COVID 19 Analysis

Nicholas Cianci

2024-08-22

COVID 19 Final Project

This is an exploratory data analysis project that dives into the John Hopkins COVID-19 data set that includes Global and US case and death data which is publicly available on GitHub.

To begin, I want to address some potential biases. COVID-19 data was a very intense time for the world, myself included. Each person has their own opinion how it was handled, what was effective vs. what was not, and what we could have done differently. It would be relatively easy to manipulate this data to support a pre-existing notion or agenda. It is important to understand and identify this potential upfront, so that I can proceed with every intention of remaining unbias in my exploration of this COVID-19 data set.

Load Packages and Data

To begin, I installed and loaded two packages - tidyverse and lubridate to help with my analysis throughout the project. I then used the github links to read in the several csv files containing our COVID-19 data.

```
#Load the tidyverse library package for use
library("tidyverse")
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library("lubridate")

#Bring in the data from the github repository
url_in <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covi
file_names <- c("time_series_covid19_confirmed_global.csv", "time_series_covid19_deaths_global.csv", "ti

urls<- str_c(url_in,file_names)

#Load each file into it's own object
global_cases <- read_csv(urls[1])
```

```
## Rows: 289 Columns: 1147
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (2): Province/State, Country/Region
## dbl (1145): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
global_deaths <- read_csv(urls[2])
```

```
## Rows: 289 Columns: 1147
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (2): Province/State, Country/Region
## dbl (1145): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
US_cases <- read_csv(urls[3])
```

```
## Rows: 3342 Columns: 1154
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (6): iso2, iso3, Admin2, Province_State, Country_Region, Combined_Key
## dbl (1148): UID, code3, FIPS, Lat, Long_, 1/22/20, 1/23/20, 1/24/20, 1/25/20...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
US_deaths <- read_csv(urls[4])
```

```
## Rows: 3342 Columns: 1155
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (6): iso2, iso3, Admin2, Province_State, Country_Region, Combined_Key
## dbl (1149): UID, code3, FIPS, Lat, Long_, Population, 1/22/20, 1/23/20, 1/24...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Tidy Data Global Data

The Global data was split up between case data, death data, and Population data. I decided to standardize and join them to give us one all-encompassing global data set. I accomplished this through the use of pivots and joins.

```
# Tidy Global Cases data
global_cases <- global_cases %>%
  pivot_longer(cols = -c('Province/State',
```

```
                        'Country/Region', Lat, Long),
               names_to = "date",
               values_to = "cases") %>%
  select(-c(Lat, Long))

# Tidy Global Death data
global_deaths <- global_deaths %>%
  pivot_longer(cols = -c('Province/State',
                         'Country/Region', Lat, Long),
               names_to = "date",
               values_to = "deaths") %>%
  select(-c(Lat,Long))

# Join Global cases and death data into one global data set
global <- global_cases %>%
  full_join(global_deaths) %>%
  rename(Country_Region = 'Country/Region',
         Province_State = 'Province/State') %>%
  mutate(date = mdy(date))
```

```
## Joining with `by = join_by(`Province/State`, `Country/Region`, date)`
```

```
# Review summary of joined data set
summary(global)
```

```
##  Province_State     Country_Region          date                cases
##  Length:330327      Length:330327      Min.   :2020-01-22   Min.   :        0
##  Class :character   Class :character   1st Qu.:2020-11-02   1st Qu.:      680
##  Mode  :character   Mode  :character   Median :2021-08-15   Median :    14429
##                                        Mean   :2021-08-15   Mean   :   959384
##                                        3rd Qu.:2022-05-28   3rd Qu.:   228517
##                                        Max.   :2023-03-09   Max.   :103802702
##      deaths
##  Min.   :      0
##  1st Qu.:      3
##  Median :    150
##  Mean   :  13380
##  3rd Qu.:   3032
##  Max.   :1123836
```

```
#Standardize global data set with US data set
global <- global %>%
  unite("Combined_Key", c(Province_State, Country_Region),
        sep = ", ",
        na.rm = TRUE,
        remove = FALSE)

#Read in additional file that has global population data
uid_lookup_url <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/U
uid <- read_csv(uid_lookup_url) %>%
  select(-c(Lat,Long_,Combined_Key,code3,iso2,iso3, Admin2))
```

```
## Rows: 4321 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): iso2, iso3, FIPS, Admin2, Province_State, Country_Region, Combined_Key
## dbl (5): UID, code3, Lat, Long_, Population
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
#Join the global data set with new data that contains the global populations
global<- global %>%
  left_join(uid, by = c("Province_State", "Country_Region")) %>%
  select(-c(UID, FIPS)) %>%
  select(Province_State, Country_Region, date, cases, deaths,
         Population, Combined_Key)

