Homework 2

Max Cheatle

5/21/23

## 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, …

## Explore the data

### Specific questions

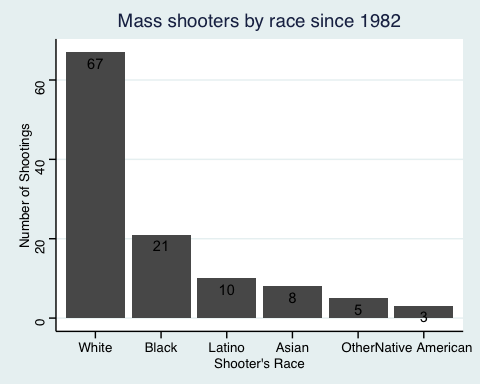
* Generate a data frame that summarizes the number of mass shootings per year.

mass\_shootings %>%   
   
 # Grouping the date by year  
 group\_by(year) %>%   
   
 # Counting the number of observations per year  
 summarise(no\_of\_shootings = n()) %>%   
 arrange(desc(no\_of\_shootings))

# A tibble: 37 × 2  
 year no\_of\_shootings  
 <dbl> <int>  
 1 2018 12  
 2 2017 11  
 3 2019 10  
 4 2012 7  
 5 2015 7  
 6 2016 6  
 7 2021 6  
 8 1999 5  
 9 2013 5  
10 1993 4  
# ℹ 27 more rows

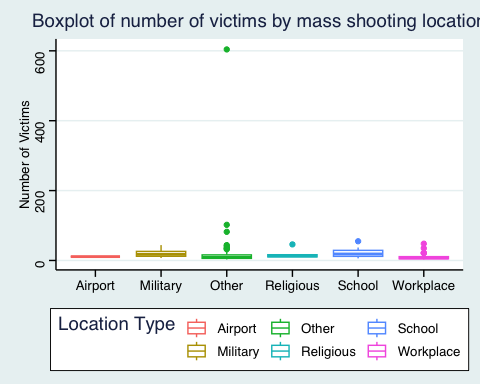
* Generate a bar chart that identifies the number of mass shooters associated with each race category. The bars should be sorted from highest to lowest and each bar should show its number.

mass\_shootings %>%   
   
 # Removing rows with no race value  
 filter(!is.na(race)) %>%   
   
 # Then following the same process as before   
 group\_by(race) %>%   
 summarise(count = n()) %>%   
 arrange(desc(count)) %>%   
   
 # Now, we plot this into bars  
 ggplot(aes(fct\_reorder(race, -count), count)) +   
 # fct\_reorder(race, -count) orders our race categories by count descending  
   
 geom\_bar(stat = "identity") +  
   
 # Let's also add text on each column  
 geom\_text(aes(label = count, vjust = 1.5)) +  
   
 # And finally aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Mass shooters by race since 1982", y="Number of Shootings", x="Shooter's Race") +  
 NULL



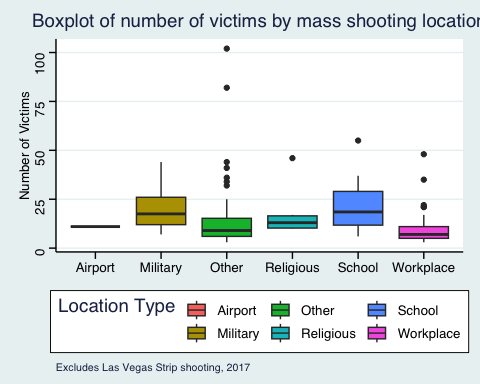
* Generate a boxplot visualizing the number of total victims, by type of location.

mass\_shootings %>%   
   
 # We want a boxplot, therefore we don't need to generate any calculations  
 ggplot(aes(x=location\_type, y=total\_victims, color = location\_type)) +  
 geom\_boxplot() +  
   
 # Now let's add some aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Boxplot of number of victims by mass shooting location", y = "Number of Victims", x = NULL, color = "Location Type") +  
 NULL



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

mass\_shootings %>%   
 # Now let's filter out the Las Vegas Strip shooting, the above plot wasn't very useful  
 filter(total\_victims < 600) %>%   
   
 # We want a boxplot, therefore we don't need to generate any calculations  
 ggplot(aes(x=location\_type, y=total\_victims, fill = location\_type)) +  
 geom\_boxplot() +  
   
 # Now let's add some aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Boxplot of number of victims by mass shooting location", caption = "Excludes Las Vegas Strip shooting, 2017", y = "Number of Victims", x = NULL, fill = "Location Type") +  
 NULL



### More open-ended questions

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

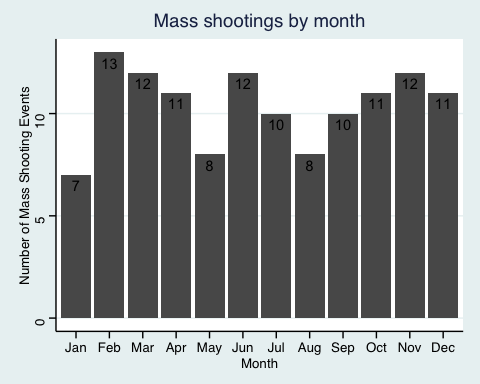
* How many white males with prior signs of mental illness initiated a mass shooting after 2000?

mass\_shootings %>%   
   
 # First we filter for only shootings committed by while males  
 filter(race == "White", male == "TRUE") %>%   
   
 # I don't filter out mental illness, since it'd be good to keep non-mental illness events for comparison. I group by prior\_mental\_illness instead  
 group\_by(prior\_mental\_illness) %>%   
   
 # Then summarise to count the number of events by prior illness category  
 summarise(n())

# A tibble: 3 × 2  
 prior\_mental\_illness `n()`  
 <chr> <int>  
1 No 9  
2 Yes 38  
3 <NA> 19

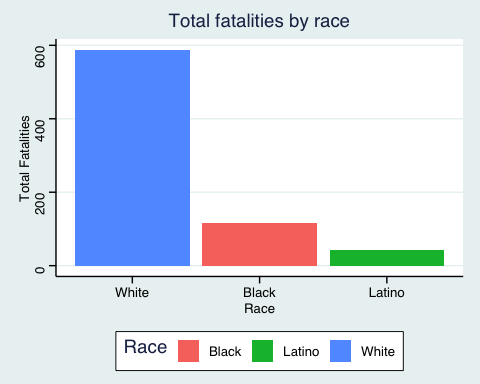
* Which month of the year has the most mass shootings? Generate a bar chart sorted in chronological (natural) order (Jan-Feb-Mar- etc) to provide evidence of your answer.
  + February has the highest number of mass shootings, with 13

mass\_shootings %>%   
   
 # First and foremost, group by month and count the number of shootings  
 group\_by(month) %>%   
 summarise(count = n()) %>%   
  
