

COVID 19 Data Analysis

2023-05-13

Introduction:

This is the final project for the Data Science as a Field course (DTSA 5301). In this project, we will analyze a COVID-19 data set available on the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University Github repository as it's a reliable source. I'll be dropping, cleaning and modifying multiple data points, I will be pointing out each change as I do it. And, I will guide you through the analysis I have conducted in this project, with a focus on analyzing the US data set, I will address the following Questions of interest for Washington state.

Objectives and Questions of Interest:

- Creating reproducible code that can be verified by my peers.
- Cleaning and analyzing the data to answer the following questions about Washington state:
 - What is the infection rate of COVID-19 per county in Washington state?
 - What are the three counties with the highest number of COVID-19 cases?
 - Predicting COVID-19 deaths in Washington state using a linear regression model.

Before we start:

Please note that this project uses the package tidyverse, if it's not installed, run the following two commands in R or R-Studio console `install.packages("tidyverse")`. If this is your first time using RStudio please note that you might also need to install tinytex using the following `install.packages("tinytex")`

Project steps:

Step 1: This step involves the following:

1- Import the following libraries:

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
## v dplyr      1.1.1      v readr      2.1.4
## v forcats    1.0.0      v stringr    1.5.0
## v ggplot2    3.4.2      v tibble     3.2.1
## v lubridate  1.9.2      v tidyr      1.3.0
## v purrr      1.0.1
```

```
## -- 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)
library(ggplot2)
library(dplyr)
```

2- Download the data set from the following source <https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/>

```
base_url <- ("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_data")

# The vector that has all the CSV file names, I'm only interested in the US cases and deaths data sets

csv_file_names <-
  c("time_series_covid19_confirmed_US.csv",
    "time_series_covid19_deaths_US.csv")

file_urls <- str_c(base_url, csv_file_names)

Raw_US_Cases <- read_csv(file_urls[1])

## 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.

Raw_US_Deaths <- read_csv(file_urls[2])
```

```
## 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, 1/24/20...
##
## 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.
```

Step 2: This step will tidy and/or transform the data to make it ready for the visualization steps:

This step will involve the following:

- Cleaning the US Cases dataset by removing unnecessary data for our analysis and viewing a summary of the data. As you can see in the summary below, the minimum number of cases at the time of importing the dataset has a negative value. To address this, I added a filter to include only cases larger than or equal to 0. Although the filter step can be combined, I am analyzing the data step by step for clarity.

```
US_cases <- Raw_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_))
```

- Print a summary of the US_cases and make sure that we have what we need.

```
US_cases <- US_cases %>%
  filter(cases >= 0)

# Print a summary of the US_cases
summary(US_cases)
```

```
##      Admin2          Province_State      Country_Region      Combined_Key
## Length:3819903      Length:3819903      Length:3819903      Length:3819903
## Class :character     Class :character     Class :character     Class :character
## Mode  :character     Mode  :character     Mode  :character     Mode  :character
##
##
##
##      date          cases
## Min.   :2020-01-22   Min.    :      0
## 1st Qu.:2020-11-02   1st Qu.:    330
## Median :2021-08-15   Median :   2272
## Mean   :2021-08-14   Mean    : 14088
## 3rd Qu.:2022-05-28   3rd Qu.:   8159
## Max.   :2023-03-09   Max.    :3710586
```

- US_cases has the following columns:
 - *Admin2*: County name.
 - *Province_State*: State.
 - *Country_Region*: US.
 - *Combined_Key*: County and state.
 - *date*: Date in Year-Month-Day format.
 - *cases*: COVID19 cases.

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

US_deaths <- US_deaths %>%
  filter(deaths >= 0)

# Print a summary of hte US_deaths
summary(US_deaths)
```

And repeat the same thing for the US Deaths.

```
##      Admin2          Province_State      Country_Region      Combined_Key
## Length:3819903      Length:3819903      Length:3819903      Length:3819903
## Class :character     Class :character     Class :character     Class :character
## Mode  :character     Mode  :character     Mode  :character     Mode  :character
##
##
##
##      Population      date          deaths
## Min.   :      0      Min.   :2020-01-22   Min.   :    0.0
## 1st Qu.:    9917      1st Qu.:2020-11-02   1st Qu.:     4.0
## Median :   24909      Median :2021-08-15   Median :    37.0
## Mean   :   99604      Mean   :2021-08-14   Mean   :   186.9
## 3rd Qu.:   64979      3rd Qu.:2022-05-28   3rd Qu.:   122.0
## Max.   :10039107      Max.   :2023-03-09   Max.   :35545.0
```

- US_deaths has the following columns:
 - *Admin2*: County name.

- *Province_State*: State.
- *Country_Region*: US.
- *Combined_Key*: County and state.
- *date*: Date in Year-Month-Day format.
- *Population*: County population
- *deaths*: COVID19 deaths.

- Finally we need to join the two data sets, we end up with a combined data set named US.

```
US <- US_cases %>%
  full_join(US_deaths)
```

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

```
# Print a summary of the US
summary(US)
```

```
##      Admin2          Province_State      Country_Region      Combined_Key
## Length:3819903      Length:3819903      Length:3819903      Length:3819903
## 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.   :      0      Min.   :      0      Min.   :      0.0
## 1st Qu.:2020-11-02      1st Qu.:    330      1st Qu.:   9917      1st Qu.:      4.0
## Median :2021-08-15      Median :   2272      Median :  24909      Median :     37.0
## Mean   :2021-08-14      Mean   :  14088      Mean   :  99604      Mean   :    186.9
## 3rd Qu.:2022-05-28      3rd Qu.:   8159      3rd Qu.:  64979      3rd Qu.:   122.0
## Max.   :2023-03-09      Max.   :3710586      Max.   :10039107      Max.   :35545.0
```

- And finally these are the columns of the US:

- *Admin2*: County name.
- *Province_State*: State.
- *Country_Region*: US.
- *Combined_Key*: County and state.
- *date*: Date in Year-Month-Day format.
- *cases*: COVID19 cases.
- *Population*: County population
- *deaths*: COVID19 deaths.

