Homework 2

Qi Ren

2023-05-22

Table of Contents

# Data Visualisation - Exploration

Now that you’ve demonstrated your software is setup, and you have the basics of data manipulation, the goal of this assignment is to practice transforming, visualising, and exploring data.

# Mass shootings in the US

In July 2012, in the aftermath of a mass shooting in a movie theater in Aurora, Colorado, [Mother Jones](https://www.motherjones.com/politics/2012/07/mass-shootings-map/) published a report on mass shootings in the United States since 1982. Importantly, they provided the underlying data set as [an open-source database](https://www.motherjones.com/politics/2012/12/mass-shootings-mother-jones-full-data/) for anyone interested in studying and understanding this criminal behavior.

## Obtain the data

## Rows: 125  
## Columns: 14  
## $ case <chr> "Oxford High School shooting", "San Jose VTA shoo…  
## $ year <dbl> 2021, 2021, 2021, 2021, 2021, 2021, 2020, 2020, 2…  
## $ month <chr> "Nov", "May", "Apr", "Mar", "Mar", "Mar", "Mar", …  
## $ day <dbl> 30, 26, 15, 31, 22, 16, 16, 26, 10, 6, 31, 4, 3, …  
## $ location <chr> "Oxford, Michigan", "San Jose, California", "Indi…  
## $ summary <chr> "Ethan Crumbley, a 15-year-old student at Oxford …  
## $ fatalities <dbl> 4, 9, 8, 4, 10, 8, 4, 5, 4, 3, 7, 9, 22, 3, 12, 5…  
## $ injured <dbl> 7, 0, 7, 1, 0, 1, 0, 0, 3, 8, 25, 27, 26, 12, 4, …  
## $ total\_victims <dbl> 11, 9, 15, 5, 10, 9, 4, 5, 7, 11, 32, 36, 48, 15,…  
## $ location\_type <chr> "School", "Workplace", "Workplace", "Workplace", …  
## $ male <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, T…  
## $ age\_of\_shooter <dbl> 15, 57, 19, NA, 21, 21, 31, 51, NA, NA, 36, 24, 2…  
## $ race <chr> NA, NA, "White", NA, NA, "White", NA, "Black", "B…  
## $ prior\_mental\_illness <chr> NA, "Yes", "Yes", NA, "Yes", NA, NA, NA, NA, NA, …

| column(variable) | description |
| --- | --- |
| case | short name of incident |
| year, month, day | year, month, day in which the shooting occurred |
| location | city and state where the shooting occcurred |
| summary | brief description of the incident |
| fatalities | Number of fatalities in the incident, excluding the shooter |
| injured | Number of injured, non-fatal victims in the incident, excluding the shooter |
| total\_victims | number of total victims in the incident, excluding the shooter |
| location\_type | generic location in which the shooting took place |
| male | logical value, indicating whether the shooter was male |
| age\_of\_shooter | age of the shooter when the incident occured |
| race | race of the shooter |
| prior\_mental\_illness | did the shooter show evidence of mental illness prior to the incident? |

## Explore the data

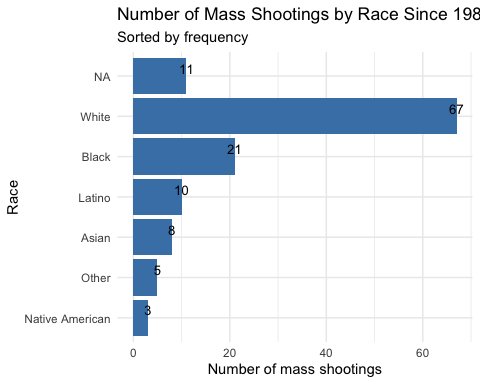
### Specific questions

* Generate a data frame that summarizes the number of mass shootings per year.

df1 <- mass\_shootings %>%   
 group\_by(year) %>%   
 summarise(count = n())

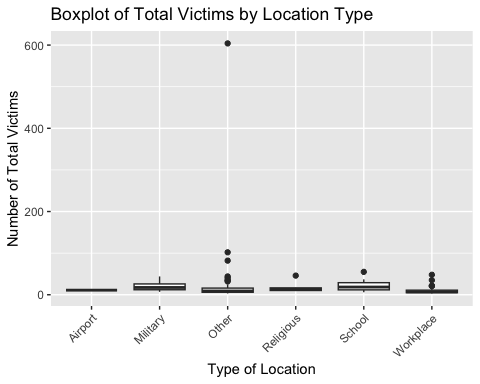
* Generate a bar chart that identifies the number of mass shooters associated with each race category. The bars should be sorted from highest to lowest and each bar should show its number.

df2 <- mass\_shootings %>%   
 group\_by(race) %>%   
 summarise(count = n()) %>%   
 arrange(desc(count))  
   
ggplot(df2, aes(x = reorder(race, count), y = count)) +  
 geom\_bar(stat = 'identity', fill = 'steelblue') +  
 geom\_text(aes(label = count), vjust = -0.3, size = 3.5) +  
 coord\_flip() +  
 labs(x = "Race",   
 y = "Number of mass shootings",   
 title = "Number of Mass Shootings by Race Since 1982",  
 subtitle = "Sorted by frequency") +  
 theme\_minimal()



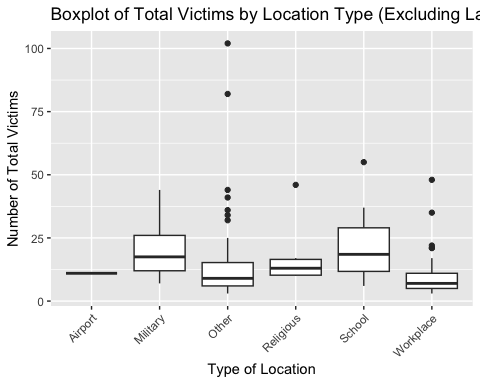
* Generate a boxplot visualizing the number of total victims, by type of location.