 # Then plot as bar chart  
 ggplot(aes(x=month, y=count)) +  
 geom\_bar(stat = "identity") +  
 geom\_text(aes(label = count, vjust = 1.5)) +  
   
 # Reordering our data by months, using month.abb since our data uses abbreviated month names  
 scale\_x\_discrete(limits = month.abb) +  
   
 # Finally, aesthetic modifications  
 ggthemes::theme\_stata() +  
 labs(title = "Mass shootings by month", x = "Month", y = "Number of Mass Shooting Events", fill = "Number of Victims")

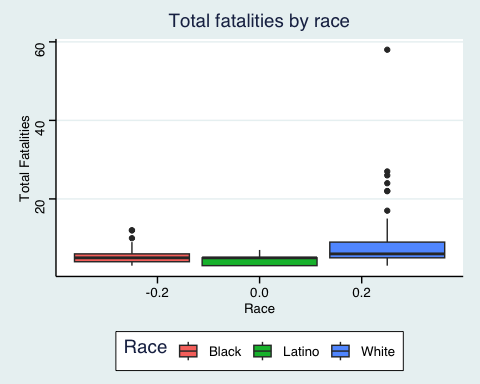


* How does the distribution of mass shooting fatalities differ between White and Black shooters? What about White and Latino shooters?
  + From the first plot, we can see that the total fatalities caused by White shooters is significantly higher than for Black and Latino shooters.
  + From the second plot, we also see that mass shootings by White shooters tend to lead to more fatalities per event, and that there are many occasions of White shooters causing more fatalities than maximum ever caused by Black or Latino shooters.

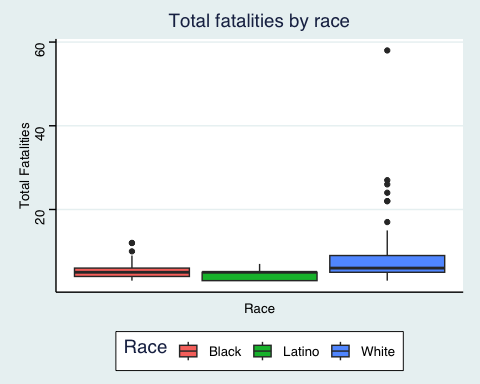
# Firstly, let's look at total fatalities in the database to gain an overview  
  
mass\_shootings %>%  
   
 # First grouping by race for counting  
 group\_by(race) %>%   
   
 # Filtering for the race groups of interest  
 filter(race == "White" | race == "Black" | race == "Latino") %>%   
 summarise(total\_fatalities = sum(fatalities)) %>%   
   
 # Now plotting to visualise the differences  
  
 ggplot(aes(x=fct\_reorder(race, -total\_fatalities), y=total\_fatalities, fill = race)) +   
 geom\_bar(stat = "identity") +  
   
 # Aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Total fatalities by race", x = "Race", y = "Total Fatalities", fill = "Race") +  
 NULL



# Now we do the same, but with a looking at the distribution of fatalities per mass shooting event, for each race of interest  
  
mass\_shootings %>%  
   
 # First grouping by race  
 group\_by(race) %>%   
   
 # Filtering for the race groups of interest  
 filter(race == "White" | race == "Black" | race == "Latino") %>%   
   
 # Now plotting to visualise the differences  
  
 ggplot(aes(y=fatalities, fill = race)) +   
 geom\_boxplot() +  
   
 # Aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Total fatalities by race", x = "Race", y = "Total Fatalities", fill = "Race") +  
 NULL



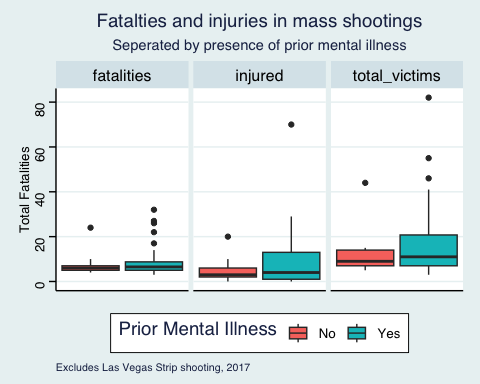
# Finally, let's repeat without the Las Vegas Shooting to create a clearer picture without the significant outlier  
  
mass\_shootings %>%  
   
 # First grouping by race  
 group\_by(race) %>%   
   
 # Filtering for the race groups of interest and to remove LV shooting  
 filter(race == "White" | race == "Black" | race == "Latino" & total\_victims < 600) %>%   
   
 # Now plotting to visualise the differences  
  
 ggplot(aes(y=fatalities, fill = race)) +   
 geom\_boxplot() +  
   
 # Aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Total fatalities by race", x = "Race", y = "Total Fatalities", fill = "Race") +  
   
 # Removing irrelevant x-axis ticks  
 theme(axis.text.x = element\_blank(), axis.ticks.x = element\_blank()) +  
 NULL



### Very open-ended

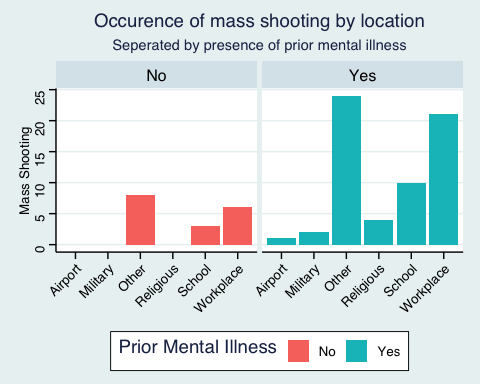
* Are mass shootings with shooters suffering from mental illness different from mass shootings with no signs of mental illness in the shooter?
  + Immediately, we see that the fatalities, injuries, and subsequently total victims in mass shooting perpetrated by those with prior mental illnesses all experience greater variability. That is, their interquartile range is wider.
  + Furthermore, the median fatality and injury count is larger when the shooter has suffered a prior mental illness. There are also much more significant outliers above Q3 in the presence of prior mental illness.
  + Regarding locations, mass shooters with prior mental illnesses are the only ones who commit said crimes in airports, military, and religious facilities.
  + Overall, it would appear that mass shootings by those with prior mental illnesses are more severe, and in a greater variety of locations (arguably more problematic locations, depending on perspective/measurement method).

