Data Visualization Workshop: Activity 3

Red Rocks Data Science Conference

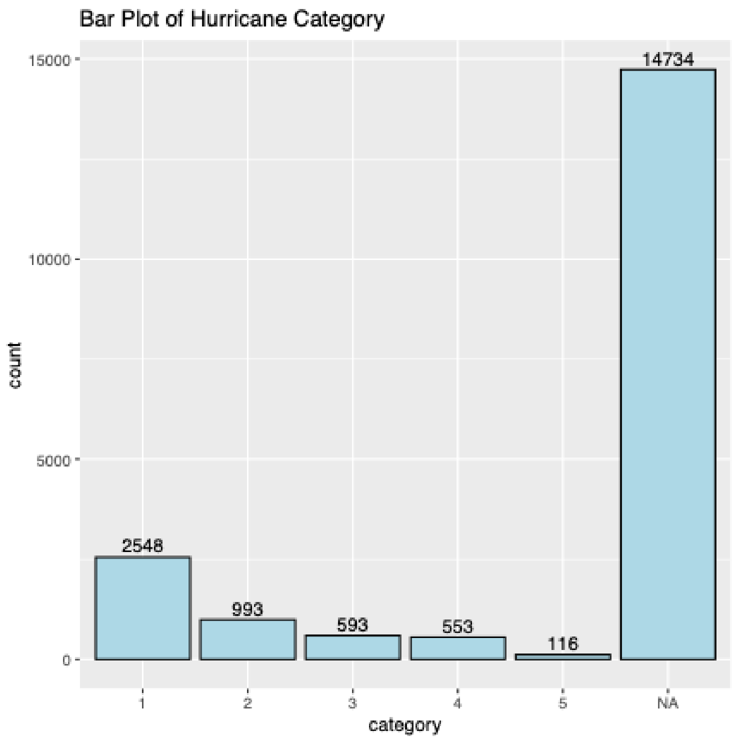
28 May 2025

This activity is going to use the **storms** dataset that is in the dplyr package. This package is loaded when loading the tidyverse. The goal here is to make plots as similar as you can to the ones given below (although, feel free to change the colors of the plots).

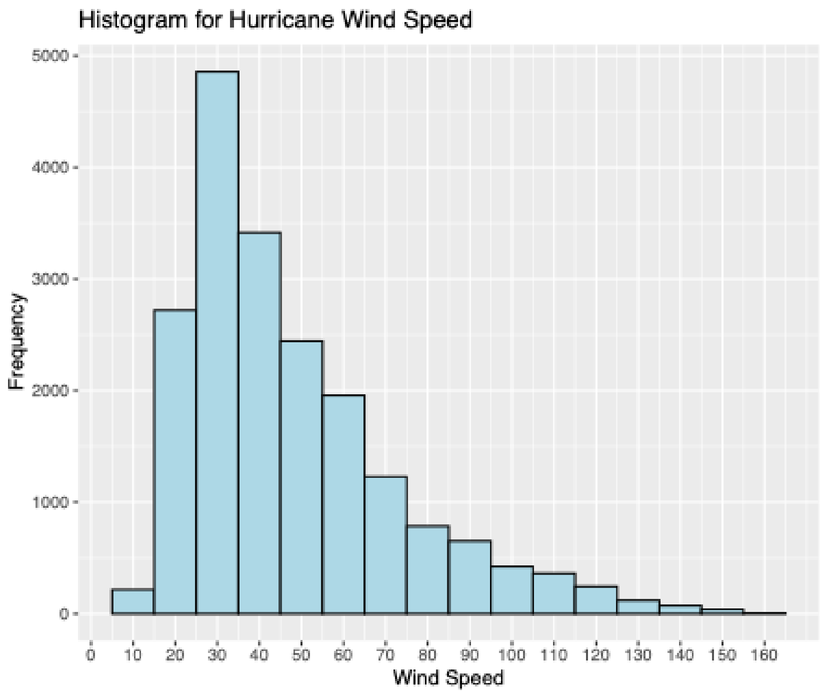
1) If using **R**, load the tidyverse. That will also load the **storms** data set. Type **?storms** to read a bit about this data set. If using Python, import pandas as pd, import all functions from plotnine, and read in the storms dataset that is on the GitHub page.

