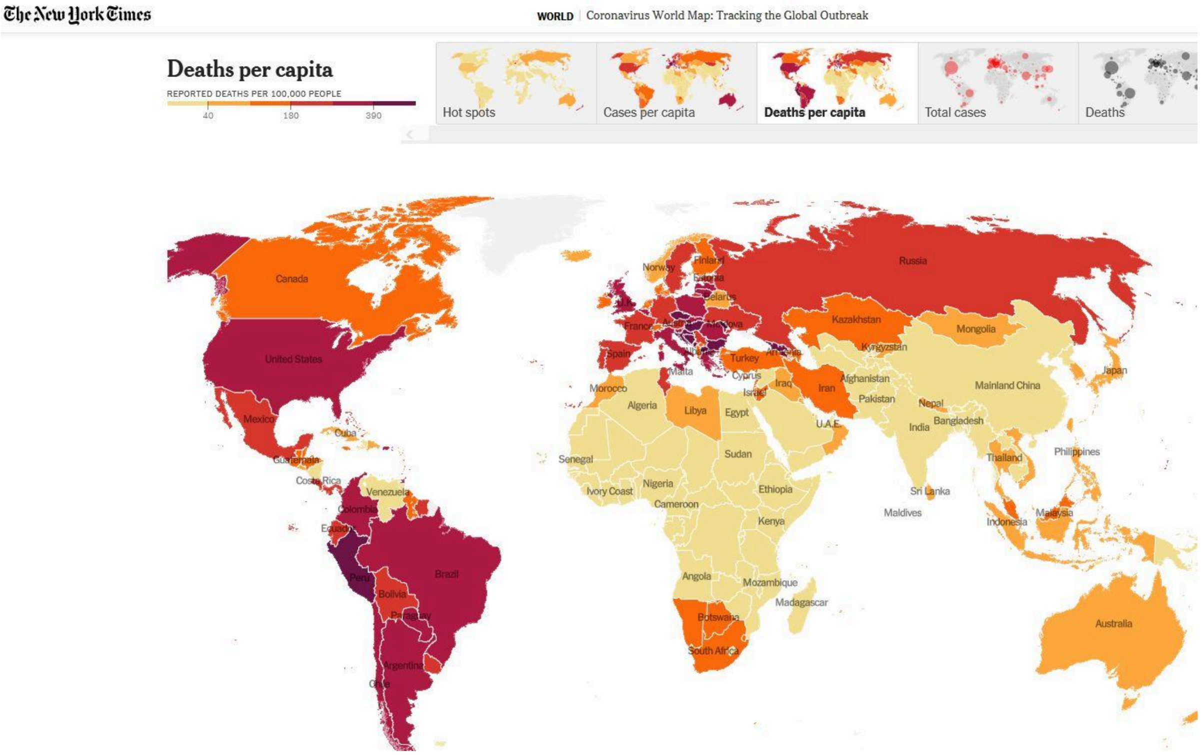
CPSC 375 Spring 2024

Mariia Nikitash, Anthony Weathersby, Abe Oueichek

Project Requirements



Though the public health emergency from the COVID-19 pandemic has ended, one question has not been adequately answered - why did countries have widely different rates of death from COVID? Only a small proportion of COVID cases lead to death, typically after a few weeks. However, the daily number of deaths in a country does not depend only on the number of cases. It also varies with several other factors including the extent of vaccination, the level of development (e.g., countries with more hospitals should see fewer deaths), age demographics (countries with a larger proportion of older people should see more deaths), pre-existing differences in medical conditions such as diabetes. The goal of this project is to use linear modeling to **predict two weeks ahead** the number of daily COVID deaths in different countries using a range of factors.

Datasets

1. Data on COVID-19 from *Our World in Data*
   Their complete dataset contains a lot of information including the number of deaths, cases, vaccinations, hospitalizations, and several other country-specific pieces of information relevant to understanding the effects of COVID. Note that you can read the "raw" CSV file from a URL directly, like so:
   ```
   read_csv("https://raw.githubusercontent.com/owid/covid-19-data/
   master/public/data/owid-covid-data.csv")
   ```
2. Population estimates from The World Bank's DataBank

Use the above web page to

a. Select all countries
b. Select a few Series (variables) that you think will be relevant to predicting death from COVID. For example, populations in certain age groups, mortality rates, and expected lifetime. At the very least, select *Population ages 80 and above, female* and *Population ages 80 and above, male*.
c. Select time: Only 2023
d. Download your selection as a CSV file (you get a .zip file, which contains the .csv file; delete the last few lines of the csv file which has the license information)