# Create the plot  
ggplot(mass\_shootings, aes(x = location\_type, y = total\_victims)) +  
 geom\_boxplot() +  
 labs(x = "Type of Location",   
 y = "Number of Total Victims",   
 title = "Boxplot of Total Victims by Location Type") +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))



* Redraw the same plot, but remove the Las Vegas Strip massacre from the dataset.

df3 <- mass\_shootings[!(mass\_shootings$case == "Las Vegas Strip massacre"),]  
  
# Create the plot  
ggplot(df3, aes(x = location\_type, y = total\_victims)) +  
 geom\_boxplot() +  
 labs(x = "Type of Location",   
 y = "Number of Total Victims",   
 title = "Boxplot of Total Victims by Location Type (Excluding Las Vegas Strip Massacre)") +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))



### More open-ended questions

Address the following questions. Generate appropriate figures/tables to support your conclusions.

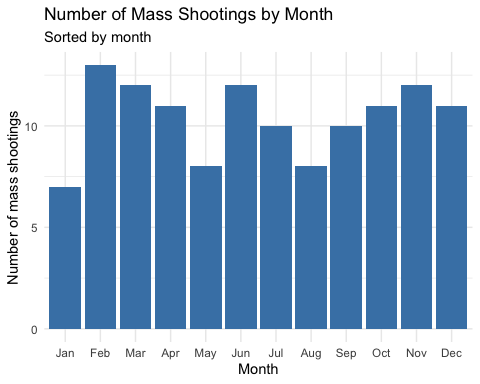
* How many white males with prior signs of mental illness initiated a mass shooting after 2000?

# Filter the dataframe  
filtered\_df <- mass\_shootings %>%   
 filter(race == "White",   
 male == "TRUE",   
 year >= 2000,   
 prior\_mental\_illness == "Yes")  
  
# Count the number of rows in the filtered dataframe  
num\_cases <- nrow(filtered\_df)  
  
print(num\_cases)

## [1] 23

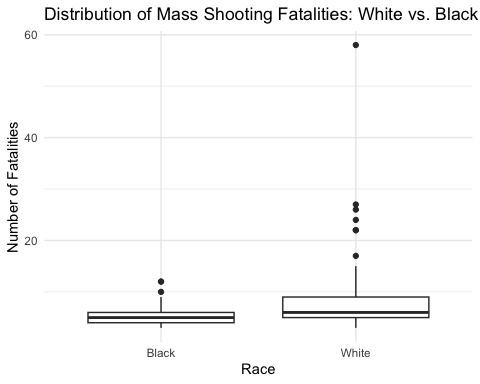
* Which month of the year has the most mass shootings? Generate a bar chart sorted in chronological (natural) order (Jan-Feb-Mar- etc) to provide evidence of your answer.

# Convert the month column to a factor and specify the levels in order  
mass\_shootings$month <- factor(mass\_shootings$month, levels = c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"))  
  
# Create a summary data frame  
summary\_df <- mass\_shootings %>%  
 group\_by(month) %>%  
 summarise(count = n(), .groups = 'drop')  
  
# Create the plot  
ggplot(summary\_df, aes(x = month, y = count)) +  
 geom\_bar(stat = 'identity', fill = 'steelblue') +  
 labs(x = "Month",   
 y = "Number of mass shootings",   
 title = "Number of Mass Shootings by Month",  
 subtitle = "Sorted by month") +  
 theme\_minimal()

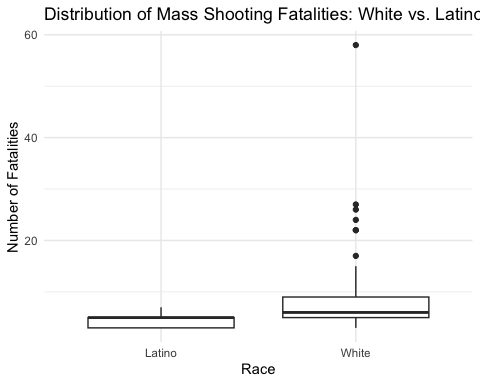


* How does the distribution of mass shooting fatalities differ between White and Black shooters? What about White and Latino shooters?

# Filter the dataframe to only include White and Black shooters  
df\_white\_black <- mass\_shootings %>% filter(race %in% c("White", "Black"))  
  
# Create the boxplot for White and Black shooters  
ggplot(df\_white\_black, aes(x = race, y = fatalities)) +  
 geom\_boxplot() +  
 labs(x = "Race",   
 y = "Number of Fatalities",   
 title = "Distribution of Mass Shooting Fatalities: White vs. Black Shooters") +  
 theme\_minimal()



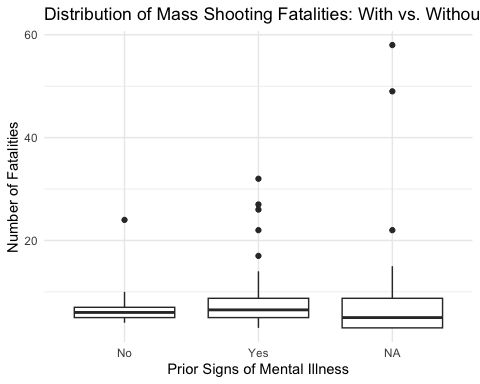
# Filter the dataframe to only include White and Latino shooters  
df\_white\_latino <- mass\_shootings %>% filter(race %in% c("White", "Latino"))  
  
# Create the boxplot for White and Latino shooters  
ggplot(df\_white\_latino, aes(x = race, y = fatalities)) +  
 geom\_boxplot() +  
 labs(x = "Race",   
 y = "Number of Fatalities",   
 title = "Distribution of Mass Shooting Fatalities: White vs. Latino Shooters") +  
 theme\_minimal()