#View sample of newly joined global data set
tail(global)
```

```
## # A tibble: 6 x 7
##   Province_State Country_Region date        cases deaths Population Combined_Key
##   <chr>          <chr>          <date>      <dbl>  <dbl>      <dbl> <chr>
## 1 <NA>           Zimbabwe       2023-03-04 264127   5668   14862927 Zimbabwe
## 2 <NA>           Zimbabwe       2023-03-05 264127   5668   14862927 Zimbabwe
## 3 <NA>           Zimbabwe       2023-03-06 264127   5668   14862927 Zimbabwe
## 4 <NA>           Zimbabwe       2023-03-07 264127   5668   14862927 Zimbabwe
## 5 <NA>           Zimbabwe       2023-03-08 264276   5671   14862927 Zimbabwe
## 6 <NA>           Zimbabwe       2023-03-09 264276   5671   14862927 Zimbabwe
```

Analyze and Visualize Global Data

In this step, I analyzed and visualized the Global data set in a variety of ways. To begin, I grouped the data so that there was 1 row for each Country/Region that included the total number of cases, deaths, and Country/Region Population along with two new fields which calculate the number of cases and deaths per 1 million people. These calculated fields will help us to see who had the highest rates of cases and deaths. If I only looked at the total number of cases and deaths, I would most likely only see the Countries/Regions with the largest populations.

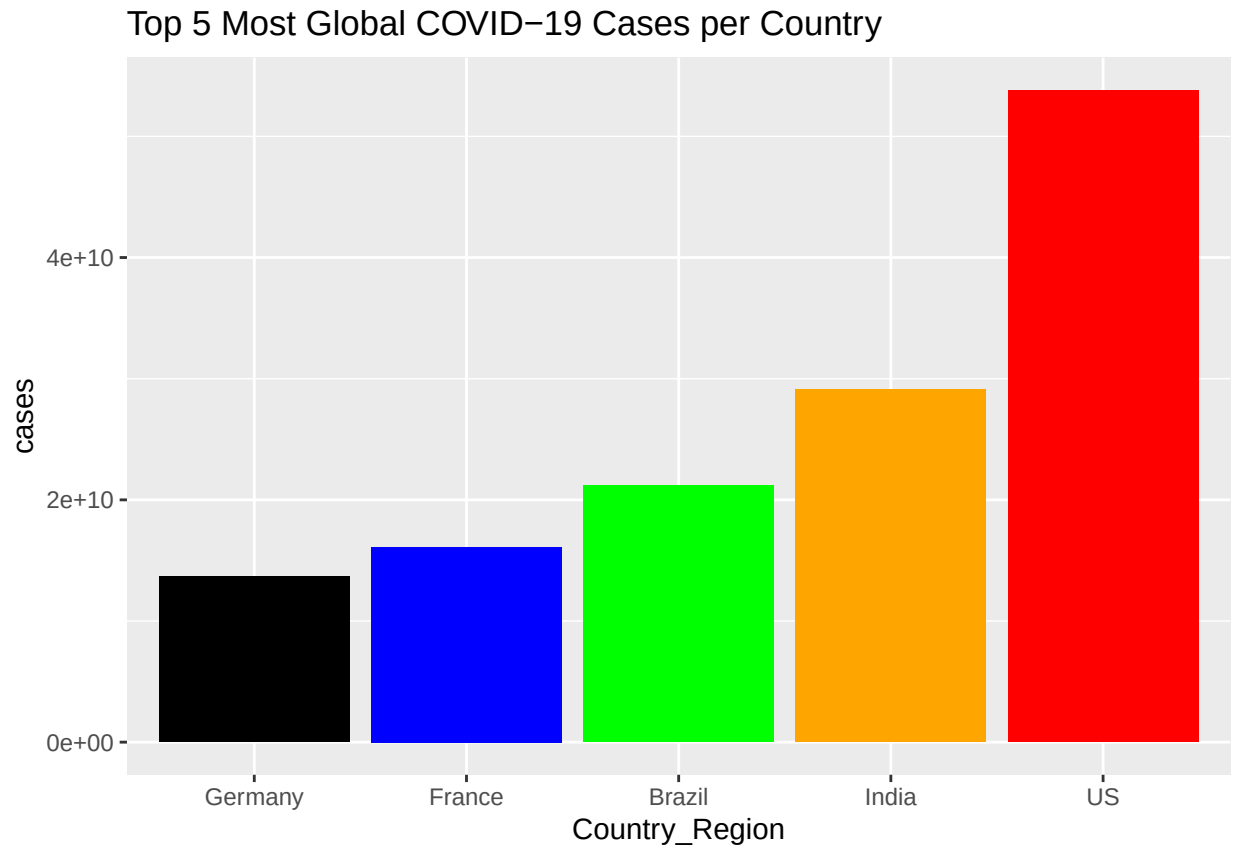
I used this new view of the data to filter for the Countries/Regions that had the Top 5 Cases, Deaths, Cases per Million, and Deaths per Million. I then visualized each of these in their own bar chart. The visualizations were created using ggplot and geom_bar.

```
Global_by_country <- global %>%
  group_by(Country_Region) %>%
  summarize(cases=sum(cases), deaths = sum(deaths),
            Population=max(Population)) %>%
  mutate(cases_per_mill = cases*1000000 / Population,
         deaths_per_mill = deaths*1000000 / Population) %>%
  select(Country_Region, cases, cases_per_mill, deaths,
         deaths_per_mill, Population)

#Top 5 Cases by Country
top_5_global_cases <- Global_by_country %>%
```

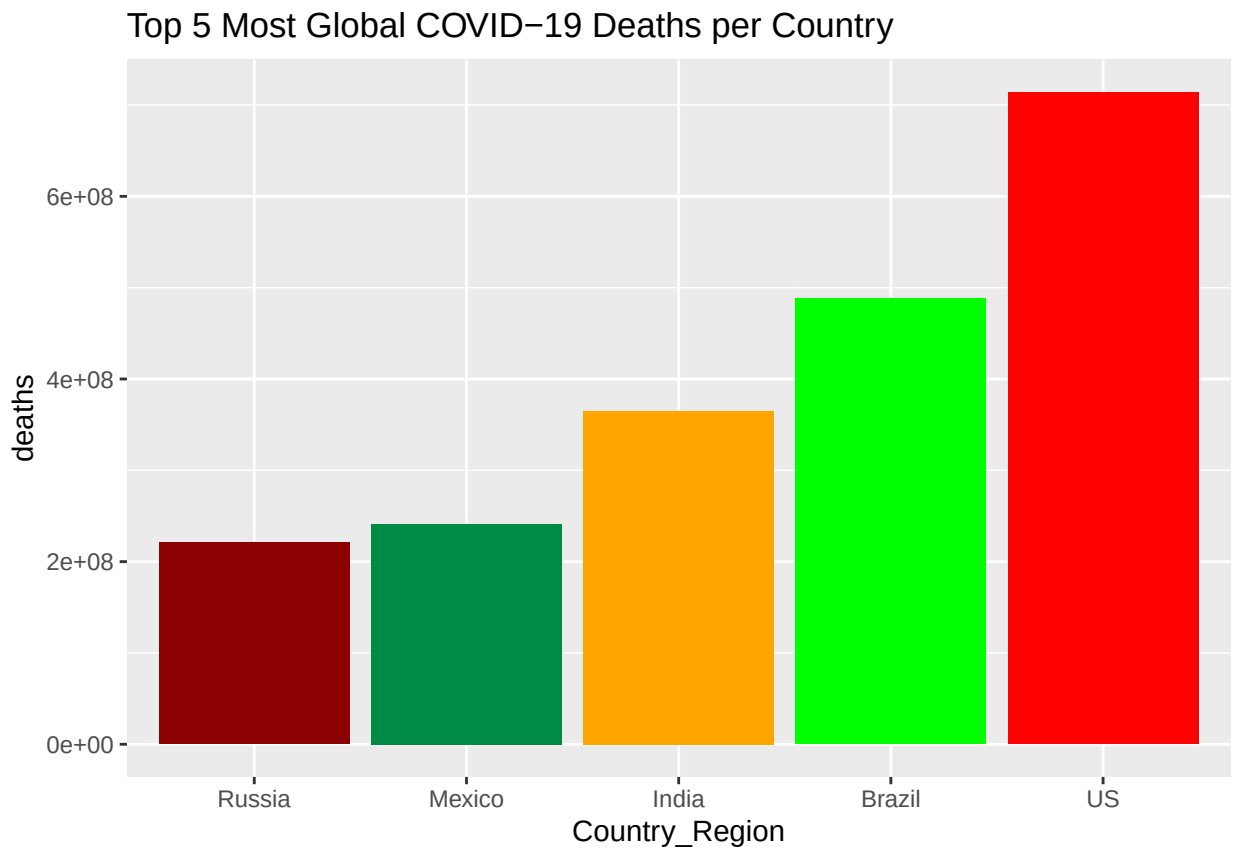
```
  top_n(5,cases) %>%
  arrange(desc(cases)) %>%
  select(Country_Region,cases,Population)