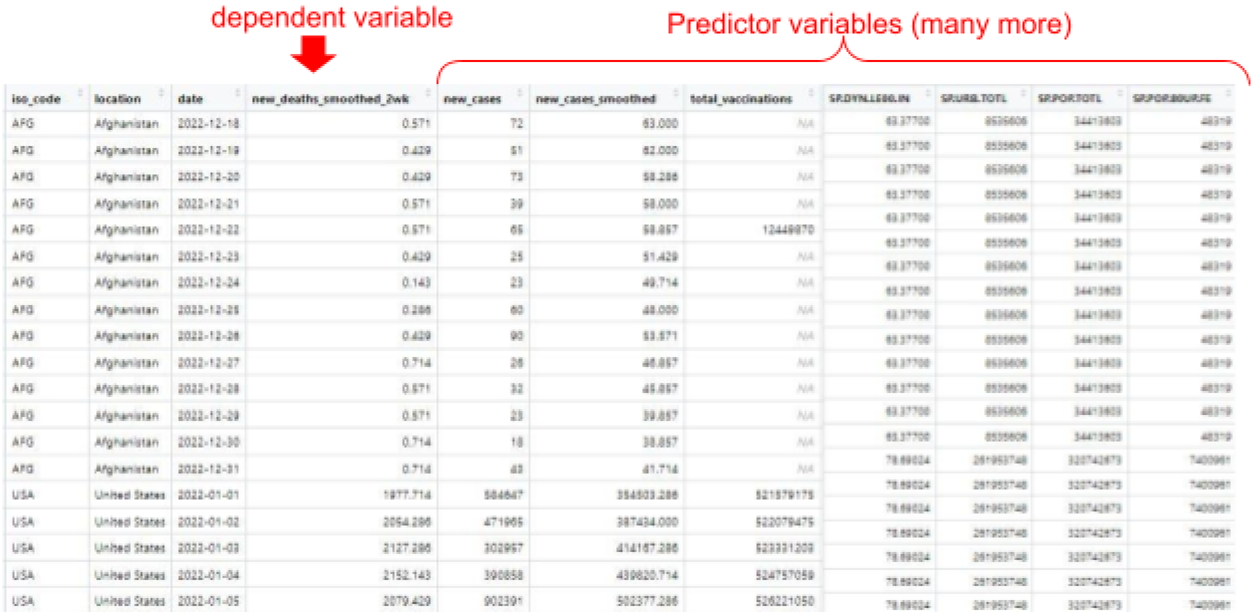
Approach

There are three steps in this project:

1) **Data wrangling** to get all the data into **one table** that can be used for linear modeling
    a) read the two data files using `read_csv()`
    b) Keep only country-level data by removing all rows where the country_code is not exactly 3 letters (these represent larger regions like continents). Hint: `nchar(string)` returns the number of characters.
    c) Remove countries whose total population is less than 1 million.
    d) Add a new column `new_deaths_smoothed_2wk` that has the same values as `new_deaths_smoothed` *but two weeks ahead* (will be used for linear modeling as described later). R has a `Date` type that enables calculations with dates like `mutate(date= date - 14)` and `filter(date >= as.Date("2023-01-01"))`.
    e) tidy tables, as needed. (Hint: only the population data is not tidy.)
    f) Merge the tables (Hint: `join` using the 3-letter ISO code)

```
#Load Required Libraries
library(tidyverse)
library(modelr)
library(ggplot2)

#Start of Part 1: Data Wrangling
#Load and wrangle the Covid Data data table
CovidData <-
read_csv("https://raw.githubusercontent.com/owid/covid-19-data/master/public/da
ta/owid-covid-data.csv") %>%
  filter(nchar(iso_code) == 3) %>%
  mutate(date = as.Date(date)) %>%
  group_by(`iso_code`) %>%
  mutate(new_deaths_smoothed_2wk = lead(new_deaths_smoothed, 14)) %>%
  filter(date >= as.Date("2022-01-01") & date <= as.Date("2023-12-31"))

#Load and wrangle the Series Info data table
SeriesInfo <- read_csv("covid2.csv") %>%
  pivot_wider(names_from = `Series Name`, values_from = `2023 [YR2023]`) %>%
  mutate(`Population ages 80 and above, female` = as.integer(`Population ages 80 and above, female`)) %>%
  mutate(`Population ages 80 and above, male` = as.integer(`Population ages 80 and above, male`)) %>%
  select(-`Series Code`) %>%
  group_by(`Country Name`, `Country Code`) %>% summarize(across(1:last_col(), ~ first(na.omit(.x)))) %>% mutate('Total Population' = `Population ages 80 and above, female` + `Population ages 80 and above, male`) %>%
  filter(`Total Population` >= 1000000) %>%
  rename('iso_code' = 'Country Code')

#Joining the two data tables together
CombinedData <- CovidData %>%
  left_join(SeriesInfo %>% select(iso_code, `Population ages 80 and above, female`,      `Population ages 80 and above, male`), by = "iso_code") %>%
  filter(date >= as.Date("2022-01-01") & date <= as.Date("2023-12-31"))

View(CombinedData)
View(CovidData)
View(SeriesInfo)
#End of Part 1: Data Wrangling
```

