2  274.  0.05    72.0  0.28  47.6  66.0  1003.
##  3 2020-01-03  52.8  64.9  45.7  362.  0.95    92.1  0.29  51    31.1   998.
##  4 2020-01-04  57.2  65.1  42.6  373   0.52    83.5  0.35  54.6  44.9   993.
##  5 2020-01-05  42.1  50.5  34.9  265.  0       57.0  0.33  48.3 135.   1005.
##  6 2020-01-06  44.1  58.5  32    265.  0       57.6  0.31  46.1 138.   1005.
##  7 2020-01-07  41.4  52    31.3  274.  0.07    75.2  0.3   44.6  40.9  1002.
##  8 2020-01-08  42.5  57.6  29.7  314.  0       58.9  0.3   43.3 136.   1010.
##  9 2020-01-09  40.4  50.5  31.3  265.  0       60.2  0.3   43.3 122.   1022.
## 10 2020-01-10  52    65.8  38.1  266.  0       73.5  0.29  46.1  74.6  1019.
## # ... with 719 more rows, and 1 more variable: IfRain <fct>
```

Month Factor variable

- Factor variable is a category or bin we can place a value in.

```
# month variable
cardinal$month<-month(cardinal$date)
cardinal$month<-as.factor(cardinal$month)

# str factor
str(cardinal)
```

```
## tibble [729 x 13] (S3: tbl_df/tbl/data.frame)
##  $ date  : Date[1:729], format: "2020-01-01" "2020-01-02" ...
##  $ AvgT  : num [1:729] 43.1 44.9 52.8 57.2 42.1 44.1 41.4 42.5 40.4 52 ...
##  $ MaxT  : num [1:729] 53.6 55.4 64.9 65.1 50.5 58.5 52 57.6 50.5 65.8 ...
##  $ MinT  : num [1:729] 35.1 35.2 45.7 42.6 34.9 32 31.3 29.7 31.3 38.1 ...
##  $ AvgLw : num [1:729] 266 274 362 373 265 ...
##  $ Tprep : num [1:729] 0 0.05 0.95 0.52 0 0 0.07 0 0 0 ...
##  $ AvgHum: num [1:729] 63.8 72 92.1 83.5 57 ...
##  $ AvgSm : num [1:729] 0.28 0.28 0.29 0.35 0.33 0.31 0.3 0.3 0.3 0.29 ...
##  $ AvgSt : num [1:729] 48.6 47.6 51 54.6 48.3 46.1 44.6 43.3 43.3 46.1 ...
##  $ AvgSr : num [1:729] 134.8 66 31.1 44.9 135.4 ...
##  $ AvgStp: num [1:729] 999 1003 998 993 1005 ...
##  $ IfRain: Factor w/ 2 levels "0","1": 1 2 2 2 1 1 2 1 1 1 ...
##  $ month : Factor w/ 12 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
```

Numerical Variable, Rain Difference

- Dollar sign + "Name of Variable"

```
cardinal$TDiff <- cardinal$MaxT-cardinal$MinT
```

Numerical Summaries

- Help us determine basic trends in data from printouts.
- Summary gives as a 5 number summary of numeric variables
- Basic counts of factor variables

