Project 2 - Brazilian Climate Data

Zachary Palmore

9/21/2020

Objective

In this part of project 2, I will be tidying and briefly analyzing climate data from the National Meteorological Institute of Brazil (IMET). It contains hourly surface weather conditions for the Southeastern states of Rio de Janeiro, São Paulo, and Minas Gerais e Espirito Santo. What we would like to know is;

- Max, min, and mean of temperatures for each station
- Variance in temperature over the study area (SE Brazil)
- Relationship between temperature and humidity

To get those answers, the data needs to be cleaned and tidyed before we can make use of it. We will start by importing the data.

```r
library(readr)
library(tidyverse)
library(infer)
library(googledrive)
```

Importing Data

The link provided was the most streamlined way to get the data without needing permission from the author. There is a package offered that would let me authorize the downloading of the file through Google drive. The file can then be saved in the working directory specified and pulled into R directly.

However, this may give some individual ideas on how best to access my personal drive. To protect myself, I have re-written a new chunk and downloaded the text file manually by following the link. This allows me to keep my personal email and other credentials safe from potential harm. For reference to how it was accessed prior to posting publicly, I have left the code in the notes.

```
# Code prior to public posting with password removed:
# This step streamlined the process and allowed me to grab
# the file directly from my drive
# drive_auth(email = zachary.palmore20@gmail.com)
# surfaceweather <- drive_download("1AU1IwlVIqQrVBqiNruoMfk4OOLlcls5O", # path =  "C:/bigdata", overwri
#
# Then download it from the big data folder
surfaceweather <-
  # Using the readr package seperate the characters by a
  # common source - "tab" in this case
  read_delim("C:/bigdata/surfaceweather.txt",
    "\t", quote = "\'", escape_double = FALSE, trim_ws = TRUE, col_names = TRUE, skip_empty_rows = FALSE
```

```
## Warning: 4979 parsing failures.
##  row col   expected    actual                            file
##   11  -- 31 columns 1 columns 'C:/bigdata/surfaceweather.txt'
##  409  -- 31 columns 1 columns 'C:/bigdata/surfaceweather.txt'
##  434  -- 31 columns 1 columns 'C:/bigdata/surfaceweather.txt'
##  746  -- 31 columns 1 columns 'C:/bigdata/surfaceweather.txt'
## 1131  -- 31 columns 1 columns 'C:/bigdata/surfaceweather.txt'
## .... ... .......... ......... .............................
## See problems(...) for more details.
```

Tidying

This data needs a few changes before we can make sense of it. Having lived in the United States, for me it is easier to read in Fahrenheit than Celsius. All the temperature observations are measured in Celsius. These will need to be converted. There are also many missing values and the data starts collecting at different times for each station in the study leaving gaps for analysis. The data also needs to be converted into data types that we can calculate with too as it is was all a string of characters once imported.

There may be other operations to perform but to summarize, in order to work with the data we need to:

- Convert temperature to Fahrenheit
- Change the data types to numeric vectors
- Remove all missing values
- Remove unreasonable measurements

The shear amount of data makes it difficult to find what we need. FOr example, trying to comprehend what the millions of observation collected need to ensure they are truthful in the analysis is difficult with so much data. Anomalies are likely present but being clouded the be millions of other observations. To begin solving this, we can select out which columns are needed.

```r
climate <- subset(surfaceweather, select = c("wsid",
                                             "mdct",
                                             "temp",
                                             "hmdy"), na.rm = TRUE)
glimpse(climate)
```

```
## Rows: 1,053,554
## Columns: 4
## $ wsid <dbl> 178, 178, 178, 178, 178, 178, 178, 178, 178, 178, NA, 178, 178...
## $ mdct <chr> "11/6/2007 0:00", "11/6/2007 1:00", "11/6/2007 2:00", "11/6/20...
## $ temp <dbl> 29.3, 29.0, 27.4, 25.8, 25.4, 23.8, 22.0, 19.7, 18.3, 22.9, NA...
## $ hmdy <dbl> 35, 39, 44, 58, 57, 62, 72, 86, 93, 75, NA, 61, 0, 0, 0, 36, 3...
```

While that did shrink the amount of data, we are still at over one million rows each with four variable types. Processing this much data take a lot of computing power. To make it easier, it might be best to take a few simple random samples. Then, rather than waiting for data to be processes, we can tidy those samples for downstream analysis, review them for potential errors, clean them up, then apply the process to the climate data overall.