- Now that we have the data cleaned up a bit, we'll work on grouping, summarizing and adding new fields that will be very useful when we start visualizing the data.

```
# Group the data by (Province_State, Country_Region, date) to get the sum of cases, deaths and population
US_by_state <- US %>%
  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.
```

- Let's print a summary of what we have so far.

```
summary(US_by_state)
```

```
## Province_State      Country_Region      date      cases
## Length:66294      Length:66294      Min.   :2020-01-22      Min.   :      0
## Class :character    Class :character    1st Qu.:2020-11-02      1st Qu.: 31115
## Mode  :character    Mode  :character    Median :2021-08-15      Median : 293146
##                                     Mean  :2021-08-15      Mean   : 811738
##                                     3rd Qu.:2022-05-28      3rd Qu.: 953450
##                                     Max.   :2023-03-09      Max.   :12129699
##
##      deaths      deaths_per_mill      Population
## Min.   :      0      Min.   : 0.0      Min.   :      0
## 1st Qu.: 555      1st Qu.: 490.2      1st Qu.: 1068778
## Median : 3849      Median :1665.9      Median : 3660113
## Mean   : 10768      Mean   :  Inf      Mean   : 5739226
## 3rd Qu.: 13695      3rd Qu.:2794.0      3rd Qu.: 6892503
## Max.   :101159      Max.   :  Inf      Max.   :39512223
##                                     NA's   :1211
```

- Next we'll group the data by the Country_Region and Date, so for each date we'll see how many cases there are.

```
US_totals <- US_by_state %>%
  group_by( Country_Region, date) %>%
  summarize(cases = sum(cases), deaths = sum(deaths), Population = sum(Population)) %>%
  mutate(deaths_per_mill = deaths * 1000000 / Population) %>%
  select(Country_Region, date, cases, deaths, deaths_per_mill, Population) %>%
  ungroup()
```

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

```
US_totals
```

```
## # A tibble: 1,143 x 6
##   Country_Region date      cases deaths deaths_per_mill Population
##   <chr>          <date>    <dbl> <dbl>          <dbl>      <dbl>
## 1 US            2020-01-22      1      1          0.00300  332875137
## 2 US            2020-01-23      1      1          0.00300  332875137
## 3 US            2020-01-24      2      1          0.00300  332875137
## 4 US            2020-01-25      2      1          0.00300  332875137
## 5 US            2020-01-26      5      1          0.00300  332875137
## 6 US            2020-01-27      5      1          0.00300  332875137
## 7 US            2020-01-28      5      1          0.00300  332875137
## 8 US            2020-01-29      6      1          0.00300  332875137
## 9 US            2020-01-30      6      1          0.00300  332875137
## 10 US           2020-01-31      8      1          0.00300  332875137
## # i 1,133 more rows
```

```
summary(US_totals)
```

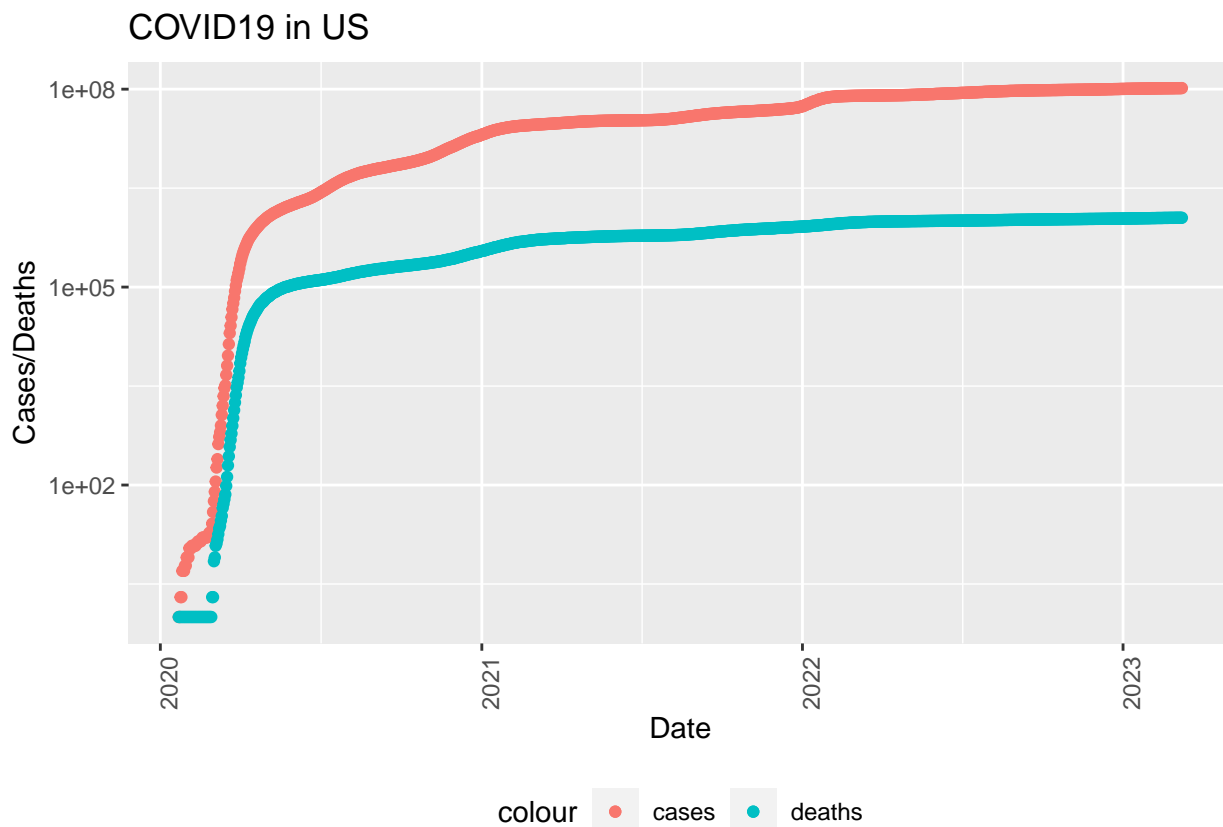
```
## Country_Region      date      cases      deaths
## Length:1143      Min.   :2020-01-22      Min.   :      1      Min.   :      1
## Class :character    1st Qu.:2020-11-02      1st Qu.: 9401880      1st Qu.: 232564
## Mode  :character    Median :2021-08-15      Median : 36845902      Median : 618029
##                                     Mean  :2021-08-15      Mean   : 47080800      Mean   : 624563
```

```
##          3rd Qu.:2022-05-27    3rd Qu.: 84083678    3rd Qu.:1006626
##          Max.      :2023-03-09    Max.      :103802702    Max.      :1123836
## deaths_per_mill      Population
## Min.      : 0.003    Min.      :332875137
## 1st Qu.: 698.652    1st Qu.:332875137
## Median :1856.639    Median :332875137
## Mean      :1876.268    Mean      :332875137
## 3rd Qu.:3024.033    3rd Qu.:332875137
## Max.      :3376.149    Max.      :332875137
```

Step 3: Visualize and analyze the data:

- We start by graphing the data in the of US_totals which will show the total cases and deaths per day.

```
US_totals %>%
  ggplot(aes(x = date, y = cases)) +
  geom_point(aes(color = "cases")) +
  geom_point(aes(y = deaths, color = "deaths")) +
  scale_y_log10() + theme(legend.position = "bottom", axis.text.x = element_text(angle = 90)) +
  labs(title = "COVID19 in US", x = "Date", y = "Cases/Deaths")
```

