HarvardX Data Science Capstone Covid Data Project

Weichen Zhou

1/8/2021

Introduction

This is the second final project in the HarvardX Data Science course series on Edx. The pandemic in 2020 has affected so much of our lives and I want to take this opportunity to explore data about number of Covid-19 cases around the world and if there are factor that affects the spread of the virus.

I did a little bit of research on the data and found the data set: "https://www.kaggle.com/shashwatwork/impact-of-covid19-pandemic-on-the-global-economy?select=raw_data.csv", credit to Shashwat Tiwari on Kaggle.

In this project, my goal is to see if I can come up with a model that consider possible factors then predict the number of Covid-19 cases. I don't have an answer to this question and I'm hoping to find out with the aforementioned dataset.

I also wanted to include continent information to the data, which is not included in the dataset above. So I found this data: "https://www.kaggle.com/statchaitya/country-to-continent?select=countryContinent.csv", credit to Chaitanya Gokhale on Kaggle.

Overview

Here, I will give a quick overview of this report. I will start by botaining the dataset and manipulating it to the format that will work the best for the analysis. Then I split the dataset into two piece, one is the training set and the other is the testing set called "validation". Some tailoring will be done at the beginning to both sets to prepare them for future analysis, then I will hold the validation set, which contains 10% of total data on hold until the final test stage. The training set will be the set I work on to "train" the model.

Then, I will extract information from the training set and make educated guesses on what could be the possible predictors. After making a list of predictors, the number of cases induced by those predictors are computed. I'll try to fit different models covered in the course and see if they produce good accuracy.

Now, we'll start by looking at the analysis step by step.

Data setup and analysis

Data setup

Package installation and loading

We start by installing necessary data processing packages if they are not already installed, then load the packages.

```r
# Install necessary packages:
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")
if(!require(dplyr)) install.packages("dplyr", repos = "http://cran.us.r-project.org")
if(!require(ggplot2)) install.packages("ggplot2", repos = "http://cran.us.r-project.org")
if(!require(lubridate)) install.packages("lubridate", repos = "http://cran.us.r-project.org")

# Load packages
library(tidyverse)
library(caret)
library(data.table)
library(dplyr)
library(ggplot2)
library(lubridate)
```

I tried to download the two files directly from Kaggle but for some reason it didn't work. So I'm including the two files with the submission. Prior running the R script or knitting this .rmd please make sure the csv files are in the working directory.

Read in data and pre-processing

We start with reading in the raw data containing information about Covid-19 and country information.

```r
# Read in the raw data containing country, covid-19 and economy information
dat <- read.csv("raw_data.csv")
```

I feel like there are way too many countries in the world and continents might be a better categorical parameter to discribe location. I want to include continent information in my analysis but the original raw_data doesn't have it. So I'm including it from another source

```r
continent <- read.csv("countryContinent.csv")
```

Here is what the raw data looks like:

```r
head(dat)
```

```
##   iso_code    location       date total_cases total_deaths stringency_index
## 1      AFG Afghanistan 2019-12-31           0            0                0
## 2      AFG Afghanistan 2020-01-01           0            0                0
## 3      AFG Afghanistan 2020-01-02           0            0                0
## 4      AFG Afghanistan 2020-01-03           0            0                0
## 5      AFG Afghanistan 2020-01-04           0            0                0
## 6      AFG Afghanistan 2020-01-05           0            0                0
##   population gdp_per_capita human_development_index     X   X.1   X.2      X.3
## 1   38928341       1803.987                   0.498 #NUM! #NUM! #NUM! 17.47723
## 2   38928341       1803.987                   0.498 #NUM! #NUM! #NUM! 17.47723
## 3   38928341       1803.987                   0.498 #NUM! #NUM! #NUM! 17.47723
## 4   38928341       1803.987                   0.498 #NUM! #NUM! #NUM! 17.47723
## 5   38928341       1803.987                   0.498 #NUM! #NUM! #NUM! 17.47723
## 6   38928341       1803.987                   0.498 #NUM! #NUM! #NUM! 17.47723
##           X.4
## 1 7.497754494
## 2 7.497754494
## 3 7.497754494
## 4 7.497754494
## 5 7.497754494
## 6 7.497754494
```