2) Change the **category** variable to a factor. In **R**, you can do this by typing **storms2 <- mutate(storms, category = factor(category))** and then using **storms2** when making these graphs. In Python, you can do
storms2 = storms
storms2.category = storms2.category.astype("category")
storms2.category = storms2.category.cat.add_categories("NA")
storms2.category = storms2.category.fillna("NA")

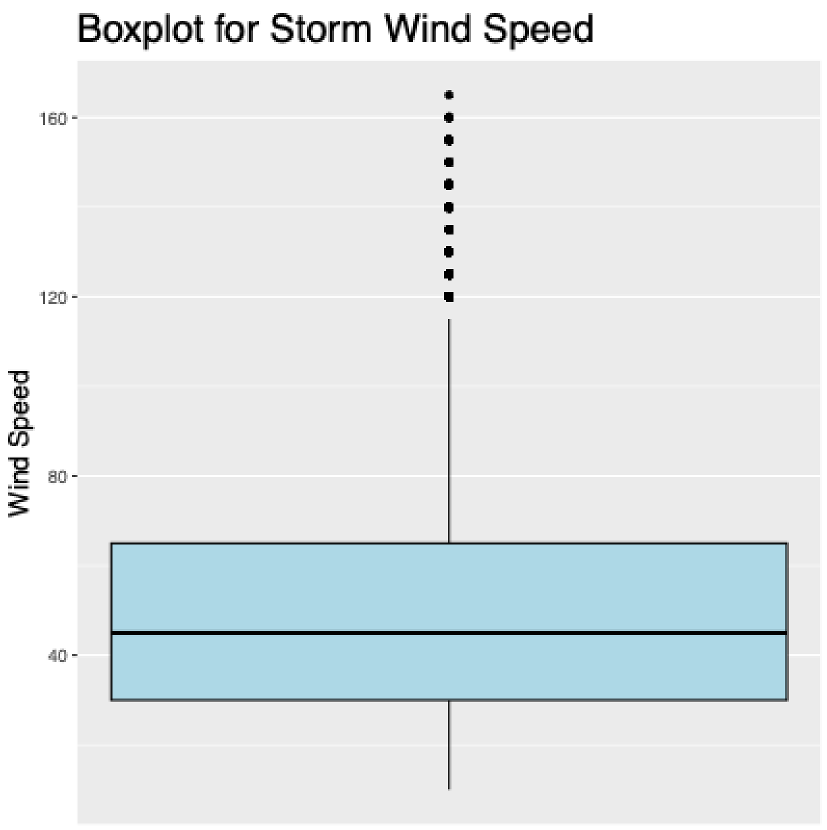
3) Using the **ggplot()** function, create a bar plot of the **category** variable in the **storms2** data set. You can add the counts on top of each bar by adding
geom_text(aes(label = ..count..), stat = "count", vjust = -0.4) in R or by adding
geom_text(aes(label = "..count.."), stat = "count", va = "bottom") in Python.
Remember that titles are centered in *plotnine* by default. Don't worry about left-aligning it.



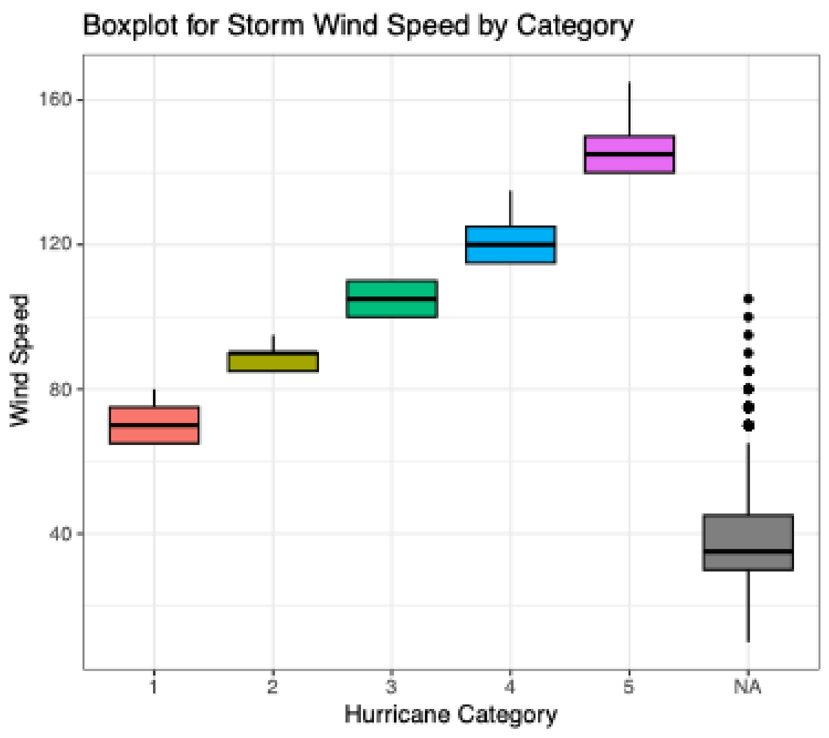
4) Now create a histogram of the **wind** variable. Note that the binwidth is 10.