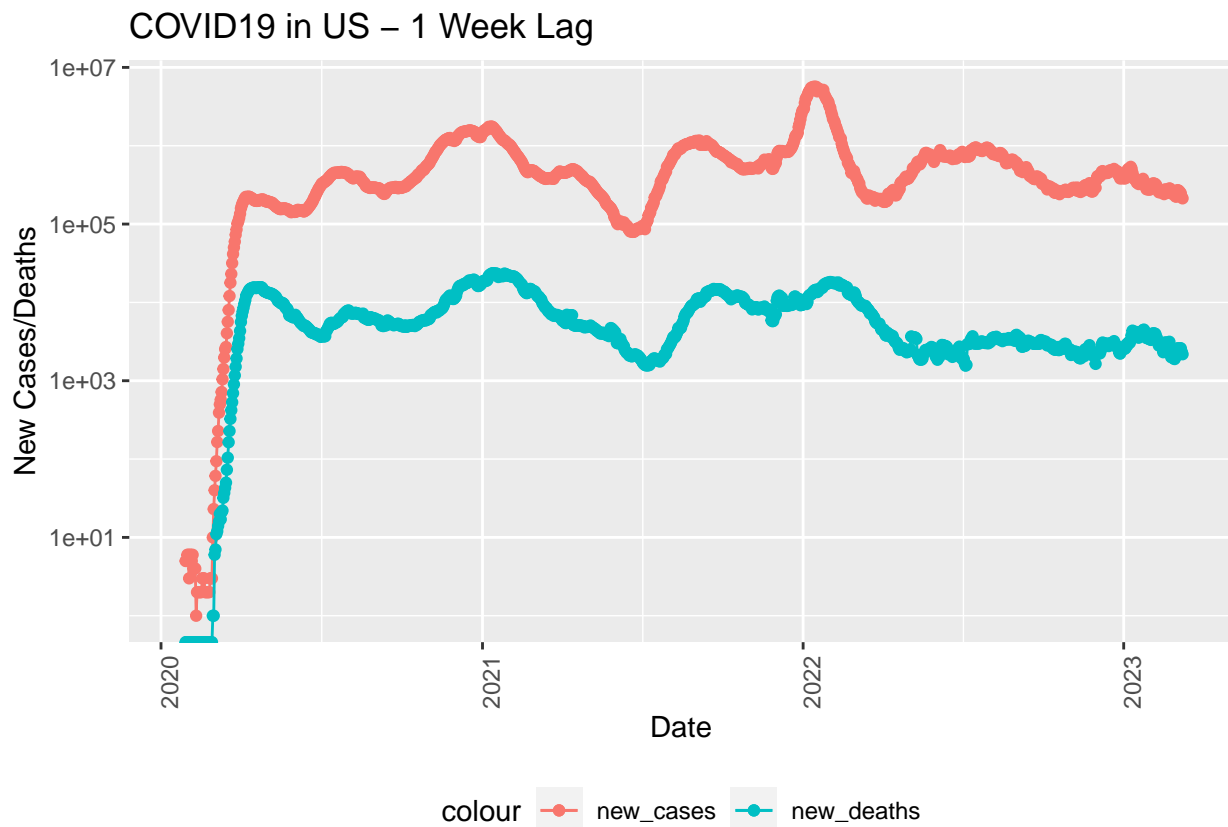


- We will now introduce additional variables to track the daily new cases and new deaths. These values will be calculated by subtracting the current number of deaths from the corresponding figure recorded one week prior. This specific choice of a one-week interval allows for a smoother graph representation.

```
lag_value = 7

US_totals <- US_totals %>%
  mutate(new_cases = cases - lag(cases, lag_value), new_deaths = deaths - lag(deaths, lag_value))
```

```
US_totals %>%
  ggplot(aes(x = date, y = new_cases)) +
  geom_line(aes(color = "new_cases")) +
  geom_point(aes(color = "new_cases")) +
  geom_line(aes(y = new_deaths, color = "new_deaths")) +
  geom_point(aes(y = new_deaths, color = "new_deaths")) +
  scale_y_log10() +
  theme(legend.position = "bottom", axis.text.x = element_text(angle = 90)) +
  labs(title = "COVID19 in US - 1 Week Lag", x = "Date", y = "New Cases/Deaths")
```