At the end of these steps, the data should be in one table, ready for linear regression (only a small sample of the data is shown below):


dependent variable
Predictor variables (many more)

iso_code	location	date	new_deaths_smoothed_2wk	new_cases	new_cases_smoothed	total_vaccinations
AFG	Afghanistan	2022-12-18	0.571	72	63.000	NA
AFG	Afghanistan	2022-12-19	0.429	51	62.000	NA
AFG	Afghanistan	2022-12-20	0.429	73	58.286	NA
AFG	Afghanistan	2022-12-21	0.571	39	58.000	NA
AFG	Afghanistan	2022-12-22	0.571	65	58.857	12448970
AFG	Afghanistan	2022-12-23	0.429	25	51.429	NA
AFG	Afghanistan	2022-12-24	0.143	23	49.714	NA
AFG	Afghanistan	2022-12-25	0.286	60	48.000	NA
AFG	Afghanistan	2022-12-26	0.429	90	53.571	NA
AFG	Afghanistan	2022-12-27	0.714	26	46.857	NA
AFG	Afghanistan	2022-12-28	0.571	32	45.857	NA
AFG	Afghanistan	2022-12-29	0.571	23	39.857	NA
AFG	Afghanistan	2022-12-30	0.714	18	38.857	NA
AFG	Afghanistan	2022-12-31	0.714	43	41.714	NA
USA	United States	2022-01-01	1977.714	584847	354503.286	521579175
USA	United States	2022-01-02	2054.286	471965	387434.000	522079475
USA	United States	2022-01-03	2127.286	302957	414167.286	523331203
USA	United States	2022-01-04	2152.143	390858	439820.714	524757059
USA	United States	2022-01-05	2079.429	902391	502377.286	526221050

SP.DYN.LE00.IN	SP.URB.TOTL	SP.POP.TOTL	SP.POP.80UP.FE
63.27700	8535606	34413603	48319
63.27700	8535606	34413603	48319
63.27700	8535606	34413603	48319
63.27700	8535606	34413603	48319
63.27700	8535606	34413603	48319
63.27700	8535606	34413603	48319
63.27700	8535606	34413603	48319
63.27700	8535606	34413603	48319
63.27700	8535606	34413603	48319
63.27700	8535606	34413603	48319
63.27700	8535606	34413603	48319
63.27700	8535606	34413603	48319
63.27700	8535606	34413603	48319
63.27700	8535606	34413603	48319
78.69024	261953748	320742673	7400981
78.69024	261953748	320742673	7400981
78.69024	261953748	320742673	7400981
78.69024	261953748	320742673	7400981
78.69024	261953748	320742673	7400981
78.69024	261953748	320742673	7400981
78.69024	261953748	320742673	7400981

2) **Linear modeling**
   The goal is to predict `new_deaths_smoothed` *two weeks in the future*. Hint: this is the dependent variable.
   a) Make a list of all predictor variables that are available. The challenge is to identify which combination of these predictors will give the best predictive model.