top_5_global_cases %>%
  mutate(Country_Region = fct_reorder(Country_Region, cases))%>%
  ggplot(aes(y=cases,x=Country_Region, fill=Country_Region))+
  geom_bar(stat = "identity")+
  scale_fill_manual(values = c("black", "blue", "green", "orange", "red"))+
  theme(legend.position="none")+
  ggtitle("Top 5 Most Global COVID-19 Cases per Country")
```



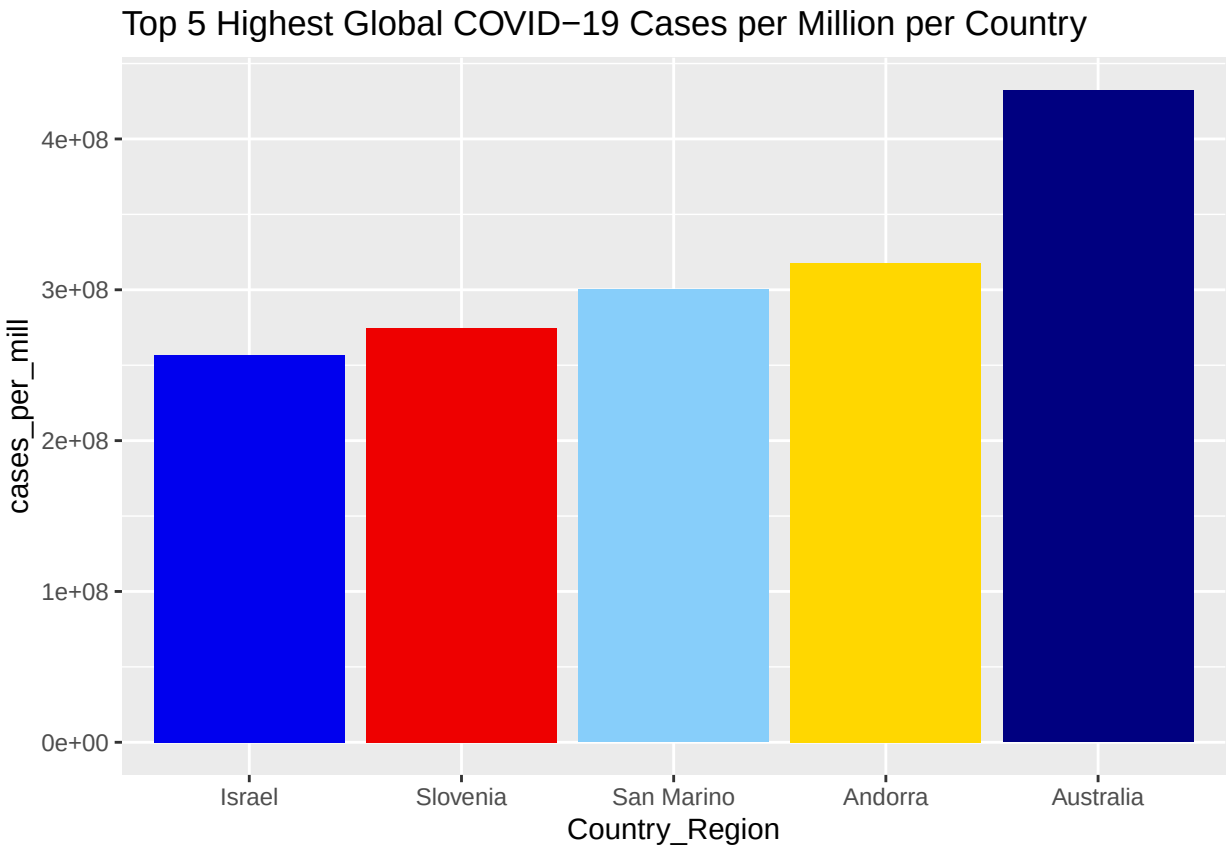

```
#Top 5 Deaths by Country
top_5_global_deaths <- Global_by_country %>%
  top_n(5,deaths) %>%
  arrange(desc(deaths)) %>%
  select(Country_Region,deaths,Population)

top_5_global_deaths %>%
  mutate(Country_Region = fct_reorder(Country_Region, deaths))%>%
  ggplot(aes(y=deaths,x=Country_Region, fill=Country_Region))+
  geom_bar(stat = "identity")+
  scale_fill_manual(values = c("red4", "springgreen4", "orange", "green", "red"))+
  theme(legend.position="none")+
  ggtitle("Top 5 Most Global COVID-19 Deaths per Country")
```



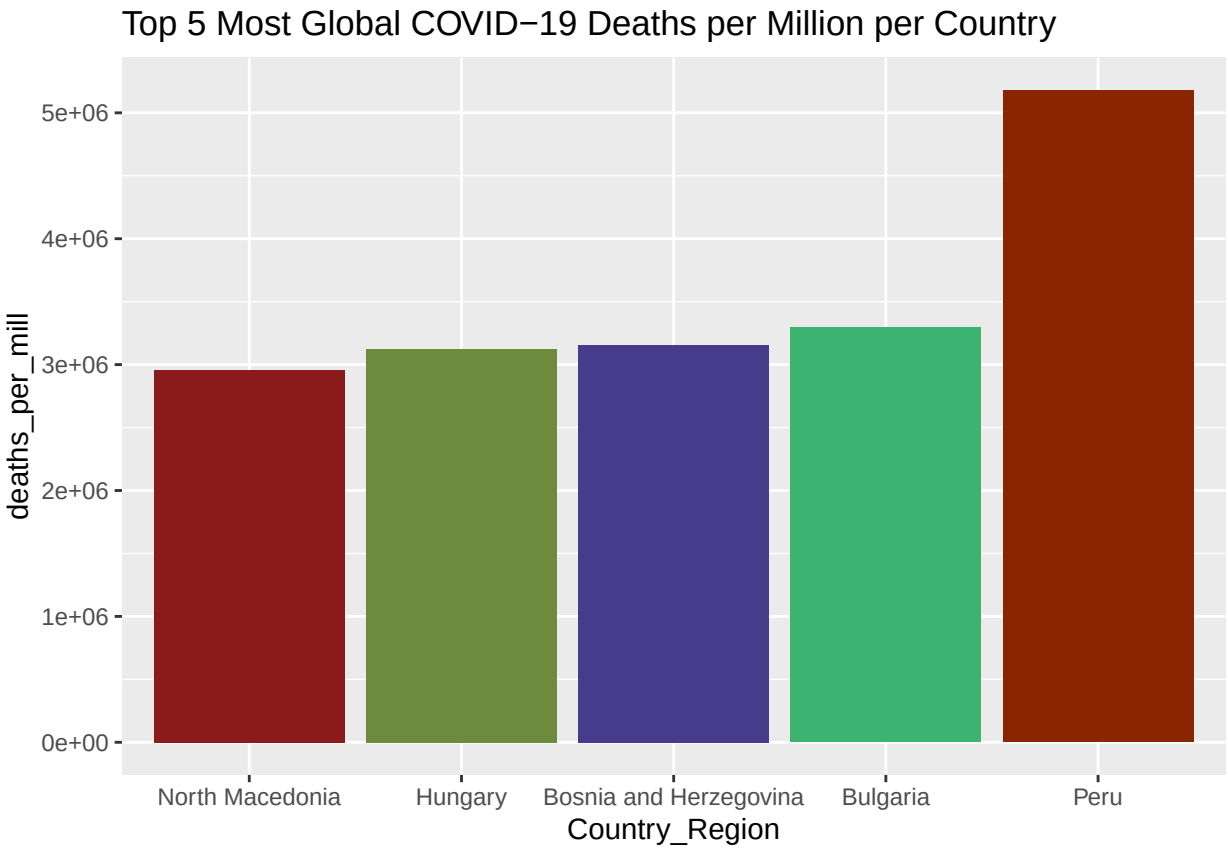