```
summary(cardinal)
```

```
##       date                 AvgT            MaxT            MinT      
##  Min.   :2020-01-01   Min.   :29.20   Min.   :34.20   Min.   :21.20  
##  1st Qu.:2020-07-03   1st Qu.:50.40   1st Qu.:60.80   1st Qu.:39.40  
##  Median :2021-01-01   Median :62.30   Median :72.90   Median :52.90  
##  Mean   :2020-12-31   Mean   :61.43   Mean   :71.61   Mean   :52.14  
##  3rd Qu.:2021-07-03   3rd Qu.:73.80   3rd Qu.:84.20   3rd Qu.:66.20  
##  Max.   :2022-01-01   Max.   :84.10   Max.   :95.00   Max.   :76.30  
##                                                                      
##      AvgLw           Tprep            AvgHum          AvgSm       
##  Min.   :260.4   Min.   :0.0000   Min.   :26.36   Min.   :0.1300  
##  1st Qu.:270.0   1st Qu.:0.0000   1st Qu.:62.10   1st Qu.:0.2700  
##  Median :287.7   Median :0.0000   Median :71.95   Median :0.3000  
##  Mean   :306.0   Mean   :0.1586   Mean   :69.93   Mean   :0.2975  
##  3rd Qu.:322.7   3rd Qu.:0.0700   3rd Qu.:79.80   3rd Qu.:0.3300  
##  Max.   :603.4   Max.   :4.3500   Max.   :95.03   Max.   :0.4500  
##                                                                   
##      AvgSt          AvgSr            AvgStp       IfRain      month    
##  Min.   :37.9   Min.   :  6.33   Min.   : 986.0   0:449   1      : 63  
##  1st Qu.:52.0   1st Qu.:107.62   1st Qu.: 999.9   1:280   3      : 62  
##  Median :64.2   Median :166.82   Median :1003.6           8      : 62  
##  Mean   :63.7   Mean   :175.05   Mean   :1003.9           10     : 62  
##  3rd Qu.:76.9   3rd Qu.:247.28   3rd Qu.:1007.5           12     : 62  
##  Max.   :86.0   Max.   :437.72   Max.   :1022.4           7      : 61  
##                                                           (Other):357  
##      TDiff      
##  Min.   : 2.90  
##  1st Qu.:15.00  
##  Median :19.50  
##  Mean   :19.47
##  3rd Qu.:23.60
##  Max.   :37.30
##
```

- Frequency Table to compare categorical / factor variables.

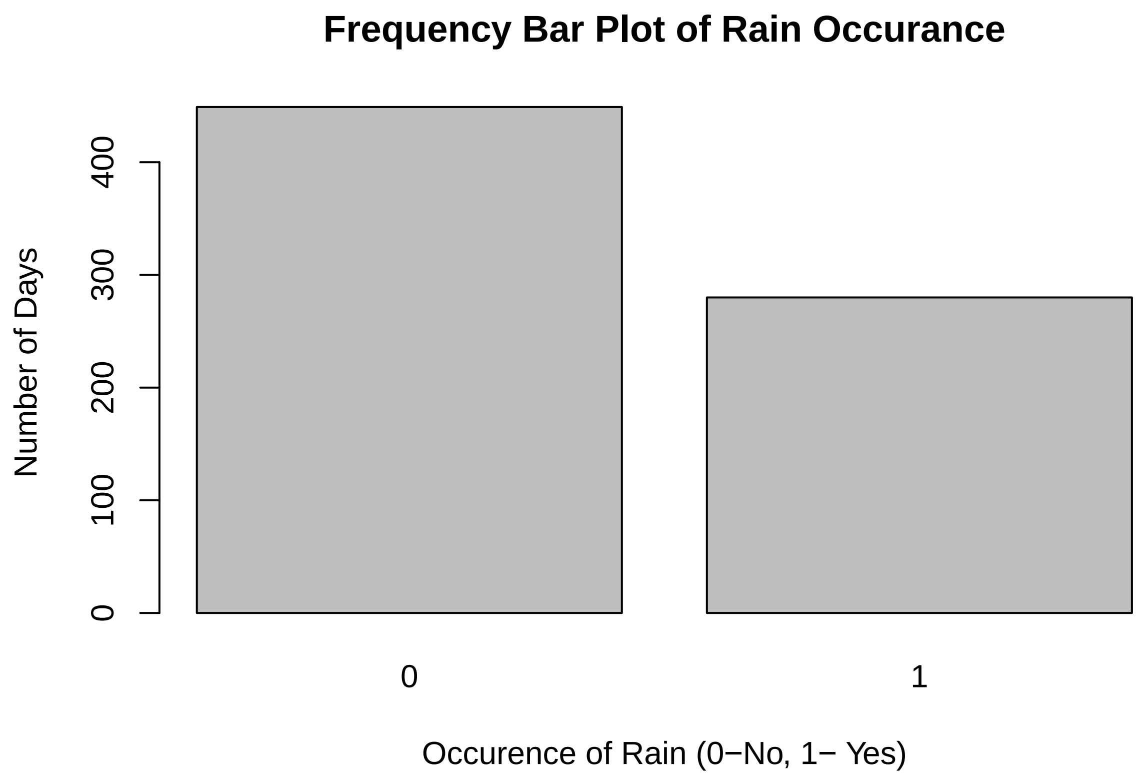
```
# Frequency Table
table(cardinal$month,cardinal$IfRain)
```