### Very open-ended

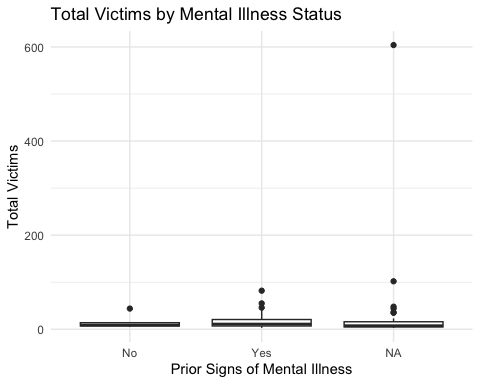
* Are mass shootings with shooters suffering from mental illness different from mass shootings with no signs of mental illness in the shooter?

# Create the boxplot  
ggplot(mass\_shootings, aes(x = prior\_mental\_illness, y = fatalities)) +  
 geom\_boxplot() +  
 labs(x = "Prior Signs of Mental Illness",   
 y = "Number of Fatalities",   
 title = "Distribution of Mass Shooting Fatalities: With vs. Without Prior Signs of Mental Illness") +  
 theme\_minimal()

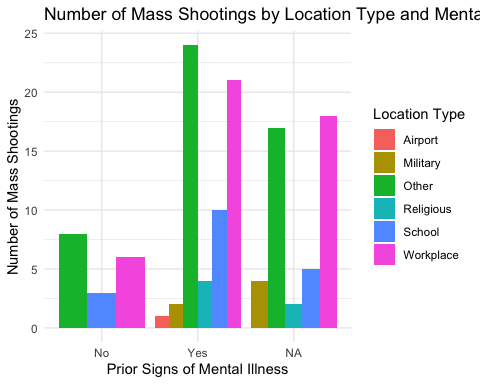


* Assess the relationship between mental illness and total victims, mental illness and location type, and the intersection of all three variables.

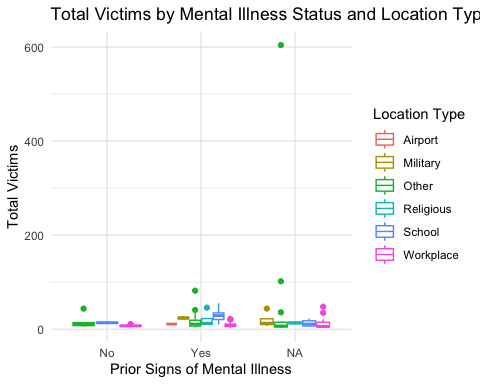
# Create a boxplot to assess the relationship between mental illness and total victims  
ggplot(mass\_shootings, aes(x = prior\_mental\_illness, y = total\_victims)) +  
 geom\_boxplot() +  
 labs(x = "Prior Signs of Mental Illness",   
 y = "Total Victims",   
 title = "Total Victims by Mental Illness Status") +  
 theme\_minimal()



# Create a bar chart to assess the relationship between mental illness and location type  
mass\_shootings %>%  
 group\_by(prior\_mental\_illness, location\_type) %>%  
 summarise(count = n(), .groups = 'drop') %>%  
 ggplot(aes(x = prior\_mental\_illness, y = count, fill = location\_type)) +  
 geom\_bar(stat = 'identity', position = 'dodge') +  
 labs(x = "Prior Signs of Mental Illness",   
 y = "Number of Mass Shootings",   
 title = "Number of Mass Shootings by Location Type and Mental Illness Status",  
 fill = "Location Type") +  
 theme\_minimal()



# Create a boxplot to assess the intersection of all three variables  
ggplot(mass\_shootings, aes(x = prior\_mental\_illness, y = total\_victims, color = location\_type)) +  
 geom\_boxplot() +  
 labs(x = "Prior Signs of Mental Illness",   
 y = "Total Victims",   
 title = "Total Victims by Mental Illness Status and Location Type",  
 color = "Location Type") +  
 theme\_minimal()



Make sure to provide a couple of sentences of written interpretation of your tables/figures. Graphs and tables alone will not be sufficient to answer this question.

# Exploring credit card fraud

We will be using a dataset with credit card transactions containing legitimate and fraud transactions. Fraud is typically well below 1% of all transactions, so a naive model that predicts that all transactions are legitimate and not fraudulent would have an accuracy of well over 99%– pretty good, no? (well, not quite as we will see later in the course)

You can read more on credit card fraud on [Credit Card Fraud Detection Using Weighted Support Vector Machine](https://www.scirp.org/journal/paperinformation.aspx?paperid=105944)

The dataset we will use consists of credit card transactions and it includes information about each transaction including customer details, the merchant and category of purchase, and whether or not the transaction was a fraud.

## Obtain the data

The dataset is too large to be hosted on Canvas or Github, so please download it from dropbox <https://www.dropbox.com/sh/q1yk8mmnbbrzavl/AAAxzRtIhag9Nc_hODafGV2ka?dl=0> and save it in your dsb repo, under the data folder