# First, let's check if prior mental illnesses lead to differences in magnitude of total victims, fatalities, or injuries  
  
mass\_shootings %>%   
   
 # Grouping my mental illness for comparative analysis  
 group\_by(prior\_mental\_illness) %>%   
 select(prior\_mental\_illness, fatalities, injured, total\_victims) %>%   
   
 # Need to remove the Las Vegas shooting again for comparison without significant outlier  
 filter(total\_victims < 600) %>%   
   
 # Also removing NAs for removal of doubt and to reduce information overload  
 filter(!is.na(prior\_mental\_illness)) %>%   
   
 # We need to long the data for the visualistion I'd like to use  
 pivot\_longer(cols = c(fatalities, injured, total\_victims), values\_to = "value", names\_to = "variable") %>%   
   
 # Plotting in facet\_wrap for overview  
 ggplot(aes(y=value, fill=prior\_mental\_illness)) +  
 geom\_boxplot() +  
 facet\_wrap(~variable) +  
  
 # Aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Fatalties and injuries in mass shootings", subtitle = "Seperated by presence of prior mental illness", x = NULL, y = "Total Fatalities", fill = "Prior Mental Illness", caption = "Excludes Las Vegas Strip shooting, 2017") +  
   
 # Removing irrelevant x-axis labelling  
 theme(axis.text.x = element\_blank(), axis.ticks.x = element\_blank()) +  
 NULL



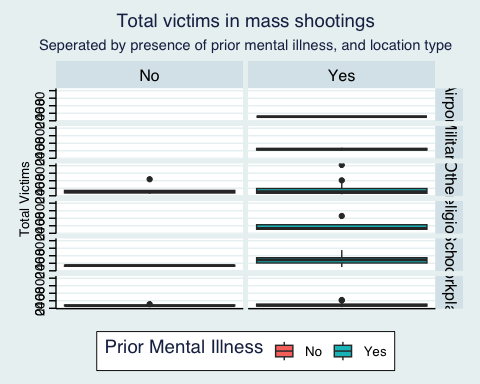
# Let's take a look from another angle. Do prior mental illnesses lead to different mass shooting locations?  
  
mass\_shootings %>%   
   
 # Grouping my mental illness for comparative analysis  
 group\_by(prior\_mental\_illness, location\_type) %>%  
   
 # Removing nulls  
 filter(!is.na(prior\_mental\_illness)) %>%   
   
 # Finding the % of occurences per location  
 summarise(count = n()) %>%   
   
 # Plotting for efficient comparison  
   
 ggplot(aes(y=count, x=location\_type, fill = prior\_mental\_illness)) +   
 geom\_bar(stat = "identity") +   
 facet\_wrap(~prior\_mental\_illness) +  
   
 # Aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Occurence of mass shooting by location", subtitle = "Seperated by presence of prior mental illness", x = NULL, y = "Mass Shooting", fill = "Prior Mental Illness") +  
   
 # Adjusting x axis for readability  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1)) +  
 NULL

`summarise()` has grouped output by 'prior\_mental\_illness'. You can override  
using the `.groups` argument.



* Assess the relationship between mental illness and total victims, mental illness and location type, and the intersection of all three variables.
  + As above, the three most significant differences here are that the only mass shootings in airports, military facilities, and religious venues, are perpetrated by those with prior mental illnesses.
  + Moreover, the variation of total fatalities is significantly wider in school mass shootings where the shooter has a prior mental illness, and slightly wider for the same case in workplace mass shootings.

# I used two of those examples in my above analysis (without looking ahead). So... I'll add here the 'intersection of all three variables' part.  
  
mass\_shootings %>%   
   
 # Grouping my mental illness for comparative analysis  
 group\_by(prior\_mental\_illness, location\_type) %>%  
   
 # Removing nulls and counting the total total\_victims in each illness & location pairing  
 filter(!is.na(prior\_mental\_illness)) %>%  
   
 # Now plotting, planning to use a facet\_grid here  
 ggplot(aes(y=total\_victims, fill = prior\_mental\_illness)) +  
 geom\_boxplot() +  
 facet\_grid(row=vars(location\_type), col=vars(prior\_mental\_illness)) +  
   
 # Aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Total victims in mass shootings", subtitle = "Seperated by presence of prior mental illness, and location type", x = NULL, y = "Total Victims", fill = "Prior Mental Illness") +  
   
 # Removing irrelevant x-axis labelling  
 theme(axis.text.x = element\_blank(), axis.ticks.x = element\_blank()) +  
 NULL



# Exploring credit card 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, …

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

card\_fraud %>%   
   
 # First grouping by year to get a year-by-year summary  
 group\_by(trans\_year) %>%   
   
 # Here, we count the total transactions, the number that were marked as fraudulent, and the subsequent fraud rate rounded to 2 decimals  
 summarise(total\_trans = n(), fraud\_trans = sum(is\_fraud == 1), pct\_fraud = round(fraud\_trans/total\_trans\*100,2))

# A tibble: 2 × 4  
 trans\_year total\_trans fraud\_trans pct\_fraud  
 <dbl> <int> <int> <dbl>  
1 2019 478646 2721 0.57  
2 2020 192382 1215 0.63

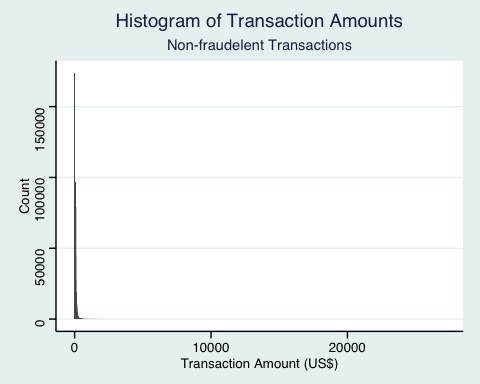
* How much money (in US$ terms) are fraudulent transactions costing the company? Generate a table that summarizes the total amount of legitimate and fraudulent transactions per year and calculate the % of fraudulent transactions, in US$ terms.

card\_fraud %>%   
   
 # First grouping by year to get a year-by-year summary  
 group\_by(trans\_year) %>%   
   
 # Now we sum the total amount of transactions, the amount of transactions where fraud is marked true, and the subsequent percentage rounded to 2 decimals  
 summarise(total\_amt = sum(amt), fraud\_amt = sum(ifelse(is\_fraud == 1, amt, 0)), pct\_fraud\_amt = round(fraud\_amt/total\_amt\*100,2))

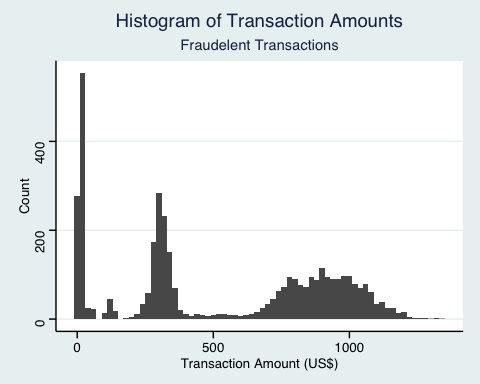
# A tibble: 2 × 4  
 trans\_year total\_amt fraud\_amt pct\_fraud\_amt  
 <dbl> <dbl> <dbl> <dbl>  
1 2019 33606041. 1423140. 4.23  
2 2020 13577863. 651949. 4.8