```
##
##       0  1
##   1  34 29
##   2  27 30
##   3  39 23
##   4  43 17
##   5  37 23
##   6  38 22
##   7  35 26
##   8  38 24
##   9  38 22
##   10 40 22
##   11 39 21
##   12 41 21
```

Plotting Basic

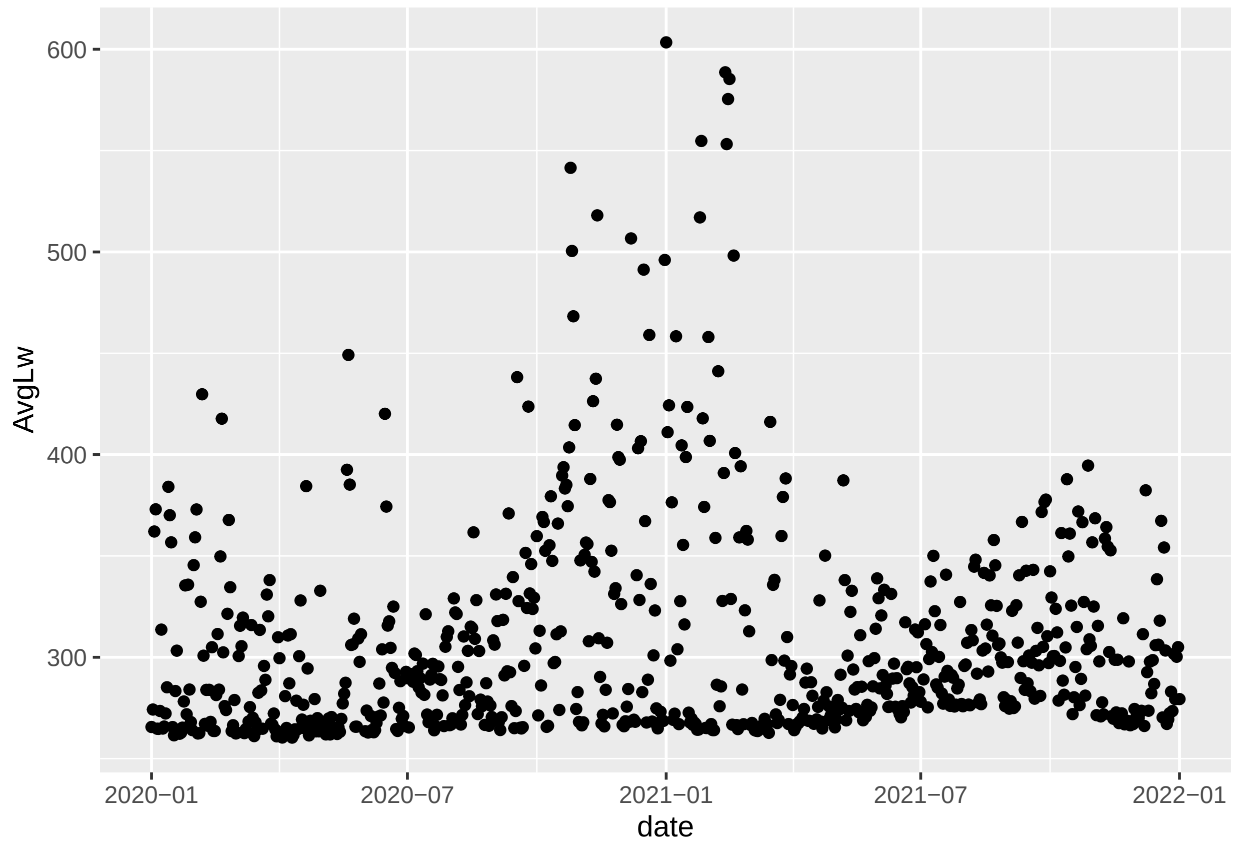
- Simple visual of frequency count

```
# base plots in R, categorical variables
#does a count
plot(cardinal$IfRain,main ='Frequency Bar Plot of Rain Occurance',xlab="Occurence of Rain (0-No, 1- Yes
```



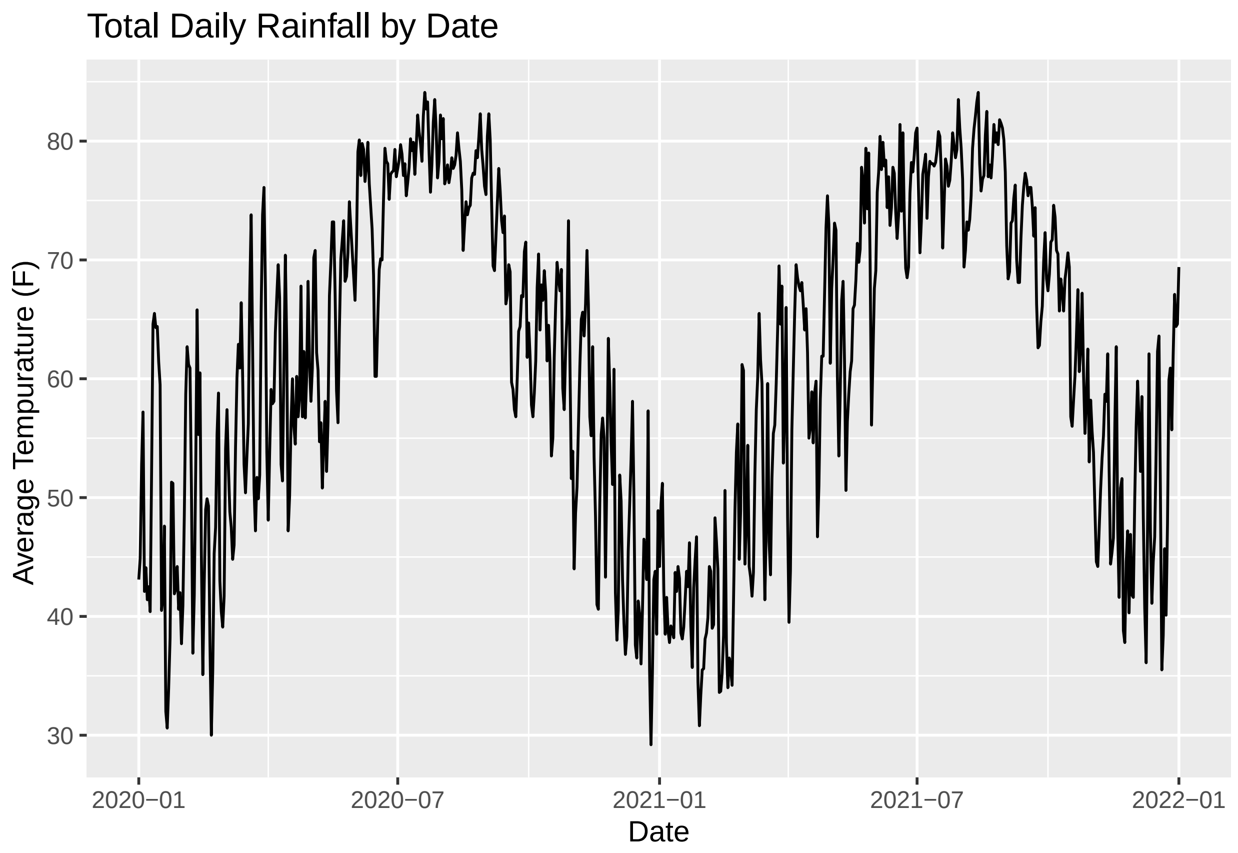
- We will be using a package called `ggplot2`.
- Here is a good link: https://www.rstudio.com/resources/cheatsheets/
- Two basic functions: ggplot() & geom_plottype
  - Note we have not even had a title or label specs yet