## Rows: 671,028  
## Columns: 14  
## $ trans\_date\_trans\_time <dttm> 2019-02-22 07:32:58, 2019-02-16 15:07:20, 2019-…  
## $ trans\_year <dbl> 2019, 2019, 2019, 2019, 2019, 2019, 2019, 2020, …  
## $ category <chr> "entertainment", "kids\_pets", "personal\_care", "…  
## $ amt <dbl> 7.79, 3.89, 8.43, 40.00, 54.04, 95.61, 64.95, 3.…  
## $ city <chr> "Veedersburg", "Holloway", "Arnold", "Apison", "…  
## $ state <chr> "IN", "OH", "MO", "TN", "CO", "GA", "MN", "AL", …  
## $ lat <dbl> 40.1186, 40.0113, 38.4305, 35.0149, 39.4584, 32.…  
## $ long <dbl> -87.2602, -80.9701, -90.3870, -85.0164, -106.385…  
## $ city\_pop <dbl> 4049, 128, 35439, 3730, 277, 1841, 136, 190178, …  
## $ job <chr> "Development worker, community", "Child psychoth…  
## $ dob <date> 1959-10-19, 1946-04-03, 1985-03-31, 1991-01-28,…  
## $ merch\_lat <dbl> 39.41679, 39.74585, 37.73078, 34.53277, 39.95244…  
## $ merch\_long <dbl> -87.52619, -81.52477, -91.36875, -84.10676, -106…  
## $ is\_fraud <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, …

The data dictionary is as follows

| column(variable) | description |
| --- | --- |
| trans\_date\_trans\_time | Transaction DateTime |
| trans\_year | Transaction year |
| category | category of merchant |
| amt | amount of transaction |
| city | City of card holder |
| state | State of card holder |
| lat | Latitude location of purchase |
| long | Longitude location of purchase |
| city\_pop | card holder’s city population |
| job | job of card holder |
| dob | date of birth of card holder |
| merch\_lat | Latitude Location of Merchant |
| merch\_long | Longitude Location of Merchant |
| is\_fraud | Whether Transaction is Fraud (1) or Not (0) |

* In this dataset, how likely are fraudulent transactions? Generate a table that summarizes the number and frequency of fraudulent transactions per year.

fraud\_cases\_by\_year <- card\_fraud %>%   
   
 # Filter only fraud cases  
 filter(is\_fraud == "1") %>%   
   
 # Group by year and summarise fraud case in each year  
 group\_by(trans\_year) %>%   
 summarise(count\_fraud\_cases = n())  
  
fraud\_cases\_by\_year

## # A tibble: 2 × 2  
## trans\_year count\_fraud\_cases  
## <dbl> <int>  
## 1 2019 2721  
## 2 2020 1215

* How much money (in US$ terms) are fraudulent transactions costing the company? Generate a table that summarizes the total amount of legitimate and fraudulent transactions per year and calculate the % of fraudulent transactions, in US$ terms.

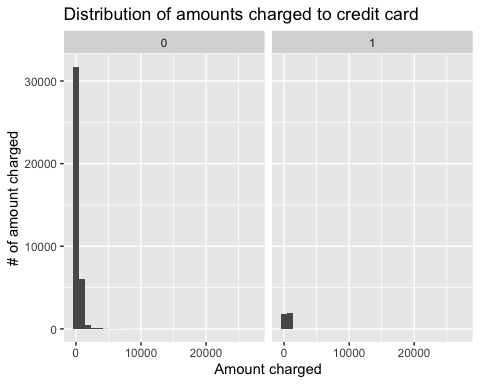
# Summarise data  
summary <- card\_fraud %>%  
 group\_by(trans\_year, is\_fraud) %>%  
 summarise(total\_amt = sum(amt, na.rm = TRUE),  
 .groups = "drop")  
  
# Calculate the yearly total  
yearly\_totals <- summary %>%  
 group\_by(trans\_year) %>%  
 summarise(yearly\_total\_amt = sum(total\_amt),  
 .groups = "drop")  
  
# Join yearly total back to summary  
summary <- summary %>%  
 left\_join(yearly\_totals, by = "trans\_year")  
  
# Calculate the percentage of fraudulent transactions  
summary <- summary %>%  
 mutate(fraud\_percentage = ifelse(is\_fraud == 1, (total\_amt / yearly\_total\_amt) \* 100, 0))  
  
# Print summary  
print(summary)

## # A tibble: 4 × 5  
## trans\_year is\_fraud total\_amt yearly\_total\_amt fraud\_percentage  
## <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 2019 0 32182901. 33606041. 0   
## 2 2019 1 1423140. 33606041. 4.23  
## 3 2020 0 12925914. 13577863. 0   
## 4 2020 1 651949. 13577863. 4.80

* Generate a histogram that shows the distribution of amounts charged to credit card, both for legitimate and fraudulent accounts. Also, for both types of transactions, calculate some quick summary statistics.

card\_fraud %>%   
   
 # Group by fraud cases and amount and summarise total amount of each case  
 group\_by(is\_fraud,amt) %>%   
 summarise(count = n()) %>%   
   
 # Plot histogram and splited facet by whether is fraud or not  
 ggplot(aes(x=amt)) +  
 geom\_histogram() +  
 labs(x= "Amount charged", y= "# of amount charged") +  
 ggtitle("Distribution of amounts charged to credit card") +  
 facet\_wrap((~is\_fraud))

## `summarise()` has grouped output by 'is\_fraud'. You can override using the  
## `.groups` argument.  
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



# Filter out fraudulent transactions  
fraudulent <- card\_fraud %>%  
 filter(is\_fraud == 1)  
  
legitimate <- card\_fraud %>%   
 filter(is\_fraud == 0)  
# Calculate summary statistics  
fraudulent\_summary <- fraudulent %>%  
 summarise(Mean = mean(amt, na.rm = TRUE),  
 Median = median(amt, na.rm = TRUE),  
 Min = min(amt, na.rm = TRUE),  
 Max = max(amt, na.rm = TRUE))  
  
legitimate\_summary <- legitimate %>%  
 summarise(Mean = mean(amt, na.rm = TRUE),  
 Median = median(amt, na.rm = TRUE),  
 Min = min(amt, na.rm = TRUE),  
 Max = max(amt, na.rm = TRUE))  
  
# Print summary statistics  
print("Fraudulent Transaction Summary:")

## [1] "Fraudulent Transaction Summary:"

print(fraudulent\_summary)