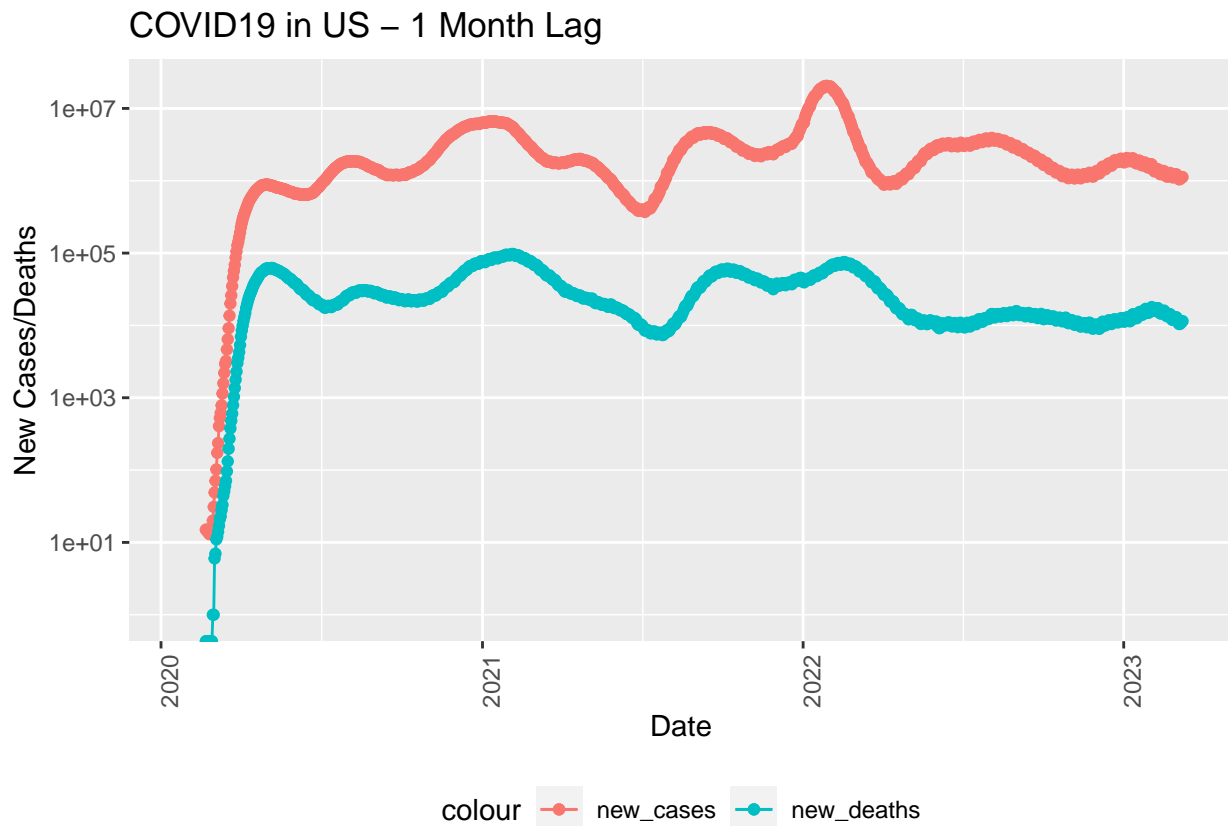


- Let's now visualize the data with a 30-day lag.

```
lag_value = 30

US_totals <- US_totals %>%
  mutate(new_cases = cases - lag(cases, lag_value), new_deaths = deaths - lag(deaths, lag_value))

US_totals %>%
  ggplot(aes(x = date, y = new_cases)) +
  geom_line(aes(color = "new_cases")) +
  geom_point(aes(color = "new_cases")) +
  geom_line(aes(y = new_deaths, color = "new_deaths")) +
  geom_point(aes(y = new_deaths, color = "new_deaths")) +
  scale_y_log10() +
  theme(legend.position = "bottom", axis.text.x = element_text(angle = 90)) +
  labs(title = "COVID19 in US - 1 Month Lag", x = "Date", y = "New Cases/Deaths")
```



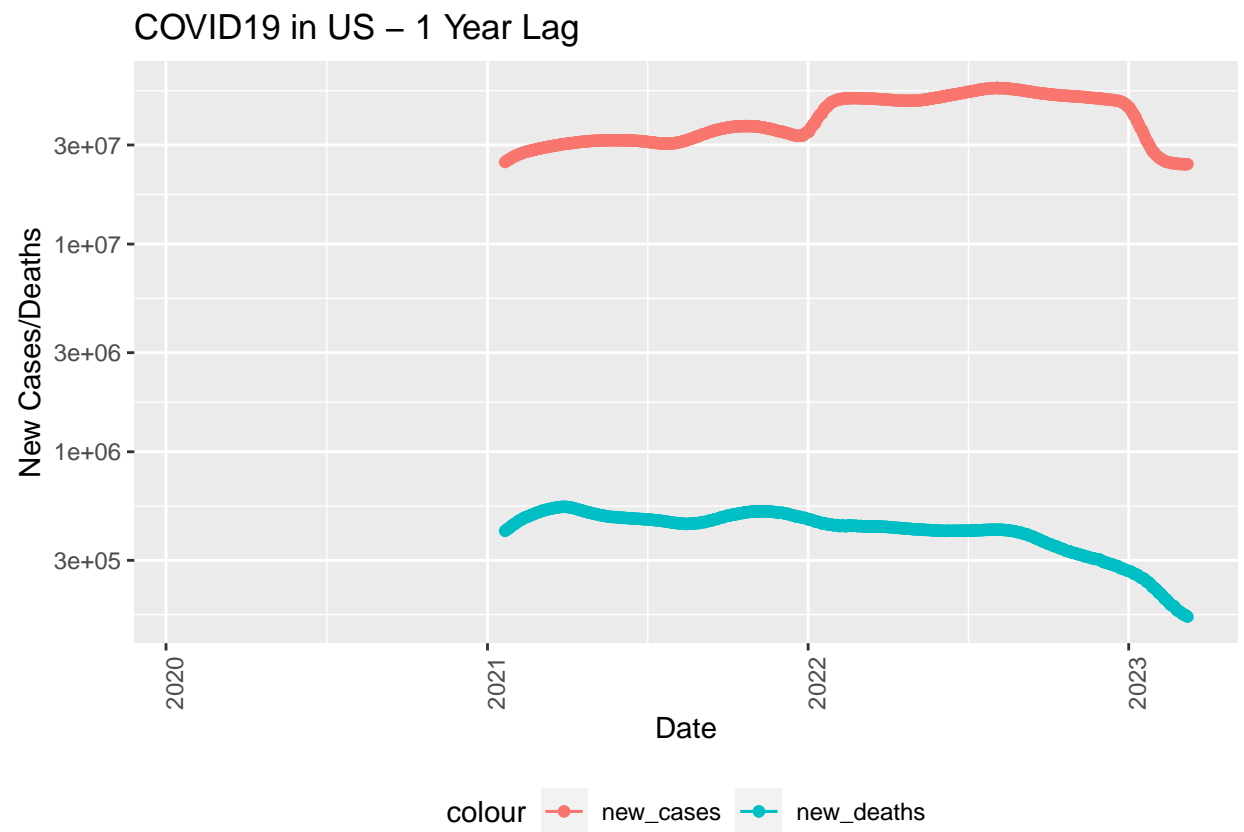
- Additionally, let's explore the yearly increase, considering that we now have data spanning multiple years. We will examine the increase in cases per year. What I noticed in the graph below is that it looks like there is a substantial decrease in the number of deaths.

```
lag_value = 365

US_by_state <- US_by_state %>%
  mutate(new_cases = cases - lag(cases, lag_value), new_deaths = deaths - lag(deaths, lag_value))

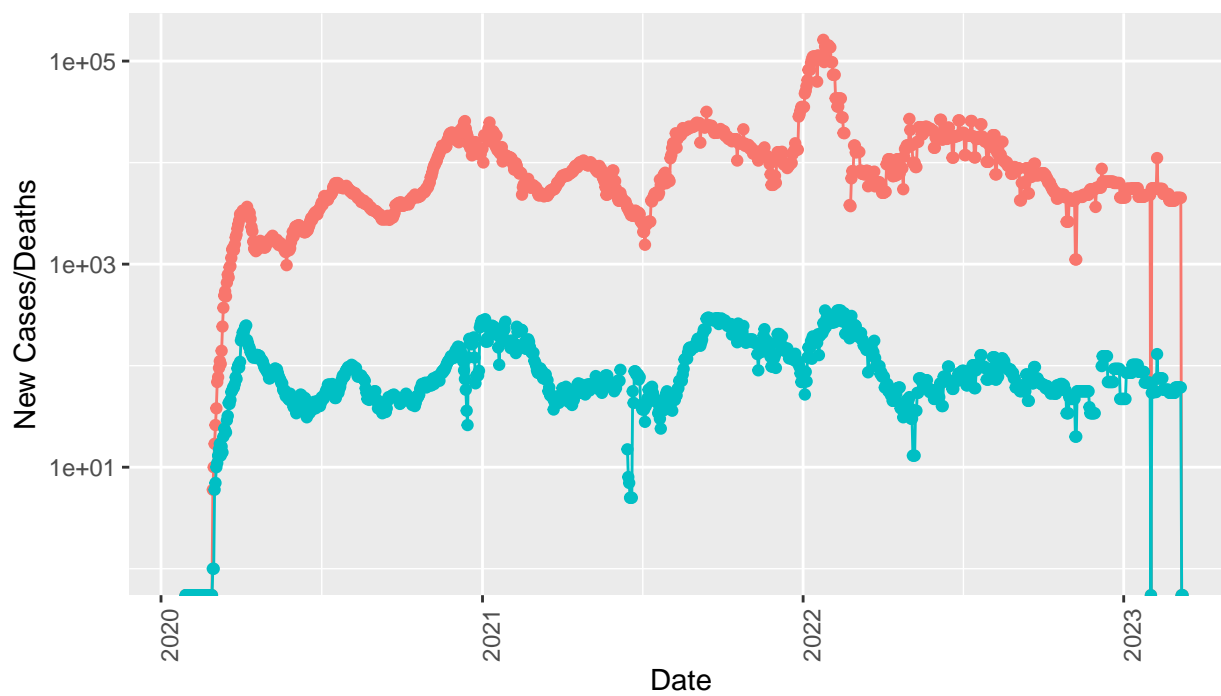
US_totals <- US_totals %>%
  mutate(new_cases = cases - lag(cases, lag_value), new_deaths = deaths - lag(deaths, lag_value))

US_totals %>%
  ggplot(aes(x = date, y = new_cases)) +
  geom_line(aes(color = "new_cases")) +
  geom_point(aes(color = "new_cases")) +
  geom_line(aes(y = new_deaths, color = "new_deaths")) +
  geom_point(aes(y = new_deaths, color = "new_deaths")) +
  scale_y_log10() +
  theme(legend.position = "bottom", axis.text.x = element_text(angle = 90)) +
  labs(title = "COVID19 in US - 1 Year Lag", x = "Date", y = "New Cases/Deaths")
```