```
#Top 5 Cases per Million by Country
top_5_global_cpm <- Global_by_country %>%
  top_n(5,cases_per_mill) %>%
  arrange(desc(cases_per_mill)) %>%
  select(Country_Region,cases_per_mill, Population)

top_5_global_cpm %>%
  mutate(Country_Region = fct_reorder(Country_Region, cases_per_mill))%>%
  ggplot(aes(y=cases_per_mill,x=Country_Region, fill=Country_Region))+
  geom_bar(stat = "identity")+
  scale_fill_manual(values = c("blue2", "red2", "lightskyblue", "gold1", "navyblue"))+
  theme(legend.position="none")+
  ggtitle("Top 5 Highest Global COVID-19 Cases per Million per Country")
```




```
#Top 5 Deaths per Million by Country
top_5_global_dpm <- Global_by_country %>%
  top_n(5,deaths_per_mill) %>%
  arrange(desc(deaths_per_mill)) %>%
  select(Country_Region,deaths_per_mill, Population)

top_5_global_dpm %>%
  mutate(Country_Region = fct_reorder(Country_Region, deaths_per_mill))%>%
  ggplot(aes(y=deaths_per_mill,x=Country_Region, fill=Country_Region))+
  geom_bar(stat = "identity")+
  scale_fill_manual(values = c("firebrick4", "darkolivegreen4", "slateblue4", "mediumseagreen", "orange
  theme(legend.position="none")+
  ggtitle("Top 5 Most Global COVID-19 Deaths per Million per Country")
```



The Top 5 total cases and deaths bar charts were filled with large Countries/Regions with big populations such as India, Brazil, France, Germany and the United States which had the highest number of both cases and deaths.

I was more interested to see which Countries/Regions had the highest cases and deaths per million, and it did not disappoint. These bar charts showed a much different picture as none of the countries with the highest totals were also in the highest cases/deaths per million charts.

The country with the highest cases per million was actually Australia and the country with the highest deaths per million was Peru. It was interesting to see smaller countries like Andorra, San Marino, and Bosnia and Herzegovina make appearances in those lists as well.

Tidy US Data

I decided to do a similar process with the US data as I did with the global set. I combined the US case and death data into one full US data set and cleared out missing data along the way. This again was done with pivots and joins.

```
# Tidy US Cases data
US_cases <- US_cases %>%
  pivot_longer(cols = -(UID:Combined_Key),
               names_to = "date",
               values_to = "cases") %>%
  select(Admin2:cases) %>%
  mutate(date=mdy(date)) %>%
  select(-c(Lat,Long_))
```

```
# Tidy US Death data
US_deaths <- US_deaths %>%
  pivot_longer(cols = -(UID:Population),
               names_to = "date",
               values_to = "deaths") %>%
  select(Admin2:deaths) %>%
  mutate(date = mdy(date)) %>%
  select(-c(Lat, Long_))

# Join US cases and death data into one US data set and clear out missing data
US_total <- US_cases %>%
  full_join(US_deaths) %>%
  filter(Province_State != "Diamond Princess")%>%
  filter(Province_State != "Grand Princess")
```

```
## Joining with `by = join_by(Admin2, Province_State, Country_Region,
## Combined_Key, date)`
```

```
# Review summary of joined data set
summary(US_total)
```

```
##     Admin2          Province_State     Country_Region     Combined_Key
##  Length:3817620     Length:3817620     Length:3817620     Length:3817620
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##       date                cases           Population          deaths
##  Min.   :2020-01-22   Min.   :  -3073   Min.   :       0   Min.   :  -82
##  1st Qu.:2020-11-02   1st Qu.:    332   1st Qu.:    9928   1st Qu.:    4
##  Median :2021-08-15   Median :   2276   Median :   24911   Median :   37
##  Mean   :2021-08-15   Mean   :  14096   Mean   :   99663   Mean   :  187
##  3rd Qu.:2022-05-28   3rd Qu.:   8167   3rd Qu.:   64998   3rd Qu.:  122
##  Max.   :2023-03-09   Max.   :3710586   Max.   :10039107   Max.   :35545
```

Analyze and Visualize US Data

The US data analysis followed the path of the Global data analysis with the key difference being I grouped the data by US State rather than by country/region and I added fields for cases/deaths per thousand rather than per million people. I did this to account for smaller state populations.

```
# Group US data by State and add calculated fields
US_by_state <- US_total %>%
  group_by(Province_State, Country_Region,date) %>%
  summarize(cases=sum(cases),deaths=sum(deaths),
            Population =sum(Population)) %>%
  mutate(deaths_per_mill = deaths *1000000 / Population) %>%
  select(Province_State, Country_Region, date,
         cases, deaths, deaths_per_mill, Population) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'Province_State', 'Country_Region'. You can
## override using the `.groups` argument.
```

```
US_by_state <- US_by_state %>%
  group_by(Province_State) %>%
  summarize(cases=sum(cases), deaths = sum(deaths),
            Population=max(Population)) %>%
  mutate(cases_per_thou = cases*1000 / Population,
         deaths_per_thou = deaths*1000 / Population) %>%
  select(Province_State, cases, cases_per_thou, deaths, deaths_per_thou, Population)