Notice that there are columns with unclear information, those column names are X, X.1,X.2,X.3 and X.4. At this point I realize there's lack of information for this dataset. I'm not sure what they do, so we remove them. And we rename the columns so the column names are shorter and easier to understand.

```
dat <- dat %>% select(-X,-X.1,-X.2,-X.3,-X.4) %>% rename(
  iso = iso_code,
  country = location,
  cases = total_cases,
  deaths = total_deaths,
  stringency = stringency_index,
  gdp = gdp_per_capita,
  human_development = human_development_index
)
head(dat)
```

```
##   iso     country       date cases deaths stringency population      gdp
## 1 AFG Afghanistan 2019-12-31     0      0          0   38928341 1803.987
## 2 AFG Afghanistan 2020-01-01     0      0          0   38928341 1803.987
## 3 AFG Afghanistan 2020-01-02     0      0          0   38928341 1803.987
## 4 AFG Afghanistan 2020-01-03     0      0          0   38928341 1803.987
## 5 AFG Afghanistan 2020-01-04     0      0          0   38928341 1803.987
## 6 AFG Afghanistan 2020-01-05     0      0          0   38928341 1803.987
##   human_development
## 1             0.498
## 2             0.498
## 3             0.498
## 4             0.498
## 5             0.498
## 6             0.498
```

Next step is to join the continent data to the above dataset. We'll start by observing what information it holds.

```
head(continent)
```

```
##            country code_2 code_3 country_code    iso_3166_2 continent
## 1      Afghanistan     AF    AFG            4 ISO 3166-2:AF      Asia
## 2 \xc5land Islands     AX    ALA          248 ISO 3166-2:AX    Europe
## 3          Albania     AL    ALB            8 ISO 3166-2:AL    Europe
## 4          Algeria     DZ    DZA           12 ISO 3166-2:DZ    Africa
## 5   American Samoa     AS    ASM           16 ISO 3166-2:AS   Oceania
## 6          Andorra     AD    AND           20 ISO 3166-2:AD    Europe
##        sub_region region_code sub_region_code
## 1   Southern Asia         142              34
## 2 Northern Europe         150             154
## 3 Southern Europe         150              39
## 4 Northern Africa           2              15
## 5       Polynesia           9              61
## 6 Southern Europe         150              39
```

We will then take the columns that we'll potentially need.

```r
continent_info <- continent %>% rename(iso = code_3) %>%
  select(iso, continent, sub_region,region_code)
```

Join the continent data

```r
covidecon <- left_join(dat,continent_info,by="iso")
head(covidecon)
```

```
##   iso     country       date cases deaths stringency population      gdp
## 1 AFG Afghanistan 2019-12-31     0      0          0   38928341 1803.987
## 2 AFG Afghanistan 2020-01-01     0      0          0   38928341 1803.987
## 3 AFG Afghanistan 2020-01-02     0      0          0   38928341 1803.987
## 4 AFG Afghanistan 2020-01-03     0      0          0   38928341 1803.987
## 5 AFG Afghanistan 2020-01-04     0      0          0   38928341 1803.987
## 6 AFG Afghanistan 2020-01-05     0      0          0   38928341 1803.987
##   human_development continent    sub_region region_code
## 1             0.498      Asia Southern Asia         142
## 2             0.498      Asia Southern Asia         142
## 3             0.498      Asia Southern Asia         142
## 4             0.498      Asia Southern Asia         142
## 5             0.498      Asia Southern Asia         142
## 6             0.498      Asia Southern Asia         142
```

To uniform a time frame, We only want to look at data in the year 2020, so we'll extract those data

```r
covidecon <- covidecon %>% mutate(date = as_datetime(date), month = month(date)) %>%
  filter(year(date) == 2020)
head(covidecon)
```

```
##   iso     country       date cases deaths stringency population      gdp
## 1 AFG Afghanistan 2020-01-01     0      0          0   38928341 1803.987
## 2 AFG Afghanistan 2020-01-02     0      0          0   38928341 1803.987
## 3 AFG Afghanistan 2020-01-03     0      0          0   38928341 1803.987
## 4 AFG Afghanistan 2020-01-04     0      0          0   38928341 1803.987
## 5 AFG Afghanistan 2020-01-05     0      0          0   38928341 1803.987
## 6 AFG Afghanistan 2020-01-06     0      0          0   38928341 1803.987
##   human_development continent    sub_region region_code month
## 1             0.498      Asia Southern Asia         142     1
## 2             0.498      Asia Southern Asia         142     1
## 3             0.498      Asia Southern Asia         142     1
## 4             0.498      Asia Southern Asia         142     1
## 5             0.498      Asia Southern Asia         142     1
## 6             0.498      Asia Southern Asia         142     1
```

If we explore this covidecon data frame, notice that there are many NA data in the file.

```r
temp <- covidecon %>% filter(is.na(cases) | is.na(deaths) | is.na(gdp) | is.na(stringency))
nrow(temp)
```

```
## [1] 17929
```

Up to this point, it is clear that I didn't get a very nice dataset, but I do not have enough time to find a new one so I'll stick with this. And see what happens in the end.