Gdp_per_capita
Hospital_beds_per_thousand

Total_vaccinations_per_hundred
People_fully_vaccinated_per_hundred
`Population ages 80 and above, female`
`Population ages 80 and above, male`

stringency_index

   b) Generate some (at least 3) transformed variables. E.g., these could combine variables (e.g., `cardiovasc_deaths= cardiovasc_death_rate*population`).

```
# 2b. 3 Transformed Variables
CombinedData <- CombinedData %>%
  mutate(
    gdp_vaccination_interaction = gdp_per_capita * people_fully_vaccinated_per_hundred,
    beds_stringency_interaction = hospital_beds_per_thousand * stringency_index,
    total_population_80_above = `Population ages 80 and above, female` + `Population ages 80 and above, male`,
    elderly_vaccination_interaction = total_population_80_above * total_vaccinations_per_hundred)
```

c) Split your dataset into train and test subsets: only data from 2022 should be used for building/training the linear models in `lm()`. (Data from 2023 will be used for evaluation as described later). Note: ***each day*** is one data point.

```
# 2c. Split Test and Train Data
train_data <- filter(CombinedData, date < as.Date("2023-01-01"))
test_data <- filter(CombinedData, date >= as.Date("2023-01-01"))
%>%
  filter(date >= as.Date("2023-01-01") & date <=
as.Date("2023-06-30"))
```

d) Run linear regression with **at least 5 different combinations of predictor variables.** Hint: each model will look like:
`new_deaths_smoothed_2wk~new_cases_smoothed+gdp_per_capita+diabetes_prevalence+icu_patients+SP.URB.TOTL`

```
# 2d. Model Creation for Linear Regression
model1 <- lm(new_deaths_smoothed_2wk ~
beds_stringency_interaction + gdp_vaccination_interaction, data
= train_data)
model2 <- lm(new_deaths_smoothed_2wk ~
beds_stringency_interaction + gdp_vaccination_interaction +
total_population_80_above, data = train_data)
model3 <- lm(new_deaths_smoothed_2wk ~
elderly_vaccination_interaction +
people_fully_vaccinated_per_hundred, data = train_data)
model4 <- lm(new_deaths_smoothed_2wk ~ new_cases_smoothed
+ beds_stringency_interaction + elderly_vaccination_interaction
+ hospital_beds_per_thousand, data = train_data)
model5 <- lm(new_deaths_smoothed_2wk ~
gdp_vaccination_interaction + total_population_80_above +
gdp_per_capita + aged_65_older, data = train_data)


View(test_data)
View(train_data)
#End of Part 2: Linear Modeling
```

3) **Evaluating the linear models**
   You should evaluate each of your linear models by predicting the number of daily deaths in each day in January-June 2023 (the test data) and comparing it with the actual number of deaths on those days. Specifically:
   a) For each of your models, calculate the Root Mean Squared Error (RMSE) over all days in January-June 2023 and all countries. Hint: use `rmse()` in library(modelr).

```
#Start of Part 3: Evaluating the Linear Models
#3a.
rmse_model1 <- rmse(model1, test_data)
rmse_model2 <- rmse(model2, test_data)
rmse_model3 <- rmse(model3, test_data)
rmse_model4 <- rmse(model4, test_data)
rmse_model5 <- rmse(model5, test_data)

rmse_data <- tibble(
  model = c("Model 1", "Model 2", "Model 3", "Model 4", "Model 5"),
  rmse = c(rmse_model1, rmse_model2, rmse_model3, rmse_model4,
rmse_model5))
View(rmse_data)

test_data <- ungroup(test_data) %>%
  mutate(
    pred_model1 = predict(model1, newdata = .),
    pred_model2 = predict(model2, newdata = .),
    pred_model3 = predict(model3, newdata = .),
    pred_model4 = predict(model4, newdata = .),
    pred_model5 = predict(model5, newdata = .)
  )

daily_summary <- test_data %>%
  select(date, iso_code, new_deaths_smoothed_2wk, pred_model1,
pred_model2, pred_model3, pred_model4, pred_model5) %>%
  pivot_longer(
    cols = starts_with("pred"),
    names_to = "model",
    values_to = "predicted_deaths",
    names_prefix = "pred_") %>%
  mutate(
    error = predicted_deaths - new_deaths_smoothed_2wk)

rmse_per_country_model <- daily_summary %>%
  group_by(iso_code, model) %>%
  summarise(
    daily_rmse = sqrt(mean(error^2, na.rm = TRUE)),
    .groups = 'drop')
daily_summary <- na.omit(daily_summary) %>% filter(error >= 0)
View(daily_summary)
```

b) For only your best model, calculate the Root Mean Squared Error for **every country**. Hint: use `group_by()` and `summarise(rmse(model= my_best_model, data=cur_data()))`. `cur_data()` gives the data in each group.

```
#3b. Model 5 seems to be the best
test_data <- test_data %>%
  mutate(pred_model5 = predict(model5, newdata = .))

# Calculate RMSE by country for model5
rmse_by_country_model5 <- test_data %>%
  group_by(iso_code) %>%
  summarise(
    actual = new_deaths_smoothed_2wk,
    predicted = pred_model5,
    rmse = sqrt(mean((actual - predicted)^2, na.rm = TRUE)),
    .groups = 'drop')
rmse_by_country_model5 <- na.omit(rmse_by_country_model5)
View(rmse_by_country_model5)

#End of Part 3: Evaluating the Linear Models
```