```
# first ggplot figure

ggplot(cardinal,aes(x= date ,y= AvgLw))+ geom_point()
```



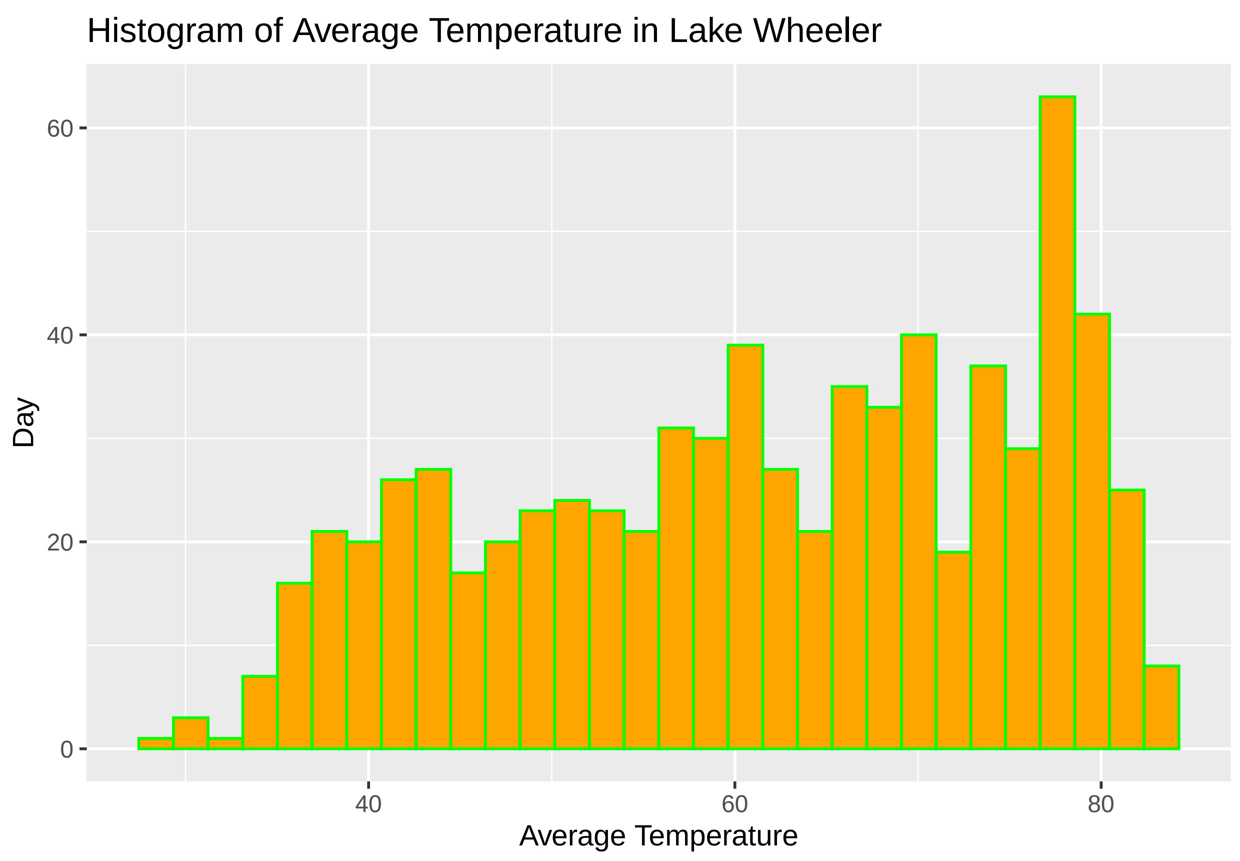
- Observe correlation and possible trend numerical variables
- Using cheatsheet, we can find a lot more plot types and options!
  - Note the use of `labs` statement