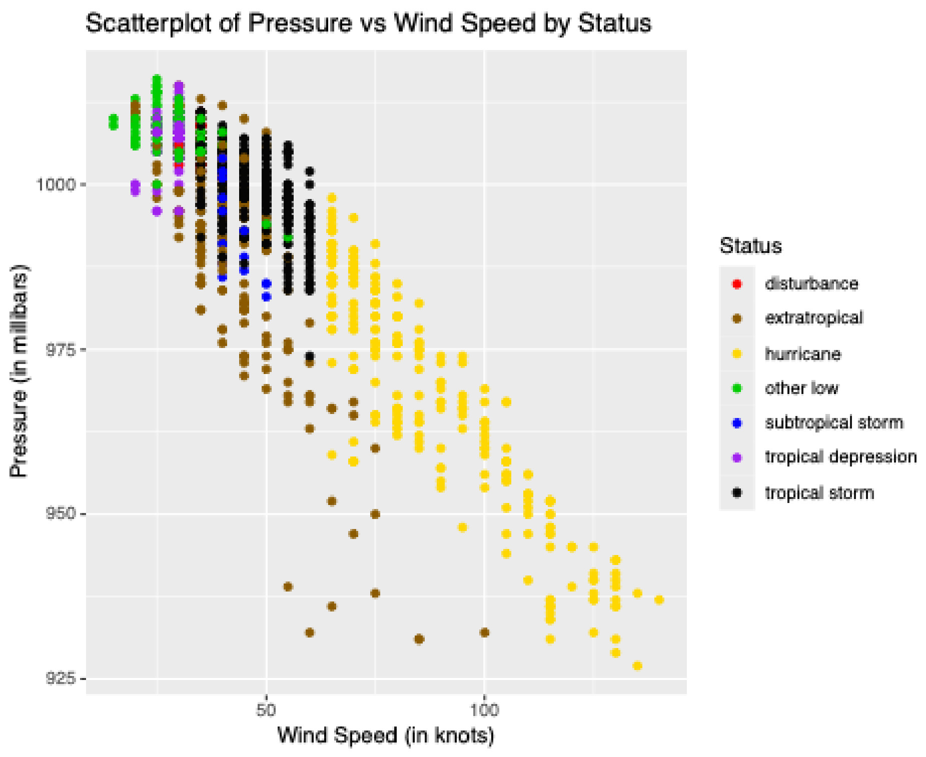
5) Create a boxplot of the **wind** variable. The text size for the axis titles is 14 and the text size for the plot title is 20. You can get rid of the numbers on the x-axis by adding on the function **scale_x_continuous(breaks = NULL)** in **R** or *scale_x_continuous(breaks = [])* in Python.