Group work

You may work in groups of 1-3. Include all group member names in the PDF reports.

Stage 2 (Final submission) Due: Friday, May 3

To submit:

1. A short report describing your work. Specifically, your report should include:

- brief description of only the important data wrangling steps

  In order to complete this assignment, we began by loading the necessary libraries. The tidyverse library is indispensable for data cleaning and transformation, enabling us to shape our data into a usable format for visualization, summarization, and modeling. Additionally, we utilized the modelr library, primarily for its capability to calculate the root mean squared error (RMSE) for our models. This is crucial for evaluating the predictive performance of our models accurately. Lastly, the ggplot2 library was used to create insightful visualizations of our data, aiding in the exploration and communication of trends and patterns.

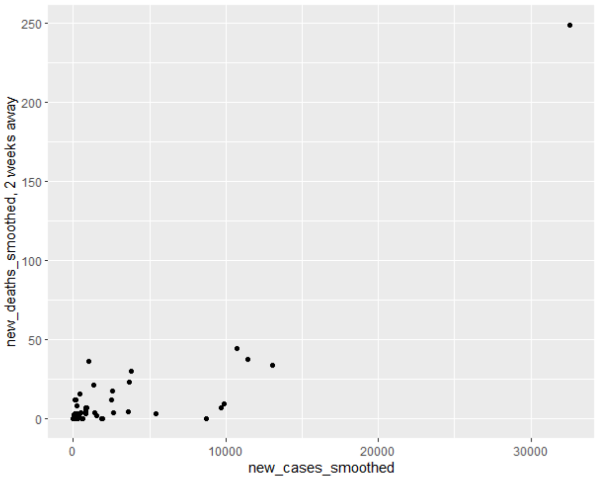
  Moving forward, we proceeded with loading and wrangling the Covid Data and Series Info data tables. The Covid Data, sourced from an online repository, contained information regarding COVID-19 cases and related metrics. Through a series of transformations including filtering, date conversion, and calculation of a 2-week smoothed average of new deaths, we prepared the data for further analysis. Similarly, the Series Info data, obtained from a local CSV file, provided additional demographic insights such as the population aged 80 and above. After consolidating and refining these datasets, we culminated the data wrangling process by joining the two tables together based on the ISO code, ensuring a comprehensive dataset for our analysis.

  In conclusion, the data wrangling phase laid the foundation for our analysis by ensuring that our data is clean, relevant, and structured appropriately.

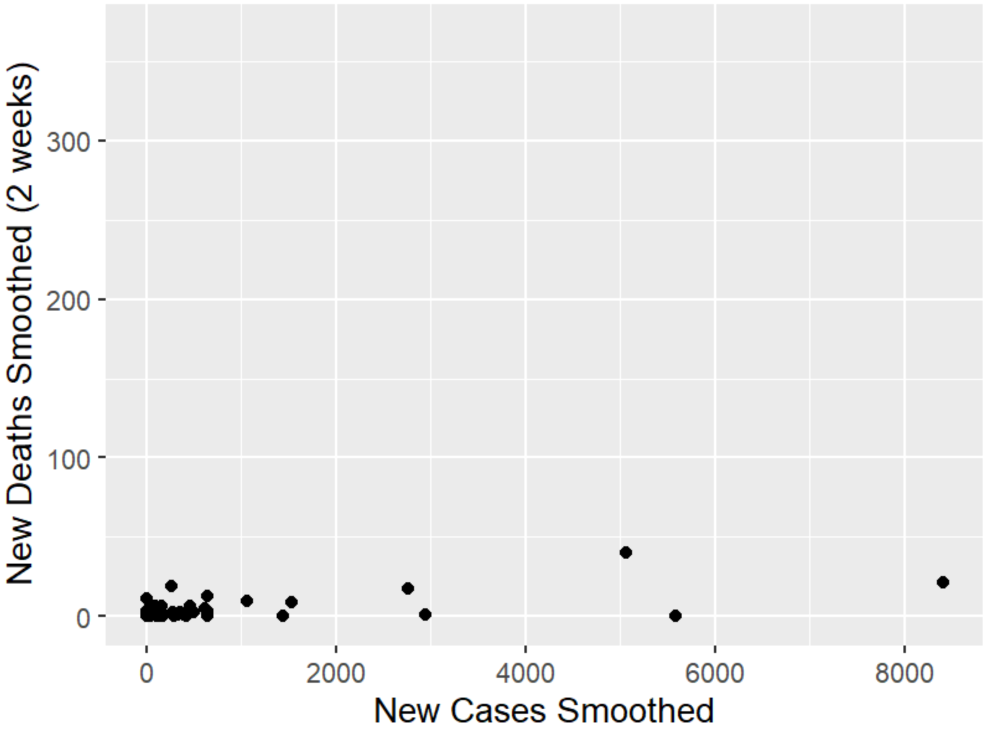
- List of the variables ("series") that you selected from the Population estimates webpage,