## # A tibble: 1 × 4  
## Mean Median Min Max  
## <dbl> <dbl> <dbl> <dbl>  
## 1 527. 369. 1.06 1334.

print("Legitimate Transaction Summary:")

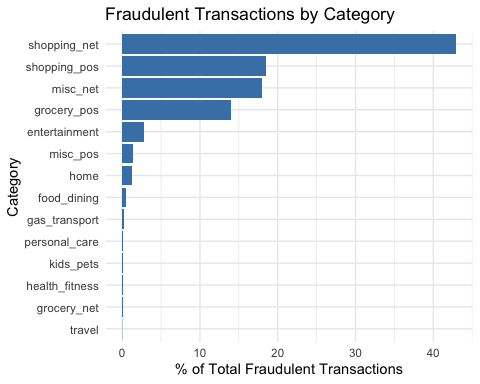
## [1] "Legitimate Transaction Summary:"

print(legitimate\_summary)

## # A tibble: 1 × 4  
## Mean Median Min Max  
## <dbl> <dbl> <dbl> <dbl>  
## 1 67.6 47.2 1 27120.

* What types of purchases are most likely to be instances of fraud? Consider category of merchants and produce a bar chart that shows % of total fraudulent transactions sorted in order.

fraudulent <- card\_fraud %>%  
 filter(is\_fraud == 1)  
  
# Calculate % of total fraudulent transactions by category  
fraud\_by\_category <- fraudulent %>%  
 group\_by(category) %>%  
 summarise(total\_amt = sum(amt), .groups = "drop") %>%  
 mutate(percent\_of\_total = (total\_amt / sum(total\_amt)) \* 100) %>%  
 arrange(desc(percent\_of\_total))  
  
# Plot a bar chart  
ggplot(fraud\_by\_category, aes(x = reorder(category, percent\_of\_total), y = percent\_of\_total)) +  
 geom\_bar(stat = "identity", fill = "steelblue") +  
 coord\_flip() +  
 theme\_minimal() +  
 labs(title = "Fraudulent Transactions by Category", x = "Category", y = "% of Total Fraudulent Transactions")



* When is fraud more prevalent? Which days, months, hours? To create new variables to help you in your analysis, we use the lubridate package and the following code

mutate(  
 date\_only = lubridate::date(trans\_date\_trans\_time),  
 month\_name = lubridate::month(trans\_date\_trans\_time, label=TRUE),  
 hour = lubridate::hour(trans\_date\_trans\_time),  
 weekday = lubridate::wday(trans\_date\_trans\_time, label = TRUE)  
 )

* Are older customers significantly more likely to be victims of credit card fraud? To calculate a customer’s age, we use the lubridate package and the following code

mutate(  
 age = interval(dob, trans\_date\_trans\_time) / years(1),  
 )

new\_card\_fraud <- card\_fraud %>%   
   
 # Change date/month/hour/weekday format  
 mutate(  
 date\_only = lubridate::date(trans\_date\_trans\_time),  
 month\_name = lubridate::month(trans\_date\_trans\_time, label=TRUE),  
 hour = lubridate::hour(trans\_date\_trans\_time),  
 weekday = lubridate::wday(trans\_date\_trans\_time, label = TRUE)) %>%   
   
 # Calculate age  
 mutate(age = interval(dob, trans\_date\_trans\_time) / years(1),)   
  
# View no. of fraud cases by year  
new\_card\_fraud %>%   
 group\_by(trans\_year) %>%   
 summarise(count = n())

## # A tibble: 2 × 2  
## trans\_year count  
## <dbl> <int>  
## 1 2019 478646  
## 2 2020 192382

#More fruad cases occurred in 2019  
  
# View no. of fraud cases by month  
new\_card\_fraud %>%   
 group\_by(month\_name) %>%   
 summarise(count = n()) %>%   
 arrange(desc(count))

## # A tibble: 12 × 2  
## month\_name count  
## <ord> <int>  
## 1 May 75801  
## 2 Mar 74478  
## 3 Jun 74214  
## 4 Dec 72986  
## 5 Apr 69876  
## 6 Jan 53806  
## 7 Feb 50660  
## 8 Aug 45280  
## 9 Jul 44974  
## 10 Sep 36533  
## 11 Nov 36333  
## 12 Oct 36087

#No visible trend in cases in months  
  
# View no. of fraud cases by date  
new\_card\_fraud %>%   
 group\_by(date\_only) %>%   
 summarise(count = n())

## # A tibble: 537 × 2  
## date\_only count  
## <date> <int>  
## 1 2019-01-01 1287  
## 2 2019-01-02 587  
## 3 2019-01-03 629  
## 4 2019-01-04 763  
## 5 2019-01-05 735  
## 6 2019-01-06 1031  
## 7 2019-01-07 1223  
## 8 2019-01-08 1167  
## 9 2019-01-09 607  
## 10 2019-01-10 626  
## # ℹ 527 more rows

#No trend observed  
  
# View no. of fraud cases by hour  
new\_card\_fraud %>%   
 group\_by(hour) %>%   
 summarise(count = n()) %>%   
 arrange(desc(count))

## # A tibble: 24 × 2  
## hour count  
## <int> <int>  
## 1 22 34674  
## 2 23 34625  
## 3 18 34131  
## 4 19 33987  
## 5 20 33978  
## 6 13 33972  
## 7 16 33960  
## 8 15 33907  
## 9 17 33842  
## 10 21 33814  
## # ℹ 14 more rows

#More fraud occur at night  
  
# View no. of fraud cases by weekday  
new\_card\_fraud %>%   
 group\_by(weekday) %>%   
 summarise(count = n()) %>%   
 arrange(desc(count))