tail(US_by_state)
```

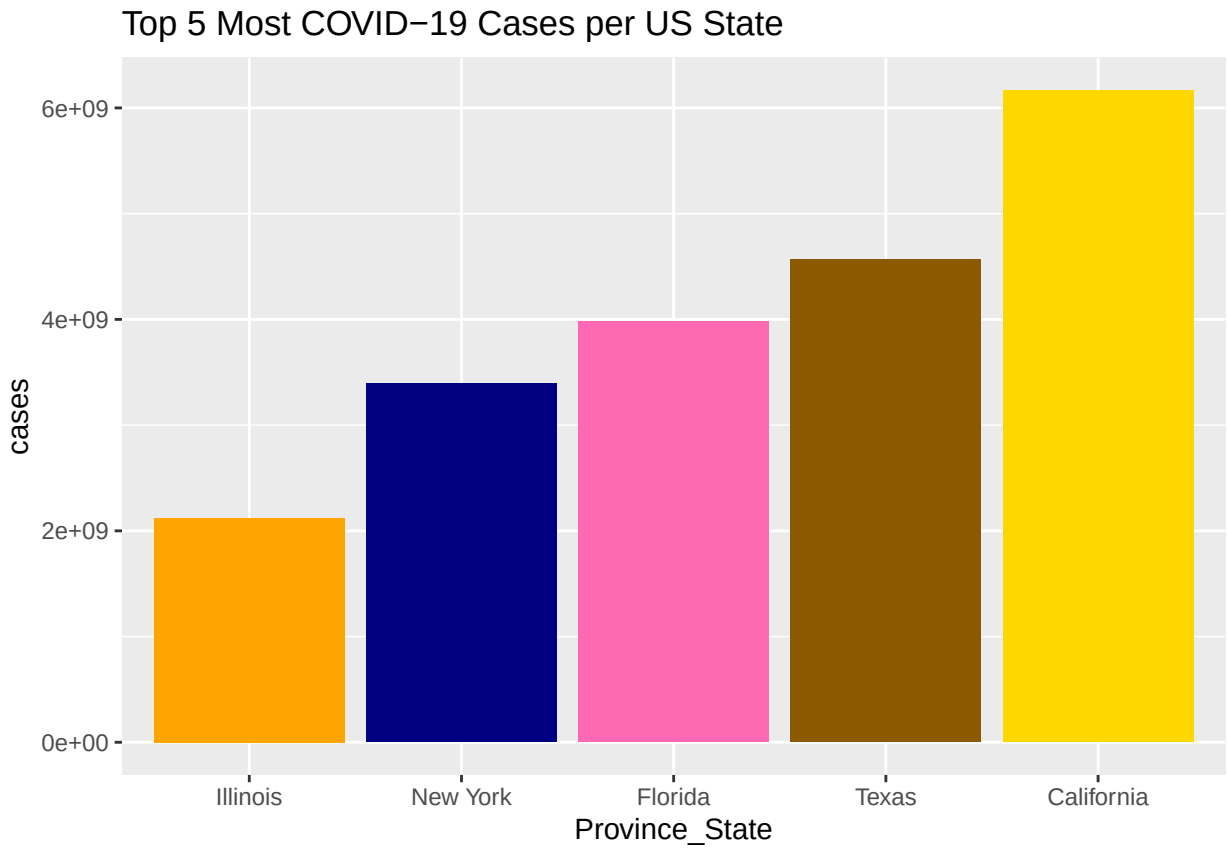
```
## # A tibble: 6 x 6
##   Province_State      cases cases_per_thou   deaths deaths_per_thou Population
##   <chr>               <dbl>          <dbl>    <dbl>           <dbl>      <dbl>
## 1 Virgin Islands   10749871        100215.    71105            663.     107268
## 2 Virginia       1127963803        132149. 13755116           1612.    8535519
## 3 Washington      924913618        121461.  8680526           1140.    7614893
## 4 West Virginia   312093419        174145.  4306523           2403.    1792147
## 5 Wisconsin      1065102633        182931.  9781470           1680.    5822434
## 6 Wyoming         101470234        175324.  1136735           1964.     578759
```

I decided to mimic the analysis and visualization performed on the global data on the us data. Meaning, I filtered the US by State data by top 5 states with the most cases, deaths, cases per thousand and deaths per thousand.

Again, each of these filtered views were visualized in the form of a bar chart which was created using ggplot and geom_bar.

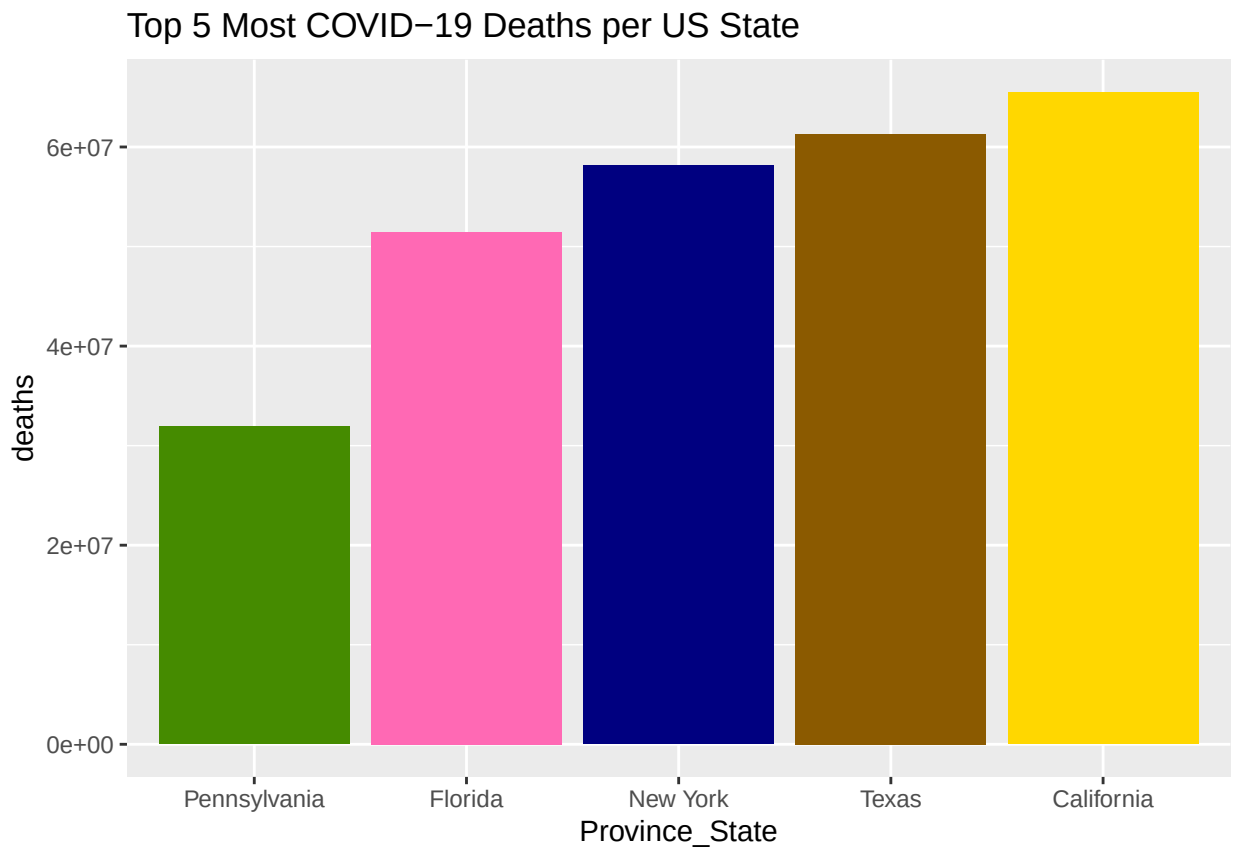
```
#Top 5 Cases by US State
top_5_us_cases <- US_by_state %>%
  top_n(5,cases) %>%
  select(Province_State,cases,Population)