```
ggplot(cardinal,aes(x=date,y=AvgT))+geom_line()+labs(title="Total Daily Rainfall by Date",y="Average Te
```



- options are very versatile inside a `+geom_statement()`
- We can use the cheatsheet to find out information about this
- Note how we change atributes inside the `aes` statment

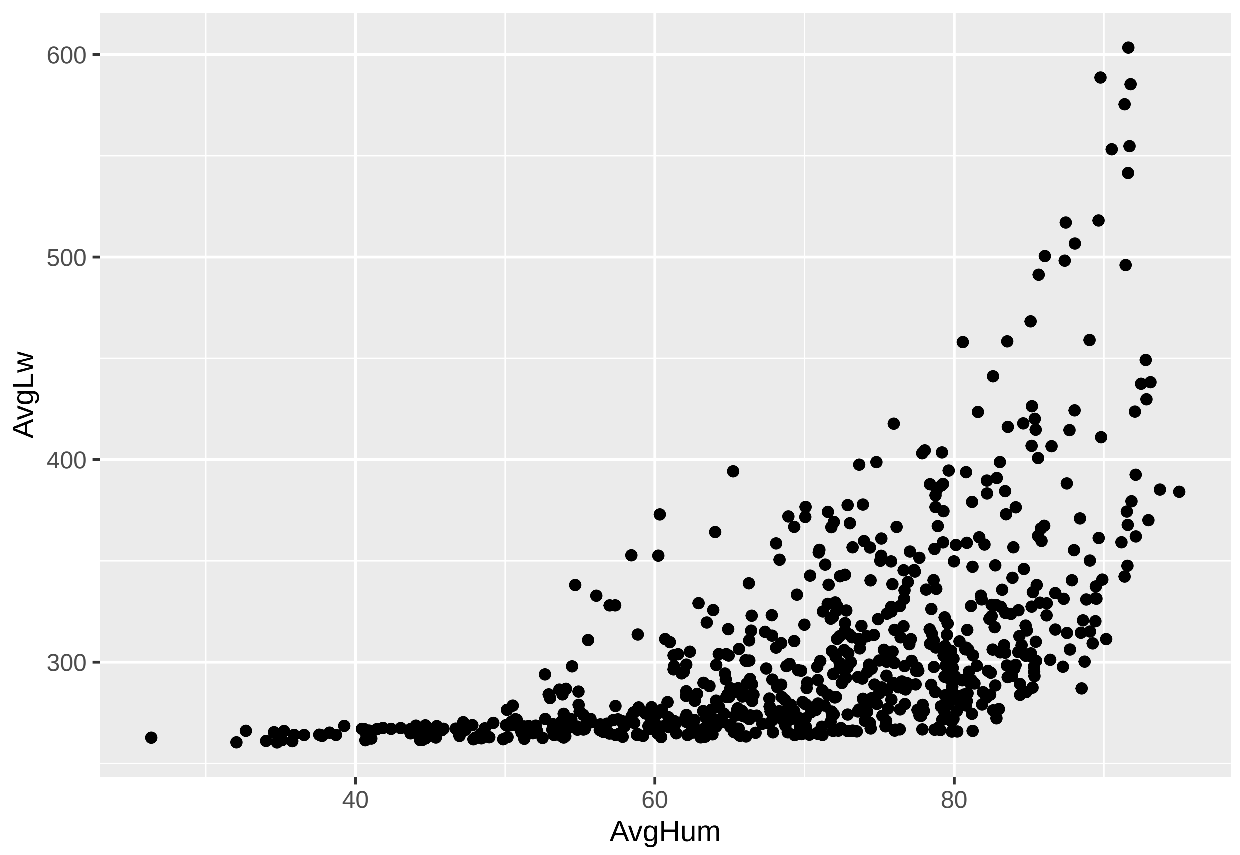
```
ggplot(cardinal,aes(x=AvgT)) + geom_histogram(color="green",fill="orange")+labs(x="Average Temperature"
```



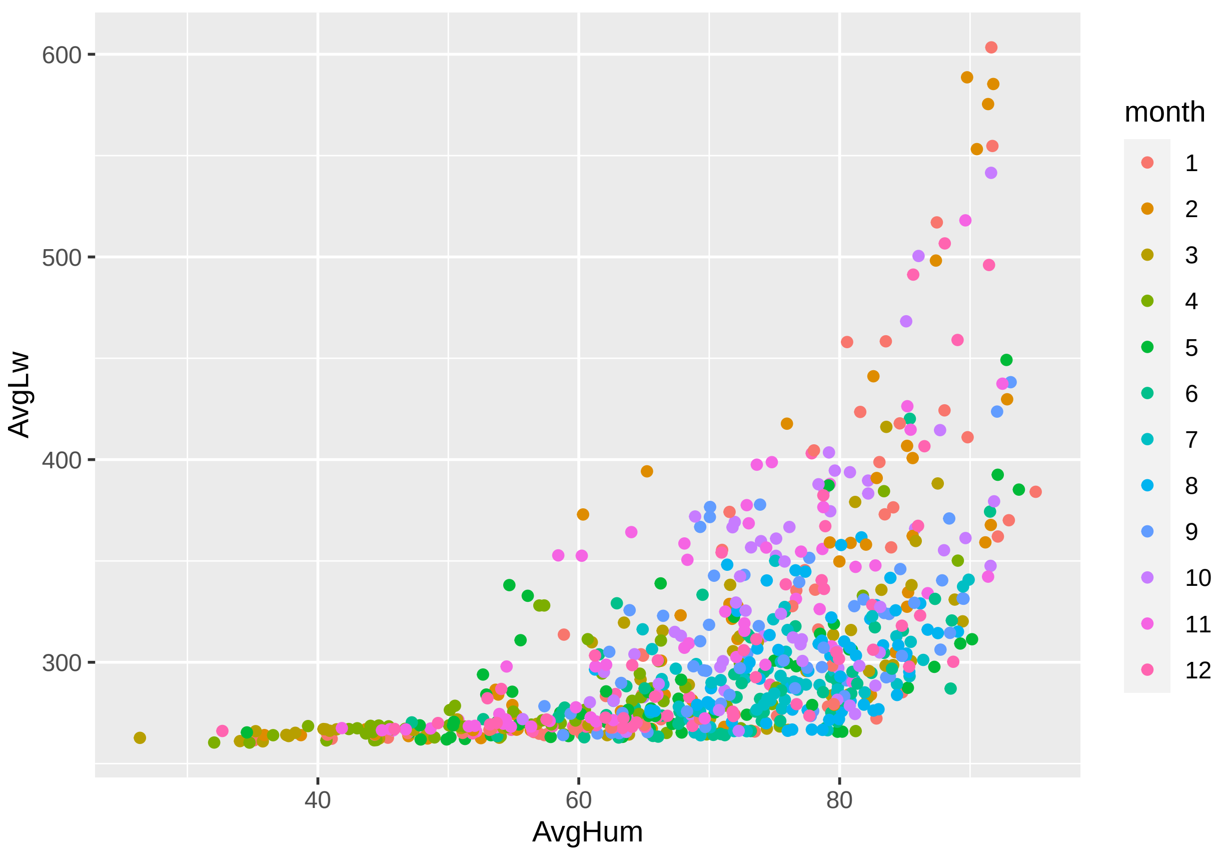
```
# title and all labels included in one more statement
```

- One popular option is to color-code based off of a categorial / factor variable
- See the difference when we include `aes(col=month)`
- Below is a scatter plot