* 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 %>%   
 filter(is\_fraud == 0) %>% # Added in response to note below RE: comparing on same axes  
 ggplot(aes(x=amt)) +  
 geom\_histogram(binwidth = 20) + # Lower binwidths are harder to interpret, but higher binwidth is less telling information, so have taken a balance here  
   
 # facet\_wrap(~ is\_fraud) +   
   
 # Frequencies are so different, it's not possible to compare on same axis. Instead, let's make a separate histrogram for each is\_fraud value  
   
 # Aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Histogram of Transaction Amounts", subtitle = "Non-fraudelent Transactions", x= "Transaction Amount (US$)", y= "Count")



# Now for fraudulent transaction  
card\_fraud %>%   
 filter(is\_fraud == 1) %>% # Added in response to note below RE: comparing on same axes  
 ggplot(aes(x=amt)) +  
 geom\_histogram(binwidth = 20) + # Lower binwidth is possible here but kept the same for comparison to above plot  
   
 # facet\_wrap(~ is\_fraud) +   
   
 # Frequencies are so different, it's not possible to compare on same axis. Instead, let's make a separate histrogram for each is\_fraud value  
   
 # Aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Histogram of Transaction Amounts", subtitle = "Fraudelent Transactions", x= "Transaction Amount (US$)", y= "Count")

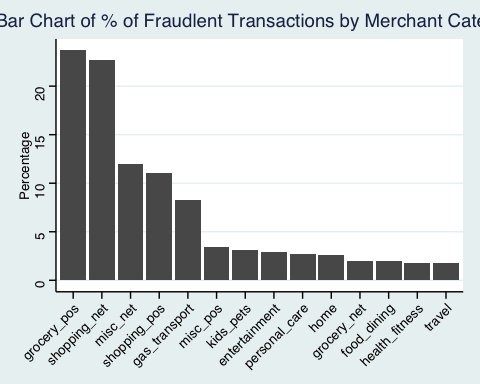


# Summary statistics  
card\_fraud %>%   
   
 # Changing fraud values 0,1 to more readable names  
 mutate(is\_fraud = case\_when(  
 is\_fraud == 1 ~ "Fraud",  
 is\_fraud == 0 ~ "Legitimate"  
 )) %>%   
 group\_by(is\_fraud) %>%   
   
 # Calculating key summary statistics; mean, median, min/max, standard deviation, q25, q75  
 summarise(mean = mean(amt), median = median(amt), min = min(amt), max = max(amt), sd = sd(amt), q25 = quantile(amt, 0.25), q75 = quantile(amt, 0.75))

# A tibble: 2 × 8  
 is\_fraud mean median min max sd q25 q75  
 <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
1 Fraud 527. 369. 1.06 1334. 391. 240. 901.   
2 Legitimate 67.6 47.2 1 27120. 155. 9.6 82.4

* 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.
  + groceries\_pos and shopping\_net are by far the most frequent categories subject to fraudulent spending, combining for more than 40% of fraudulent transactions
  + misc\_net, shopping\_pos, and gas\_transport are all significant categories too, around the 10% mark respectively
  + The rest of the categories all perform very similarly, less than 5%

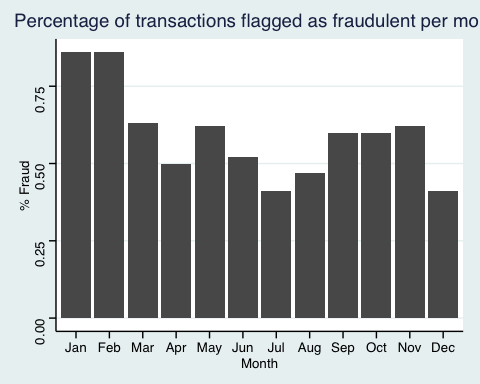
# First I want the total number of transcations stored as a value, to make life easier for our ggplot code  
total\_fraud\_trans <- card\_fraud %>%   
 filter(is\_fraud == 1) %>%  
 summarise(total\_fraud\_trans = n()) %>%   
   
 # Use pull() to retreive the number only (not as a tibble)  
 pull(total\_fraud\_trans)  
  
card\_fraud %>%   
 filter(is\_fraud == 1) %>%   
 group\_by(category) %>%   
   
 # Now we count the fraudulent transaction per category, and divide by the total value that we saved previously  
 summarise(fraud\_trans = n(), pct = round(fraud\_trans/total\_fraud\_trans\*100,2)) %>%   
   
 # Finally, plotting to bar chart  
 ggplot(aes(y=pct, x=reorder(category, -pct))) +  
 geom\_bar(stat = "identity") +  
  
 # Aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Bar Chart of % of Fraudlent Transactions by Merchant Category", x = NULL, y = "Percentage", fill = NULL) +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))



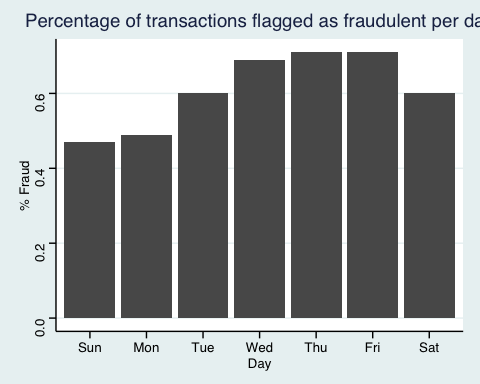
* When is fraud more prevalent? Which days, months, hours? To create new variables to help you in your analysis, we use the lubridate package and the following code
  + Fraud as a percentage of total transactions:
    - January and February are the most common months for fraud, with over 0.75% of transactions being fraudulent
      * In terms of seasonality, it would appear that fraud is more frequent in the winter months, excluding December which is the lowest month for fraud all year
      * Perhaps something to explore is the raw number of fraudulent transactions, since the number of total transactions may be higher in December, therefore lowering the percentage of fraud even if fraud is just as prevalent
    - Fraud is most common at the end of the working week (Wed, Thur, Fri), and lowest over the weekends
      * Similar to above, perhaps the total number of transactions is highest on weekends, so it may be useful to check the raw number of fraudulent transactions
    - By a significant margin, fraud is most common between 10pm and 3am (overnight)
  + Fraud measured by number of instances of fraudulent transactions:
    - January to June are now dominant for fraudulent transactions, as opposed to only January and February previously
      * December also sees a higher number of fraudulent transactions, suggesting that the low % was driven by a large total number of transactions
    - In terms of days, our second plot shows that fraud is more likely on the weekends, and Monday - the trend here has reversed
    - Nil change for hours of the day when looking at instances rather than percentage