## # A tibble: 7 × 2  
## weekday count  
## <ord> <int>  
## 1 Mon 131419  
## 2 Sun 129550  
## 3 Sat 104039  
## 4 Tue 82930  
## 5 Fri 78951  
## 6 Thu 76200  
## 7 Wed 67939

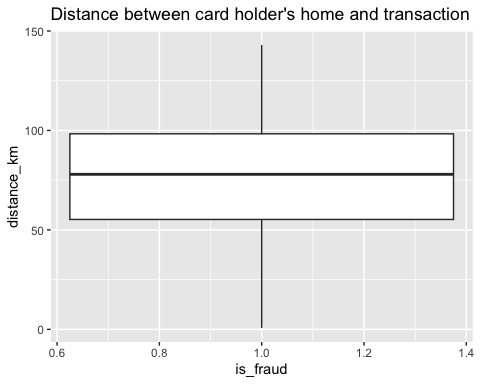
#More fraud occur over the weekend, from Sat to Mon.  
  
# View no. of fraud cases by age  
new\_card\_fraud %>%   
 group\_by(age) %>%   
 summarise(count = n())

## # A tibble: 670,944 × 2  
## age count  
## <dbl> <int>  
## 1 13.9 1  
## 2 13.9 1  
## 3 13.9 1  
## 4 13.9 1  
## 5 13.9 1  
## 6 13.9 1  
## 7 13.9 1  
## 8 13.9 1  
## 9 13.9 1  
## 10 13.9 1  
## # ℹ 670,934 more rows

#No trend observed

* Is fraud related to distance? The distance between a card holder’s home and the location of the transaction can be a feature that is related to fraud. To calculate distance, we need the latidue/longitude of card holders’s home and the latitude/longitude of the transaction, and we will use the [Haversine formula](https://en.wikipedia.org/wiki/Haversine_formula) to calculate distance. I adapted code to [calculate distance between two points on earth](https://www.geeksforgeeks.org/program-distance-two-points-earth/amp/) which you can find below

# distance between card holder's home and transaction  
# code adapted from https://www.geeksforgeeks.org/program-distance-two-points-earth/amp/  
  
  
card\_fraud <- card\_fraud %>%  
 mutate(  
   
 # convert latitude/longitude to radians  
 lat1\_radians = lat / 57.29577951,  
 lat2\_radians = merch\_lat / 57.29577951,  
 long1\_radians = long / 57.29577951,  
 long2\_radians = merch\_long / 57.29577951,  
   
 # calculate distance in miles  
 distance\_miles = 3963.0 \* acos((sin(lat1\_radians) \* sin(lat2\_radians)) + cos(lat1\_radians) \* cos(lat2\_radians) \* cos(long2\_radians - long1\_radians)),  
  
 # calculate distance in km  
 distance\_km = 6377.830272 \* acos((sin(lat1\_radians) \* sin(lat2\_radians)) + cos(lat1\_radians) \* cos(lat2\_radians) \* cos(long2\_radians - long1\_radians))  
  
 ) %>%   
   
 # Filter only fraud cases  
 filter(is\_fraud == "1") %>%   
   
 # Plot boxplot  
 ggplot(aes(x= is\_fraud,y=distance\_km)) +  
 geom\_boxplot() +  
 ggtitle("Distance between card holder's home and transaction")  
  
card\_fraud



#Most cases occur between 50-100km from the victim. Fraud cases seem to correlate with distance.

Plot a boxplot or a violin plot that looks at the relationship of distance and is\_fraud. Does distance seem to be a useful feature in explaining fraud?

# Exploring sources of electricity production, CO2 emissions, and GDP per capita.

There are many sources of data on how countries generate their electricity and their CO2 emissions. I would like you to create three graphs:

## 1. A stacked area chart that shows how your own country generated its electricity since 2000.

You will use

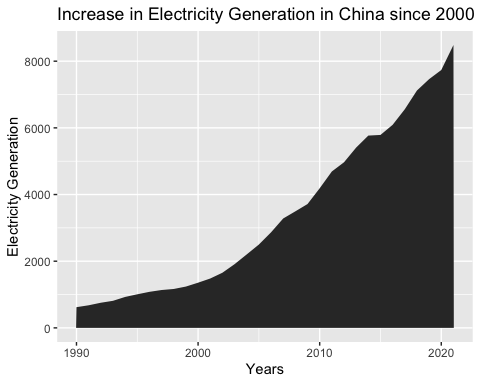
geom\_area(colour="grey90", alpha = 0.5, position = "fill")

## 2. A scatter plot that looks at how CO2 per capita and GDP per capita are related

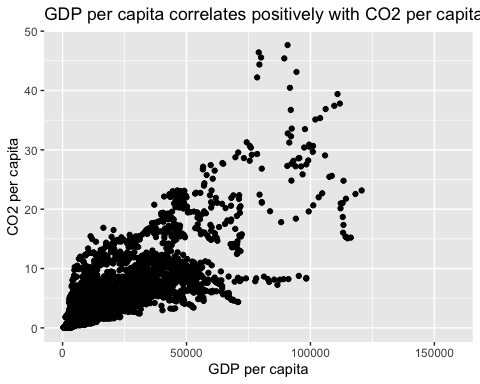
## 3. A scatter plot that looks at how electricity usage (kWh) per capita/day GDP per capita are related

We will get energy data from the Our World in Data website, and CO2 and GDP per capita emissions from the World Bank, using the wbstatspackage.