We'll treat the NA's for cases, deaths, and stringency as one's (not zeros but close enough to zeros).

```r
covidecon$cases[is.na(covidecon$cases)] <- 1
covidecon$deaths[is.na(covidecon$deaths)] <- 1
```

```
covidecon$stringency[is.na(covidecon$stringency)] <- 1
```

Then we consider gdp in particular, if we have absolutely no gdp info, we'll have to remove the country.

```
temp <- covidecon %>% filter(is.na(gdp))
head(temp)
```

```
##   iso country       date cases deaths stringency population gdp
## 1 AND Andorra 2020-03-03     1      1       0.00      77265  NA
## 2 AND Andorra 2020-03-09     1      1       0.00      77265  NA
## 3 AND Andorra 2020-03-10     1      1       0.00      77265  NA
## 4 AND Andorra 2020-03-11     1      1       0.00      77265  NA
## 5 AND Andorra 2020-03-12     1      1       0.00      77265  NA
## 6 AND Andorra 2020-03-13     1      1       9.26      77265  NA
##   human_development continent      sub_region region_code month
## 1             0.858    Europe Southern Europe         150     3
## 2             0.858    Europe Southern Europe         150     3
## 3             0.858    Europe Southern Europe         150     3
## 4             0.858    Europe Southern Europe         150     3
## 5             0.858    Europe Southern Europe         150     3
## 6             0.858    Europe Southern Europe         150     3
```

```
# See the list of contries with no gdp information
country_no_gdp <- temp %>% distinct(country)
# Get the list of countries to remove.
country_to_rm <-  covidecon %>% group_by(country) %>%
  summarize(temp_gpd = max(gdp,na.rm=TRUE)) %>%
  filter(temp_gpd == -Inf) %>% select(country)
```

We find that the country to remove and country with no gdp are the same.

```
country_no_gdp$country == country_to_rm$country
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [16] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

So we'll just remove these countries . We start by defining a not-in operator, and we remove those countries from our data frame.

```
`%!in%` = Negate(`%in%`)
covidecon <- covidecon %>% filter(country %!in% country_to_rm$country)
```

Now our data set should be good and we can explore some more. First, we split our data into a training set and a test set

```
set.seed(1, sample.kind="Rounding")
test_index <- createDataPartition(y = covidecon$cases, times = 1, p = 0.1, list = FALSE)
train_set <- covidecon[-test_index,]
temp <- covidecon[test_index,]
```

We need to make sure the countries in the validation set are all included in the train set. It does not make sense to predict data that we have no information on. We can do the prediction, but the accuracy would be really bad.