  Population ages 80 and above, female
  Population ages 80 and above, male

- a scatterplot of only the most recent new deaths per day two weeks ahead (new_deaths_smoothed_2wk) in the test dataset (i.e., 2023-06-30) and the corresponding new cases per day (new_cases_smoothed) for every country (i.e., one point per country), like so:

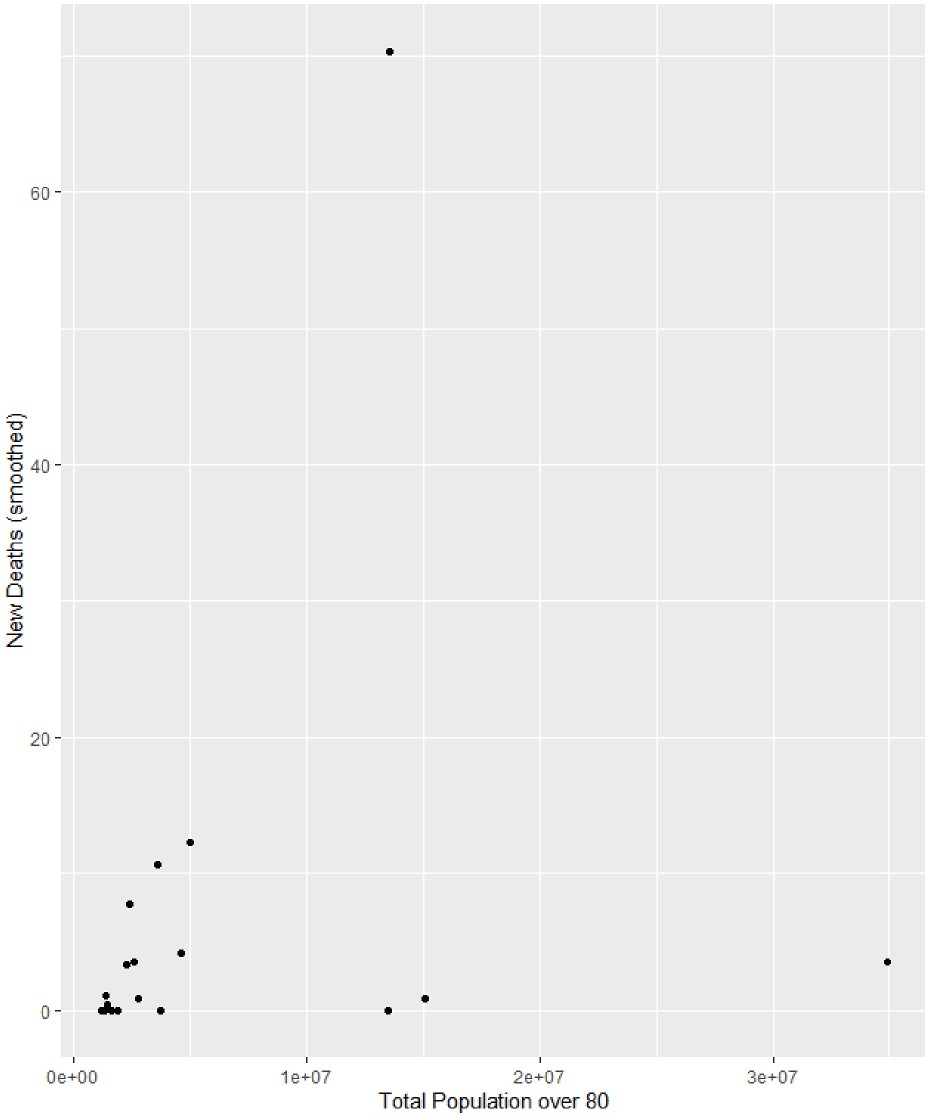


```
recent_data <- CombinedData %>%
    group_by(iso_code) %>%
    slice_max(order_by = date) %>%
    ungroup()

ggplot(recent_data, aes(x = new_cases_smoothed, y =
new_deaths_smoothed_2wk)) +
    geom_point(aes()) +
    labs(title = "Most Recent COVID-19 Deaths vs. Cases by Country",
         x = "New Cases Smoothed",
         y = "New Deaths Smoothed (2 weeks)")
```