top_5_us_cases %>%
  mutate(Province_State = fct_reorder(Province_State, cases))%>%
  ggplot(aes(y=cases,x=Province_State, fill=Province_State))+
  geom_bar(stat = "identity")+
  scale_fill_manual(values = c("orange", "navyblue", "hotpink", "orange4", "gold"))+
  theme(legend.position="none")+
  ggtitle("Top 5 Most COVID-19 Cases per US State")
```



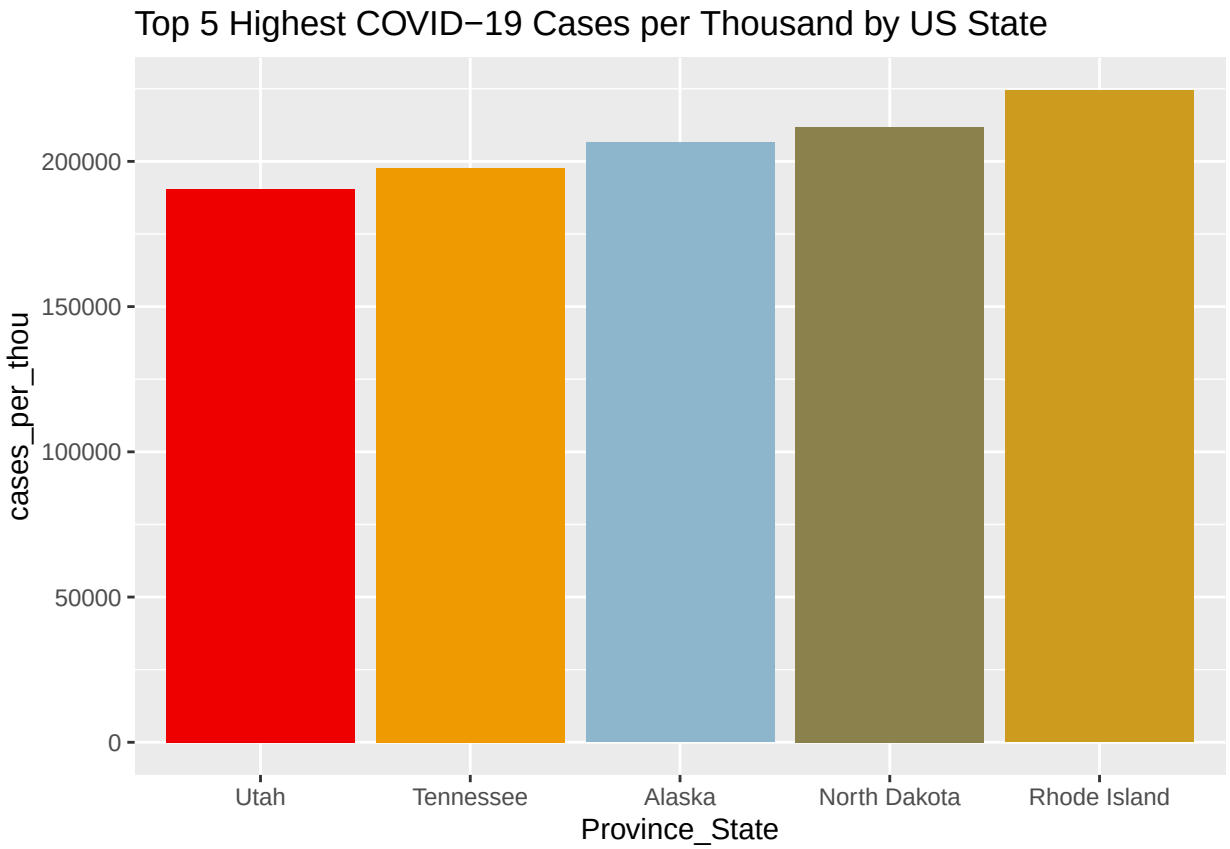

```
#Top 5 Deaths by US State
top_5_us_deaths <- US_by_state %>%
  top_n(5,deaths) %>%
  select(Province_State,deaths,Population)

top_5_us_deaths %>%
  mutate(Province_State = fct_reorder(Province_State, deaths))%>%
  ggplot(aes(y=deaths,x=Province_State, fill=Province_State))+
  geom_bar(stat = "identity")+
  scale_fill_manual(values = c("chartreuse4", "hotpink", "navyblue","orange4", "gold"))+
  theme(legend.position="none")+
  ggtitle("Top 5 Most COVID-19 Deaths per US State")
```



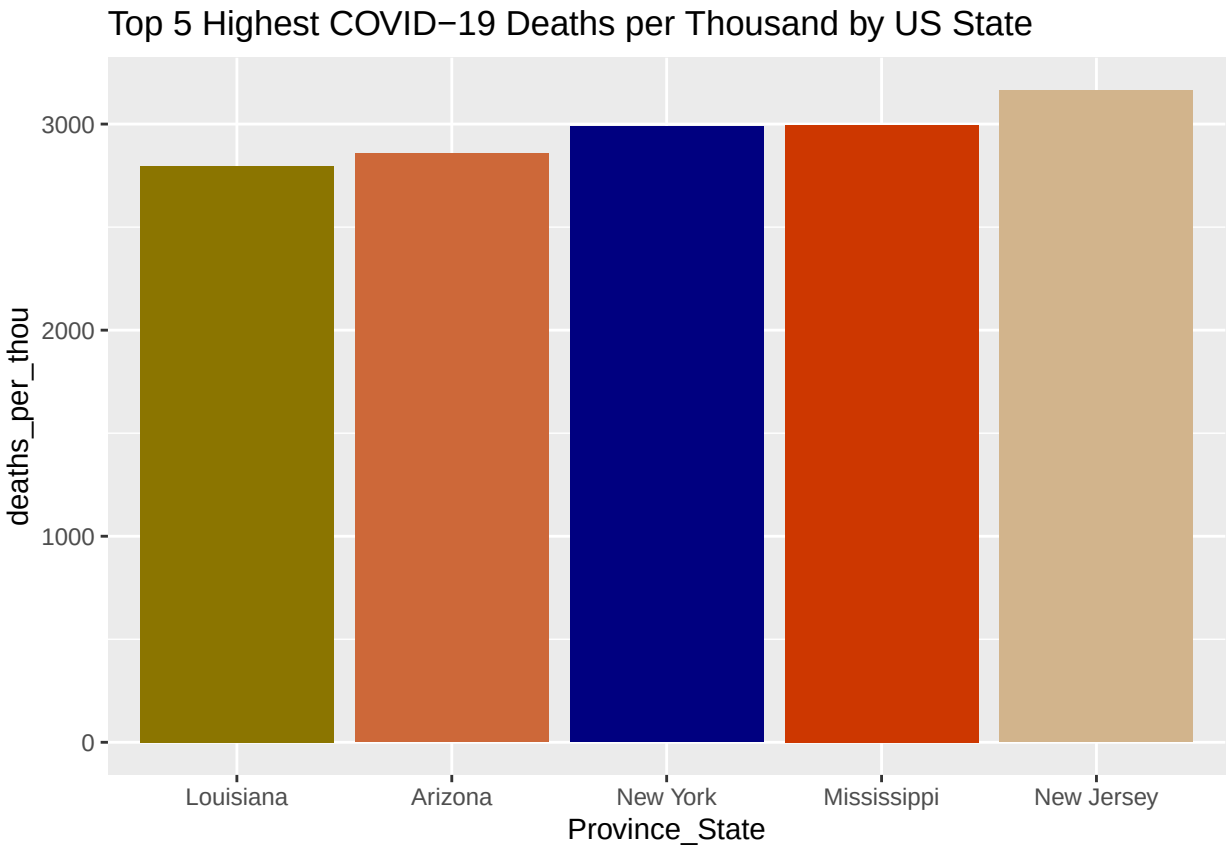