We will start with 3 samples of the variables we have already selected for manipulation. To check for how likely these data are to the real set, we will observe their mean and median of temperature and humidity. This will help give an idea of what to expect in the results, although, even if they do not match closely, we will be running the final example on all the climate data once it is prepared.

```r
set.seed(10012020)
sample1 <- climate %>%
                    rep_sample_n(size = 50,
                                 reps = 100,
                                 replace = TRUE)
set.seed(10022020)
sample2 <- climate %>%
                    rep_sample_n(size = 50,
                                 reps = 100,
                                 replace = TRUE)
set.seed(10032020)
sample3 <- climate %>%
                    rep_sample_n(size = 50,
                                 reps = 100,
                                 replace = TRUE)
climate_sample <- rbind(sample1, sample2, sample3)
```

Before we can calculate the mean and median of temperature and humidity in the sample, they need to be the right data type. In this case, we want numeric vectors for both temperature and humidity. Since there are only 304 missing values for the entire sample (1500 total observations), we can also omit them from the data completely as the loss of any data within the rows will be negligible overall. Then we can run the summary statistics.

```
# find total missing values
sum(is.na(climate_sample))
```

```
## [1] 304
```

```
# converting data types
climate_sample$temp <- as.numeric(climate_sample$temp)
climate_sample$hmdy <- as.numeric(climate_sample$hmdy)
# remove the missing values
climate_sample <- na.omit(climate_sample)
summary(climate_sample$temp)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   18.90   22.30   21.71   25.70   42.50
```

```
summary(climate_sample$hmdy)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   59.00   77.00   71.15   90.00  100.00
```

Our sum of missing values is now zero but given that our study area is of Southeast Brazil, a relatively warm and moist country, the minimum of both categories should not be zero. these may be from when they were setting up the devices or collaborating sensors.

To make sense of these measurements, we can convert them to farenheit. We will place the results in a new column and call in tempf. We will also select all temperature and humidity observations that are greater than 0 for two reasons.

Humidity cannot be 0, especially in the tropical and semi-tropical areas of Brazil. If this were the case, there would be no moisture in the air. On a planet that has over 70% of its surface covered in water, finding a true zero relative humidity is very improbable.

Secondly, temperatures rarely drop below zero degrees Celsius in any parts of Brazil. The only place this could occur is at much higher elevations. Due to their location on and next to the equator, measuring a minimum less than zero degrees Fahrenheit, would also be very improbable.

```r
# Create a function that converts degrees C to F
degctof <- function(c) {
  f <- (c * (9/5)) + 32
  return(f)
}
# Apply the function to all observations of the temperature # column and select those greater than 0
climate_sample <- climate_sample %>%
  mutate(tempf = degctof(temp)) %>%
  filter(tempf > 0
         & hmdy > 0)
# select column for analysis
climate_sample <- climate_sample[,c(2,3,5,6)]
# for comparisson to the overall data
summary(climate_sample$tempf)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   32.00   66.92   72.68   72.96   78.62  108.50
```

```r
summary(climate_sample$hmdy)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.00   62.00   79.00   74.71   90.00  100.00
```

The new sample mean is 22.76 for temperature in degrees Fahrenheit and the median is 22.60. The new sample mean is 74.71 for humidity and its median is 79.00. Now we repeat the process on these on the larger source of climate data.

```r
# find total missing values
sum(is.na(climate))
```

```
## [1] 19916
```

```r
# converting data types
climate$temp <- as.numeric(climate$temp)
climate$hmdy <- as.numeric(climate$hmdy)
# remove the missing values
climate <- na.omit(climate)
climate <- climate %>%
  mutate(tempf = degctof(temp)) %>%
  filter(tempf > 0
         & hmdy > 0)
# select column for analysis
climate <- climate[,c(1,2,4,5)]
# Change column names
climate <- climate %>%
  dplyr::rename(StationID = wsid,
                Time = mdct,
                Humidity = hmdy,
                Temperature = tempf)
# recalculate total missing values
# Compare sample and overall
summary(climate$Temperature)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   32.00   67.10   72.86   72.95   78.44  110.30
```

```
summary(climate$Humidity)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.00   62.00   78.00   74.54   90.00  100.00
```

```
sum(is.na(climate))
```

```
## [1] 0
```

We successfully reduced all 19916 missing values in and its affiliated observations in the climate data to zero. Temperature measurements have been converted from Celsius to Fahrenheit and filtered to have only reasonable observations stored in the data frame. The data is also in a workable format for analysis.

For comparison, it turns out the sample mean and climate data mean were very similar. The climate had a mean temperature of 72.95 while the sample mean was 72.96. Their medians were also very similar at 72.86 and 72.68 respectively. Humidity followed the same pattern so we probably could have used the sample alone for this analysis.