```
# general plot
ggplot(cardinal,aes(x=AvgHum,y=AvgLw))+geom_point()
```

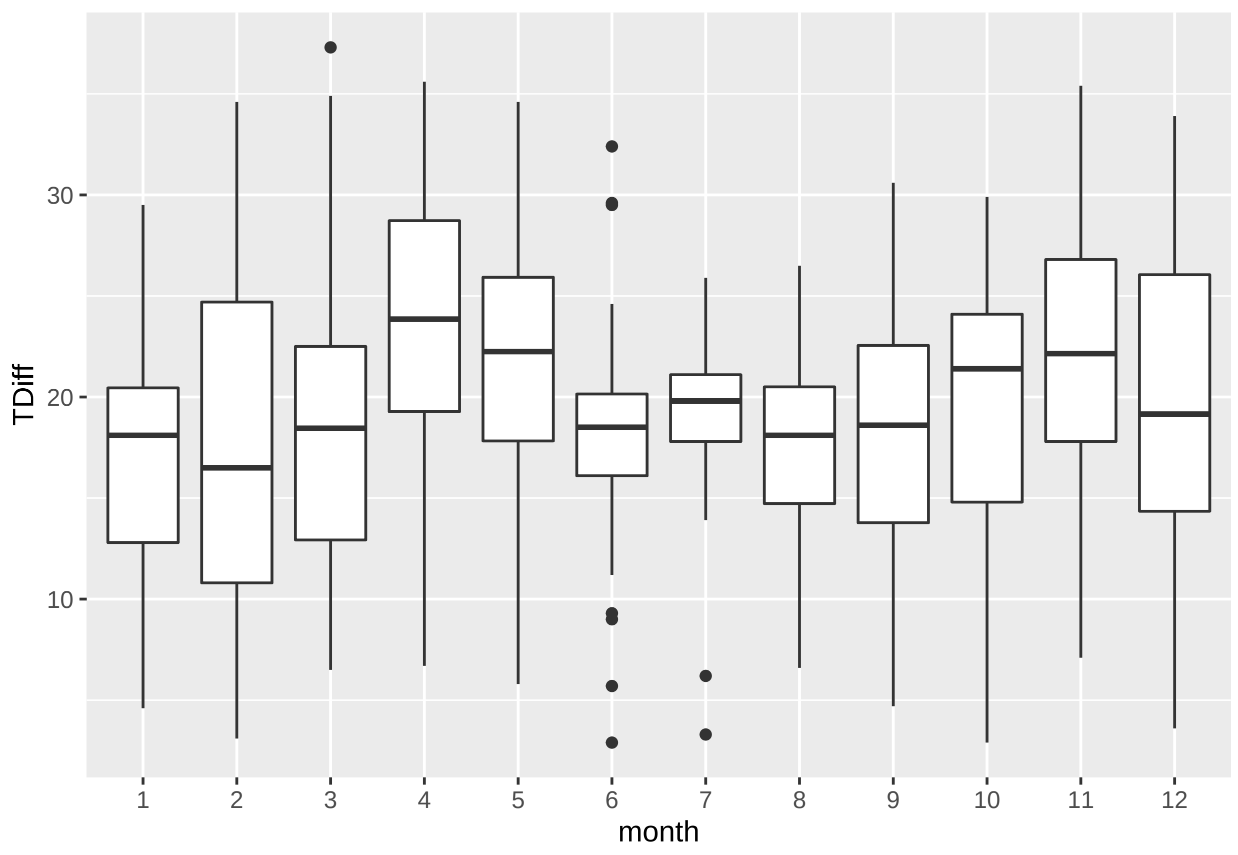