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)  
 )

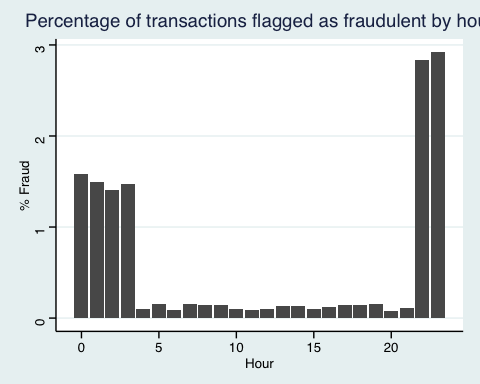
# Let's find out which days, months, and hours experience the most prevelant fraud  
  
card\_fraud\_times <- card\_fraud %>%   
 mutate(  
 date\_only = lubridate::date(trans\_date\_trans\_time),  
 month\_name = lubridate::month(trans\_date\_trans\_time, label=TRUE),  
 hour = lubridate::hour(trans\_date\_trans\_time),  
 weekday = lubridate::wday(trans\_date\_trans\_time, label = TRUE)  
 ) %>%   
   
 # Let's reduce our selection to only the columns of interest for clarity  
 select(date\_only, month\_name, hour, weekday, is\_fraud)   
  
# Now let's group by months, and find out which are the worst for fraud  
  
card\_fraud\_times %>%   
 group\_by(month\_name) %>%   
   
 # Here I summarise to find the number of transactions in each month, the number of fraudulent transactions, and the subsequent percentage  
 summarise(total\_trans = n(), fraud\_trans = sum(is\_fraud == 1), pct\_fraud = round(fraud\_trans/total\_trans\*100,2)) %>%   
   
 # Now I'm going to plot for clarity, it will also help see if there is any seasonality  
 ggplot(aes(x = month\_name, y = pct\_fraud)) +  
 geom\_bar(stat = "identity") +  
   
 # Now aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Percentage of transactions flagged as fraudulent per month", y = "% Fraud", x = "Month")



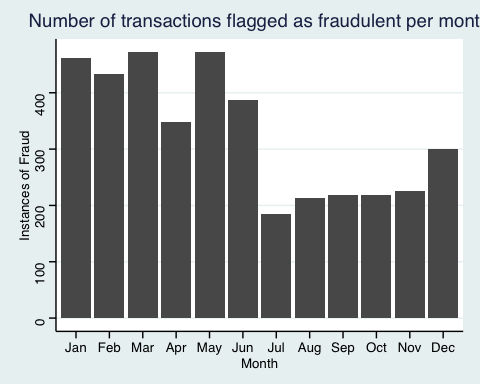
# Repeating for days instead of months  
   
card\_fraud\_times %>%   
 group\_by(weekday) %>%   
 summarise(total\_trans = n(), fraud\_trans = sum(is\_fraud == 1), pct\_fraud = round(fraud\_trans/total\_trans\*100,2)) %>%   
 ggplot() +  
 geom\_bar(aes(x = weekday, y = pct\_fraud), stat = "identity") +  
 ggthemes::theme\_stata() +  
 labs(title = "Percentage of transactions flagged as fraudulent per day", y = "% Fraud", x = "Day")



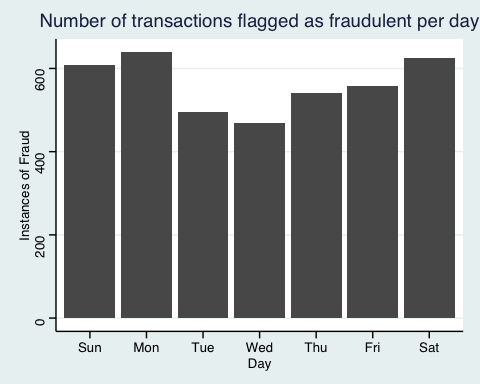
# Repeating for hours instead of days  
  
card\_fraud\_times %>%   
 group\_by(hour) %>%   
 summarise(total\_trans = n(), fraud\_trans = sum(is\_fraud == 1), pct\_fraud = round(fraud\_trans/total\_trans\*100,2)) %>%   
 ggplot(aes(x = hour, y = pct\_fraud)) +  
 geom\_bar(stat = "identity") +  
 ggthemes::theme\_stata() +  
 labs(title = "Percentage of transactions flagged as fraudulent by hour", y = "% Fraud", x = "Hour")



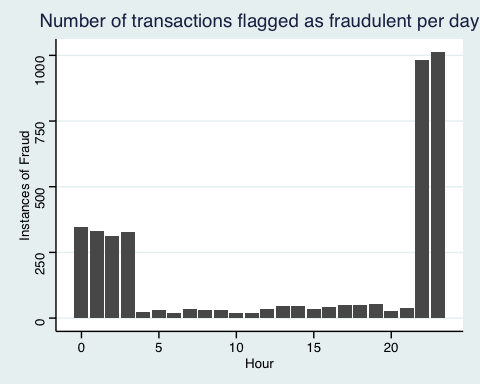
# As discussed, lets also look at raw fraudulent transactions data, just in case our percentages were biased by the total number of transactions (denominator), rather than any change to fraud behaviour  
  
# First by month, using the same code but with ggplot changes  
  
card\_fraud\_times %>%   
 group\_by(month\_name) %>%   
 summarise(total\_trans = n(), fraud\_trans = sum(is\_fraud == 1), pct\_fraud = round(fraud\_trans/total\_trans\*100,2)) %>%   
   
 # This is the only line of code I have altered from above  
 ggplot(aes(x = month\_name, y = fraud\_trans)) +  
 geom\_bar(stat = "identity") +  
 ggthemes::theme\_stata() +  
 labs(title = "Number of transactions flagged as fraudulent per month", y = "Instances of Fraud", x = "Month")



# Repeating for days instead of months  
   
card\_fraud\_times %>%   
 group\_by(weekday) %>%   
 summarise(total\_trans = n(), fraud\_trans = sum(is\_fraud == 1), pct\_fraud = round(fraud\_trans/total\_trans\*100,2)) %>%   
   
 # This is the only line of code I have altered from above  
 ggplot(aes(x = weekday, y = fraud\_trans)) +  
 geom\_bar(stat = "identity") +  
 ggthemes::theme\_stata() +  
 labs(title = "Number of transactions flagged as fraudulent per day", y = "Instances of Fraud", x = "Day")



# Repeating for hours instead of days  
  
card\_fraud\_times %>%   
 group\_by(hour) %>%   
 summarise(total\_trans = n(), fraud\_trans = sum(is\_fraud == 1), pct\_fraud = round(fraud\_trans/total\_trans\*100,2)) %>%   
   