New Deaths Smoothed (2 weeks)
300
200
100
0
0
2000
4000
6000
8000
New Cases Smoothed

- a scatterplot of only the most recent new deaths (new_deaths_smoothed) in the test dataset (i.e., 2023-06-30) and the total (female+male) population over 80 for every country (i.e., one point per country), like so:

```
> test_data_june30 <- filter(test_data, date ==
as.Date("2023-06-30"))
> ggplot(test_data_june30, aes(x = total_population_80_above, y =
new_deaths_smoothed_2wk)) +
+     geom_point() +
+     labs(x = "Total Population over 80", y = "New Deaths
(smoothed)")
```



- descriptions of variable transforms,
- list of the different combinations of predictor variables in your models,

Model 1:

- Dependent variable: `new_deaths_smoothed_2wk`

- Independent variables: `beds_stringency_interaction`, `gdp_vaccination_interaction`

- R code:

```
model1 <- lm(new_deaths_smoothed_2wk ~ beds_stringency_interaction + gdp_vaccination_interaction, data = train_data)
```

Model 2:

- Dependent variable: `new_deaths_smoothed_2wk`

- Independent variables: `beds_stringency_interaction`, `gdp_vaccination_interaction`, `total_population_80_above`

- R code:

```
model2 <- lm(new_deaths_smoothed_2wk ~ beds_stringency_interaction + gdp_vaccination_interaction + total_population_80_above, data = train_data)
```

Model 3:

- Dependent variable: `new_deaths_smoothed_2wk`

- Independent variables: `elderly_vaccination_interaction`, `people_fully_vaccinated_per_hundred`

- R code:

```
model3 <- lm(new_deaths_smoothed_2wk ~ elderly_vaccination_interaction + people_fully_vaccinated_per_hundred, data = train_data)
```

Model 4:

- Dependent variable: `new_deaths_smoothed_2wk`

- Independent variables: `new_cases_smoothed`, `beds_stringency_interaction`, `elderly_vaccination_interaction`, `hospital_beds_per_thousand`

- R code:

```
        model4 <- lm(new_deaths_smoothed_2wk ~ new_cases_smoothed +
beds_stringency_interaction + elderly_vaccination_interaction +
hospital_beds_per_thousand, data = train_data)
```

Model 5:

- Dependent variable: `new_deaths_smoothed_2wk`

- Independent variables: `gdp_vaccination_interaction`,`total_population_80_above`, `gdp_per_capita`, `aged_65_older`

- R code:

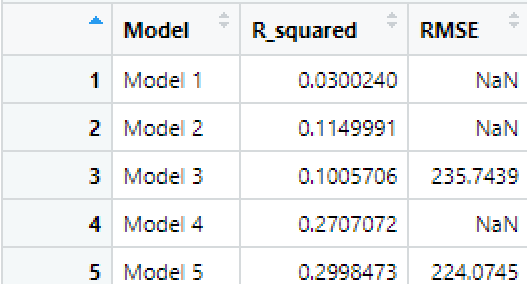
```
        model5 <- lm(new_deaths_smoothed_2wk ~ gdp_vaccination_interaction
+ total_population_80_above + gdp_per_capita + aged_65_older, data =
train_data)
```

○ brief reasons for *why* you chose these predictor variables (e.g., your prior knowledge, or a plot showed a trend),

GDP per Capita: Economic status often reflects healthcare infrastructure and resources.
Hospital Beds per Thousand: Availability of beds impacts healthcare capacity during surges.
Total Vaccinations per Hundred: Indicates vaccination coverage in the population.