```
# coloring by month to observe trends
ggplot(cardinal,aes(x=AvgHum,y=AvgLw))+geom_point(aes(col=month))
```



- We can also use categorical variables on the x-axis

```
# + geom_boxplot is a great tool to observe spreads

ggplot(cardinal,aes(x=month , y = TDiff))+geom_boxplot()
```

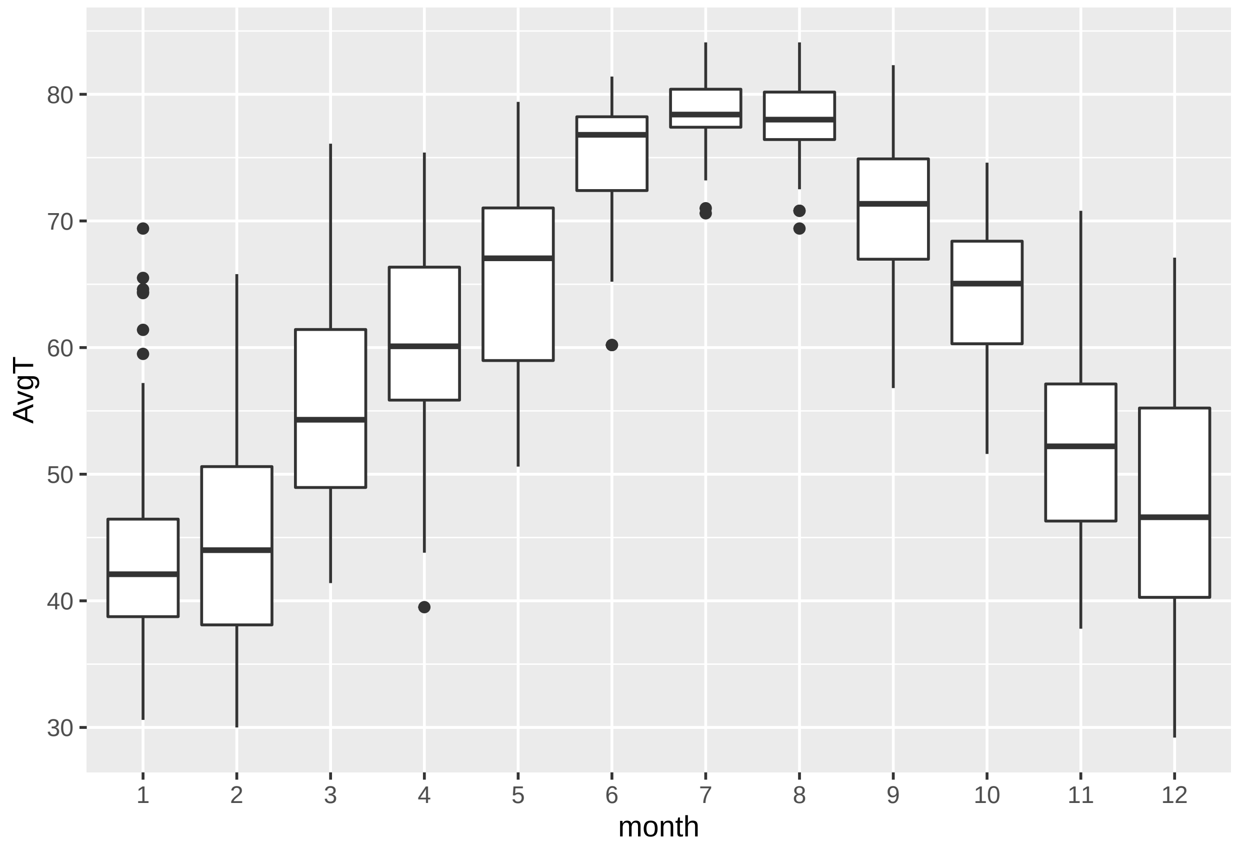


- Box-plots are cumbersome and not super helpful in visalization.

How could we make this boxplot better?

- Note we are looking at average daily tempurature