```
#Top 5 Cases per Thousand by US State
top_5_us_cpt <- US_by_state %>%
  top_n(5,cases_per_thou) %>%
  select(Province_State,cases_per_thou, Population)

top_5_us_cpt %>%
  mutate(Province_State = fct_reorder(Province_State, cases_per_thou))%>%
  ggplot(aes(y=cases_per_thou,x=Province_State, fill=Province_State))+
  geom_bar(stat = "identity")+
  scale_fill_manual(values = c("red2", "orange2", "lightskyblue3", "lightgoldenrod4", "goldenrod3"))+
  theme(legend.position="none")+
  ggtitle("Top 5 Highest COVID-19 Cases per Thousand by US State")
```




```
#Top 5 Deaths per Thousand by US State
top_5_us_dpt <- US_by_state %>%
  top_n(5,deaths_per_thou) %>%
  select(Province_State,deaths_per_thou, Population)

top_5_us_dpt %>%
  mutate(Province_State = fct_reorder(Province_State, deaths_per_thou))%>%
  ggplot(aes(y=deaths_per_thou,x=Province_State, fill=Province_State))+
  geom_bar(stat = "identity")+
  scale_fill_manual(values = c("gold4", "sienna3", "navyblue", "orangered3", "tan"))+
  theme(legend.position="none")+
  ggtitle("Top 5 Highest COVID-19 Deaths per Thousand by US State")
```



Similarly to the Global data, the states with the highest number of cases and deaths were states with large populations like California and Texas.

The state with the highest deaths per thousand people was New Jersey and the state with the highest cases per thousand people was Rhode Island. This too follows the pattern of the global data which saw smaller populations have higher cases/deaths per thousand or million people. There were a few cases that didn't quite follow this pattern such as New York which was in the highest deaths per thousand list and Tennessee which was in the highest cases per thousand list.

Model Creation and Visualization

I then created two models using linear regression. The first model was more to confirm the obvious, which was that the number of cases per thousand is a good predictor for the number of deaths per thousand. This was proven to be true by the visualization which shows the predicted values following the same pattern as the actual values on the plot. Additional evidence was provided by the very low p-value of the model which proves that cases per thousand is a statistically significant predictor of deaths per thousand.

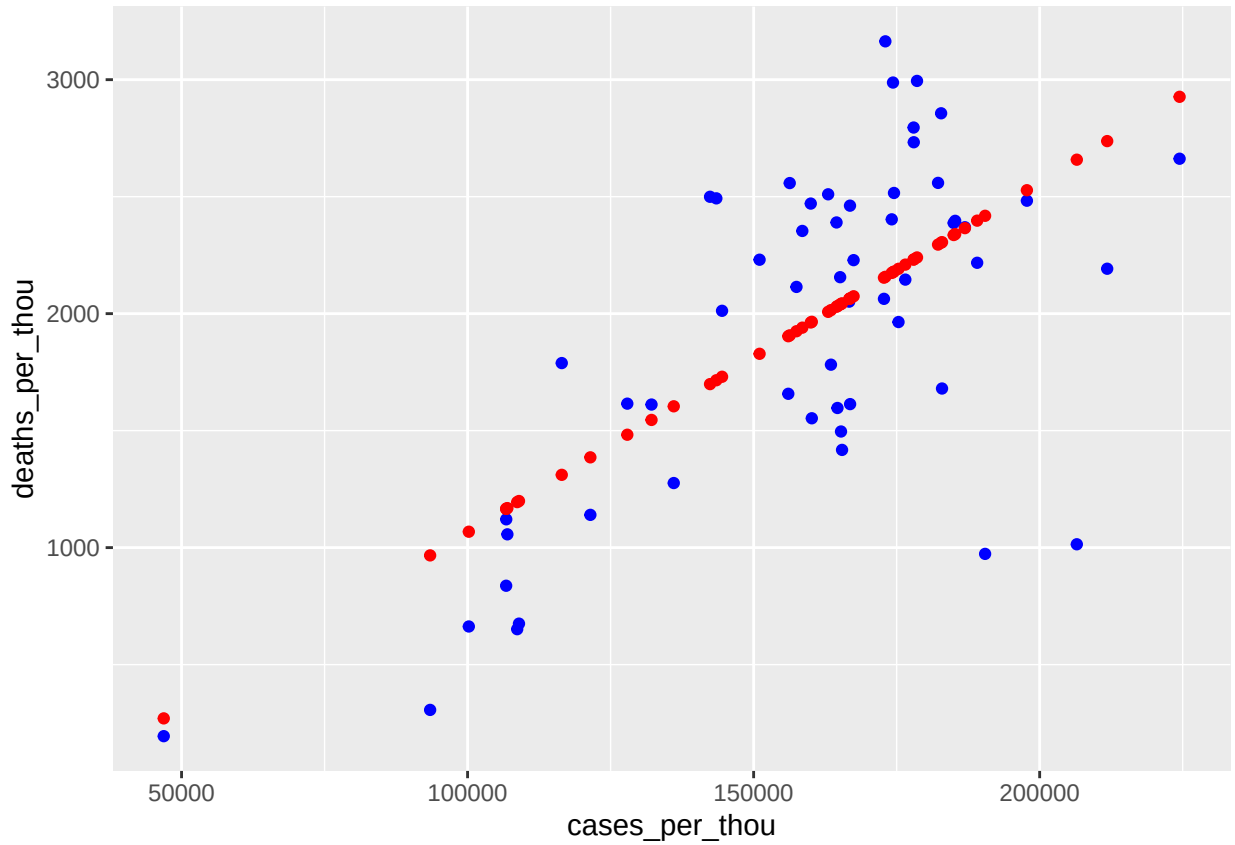
The second model, I created as more of an exploratory exercise. I wanted to see if the overall Population was a predictor of cases per thousand. I approached this in the same way, by creating a linear regression, adding a new field filled with predicted values and plotting the predicted values against the actual values. As it turns out, Population is NOT a good predictor of cases per thousand people! We can see this in the visualization, where the predicted values does not follow the pattern of the actual values and is further proven by a high p-value indicating this model is not statistically significant.

```
# Using a linear regression model to predict deaths per thousand using
# cases per thousand.
```

```r
us_mod1 <- lm(deaths_per_thou ~ cases_per_thou, data = US_by_state)
summary(us_mod1)
```