```
validation <- temp %>%
  semi_join(train_set, by = "country")
```

Add the removed rows from the validation set back to the train set.

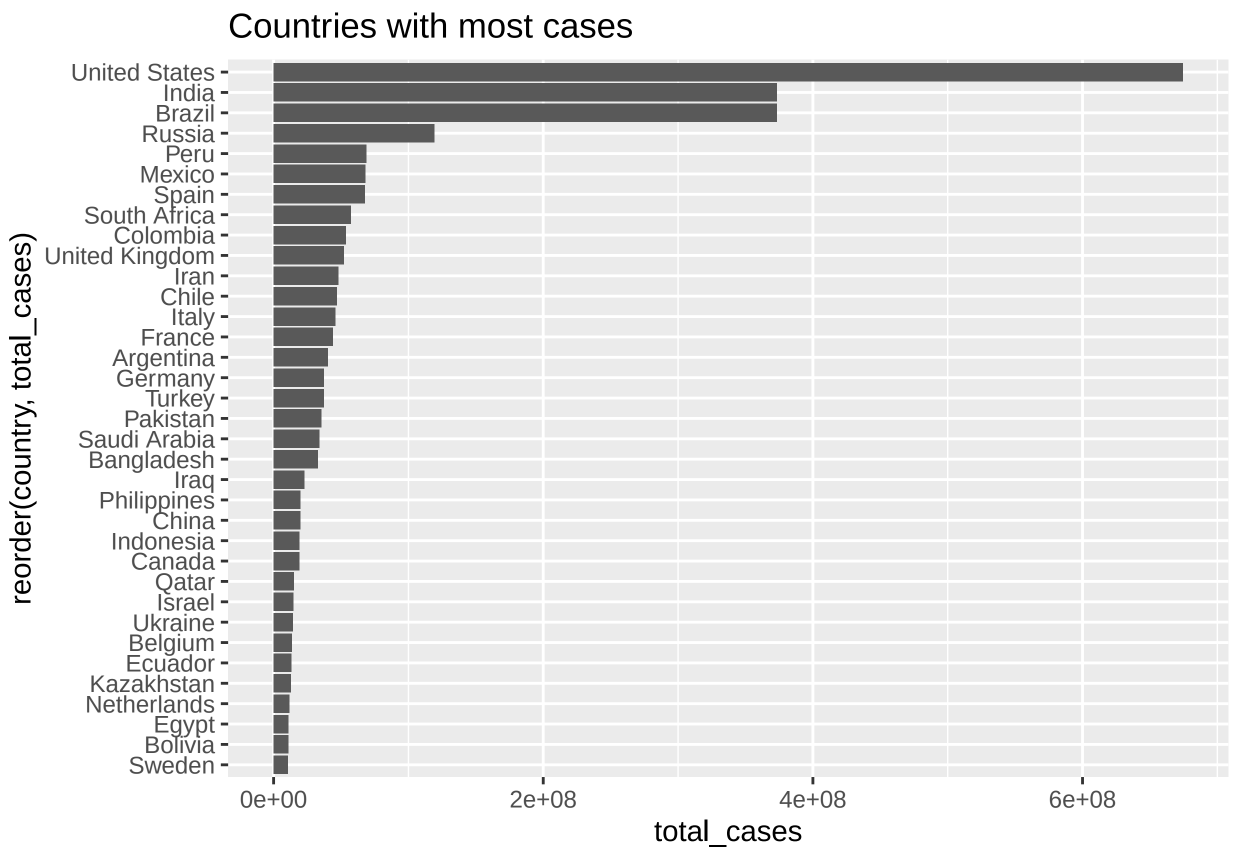
```
removed <- anti_join(temp, validation)
train_set <- rbind(train_set, removed)
```

Analysis

Total covid-cases by country

We'll first look at the number of total Covid-cases by country. We'll compute the total and graph it for visualization.

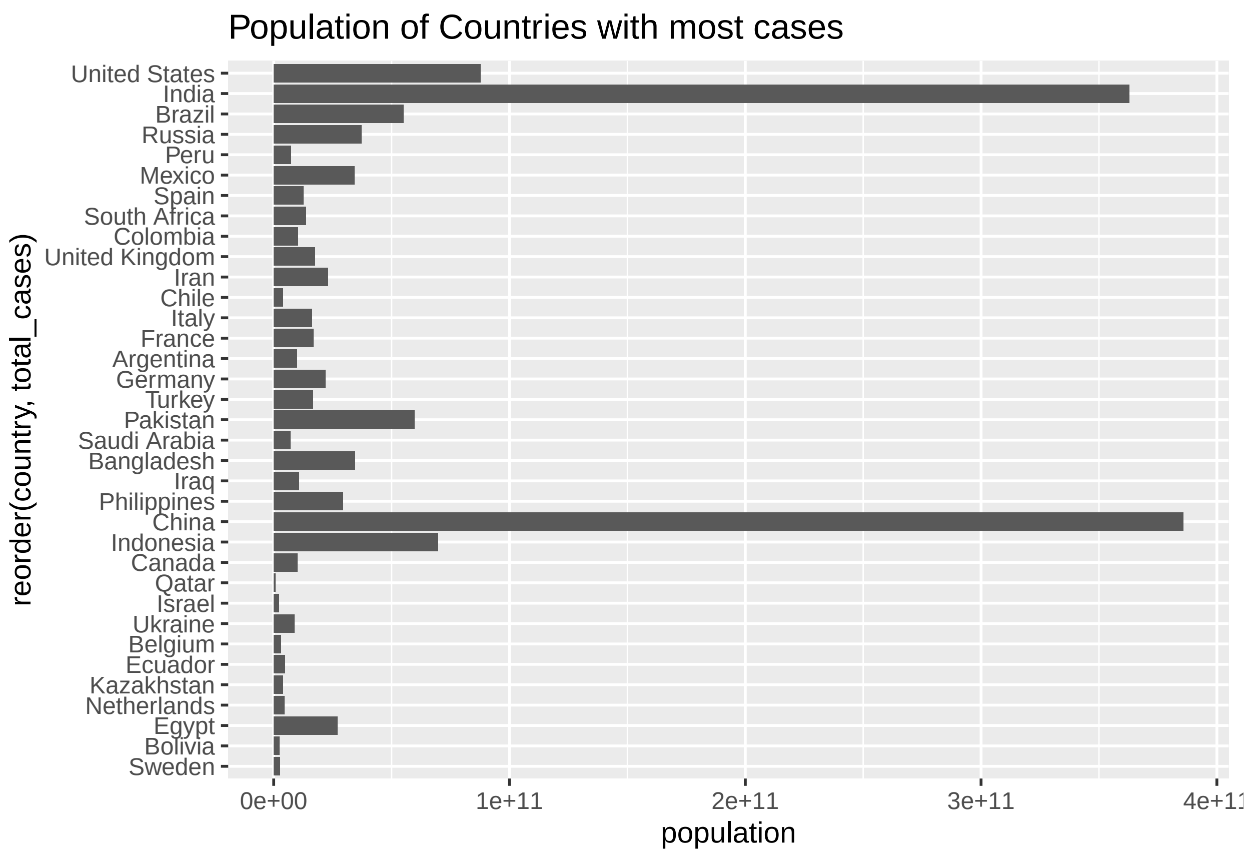
```
total_by_country <- train_set %>% group_by(country) %>% summarize(total_cases = sum(cases))
total_by_country %>% filter(total_cases > 10000000) %>%
  ggplot(aes(reorder(country,total_cases),total_cases)) +
  geom_bar(stat = "identity") +coord_flip() + ggtitle("Countries with most cases")