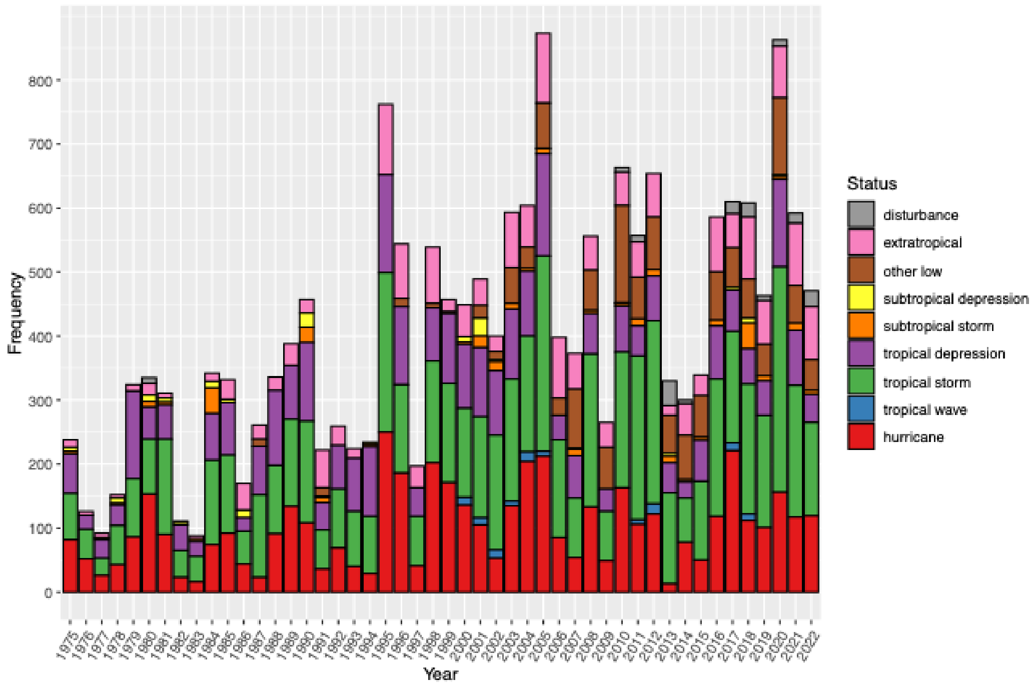
6) Now create side-by-side boxplots for wind speed.



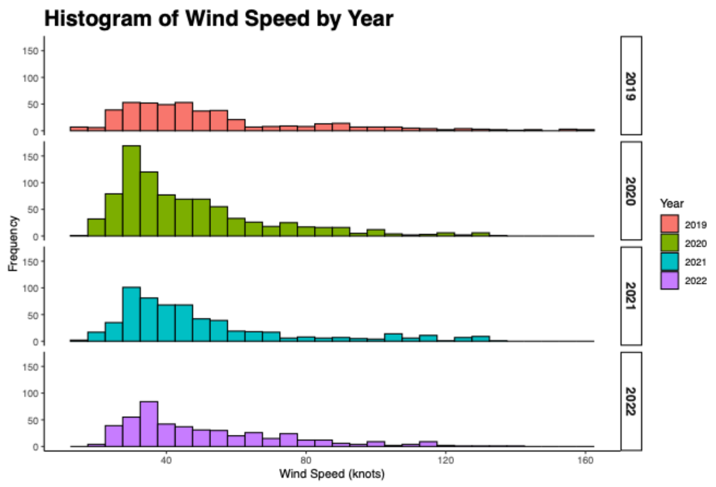
7) Now, take the **storms2** data set, filter it so it only shows storms after the year 2020, then create a scatterplot with wind speed on the x-axis and pressure on the y-axis. Each color is different based on the **status** variable.



8) The plot below is a stacked bar plot of **year** grouped by **status**. The colors can be set by adding on **scale_fill_brewer()**. This color **palette** is called "Set1" and **direction** is set to -1. Notice that the hurricane category is on the bottom to make it easiest to compare across years. This can be accomplished in **R** using **fct_relevel(status, "hurricane", after = Inf)** either in the **aes()** function or before the dataset is put into **ggplot()**. In Python using pandas, it is a little more complicated. The code for that is not given here. Also, the *scale_fill_brewer()* function in *plotnine* does not have “Set1”. Just use a number instead like palette = 1 or palette = 8 (but those palettes to not match this one exactly).



9) Time for a facet grid! Here we have histograms for the wind speed variable for years 2019-2022 (2022 is the last year in the dataset). Each histogram has 20 bins, the background theme is classic, and the title is size 20 and bold. The facet grid labels are size 12 and bold. Those can be changed with the **strip.text** option (*strip_text* in *plotnine*) in the **theme()** function. Note that the theme used here is classic. The title and strip text are bolded, but bolding may not work in plotnine.



10) If you are using **R**, save one of the plots you made above as an **R** object. Install and load the **plotly** library. Then use the **ggplotly()** function on the saved plot to change it to an interactive plot. While Python does have the plotly library, there is no ggplotly() function in Python, so this question can be skipped if you're using Python.