```
# + geom_boxplot is a great tool to observe spreads

ggplot(cardinal,aes(x=month , y = AvgT))+geom_boxplot()
```

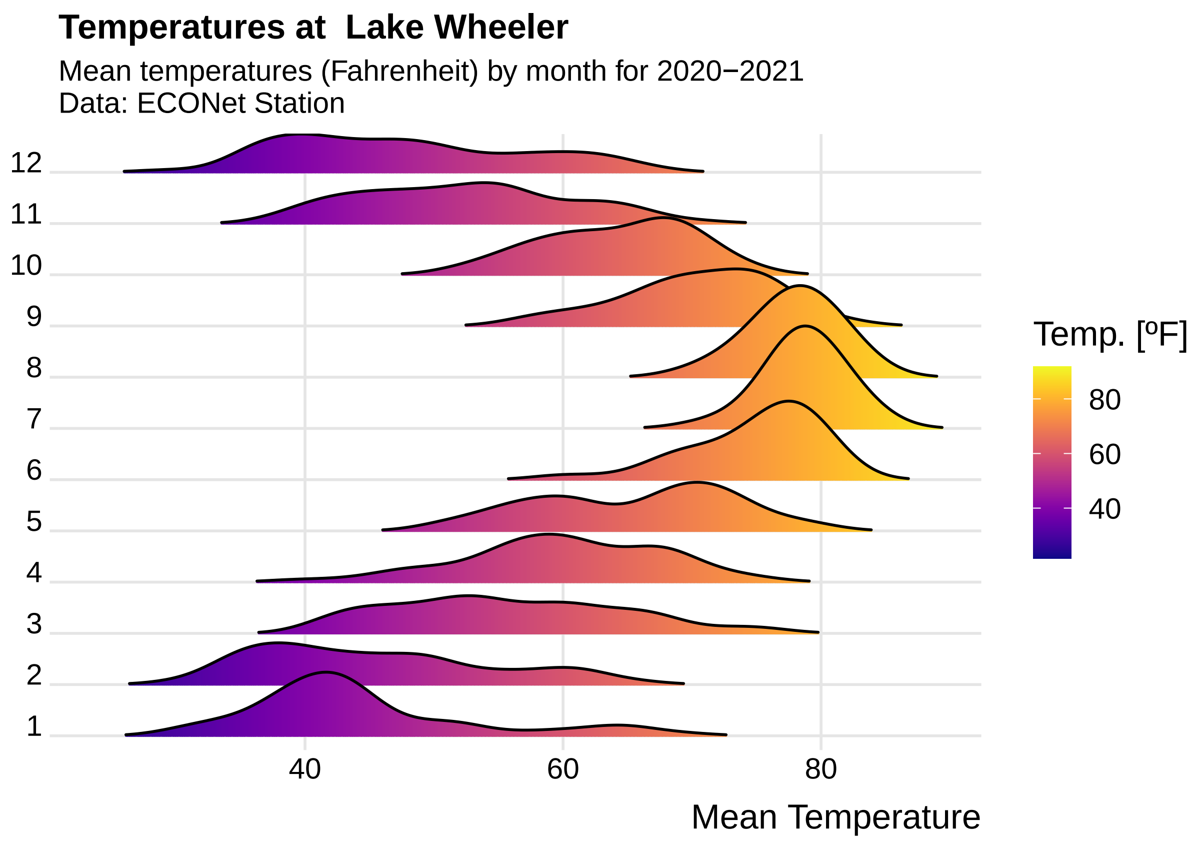


Fancy Plot Time

- Inspiration from ggridges documentation
- We want to stylize the boxplot from the above statement
- We will be using a few libraries here: remeber to use `install.packages("library_name")` first before running the library statment.

```
library(viridis)      ## color palette
library(ggridges)     ## ridges
library(hrbrthemes)   ## plot theme
```

```
ggplot(cardinal, aes(x = AvgT, y = month, fill = ..x..)) +

  geom_density_ridges_gradient(scale = 2, rel_min_height = 0.01, gradient_lwd = 1.) +

  scale_x_continuous(expand = c(0.01, 0)) +

  scale_y_discrete(expand = c(0.01, 0)) +

  scale_fill_viridis(name = "Temp. [ºF]", option = "C") +

  labs(title = 'Temperatures at  Lake Wheeler',
       subtitle = 'Mean temperatures (Fahrenheit) by month for 2020-2021\nData: ECONet Station',
     x = "Mean Temperature") +

 theme_ridges(font_size = 13, grid = TRUE) + theme(axis.title.y = element_blank())
```



```
#
```

After Completing This Tutorial You Can:

- *Define* open data and reproducible science
- *Describe* how to navigate important aspects of the R/RStudio user-interface
- *Recall* how to extract public data from a web portal (Cardinal) and import it into a data software (R/RStudio)
- *Demonstrate* understanding of dataset and statistical software through exploratory data analysis plots and numerical summaries