```



The plot only present the result for countries with more than 10000000 cases. By observing the countries name, we will have an idea of where they are located.

Now, we see if there's correlation between the Covid-19 cases and population of the country.

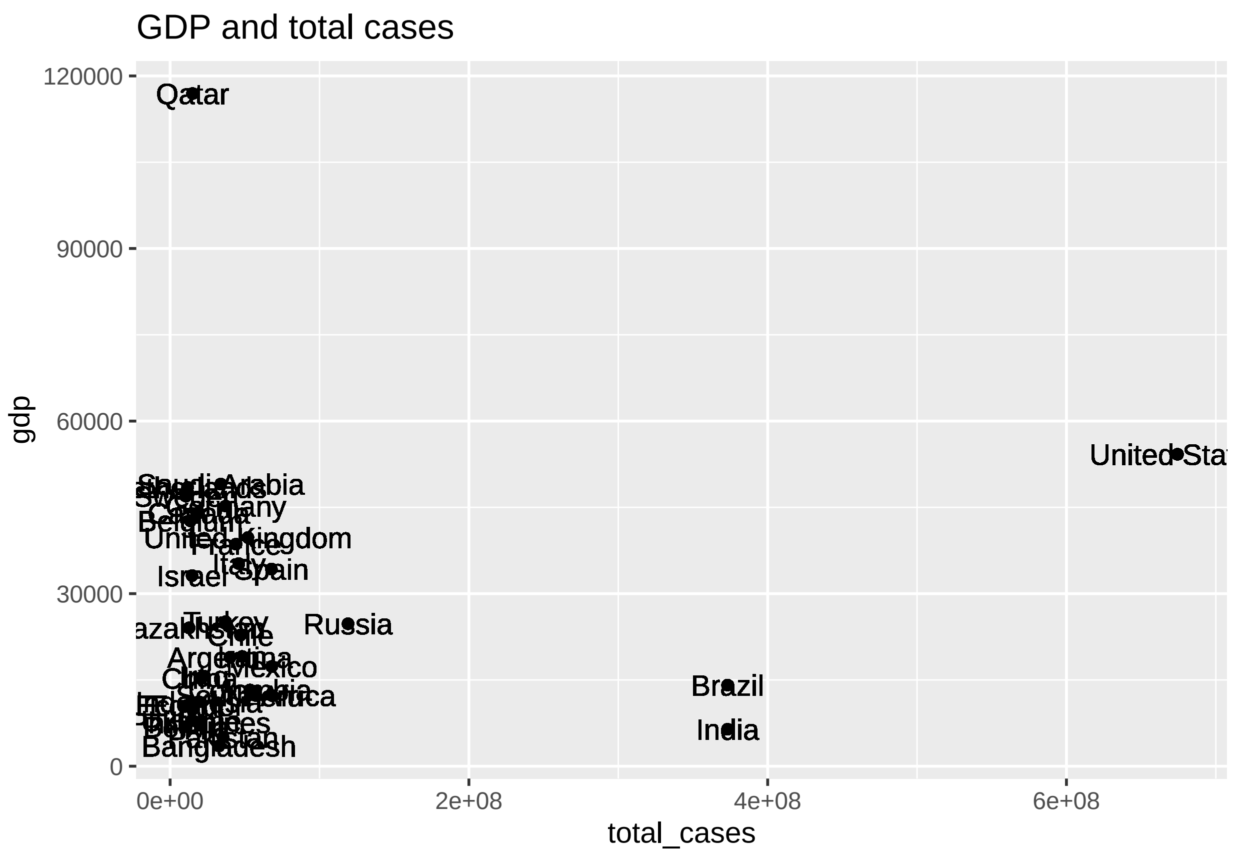
```
train_set<-left_join(train_set,total_by_country,by="country")
train_set %>% filter(total_cases > 10000000) %>%
  ggplot(aes(reorder(country,total_cases),population)) +
  geom_bar(stat = "identity") +coord_flip() +
  ggtitle("Population of Countries with most cases")
```



We see that countries with larger population tends to have more cases, which intuitively makes sense.

Now, we see if there's correlation between the Covid-19 cases and gdp of the country.