Analysis

At this time, we could export the data as a spreadsheet (.csv) and share. Instead of exporting here, that will be left up to the discretion of the user. If you wanted to, you could simply use the function write.csv and specify the data frame 'climate' and where you want it to go. For now, we will continue with the analysis.

In our objective we would like to find:

- Max, min, and mean of temperatures for each station

- Variance in temperature over the study area (SE Brazil)
- Relationship between temperature and humidity

To start, we have already found the maximum, minimum and mean of temperatures for the entire region. What we need now is to aggregate the climate data to find the statistics we need at based on each StationID. For example;

```
# finding averages of all variables
Station_Stats <- aggregate(climate, by=list(climate$StationID), FUN=mean)
```

```
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
```

```
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA
```

```r
# Removing variables that are not needed by selecting what
# is then displaying the results
Station_Stats[c(2,4,5)]
```

```
##    StationID Humidity Temperature
## 1        178 60.36472    81.21826
## 2        303 76.41116    75.69532
## 3        304 89.40095    64.77404
## 4        305 76.88934    75.56787
## 5        306 72.50111    75.47817
## 6        307 81.36915    75.03731
## 7        308 72.45186    75.08928
## 8        309 76.27279    74.47403
## 9        310 75.90453    74.82425
## 10       311 71.37528    72.36713
## 11       312 77.88482    65.28144
## 12       313 65.61311    70.08601
## 13       314 66.50168    73.27294
```

These statistics can give us an idea of what to expect when observing weather and climate patterns at this station. With the average over time, we can also observe how temperate patterns are change and with that, begin to predict weather. Climate predictions, however, are more involved. For that, we would need more statistics, like what the extremes are for each station.

```r
# finding max of all variables
Station_Stats <- aggregate(climate, by=list(climate$StationID), FUN=max)
# Removing variables that are not needed by selecting what
# is then displaying the results
Station_Stats[c(2,4,5)]
```

```
##    StationID Humidity Temperature
## 1        178      100      101.48
## 2        303      100       98.96
## 3        304      100       91.22
## 4        305       99       99.50
## 5        306       97      102.92
## 6        307      100       96.80
## 7        308       98      105.44
## 8        309      100      102.20
## 9        310       98      102.92
## 10       311      100      110.30
## 11       312      100       90.68
## 12       313      100       95.90
## 13       314       98       98.24
```

We can repeat the process to the find minimum of each station too.

```
# finding min of all variables
Station_Stats <- aggregate(climate, by=list(climate$StationID), FUN=min)
# Removing variables that are not needed by selecting what
# is then displaying the results
Station_Stats[c(2,4,5)]
```

```
##    StationID Humidity Temperature
## 1        178       12       57.56
## 2        303       23       53.60
## 3        304       30       46.58
## 4        305       18       54.86
## 5        306       17       53.60
## 6        307       31       53.96
## 7        308       10       48.92
## 8        309       18       53.96
## 9        310       14       52.16
## 10       311       16       49.64
## 11       312       16       39.02
## 12       313       10       32.00
## 13       314       11       47.30
```

We can see that the humidity and temperature at these stations vary slightly at the maximum and minimum levels. However, the difference between them is much larger. Over time, these difference can be good indicators of climate patterns. We can begin to answer questions such as, how stable is our climate? Or perhaps, how long before another extreme weather event occurs?

These are just a few examples of statistics that help us understand our environment. we can also calculate the variance in the temperature. It will directly tell us how variable the temperatures are over the entire area in this study.

```
var(climate$Temperature)
```

```
## [1] 74.50298
```