- Now, let's shift our focus to Washington state. As we examine the graphs below, we can see that Washington state is following the same overall trend as depicted earlier for the entire United States.

COVID19 in WA State – 1 Week Lag

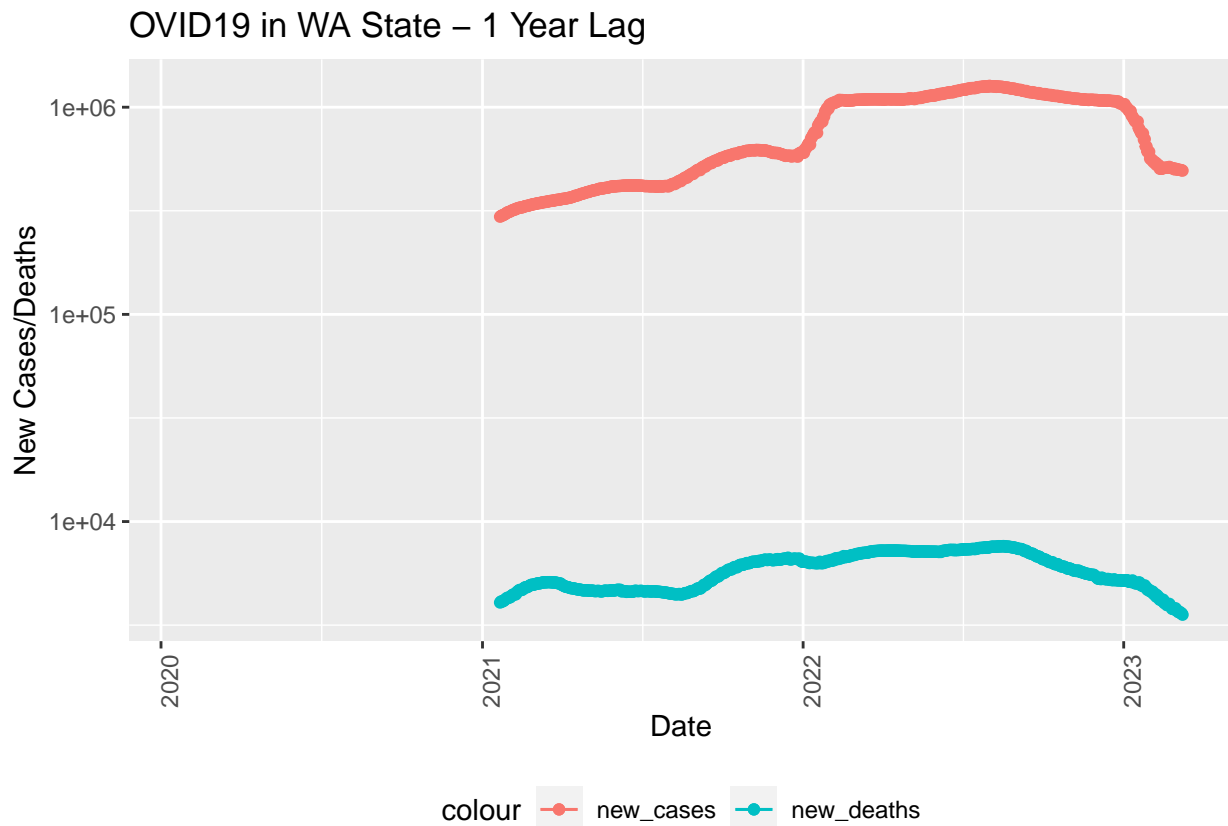


colour new_cases new_deaths

COVID19 in WA State – 1 Month Lag



colour new_cases new_deaths



- Let's proceed by grouping the counties and aggregating the number of cases. In addition, I'll apply a filter to exclude some data from Washington state, such as "Unassigned" and "Out of WA," as they seem to contain some missing data.

```
# Filter Washington state only, and remove "Unassigned" and "Out of WA" data.
WA_state <- US %>%
  filter(Province_State == "Washington") %>%
  filter(Admin2 != "Unassigned") %>%
  filter(Admin2 != "Out of WA") %>%
  mutate(new_cases = cases - lag(cases), new_deaths = deaths - lag(deaths), death_rate = deaths/cases)

# Check the data.
tail(WA_state)
```

```
## # A tibble: 6 x 11
##   Admin2 Province_State Country_Region Combined_Key date      cases Population
##   <chr>   <chr>          <chr>         <chr>    <date>    <dbl>      <dbl>
## 1 Yakima Washington      US          Yakima, Wash~ 2023-03-04 83734    250873
## 2 Yakima Washington      US          Yakima, Wash~ 2023-03-05 83734    250873
## 3 Yakima Washington      US          Yakima, Wash~ 2023-03-06 83734    250873
## 4 Yakima Washington      US          Yakima, Wash~ 2023-03-07 83734    250873
## 5 Yakima Washington      US          Yakima, Wash~ 2023-03-08 83734    250873
## 6 Yakima Washington      US          Yakima, Wash~ 2023-03-09 83734    250873
## # i 4 more variables: deaths <dbl>, new_cases <dbl>, new_deaths <dbl>,
## #   death_rate <dbl>
```

```
# Group by counties
WA_state_by_counties <- WA_state %>%
  group_by(Admin2) %>%
```

```

summarize(total_cases = sum(cases),
          total_deaths = sum(deaths),
          Population = sum(Population),
          ) %>%
ungroup()

# Print a summary of the data we have so far and make sure it's good
summary(WA_state)