```
##
## Call:
## lm(formula = deaths_per_thou ~ cases_per_thou, data = US_by_state)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1643.74  -328.08    -3.83   398.02  1006.44
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -4.313e+02  3.488e+02  -1.236    0.222
## cases_per_thou  1.496e-02  2.164e-03   6.915 5.66e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 527.7 on 54 degrees of freedom
## Multiple R-squared:  0.4696, Adjusted R-squared:  0.4598
## F-statistic: 47.82 on 1 and 54 DF,  p-value: 5.665e-09
```

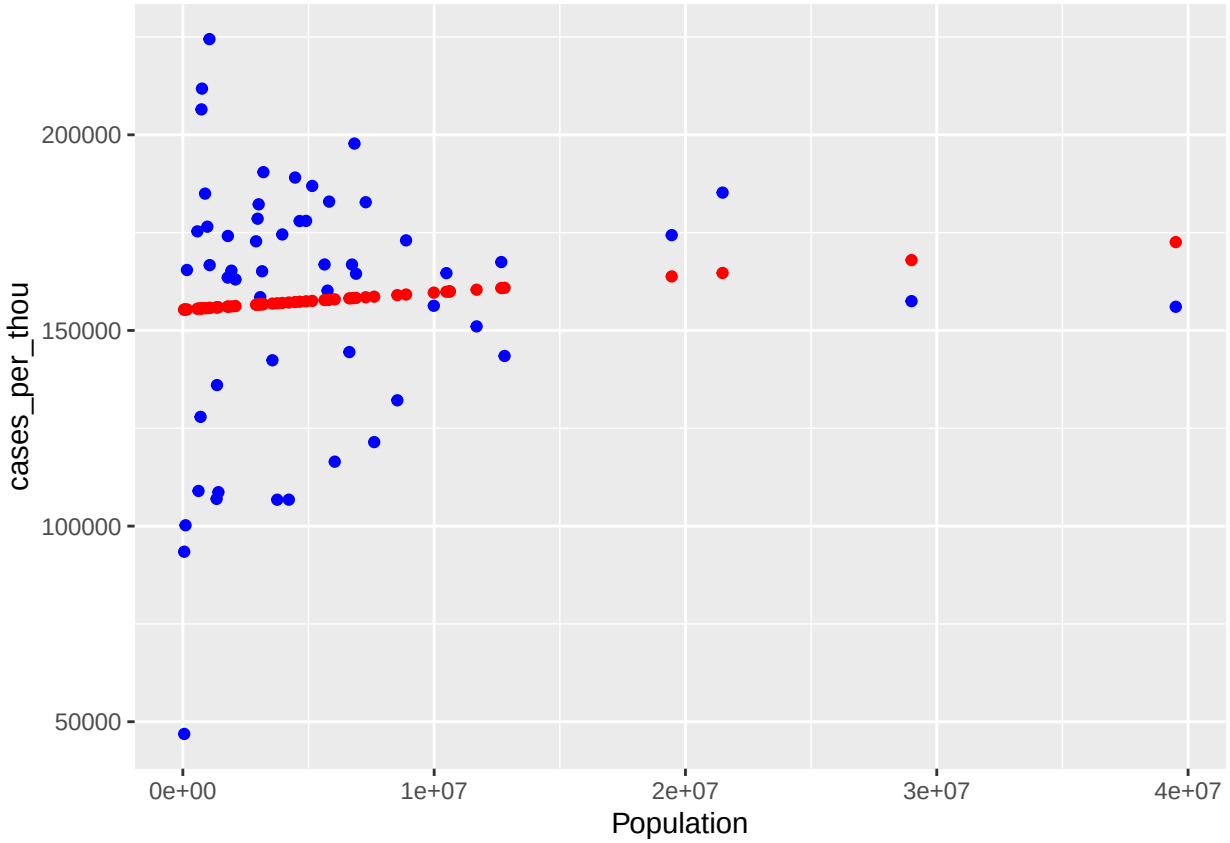
```r
#Visualize actuals vs. predicted values using our first model
us_state_w_pred1 <- US_by_state %>% mutate(pred = predict(us_mod1))
us_state_w_pred1 %>% ggplot() +
  geom_point(aes(x=cases_per_thou, y=deaths_per_thou), color="blue")+
  geom_point(aes(x=cases_per_thou, y=pred), color="red")
```




```
# Using a linear regression model to predict cases per thousand using
# total state Population.
us_mod2 <- lm(cases_per_thou ~ Population, data=US_by_state)
summary(us_mod2)
```

```
##
## Call:
## lm(formula = cases_per_thou ~ Population, data = US_by_state)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -108439  -16745    7942   20573   68696
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.553e+05  5.741e+03  27.050   <2e-16 ***
## Population  4.373e-04  6.174e-04   0.708    0.482
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 33040 on 54 degrees of freedom
## Multiple R-squared:  0.009202,   Adjusted R-squared:  -0.009146
## F-statistic: 0.5015 on 1 and 54 DF,  p-value: 0.4819
```

```
#Visualize actuals vs. predicted values using our second model
us_state_w_pred2 <- US_by_state %>% mutate(pred_cpt = predict(us_mod2))
us_state_w_pred2 %>% ggplot() +
  geom_point(aes(x=Population, y=cases_per_thou), color="blue")+
  geom_point(aes(x=Population, y=pred_cpt), color="red")
```



Conclusion

In conclusion, this was a great exercise in working with multiple data sets. It was certainly a challenge to load, join, tidy, analyze, visualize, and model soooo much data. There seemed to be many different paths I could have taken. Ultimately, I had to choose one path as there was simply too much to work with to analyze everything.

Throughout the analysis, it was interesting to see some initial thoughts confirmed such as the largest countries and states having the most cases and deaths. It was also interesting to see some surprises come into play when investigating the cases/deaths per million/thousand. This was an important lesson to learn that raw totals do not always tell the full story, often times normalizing the data can provide a new perspective.

Lastly, I was able to confirm one obvious suspicious using a linear regression model. That being that the number of cases per thousand directly predicts the number of deaths per thousand. This makes sense and is supported visually on our plot of actuals vs. predictions and with the low p-value of the first model.

We also learned that Population size is NOT a good predictor of cases per thousand, at least not by using a linear regression model. While it was a little disappointing that my guess was incorrect, it was an important part of the project nonetheless. It goes to show that not every chart or model we make is going to be significant. And sometimes finding what doesn't work, can help lead us to what does.

```r
sessionInfo()
```

```
## R version 4.3.2 (2023-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 11 x64 (build 22631)
##
## Matrix products: default
##
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## time zone: America/New_York
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] lubridate_1.9.3 forcats_1.0.0   stringr_1.5.1   dplyr_1.1.4
##  [5] purrr_1.0.2     readr_2.1.4     tidyr_1.3.0     tibble_3.2.1
##  [9] ggplot2_3.4.4   tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] bit_4.0.5         gtable_0.3.4      highr_0.10        crayon_1.5.2
##  [5] compiler_4.3.2    tidyselect_1.2.0  parallel_4.3.2    scales_1.3.0
##  [9] yaml_2.3.8        fastmap_1.1.1     R6_2.5.1          labeling_0.4.3
## [13] generics_0.1.3    curl_5.2.0        knitr_1.45        munsell_0.5.0
## [17] pillar_1.9.0      tzdb_0.4.0        rlang_1.1.2       utf8_1.2.4
## [21] stringi_1.8.3     xfun_0.41         bit64_4.0.5       timechange_0.2.0
## [25] cli_3.6.2         withr_2.5.2       magrittr_2.0.3    digest_0.6.33
## [29] grid_4.3.2        vroom_1.6.5       rstudioapi_0.15.0 hms_1.1.3
## [33] lifecycle_1.0.4   vctrs_0.6.5       evaluate_0.23     glue_1.6.2
## [37] farver_2.1.1      fansi_1.0.6       colorspace_2.1-0  rmarkdown_2.25
## [41] tools_4.3.2       pkgconfig_2.0.3   htmltools_0.5.7
```