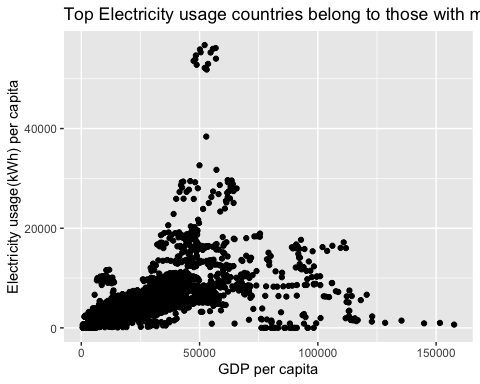
# Download electricity data  
url <- "https://nyc3.digitaloceanspaces.com/owid-public/data/energy/owid-energy-data.csv"  
  
energy <- read\_csv(url) %>%   
 filter(year >= 1990) %>%   
 drop\_na(iso\_code) %>%   
 select(1:3,  
 biofuel = biofuel\_electricity,  
 coal = coal\_electricity,  
 gas = gas\_electricity,  
 hydro = hydro\_electricity,  
 nuclear = nuclear\_electricity,  
 oil = oil\_electricity,  
 other\_renewable = other\_renewable\_exc\_biofuel\_electricity,  
 solar = solar\_electricity,  
 wind = wind\_electricity,   
 electricity\_demand,  
 electricity\_generation,  
 net\_elec\_imports, # Net electricity imports, measured in terawatt-hours  
 energy\_per\_capita, # Primary energy consumption per capita, measured in kilowatt-hours Calculated by Our World in Data based on BP Statistical Review of World Energy and EIA International Energy Data  
 energy\_per\_gdp, # Energy consumption per unit of GDP. This is measured in kilowatt-hours per 2011 international-$.  
 per\_capita\_electricity, # Electricity generation per capita, measured in kilowatt-hours  
 )   
  
# Download data for C02 emissions per capita https://data.worldbank.org/indicator/EN.ATM.CO2E.PC  
co2\_percap <- wb\_data(country = "countries\_only",   
 indicator = "EN.ATM.CO2E.PC",   
 start\_date = 1990,   
 end\_date = 2022,  
 return\_wide=FALSE) %>%   
 filter(!is.na(value)) %>%   
 #drop unwanted variables  
 select(-c(unit, obs\_status, footnote, last\_updated)) %>%   
 rename(year = date,  
 co2percap = value)  
  
  
# Download data for GDP per capita https://data.worldbank.org/indicator/NY.GDP.PCAP.PP.KD  
gdp\_percap <- wb\_data(country = "countries\_only",   
 indicator = "NY.GDP.PCAP.PP.KD",   
 start\_date = 1990,   
 end\_date = 2022,  
 return\_wide=FALSE) %>%   
 filter(!is.na(value)) %>%   
 #drop unwanted variables  
 select(-c(unit, obs\_status, footnote, last\_updated)) %>%   
 rename(year = date,  
 GDPpercap = value)  
  
#Q1: A stacked area chart that shows how your own country generated its electricity since 2000.  
energy %>%  
   
 # Filter country China that the info is not empty  
 filter(country == "China" & !is.na(electricity\_generation)) %>%   
   
 # Plot a stacked area chart  
 ggplot(aes(x=year,y=electricity\_generation)) +  
 geom\_area() +  
 labs(x="Years",y="Electricity Generation") +  
 ggtitle("Increase in Electricity Generation in China since 2000")



#Q2: A scatter plot that looks at how CO2 per capita and GDP per capita are related  
gdp\_percap %>%   
   
 # Merge CO2 and GDP data  
 left\_join(co2\_percap,by = c("iso3c","year")) %>%   
   
 # Plot a chart  
 ggplot(aes(x=GDPpercap,y=co2percap)) +   
 geom\_point() +   
 labs(x="GDP per capita",y="CO2 per capita") +  
 ggtitle("GDP per capita correlates positively with CO2 per capita")