```

```

##      Admin2      Province_State      Country_Region      Combined_Key
## Length:44577      Length:44577      Length:44577      Length:44577
## 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.   :    0  Min.   :   2225  Min.   :   0.0
## 1st Qu.:2020-11-02  1st Qu.:   437  1st Qu.:  22425  1st Qu.:   6.0
## Median :2021-08-15  Median :  3463  Median :  66768  Median :  42.0
## Mean   :2021-08-15  Mean   : 20702  Mean   : 195254  Mean   : 194.6
## 3rd Qu.:2022-05-28  3rd Qu.: 15105  3rd Qu.: 204390  3rd Qu.: 174.0
## Max.   :2023-03-09  Max.   :549865  Max.   :2252782  Max.   :3512.0
##
##      new_cases      new_deaths      death_rate
## Min.   : -549865.0  Min.   : -3512.000  Min.   : 0.0000
## 1st Qu.:    0.0    1st Qu.:    0.000  1st Qu.: 0.0080
## Median :    0.0    Median :    0.000  Median : 0.0113
## Mean   :    1.9    Mean   :    0.019  Mean   : 0.0146
## 3rd Qu.:   15.0    3rd Qu.:    0.000  3rd Qu.: 0.0152
## Max.   : 19214.0    Max.   :   45.000  Max.   : 2.0000
## NA's   :1          NA's   :1          NA's   :2230

```

```
summary(WA_state_by_counties)
```

```

##      Admin2      total_cases      total_deaths      Population
## Length:39      Min.   :   249039  Min.   :    935  Min.   :2.543e+06
## Class :character  1st Qu.:  2176342  1st Qu.:   27812  1st Qu.:2.566e+07
## Mode  :character  Median :  6996800  Median :   82129  Median :7.632e+07
##      Mean   : 23663006  Mean   :  222468  Mean   :2.232e+08
##      3rd Qu.: 21871165  3rd Qu.:  196412  3rd Qu.:1.906e+08
##      Max.   :248811536  Max.   : 2077113  Max.   :2.575e+09

```

- Let's create a visualization by graphing the population and the total number of cases per county. This will provide a clear visual representation of how the cases are distributed across different counties based on their respective populations.

First we need to pivot the data so we can group the Population and the total cases together

```

WA_state_by_counties_pivoted <- WA_state_by_counties %>%
  pivot_longer(cols=c('total_cases', 'Population'), names_to='variable',
               values_to="value")

```

```
WA_state_by_counties_pivoted
```

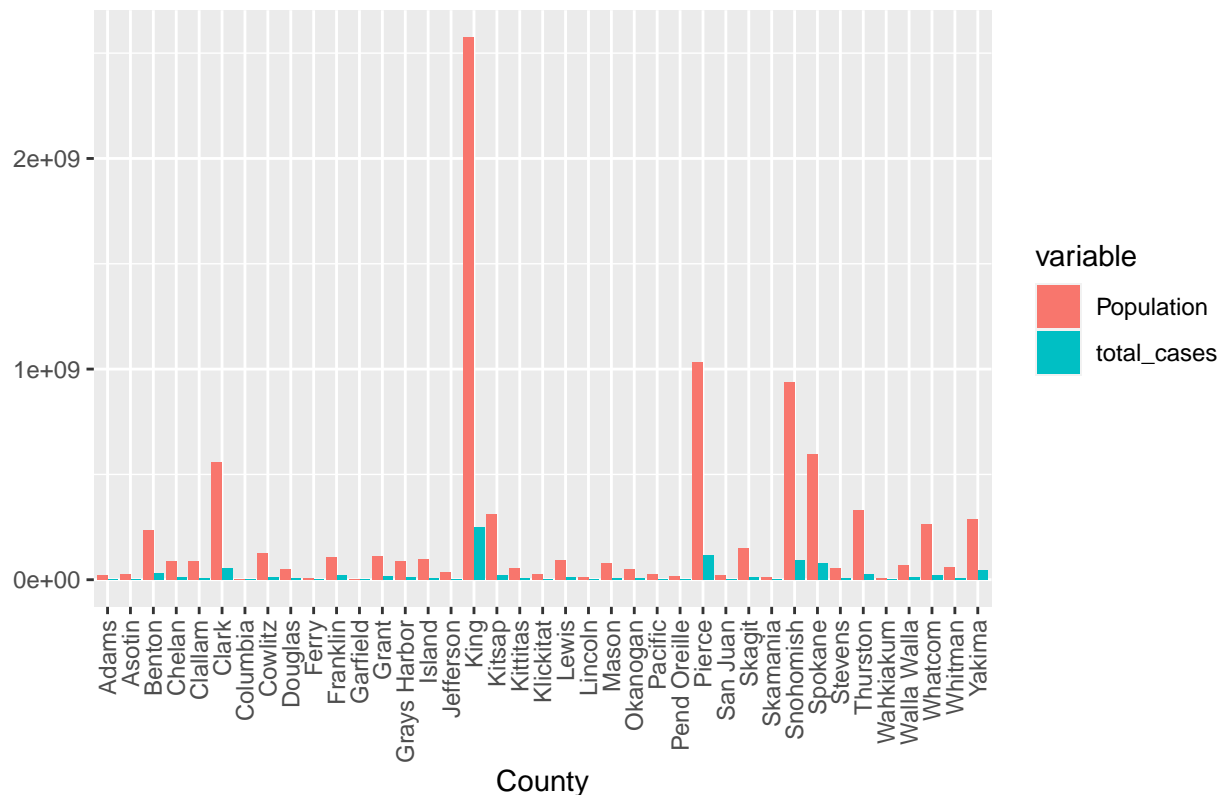
```
## # A tibble: 78 x 4
```

```
##   Admin2  total_deaths variable      value
##   <chr>      <dbl> <chr>      <dbl>
## 1 Adams      28237 total_cases  3293622
## 2 Adams      28237 Population  22840569
## 3 Asotin      44704 total_cases   2835680
## 4 Asotin      44704 Population  25811226
## 5 Benton     335603 total_cases  33040240
## 6 Benton     335603 Population  233617770
## 7 Chelan     107028 total_cases  12740435
## 8 Chelan     107028 Population  88239600
## 9 Clallam     78786 total_cases   6689251
##10 Clallam     78786 Population  88389333
## # i 68 more rows
```

```
# Graph both the total number of cases and deaths by county.
```

```
WA_state_by_counties_pivoted %>%
  ggplot(aes(fill=variable, x = Admin2, y = value)) +
  geom_bar(position="dodge", stat="identity") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  labs(title = "Cases/Deaths per county in Washington", x = "County", y = NULL)
```