 # This is the only line of code I have altered from above  
 ggplot(aes(x = hour, y = fraud\_trans)) +  
 geom\_bar(stat = "identity") +  
 ggthemes::theme\_stata() +  
 labs(title = "Number of transactions flagged as fraudulent per day", y = "Instances of Fraud", x = "Hour")



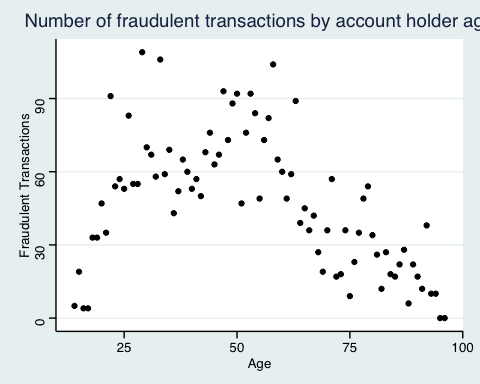
* Are older customers significantly more likely to be victims of credit card fraud? To calculate a customer’s age, we use the lubridate package and the following code
  + When looking at the total number of fraudulent transactions by age, it would appear that older customers are not more likely to be victims
  + However, when looking in percentage terms, older customers are more likely to be victims of fraud
    - But, not all older customers are victims of fraud. Rather, some older customers are more likely to be victims of fraud more often than those who are younger
    - The upwards trend in fraud susceptibility starts around age 50

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

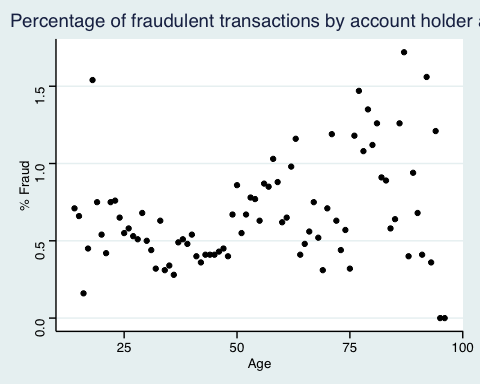
# First let me check the output of this age code, as I'm unfamiliar  
  
card\_fraud %>%   
 mutate(  
 age = interval(dob, trans\_date\_trans\_time) / years(1),  
 ) %>%   
 select(age)

# A tibble: 671,028 × 1  
 age  
 <dbl>  
 1 59.3  
 2 72.9  
 3 34.7  
 4 28.1  
 5 33.9  
 6 44.3  
 7 19.8  
 8 61.0  
 9 37.5  
10 22.3  
# ℹ 671,018 more rows

# Looks like the age generated is super specific. I want to generalise a bit, so I am going to round to the nearest whole number  
  
card\_fraud %>%   
 mutate(  
   
 # Rounding to 0 decimals, as discussed  
 age = round(interval(dob, trans\_date\_trans\_time) / years(1), 0)  
 ) %>%   
   
 # Grouping by age and calculating variables of interest  
 group\_by(age) %>%   
 summarise(total\_trans = n(), fraud\_trans = sum(is\_fraud == 1), pct\_fraud = round(fraud\_trans/total\_trans\*100,2)) %>%   
   
 # Let's look at the number of frauds first  
 ggplot(aes(x=age,y=fraud\_trans)) +  
 geom\_point() +  
  
 # And keep our aesthetic consistency  
 ggthemes::theme\_stata() +  
 labs(title = "Number of fraudulent transactions by account holder age", x = "Age", y = "Fraudulent Transactions")

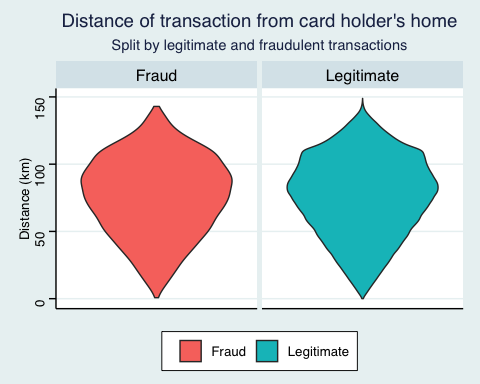


# Let's repeat for percentage of transactions being fraudulent  
  
card\_fraud %>%   
 mutate(  
 age = round(interval(dob, trans\_date\_trans\_time) / years(1), 0)  
 ) %>%   
 group\_by(age) %>%   
 summarise(total\_trans = n(), fraud\_trans = sum(is\_fraud == 1), pct\_fraud = round(fraud\_trans/total\_trans\*100,2)) %>%   
   
 # Here, we change our y value  
 ggplot(aes(x=age,y=pct\_fraud)) +  
 geom\_point() +  
   
 # Aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Percentage of fraudulent transactions by account holder age", x = "Age", y = "% Fraud")



* 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/  
  
  
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))  
  
 )  
  
  
fraud %>%   
   
 # Changing fraud values 0,1 to more readable names  
 mutate(is\_fraud = case\_when(  
 is\_fraud == 1 ~ "Fraud",  
 is\_fraud == 0 ~ "Legitimate"  
 )) %>%   
   
 # Now plotting a violin plot, faceted by legitimate and fraudulent transactions  
 ggplot(aes(x = is\_fraud, y = distance\_km, fill = is\_fraud)) +  
 geom\_violin() +  
   
 # Faceting with scales = "free\_x" to remove redundant x\_axis space  
 facet\_wrap(~ is\_fraud, scales = "free\_x") +  
   
 # Aesthetic modifications  
 ggthemes::theme\_stata() +  
 labs(title = "Distance of transaction from card holder's home", subtitle = "Split by legitimate and fraudulent transactions", x = NULL, y = "Distance (km)", fill = NULL) +  
 theme(axis.text.x = element\_blank(), axis.ticks.x = element\_blank())



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?