People Fully Vaccinated per Hundred: Reflects immunity levels in the population.
Population Ages 80 and Above (Female and Male): Older adults are at higher risk of severe illness.
Stringency Index: Measures government interventions' impact on disease spread.

- a table listing the R2 and RMSE of **all** your models
-
```
> model_performance <- tibble(
    Model = c("Model 1", "Model 2", "Model 3", "Model 4", "Model
5"), R_squared = c(summary(model1)$r.squared,
summary(model2)$r.squared, summary(model3)$r.squared,
summary(model4)$r.squared, summary(model5)$r.squared), RMSE =
c(rmse_model1, rmse_model2, rmse_model3, rmse_model4, rmse_model5))
```



	Model	R_squared	RMSE
1	Model 1	0.0300240	NaN
2	Model 2	0.1149991	NaN
3	Model 3	0.1005706	235.7439
4	Model 4	0.2707072	NaN
5	Model 5	0.2998473	224.0745

- a table showing the RMSE of only your best model for the 20 most populous countries arranged in decreasing order of population, like so:

```
> top_20_populous_countries <- CovidData %>% filter(!is.na(population)) %>%
select(iso_code, population) %>% distinct() %>% arrange(desc(population))
%>% head(20)
> top_20_countries_rmse <- rmse_by_country_model5 %>% filter(iso_code %in%
top_20_populous_countries$iso_code) %>% arrange(match(iso_code,
top_20_populous_countries$iso_code))
> top_20_countries_rmse
# A tibble: 730 × 4
   iso_code   actual predicted  rmse
   <chr>       <dbl>     <dbl> <dbl>
 1 CHN         0          392. 2887.
 2 CHN         0          392. 2887.
 3 CHN      6812.         392. 2887.
 4 CHN       194.         392. 2887.
 5 CHN        76.7        392. 2887.
 6 IND         0.857      174.  168.
 7 IND         0.857      174.  168.
 8 IND         0.857      174.  168.
 9 IND         0.857      174.  168.
10 IND         0.857      174.  168.
# ℹ 720 more rows
# ℹ Use `print(n = ...)` to see more rows
```

- a conclusion that describes in words the implication of your most accurate model.

Based on our analysis, Model 5 stands out as the most accurate, with the highest R-squared value and the lowest RMSE. This model, incorporating GDP per capita, vaccination metrics, and demographic factors, provides valuable insights into the factors influencing COVID-19 mortality rates. The low RMSE values for the top 20 most populous countries indicate the validity of the model in predicting new deaths smoothed over a 2-week period. Overall, the findings from Model 5 underscore the importance of considering a combination of socioeconomic, healthcare, and demographic factors in understanding and predicting COVID-19 outcomes.

2. A listing of your R code in one file [.R file]

Project checklist/grading rubric

1. Draft submission (approximately 10% of total grade)
    a. Data wrangling is at least partially complete
    b. Brief report of completed steps
    c. Group member names are included in the report
    d. R code for completed data wrangling
    e. Submission on time
2. Data wrangling (final)
    a. Code to load and wrangle OWID data
    b. Code to load and wrangle demographics data
    c. Code to join datasets to one table
3. Modeling:
    a. Tried at least 5 different combinations of variables for modeling
    b. Included at least 3 variable transformations
    c. Code that correctly implements the above
4. Evaluation:
    a. Generate the R2 and RMSE of all models
    b. Identified the best model and calculated its RMSE for all countries
    c. Code that correctly implements the above
    d. Note: having a high R2/low RMSE is *not* important for grading
5. Written report (final)
    a. Brief descriptions of the data wrangling steps
    b. Brief description of how variables were chosen for data modeling
    c. Descriptions of variable transformations
    d. Scatterplot of only the most recently available new_deaths_smoothed_2wk and new_cases_smoothed for every country
    e. Scatterplot of only the most recent new deaths per day and the urban population
    f. A table that shows the R2 and RMSE of the different models
    g. A table that shows the RMSE of the best model for 20 most populous countries
    h. A conclusion – what does your modeling say about death rates (e.g., what are the significant factors and what are not)
    i. Clarity of the report (e.g., appropriate section headings)
6. Code
    a. Readability: use of `tidyverse,` no unnecessary use of complex functions.
    b. Code has adequate comments

c. Note: include only the final code, i.e., do not submit just the RStudio history