Cases/Deaths per county in Washington



- Next, we will proceed to display the infection rate per county and highlight the top 3 counties. This visualization allows us to identify the counties with the highest infection rates and gain insights into the distribution of COVID-19 cases across different regions.

```
# Calculate the rate of infection
```

```
WA_state_by_counties <- WA_state_by_counties %>%
  mutate(infection_rate = total_cases * 100 / Population)
```

```
# Print the max infection rate to make sure mutate worked.
```

```
max(WA_state_by_counties$infection_rate)
```

```
## [1] 19.27053
```

```
WA_state_by_counties_pivoted <- WA_state_by_counties %>%  
  pivot_longer(cols=c('total_cases', 'infection_rate'), names_to='variable',  
    values_to="value")
```

```
WA_state_by_counties_pivoted
```

```
## # A tibble: 78 x 5
```

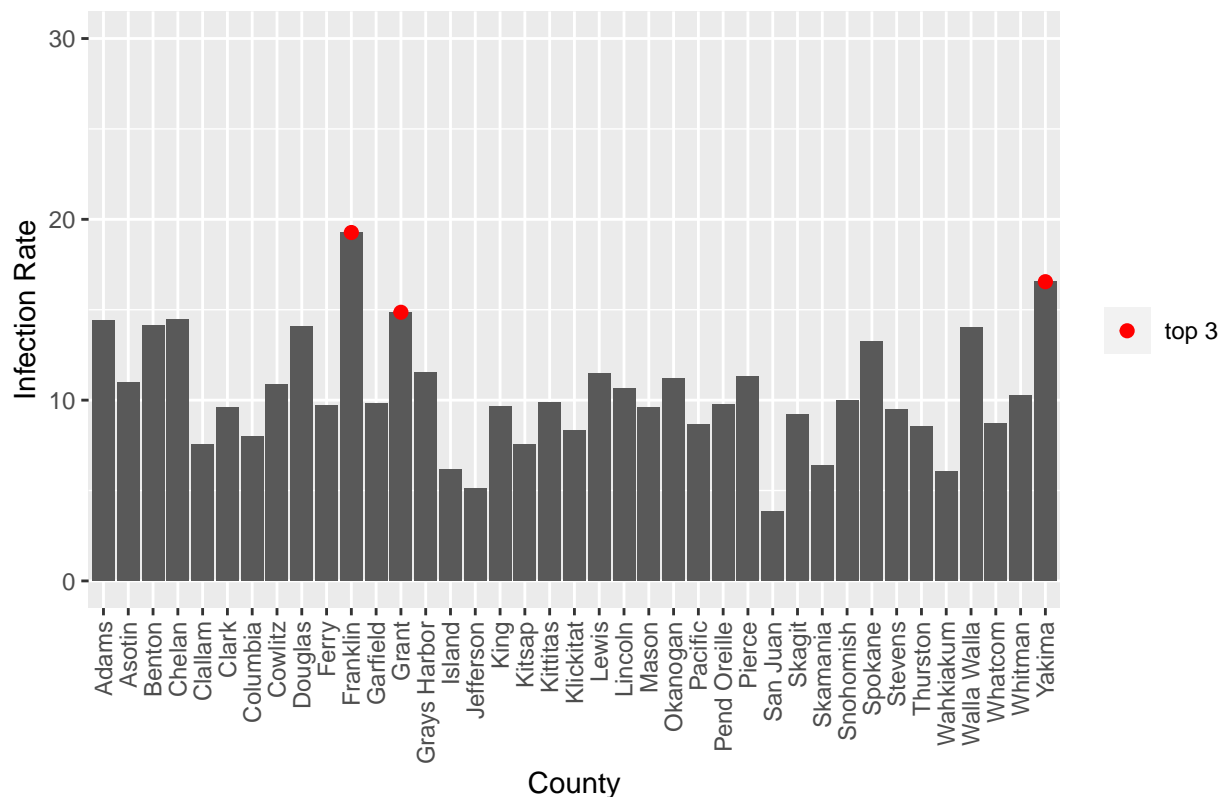
	Admin2	total_deaths	Population	variable	value
	<chr>	<dbl>	<dbl>	<chr>	<dbl>
##	1 Adams	28237	22840569	total_cases	3293622
##	2 Adams	28237	22840569	infection_rate	14.4
##	3 Asotin	44704	25811226	total_cases	2835680
##	4 Asotin	44704	25811226	infection_rate	11.0
##	5 Benton	335603	233617770	total_cases	33040240
##	6 Benton	335603	233617770	infection_rate	14.1
##	7 Chelan	107028	88239600	total_cases	12740435
##	8 Chelan	107028	88239600	infection_rate	14.4
##	9 Clallam	78786	88389333	total_cases	6689251
##	10 Clallam	78786	88389333	infection_rate	7.57

```
## # i 68 more rows
```

```
top_3_counties <- WA_state_by_counties %>%  
  arrange(desc(infection_rate)) %>%  
  top_n(3, infection_rate)
```

```
WA_state_by_counties %>%  
  arrange(WA_state_by_counties, desc(infection_rate)) %>%  
  ggplot(aes(x = Admin2, y = infection_rate)) +  
    geom_bar(stat="identity") +  
    geom_point(data = top_3_counties, aes(x = Admin2, y = infection_rate, color = "top 3"), size = 2) +  
    scale_color_manual(name = "", values = c("top 3" = "red")) +  
    theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +  
    ylim(0, 30)+  
    labs(title = "Infection Rate in WA State - Top 3 counties ", x = "County", y = "Infection Rate")
```

Infection Rate in WA State – Top 3 counties



- To predict the number of COVID-19 deaths based on the reported cases, we will employ a linear regression model. By utilizing this linear model, we can predict the deaths based on the cases.

Let's see if we can predict the deaths per thousand

```
WA_state <- US %>%
  filter(Province_State == "Washington", cases > 0, Population > 0) %>%
  group_by(Admin2) %>%
  summarize(deaths = max(deaths),
            cases = max(cases),
            Population = max(Population),
            cases_per_thousand = cases * 1000 / Population,
            deaths_per_thousand = deaths * 1000 / Population,
            ) %>%
  ungroup()
```