* Distance seemingly has almost no effect on fraudulent transactions, as the below violin plot shows the distribution of transaction distances are nearly identical for the respective types of activity

# 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:

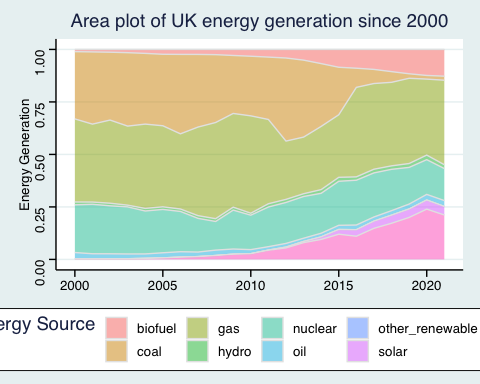
# Download electricity data  
url <- "https://nyc3.digitaloceanspaces.com/owid-public/data/energy/owid-energy-data.csv"  
  
energy <- read\_csv(url) %>%   
 filter(year >= 1990) %>%   
 drop\_na(iso\_code) %>%   
 select(1:3,  
 biofuel = biofuel\_electricity,  
 coal = coal\_electricity,  
 gas = gas\_electricity,  
 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)  
  
view(energy)  
view(co2\_percap)  
view(gdp\_percap)

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

uk\_energy <- energy %>%   
   
 # First let's filter for my country, the UK  
 filter(country == "United Kingdom" & year >= 2000) %>%   
   
 # Now pivoting the data such that all the energy sources are in one column, as are their respective generated amounts  
 pivot\_longer(cols = biofuel:wind, values\_to = "energy\_generated", names\_to = "energy\_source") %>%   
   
 # Selecting only relevant values for visual check of accuracy  
 select(country, year, energy\_source, energy\_generated)  
  
glimpse(uk\_energy) # Looks good

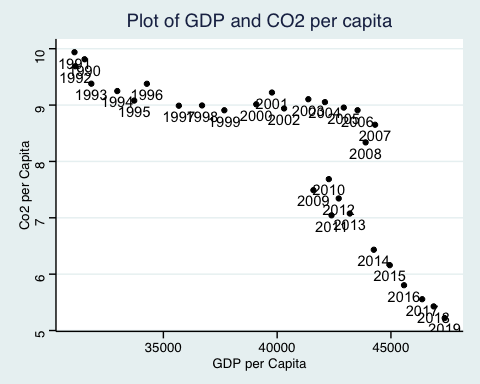
Rows: 198  
Columns: 4  
$ country <chr> "United Kingdom", "United Kingdom", "United Kingdom",…  
$ year <dbl> 2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000,…  
$ energy\_source <chr> "biofuel", "coal", "gas", "hydro", "nuclear", "oil", …  
$ energy\_generated <dbl> 3.94, 119.95, 148.08, 5.09, 85.06, 11.31, 0.00, 0.00,…

# Now let's plot it  
  
uk\_energy %>%   
 ggplot(aes(x = year,y = energy\_generated, fill = energy\_source)) +  
 geom\_area(colour="grey90", alpha = 0.5, position = "fill") +  
   
 # Keeping our aesthetics consistent from before  
 ggthemes::theme\_stata() +  
 labs(title = "Area plot of UK energy generation since 2000", x = NULL, y = "Energy Generation", fill = "Energy Source")



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

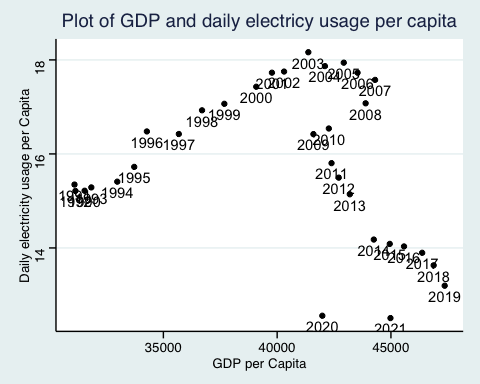
# First, lets join the relevant tables and check it worked as expected  
  
co2\_gdp <- left\_join(co2\_percap, gdp\_percap, by = c("iso3c", "year")) %>%   
 select(iso3c, year, co2percap, GDPpercap)  
  
view(co2\_gdp) # Looks good  
  
# Now lets plot a scatter plot  
co2\_gdp %>%   
 filter(iso3c == "GBR") %>%   
 ggplot(aes(y = co2percap, x = GDPpercap)) +  
 geom\_point() +  
 geom\_text(aes(label = year, hjust = 0.5, vjust = 1.5)) +  
   
 # Aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Plot of GDP and CO2 per capita", x = "GDP per Capita", y = "Co2 per Capita")



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

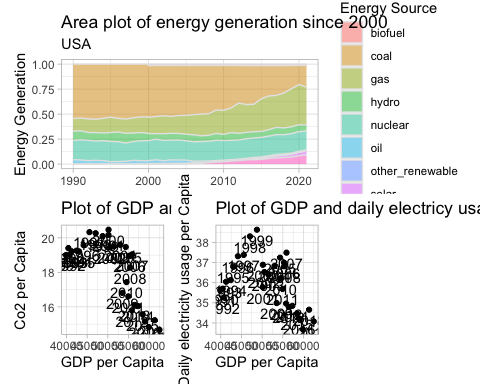
We will get energy data from the Our World in Data website, and CO2 and GDP per capita emissions from the World Bank, using the wbstatspackage.

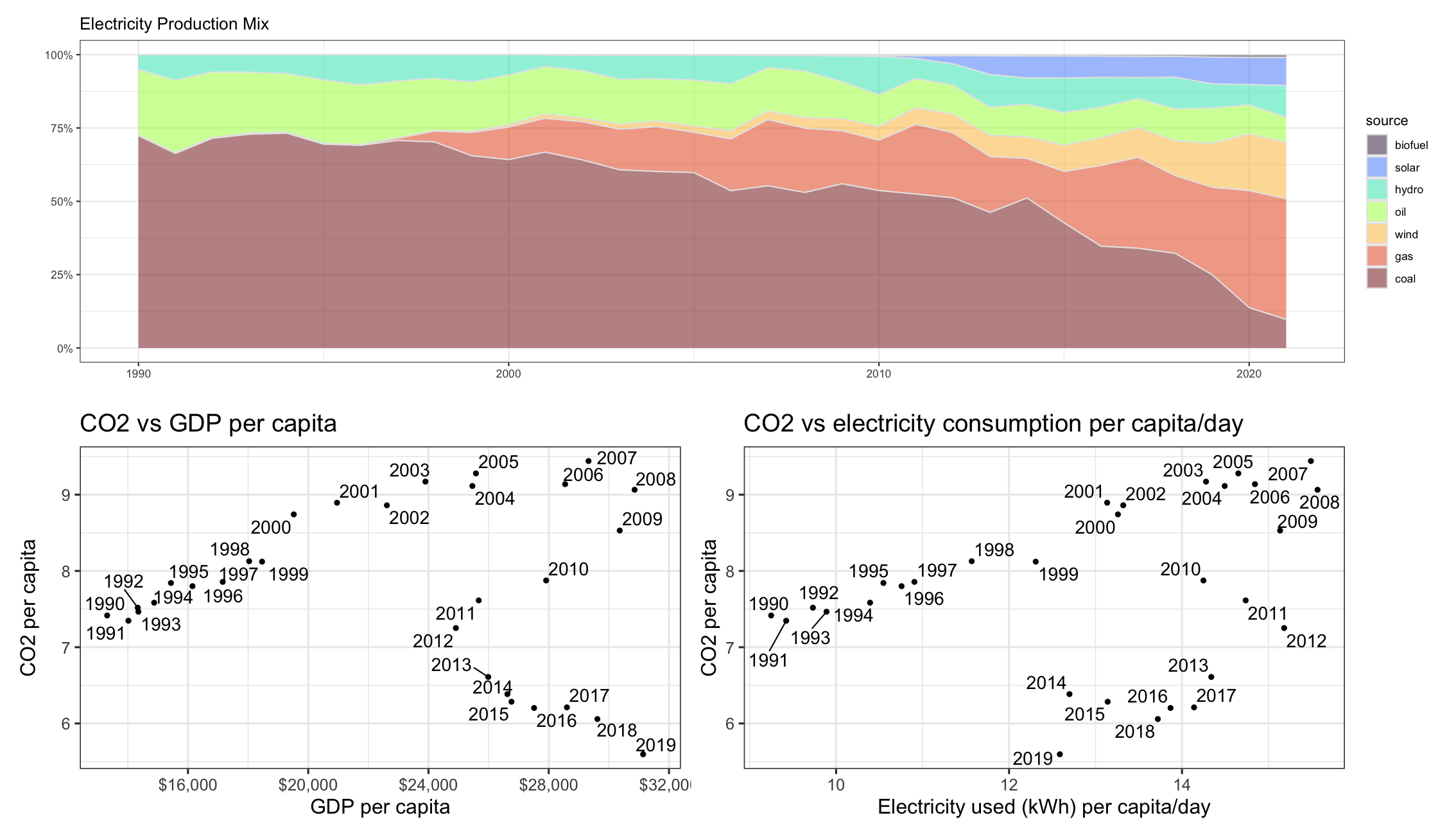
# I'm a bit confused by this title, and the plot shown below. I have decided to follow the title as 'electricity usage per capita per day, and gdp per capita', which is slightly different to the plot shown below.  
  