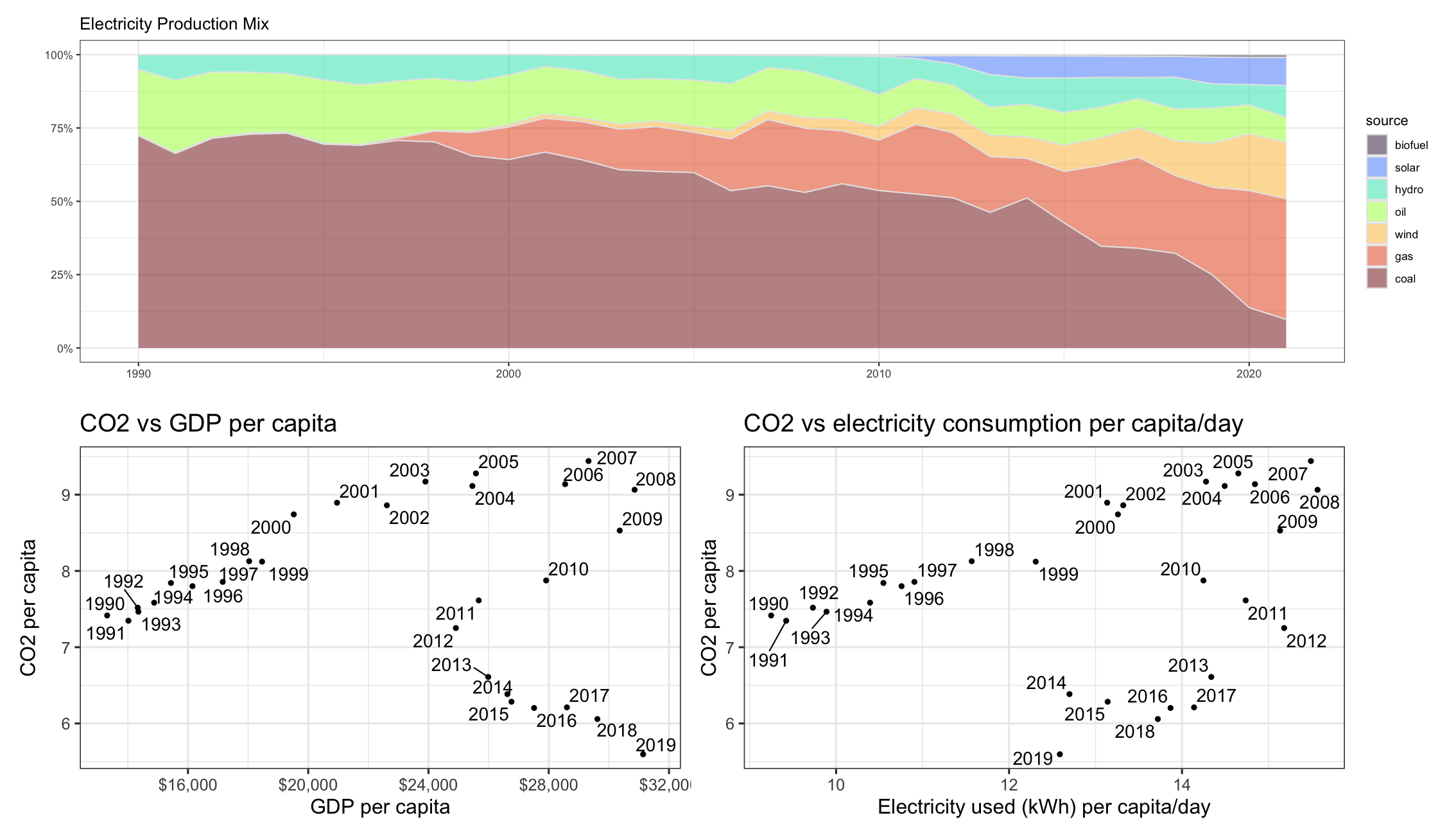


#GDP correlates with CO2 released  
  
#Q3: A scatter plot that looks at how electricity usage (kWh) per capita/day GDP per capita are related   
energy %>%   
   
 # Merge Energy and GDP data  
 mutate(iso3c = iso\_code) %>%   
 left\_join(gdp\_percap,by = c("iso3c","year")) %>%   
   
 # Plot a chart  
 ggplot(aes(x=GDPpercap,y=per\_capita\_electricity)) +   
 geom\_point() +   
 labs(x="GDP per capita",y="Electricity usage(kWh) per capita") +  
 ggtitle("Top Electricity usage countries belong to those with medium GDP per capita")



Specific questions:

1. How would you turn energy to long, tidy format?
2. You may need to join these data frames
   * Use left\_join from dplyr to [join the tables](http://r4ds.had.co.nz/relational-data.html)
   * To complete the merge, you need a unique *key* to match observations between the data frames. Country names may not be consistent among the three dataframes, so please use the 3-digit ISO code for each country
   * An aside: There is a great package called [countrycode](https://github.com/vincentarelbundock/countrycode) that helps solve the problem of inconsistent country names (Is it UK? United Kingdom? Great Britain?). countrycode() takes as an input a country’s name in a specific format and outputs it using whatever format you specify.
3. Write a function that takes as input any country’s name and returns all three graphs. You can use the patchwork package to arrange the three graphs as shown below



## # A tibble: 104,265 × 5  
## country year iso\_code Data Value  
## <chr> <dbl> <chr> <chr> <dbl>  
## 1 Afghanistan 1990 AFG biofuel NA  
## 2 Afghanistan 1990 AFG coal NA  
## 3 Afghanistan 1990 AFG gas NA  
## 4 Afghanistan 1990 AFG hydro NA  
## 5 Afghanistan 1990 AFG nuclear NA  
## 6 Afghanistan 1990 AFG oil NA  
## 7 Afghanistan 1990 AFG other\_renewable NA  
## 8 Afghanistan 1990 AFG solar NA  
## 9 Afghanistan 1990 AFG wind NA  
## 10 Afghanistan 1990 AFG electricity\_demand NA  
## # ℹ 104,255 more rows

# Deliverables

There is a lot of explanatory text, comments, etc. You do not need these, so delete them and produce a stand-alone document that you could share with someone. Knit the edited and completed R Markdown (qmd) file as a Word or HTML document (use the “Knit” button at the top of the script editor window) and upload it to Canvas. You must be comitting and pushing your changes to your own Github repo as you go along.

# Details

* Who did you collaborate with: GET
* Approximately how much time did you spend on this problem set: 4 Days
* What, if anything, gave you the most trouble: Plotting Graphs! Have to visualise from the data/codes

**Please seek out help when you need it,** and remember the [15-minute rule](https://dsb2023.netlify.app/syllabus/#the-15-minute-rule). You know enough R (and have enough examples of code from class and your readings) to be able to do this. If you get stuck, ask for help from others, post a question on Slack– and remember that I am here to help too!

As a true test to yourself, do you understand the code you submitted and are you able to explain it to someone else?

# Rubric

13/13: Problem set is 100% completed. Every question was attempted and answered, and most answers are correct. Code is well-documented (both self-documented and with additional comments as necessary). Used tidyverse, instead of base R. Graphs and tables are properly labelled. Analysis is clear and easy to follow, either because graphs are labeled clearly or you’ve written additional text to describe how you interpret the output. Multiple Github commits. Work is exceptional. I will not assign these often.

8/13: Problem set is 60–80% complete and most answers are correct. This is the expected level of performance. Solid effort. Hits all the elements. No clear mistakes. Easy to follow (both the code and the output). A few Github commits.

5/13: Problem set is less than 60% complete and/or most answers are incorrect. This indicates that you need to improve next time. I will hopefully not assign these often. Displays minimal effort. Doesn’t complete all components. Code is poorly written and not documented. Uses the same type of plot for each graph, or doesn’t use plots appropriate for the variables being analyzed. No Github commits.