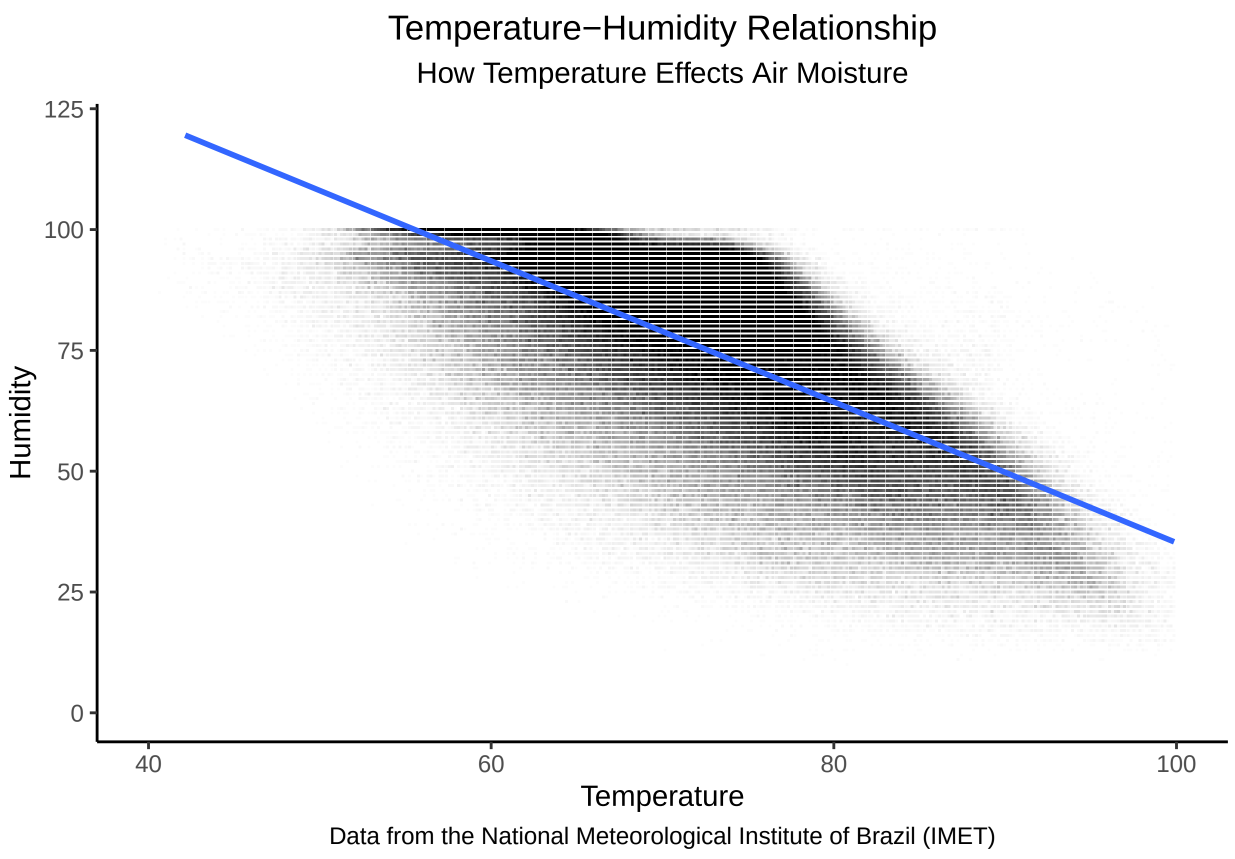
Since this data was accumulated hourly, this is a good statistic to look at for understanding how temperature varies over time. In some cases, being able to predict where there will be a high variations in temperature (low to high or high to low), we can be prepared for thing like heat stress, agricultural failure, and many other things that rely on a stable climate.

As a quick check to review the validity of the data and the relationship between temperature and humidity, we can create a scatter plot. As the temperature increases, we should see a decrease in the relative humidity measurement.

```
ggplot(data = climate, aes(x = Temperature, y = Humidity)) +
  geom_point(colour = "black",
             size = 1.5,
             shape = 46,
             alpha = 1/100,
             na.rm = TRUE
             ) +
  xlim(40, 100) +
  ylim(0, 120) +
  xlab("Temperature") +
  ylab("Humidity") +
  labs( title = "Temperature-Humidity Relationship",
        subtitle = "How Temperature Effects Air Moisture",
        caption = "Data from the National Meteorological Institute of Brazil (IMET)") +
  geom_smooth(method = lm, formula = y ~ x) +
  theme_classic() +
  theme(plot.title = element_text(hjust = 0.5), plot.subtitle = element_text(hjust = 0.5), plot.caption
```

```
## Warning: Removed 1483 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 2 rows containing missing values (geom_smooth).
```



As expected, there is a distinct negative relationship between temperature and humidity. There are two reasons for this, the first that the measure we read as humidity on weather apps, is actually a measurement of the relative humidity. This measures the relative moisture content of the air as a fraction of 100 (percent). On Earth, it can never go below zero, or be much above 100 without it raining. Higher temperatures can also hold more water than lower temperatures. When the temperature increases, and the amount of moisture in the air stays the same, the relative humidity must decrease.

Conclusion

Given the implications for weather on humanity, being able to properly estimate temperature and humidity over large areas could save lives. In this study, we observed how the mean temperature for each station is only slightly different compared to the variations in maximum and minimum temperatures. With additional data in real time, we could develop an algorithm that helped predict what the temperature would be and perhaps advise the public to potentially save people from fatal heat stress. Humidity has a big part to play in what it the temperature feels due to it relationship with temperature. Together, these calculations form the building blocks of weather prediction.