# We need to mutate the energy table such that the iso\_code column is called iso3c, the same as in our gdp\_percap table  
  
elec\_gdp <- energy %>%   
 mutate(iso3c = iso\_code) %>%   
  
# Now lets join the relevant tables and check it worked as expected  
  
left\_join(gdp\_percap, energy, by = c("iso3c", "year")) %>%  
 select(iso3c, year, GDPpercap, per\_capita\_electricity)  
  
# Im going to manipulate the per\_capita\_electricity table first, into per day form  
  
elec\_gdp %>%   
   
 # Divide by 365 to get daily usage, rather than yearly  
 mutate(per\_cap\_day\_electricity = per\_capita\_electricity/365) %>%   
 filter(iso3c == "GBR") %>%   
  
 # Now plotting  
 ggplot(aes(x = GDPpercap, y = per\_cap\_day\_electricity)) +  
 geom\_point() +  
 geom\_text(aes(label = year, hjust = 0.5, vjust = 1.5)) +  
   
 # Aesthetics  
 ggthemes::theme\_stata() +  
 labs(title = "Plot of GDP and daily electricy usage per capita", x = "GDP per Capita", y = "Daily electricity usage per Capita")



Specific questions:

1. How would you turn energy to long, tidy format?
   * As shown, I used pivot\_longer(cols = biofuel:wind, values\_to = “energy\_generated”, names\_to = “energy\_source”) to convert energy into long, tidy format
     + This essentially transforms multiple columns into one, hence ‘long’ data, and pairs them with their subsequent values in the adjacent column
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 Kingdon? 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

* # First making the iso\_code column name consistent for joining  
    
  energy\_iso <- energy %>%   
   rename(iso3c = iso\_code)  
    
  # Now joining the two per capita tables  
  percap\_data <- left\_join(co2\_percap, gdp\_percap, by = c("iso3c", "year")) %>%   
   select(country.x, iso3c, year, co2percap, GDPpercap) %>%   
   rename(country = country.x)  
    
  # Then, I need to add the per capita electricity usage as we did in Part 3, then save for use in the function  
    
  percap\_plot\_data <- left\_join(percap\_data, energy\_iso, by = c("iso3c", "year")) %>%   
   select(country.x, iso3c, year, co2percap, GDPpercap, per\_capita\_electricity) %>%   
   rename(country = country.x)  
    
    
  # Next, I'm tidying the energy table, using the same code as in Part 1, and saving for using in the function  
  energy\_tidy <- energy\_iso %>%   
   pivot\_longer(cols = biofuel:wind, values\_to = "energy\_generated", names\_to = "energy\_source") %>%   
   select(country, iso3c, year, energy\_source, energy\_generated)   
    
    
    
  # Now we can begin creating our function  
  country\_plots <- function(iso\_code) {  
    
   plot\_1 <- energy\_tidy %>%   
   filter(iso3c == iso\_code) %>%   
    
   # Plotting  
   ggplot(aes(x = year,y = energy\_generated, fill = energy\_source)) +  
   geom\_area(colour="grey90", alpha = 0.5, position = "fill") +  
    
   # Keeping our aesthetics consistent from before  
   theme\_light() +  
   labs(title = "Area plot of energy generation since 2000", subtitle = iso\_code, x = NULL, y = "Energy Generation", fill = "Energy Source")  
    
   plot\_2 <- percap\_plot\_data %>%   
   filter(iso3c == iso\_code) %>%   
   ggplot(aes(y = co2percap, x = GDPpercap)) +  
   geom\_point() +  
   geom\_text(aes(label = year, hjust = 0.5, vjust = 1.5)) +  
    
   # Aesthetics  
   theme\_light() +  
   labs(title = "Plot of GDP and CO2 per capita", x = "GDP per Capita", y = "Co2 per Capita")  
    
   plot\_3 <- percap\_plot\_data %>%   
    
   # Divide by 365 to get daily usage, rather than yearly  
   mutate(per\_cap\_day\_electricity = per\_capita\_electricity/365) %>%   
   filter(iso3c == iso\_code) %>%   
    
   # Now plotting  
   ggplot(aes(x = GDPpercap, y = per\_cap\_day\_electricity)) +  
   geom\_point() +  
   geom\_text(aes(label = year, hjust = 0.5, vjust = 1.5)) +  
    
   # Aesthetics  
   theme\_light() +  
   labs(title = "Plot of GDP and daily electricy usage per capita", x = "GDP per Capita", y = "Daily electricity usage per Capita")  
    
   plot\_1 / (plot\_2 + plot\_3)  
    
   }  
    
  country\_plots("USA")
* Warning: Removed 20 rows containing non-finite values (`stat\_align()`).
* 



# Details

* Who did you collaborate with: NA
* Approximately how much time did you spend on this problem set: 6 hrs
* What, if anything, gave you the most trouble: Getting the right tables for the function without creating a mess of manipulation within the function

**Please seek out help when you need it,** and remember the [15-minute rule](https://mam2022.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?