```
train_set %>% filter(total_cases > 10000000) %>%
  ggplot(aes(total_cases,gdp, label = country)) +
  geom_point() + geom_text(aes(label=country))+
  ggtitle("GDP and total cases")
```

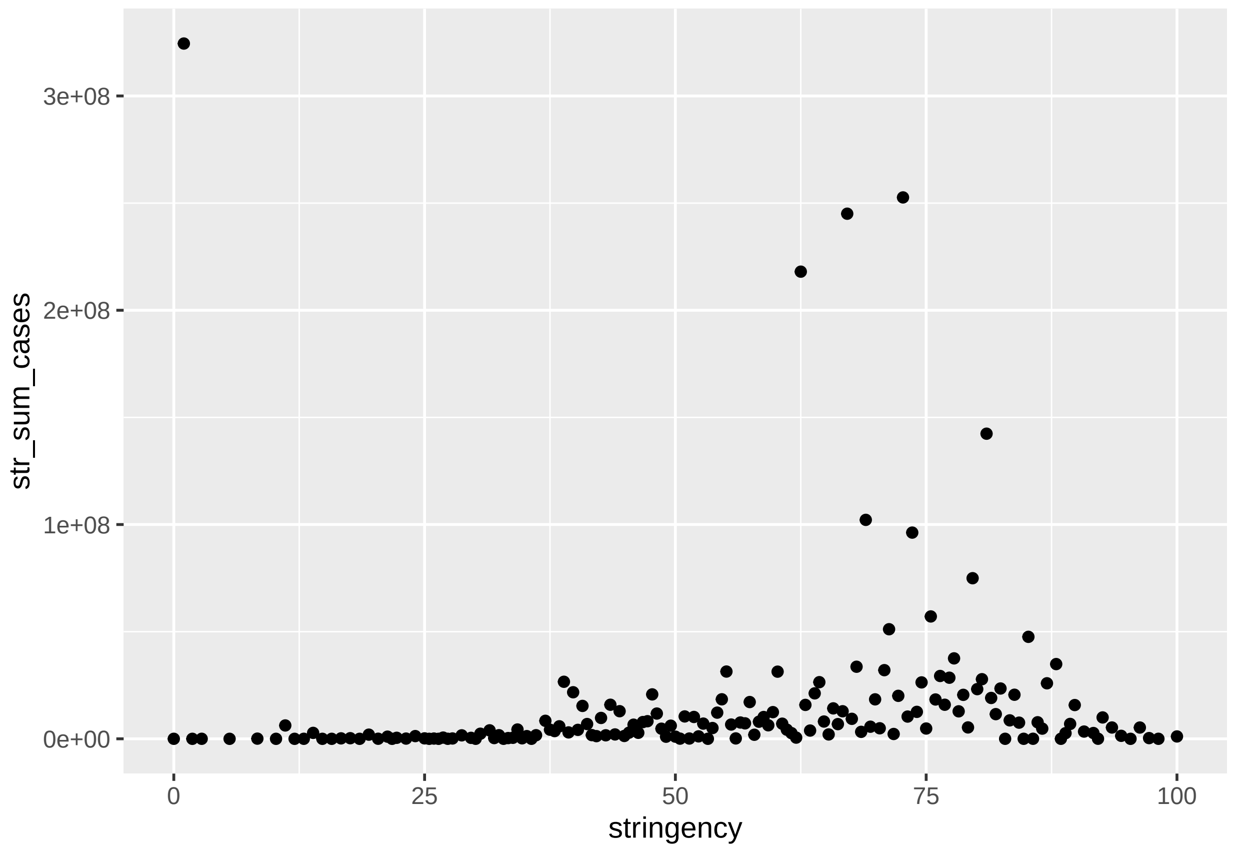


Although the labels are not that clear to read on the plot. We can still get an idea that GDP and number of Covid cases are not that related, since the majority of countries are clustered at the bottom left corner and very few "special case" countries could be considered as outliers.

Cases and stringency

Graph of cases versus stringency.