WA_state

```
## # A tibble: 39 x 6
##   Admin2  deaths  cases Population cases_per_thousand deaths_per_thousand
##   <chr>    <dbl> <dbl>    <dbl>          <dbl>          <dbl>
## 1 Adams      45   5647   19983          283.            2.25
## 2 Asotin     86   5606   22582          248.            3.81
## 3 Benton   539  62042  204390          304.            2.64
## 4 Chelan    196  24171   77200          313.            2.54
## 5 Clallam   209  15809   77331          204.            2.70
## 6 Clark    1047 111985  488241          229.            2.14
## 7 Columbia   18    751    3985          188.            4.52
## 8 Cowlitz   424  27341  110593          247.            3.83
```

```
## 9 Douglas      87 13333      43429      307.      2.00
## 10 Ferry       34  1825       7627      239.      4.46
## # i 29 more rows
```

```
# let's model the data
```

```
model <- lm(deaths_per_thousand ~ cases_per_thousand, data = WA_state)
```

```
summary(model)
```

```
##
## Call:
## lm(formula = deaths_per_thousand ~ cases_per_thousand, data = WA_state)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.6549 -0.6804 -0.2842  0.6227  2.2245
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.262704   0.625170   2.020  0.0507 .
## cases_per_thousand 0.005464   0.002549   2.144  0.0387 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9021 on 37 degrees of freedom
## Multiple R-squared:  0.1105, Adjusted R-squared:  0.08644
## F-statistic: 4.595 on 1 and 37 DF,  p-value: 0.0387
```

```
WA_state_with_predictions <- WA_state %>% mutate(pred=predict(model))
```

```
WA_state_with_predictions
```

```
## # A tibble: 39 x 7
##   Admin2 deaths cases Population cases_per_thousand deaths_per_thousand pred
##   <chr>   <dbl> <dbl>      <dbl>          <dbl>          <dbl> <dbl>
## 1 Adams      45   5647    19983          283.          2.25  2.81
## 2 Asotin     86   5606    22582          248.          3.81  2.62
## 3 Benton    539  62042   204390          304.          2.64  2.92
## 4 Chelan    196  24171   77200          313.          2.54  2.97
## 5 Clallam   209  15809   77331          204.          2.70  2.38
## 6 Clark    1047 111985  488241          229.          2.14  2.52
## 7 Columb~   18   751     3985          188.          4.52  2.29
## 8 Cowlitz   424  27341  110593          247.          3.83  2.61
## 9 Douglas   87  13333   43429          307.          2.00  2.94
## 10 Ferry    34   1825    7627          239.          4.46  2.57
## # i 29 more rows
```

```
WA_state_with_predictions %>%
```

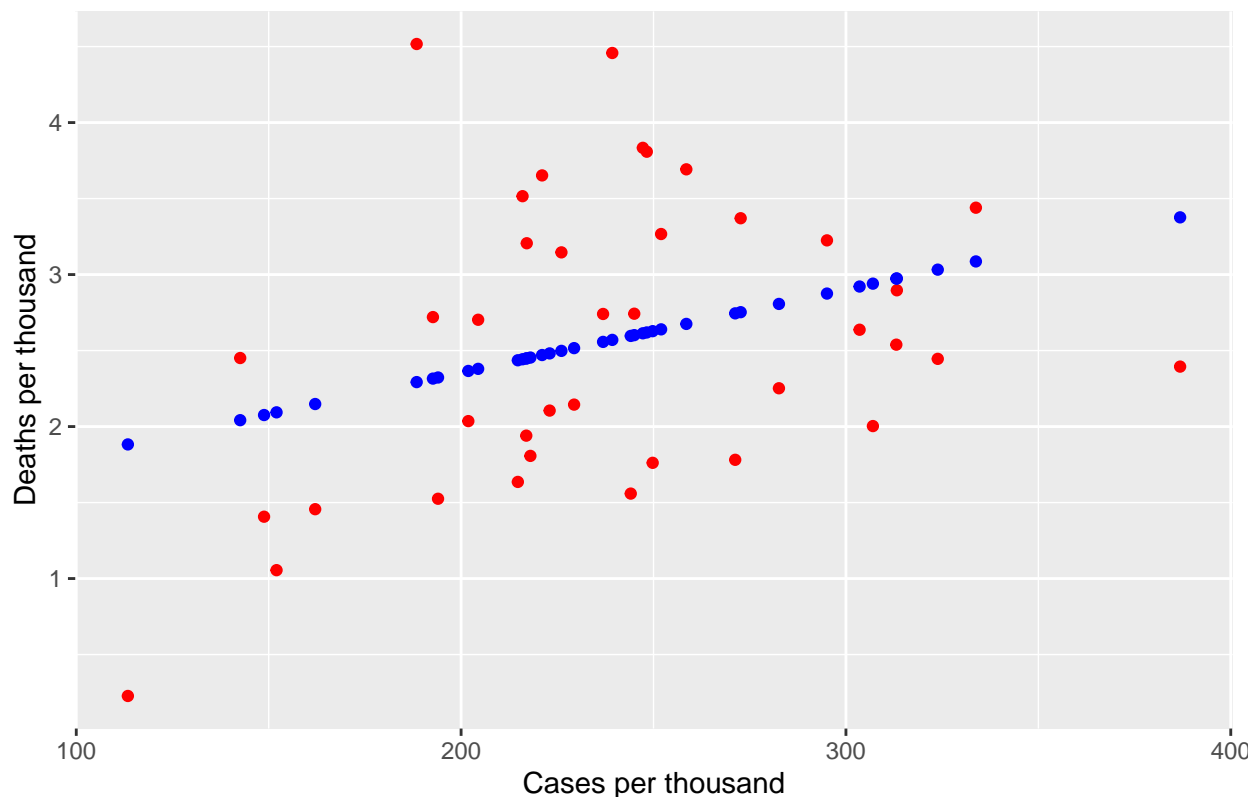
```
ggplot() +
```

```
  geom_point(aes(x = cases_per_thousand, y = deaths_per_thousand), color = "red") +
```

```
  geom_point(aes(x = cases_per_thousand, y = pred), color = "blue") +
```

```
  labs(title = "Linear Model - WA State - Predicting deaths per thousand", x = "Cases per thousand", y = "Deaths per thousand")
```


Linear Model – WA State – Predicting deaths per thousand



Conclusion

- Based on the infection rate, the top 3 counties are Franklin, Grant, and Yakima.
- The number of deaths shows a downward trend. It would be valuable to incorporate additional data such as vaccination records to evaluate the potential positive impact of vaccines.
- Although the linear regression model fits the data, a more appropriate model, such as logistic regression, may better capture the underlying patterns.

Bias

1- It is crucial to approach the analysis of such data/reports with caution, as they can be subject to various biases. Questions arise regarding the data collection process, such as who is responsible for gathering the data and whether there are established data compliance protocols in place. Are the reported deaths accurately documented across all counties? Additionally, we must consider the accuracy of data entry and potential inconsistencies. If certain groups, counties, or regions are systematically favored in the data collection process, sampling bias can be introduced. Similarly, selective reporting of data can introduce reporting bias, impacting the overall analysis.

2- Another important consideration is to focus on rates rather than absolute numbers. Comparing the number of cases or deaths between different regions or counties without considering the infection or death rates can lead to erroneous conclusions. It is essential to assess the rate of infection and/or deaths to make informed decisions and avoid misinterpretations based solely on the raw numbers.

3- A personal bias might also affect how we analyze the data, for example, my political beliefs might influence my analysis, another one is confirmation bias, where we selectively focus on information that aligns with our beliefs.

We need to be aware of our biases so that we can avoid them and provide an objective analysis.