```r
train_set %>% group_by(stringency) %>% summarise(str_sum_cases=sum(cases)) %>%
  ggplot(aes(stringency,str_sum_cases)) + geom_point()
```

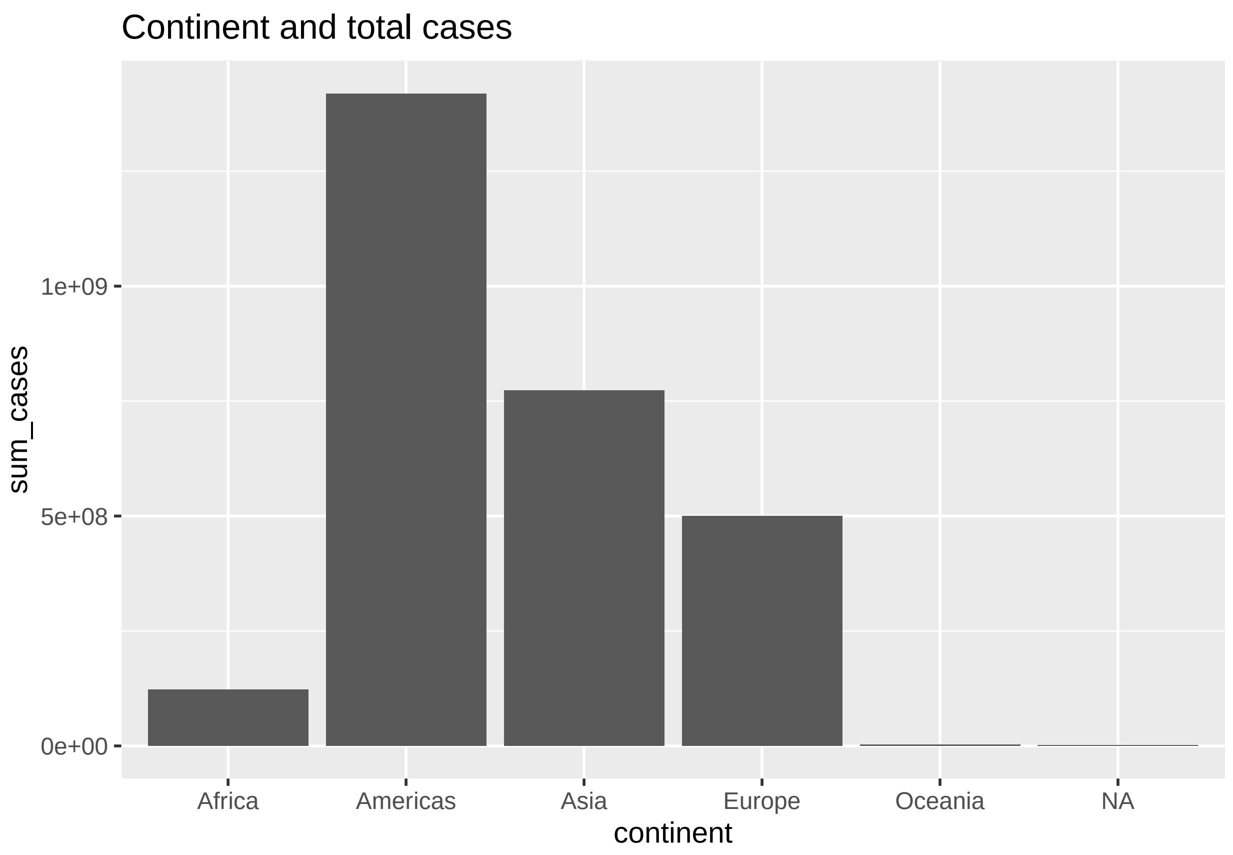


The plot shows that stringency does not help with controling Covid cases that much. But in reality, we know that it should. This is probably cause by the fact that countries with higher stringency also have more population, hence more cases. But that we cannot see in the plot.

Continents and cases

Lets look at how continents affect the case number.

```
continent_avg <- train_set %>% group_by(continent) %>% summarise(sum_cases = sum(cases))
continent_avg %>% ggplot(aes(continent,sum_cases)) +
  geom_bar(stat="identity") + ggtitle("Continent and total cases")
```



Looking at the plot, it is very hard to tell if the case total is low because of not that many infection or because we lost data. Notice that a lot of the data missing are in countries in Africa. And this plot here agrees with the country total plot we created above.

Result

We want to predict the number of covid-19 cases by building a model considering the effect of population, gdp, and continent here we'll use a few models we learned to see which one gives the highest accuracy, or if any of them gives any good accuracy.

Linear model

We will fit a linear model and compute accuracy.

```
lm_fit <- train(cases ~ population+gdp+continent+stringency,method = "lm", data=train_set,na.action=na.
y_hat_lm <- predict(lm_fit,newdata = na.omit(validation))
mean(y_hat_lm == na.omit(validation)$cases)
```

```
## [1] 0
```

We have 0 accuracy. This is definitely not a good model for fitting.

General Linear model

Moving from lm, I wanted to test out a general linear model.

```
glm_fit <- train(cases ~ population+gdp+continent+stringency,method = "glm", data=train_set,na.action=na
y_hat_glm <- predict(glm_fit,newdata = na.omit(validation))
mean(round(y_hat_glm) == na.omit(validation)$cases)
```

```
## [1] 0
```

Recursive Partitioning And Regression Trees

Then I tried to fit a rpart model:

```
rpart_fit <- train(cases ~ population+gdp+continent+stringency,method = "rpart", data=train_set,na.acti
y_hat_rpart <- predict(rpart_fit,newdata = na.omit(validation))
mean(round(y_hat_rpart) == na.omit(validation)$cases)
```

```
## [1] 0.0002329373
```

The accuracy here is not zero but very close to zero. Indicating that this is not a good model to fit as well.

Random forest

The last fitting I tried is the random forest. The code takes very long time to run and it ended up crashing R. given the previous few tries, I'm not sure if this is the right model to try or if it is even worth it to implement. The code is here, but I wouldn't recommend running it. rf_fit <- train(cases ~ population+gdp+continent+stringency,method = "rf", data=train_set,na.action=na.omit) y_hat_rf <- predict(rf_fit,newdata = na.omit(validation)) mean(round(y_hat_rf) == na.omit(validation)$cases)

Conclusion

I have very limited time to spend on this project and it is sad that I couldn' work on it more before the due time. The analysis above didn't result in a model worth noting to predict the number of Covid-19 cases in the world factoring in the effect of country population, continent, stringency and gdp. So here in this session, I want to talk about the two possible reason for not being able to find a model.

The first reason would be the constraint on the dataset. The data set I decided to work on has really limited information and maybe the quality of information has limited the amount of analysis I could perform. And the X's I took out at the very beginning might have special meanings that I do not know. Also, there were values missing and I had smooth out the data by adding in my guesses, which increased the uncertainty in our prediction. And the NA values have been creating a lot of obstracles when implementing the code. Also, Covid-19 is a virus, so the number of cases may also be affected by the medication quality in the country, and may also be affected by biological differences resulted on people with different living habits. However, with the limited data from the dataset I found, we wouldn't be able to tell. Another reason is the models I tested on. I only had the time to test limited amount of models that are listed above and none of them seems like a good fit.

I do believe that with more work done, more accurate data collected, better analysis on predictors, maybe testing with more complicated models , there should be be a model that describes the number of cases better.