Data Science: Capstone Covid-19 Project

David Sanchez Plana

18 junio, 2020

Contents

Executive summary

This is a project report for the *Data Science: Capstone* course. It is asked to apply machine learning techniques to a chosen dataset, with the purpose of study the data, predict the values of some variables and get the model accuracy. To make this report, *R Markdown* guide has been followed.

Applying decision trees modeling we achieve an accuracy of about **80%** and applying regularization we obtain a RMSE of **0.04**

1 Introduction

From the *Historical development of epidemiology: its training as a scientific discipline* article epidemiology is described the branch of public health that aims to describe and explain the dynamics of population health, identify the elements that comprise it, and understand the forces that govern it, in order to intervene in the course of its natural development. Currently, it is accepted that to fulfill its mission, epidemiology investigates the distribution, frequency and determinants of health conditions in human populations, as well as the modalities and impact of the social responses established to attend to them.

In the last decade, due to the evolution and the impulse that Data Science or Big Data have suffered, these epidemiological studies have been helped by a powerful ally. Thanks to programming tools such as R, which have a flexible set of tools to analyze large volumes of data, evaluations (and in some cases predictions) of how these epidemics or viruses will unfold can be made.

That is why the opportunity to apply data science knowledge to the case of the current covid-19 pandemic arises.

1.1 Input Data

The data obtained from Covid-19 has been extracted from the official WHO website. This project is not going to enter into the debate of the veracity of the data or the methods of measurement of the infected. Depending on the source being consulted, there may be variations.

To add more useful information to the WHO dataset, we have chosen to include information about the population density and life expectancy of each country, as this variables may influence the spread of a pandemic.

Population data has been obtained from here while life expectancy data has been obtained from here.

Anyway, if you want to use another input data set with information from the Covid-19 to perform other studies, you can do it. Simply define the countries with the exact name that have the population and life expectancy ones.

The following provided code generates the datasets.

```r
if(!require(dplyr)) install.packages("dplyr",
                                     repos = "http://cran.us.r-project.org")
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(dslabs)) install.packages("dslabs",
                                     repos = "http://cran.us.r-project.org")
if(!require(lubridate)) install.packages("lubridate",
                                     repos = "http://cran.us.r-project.org")
if(!require(ggplot2)) install.packages("ggplot2",
                                     repos = "http://cran.us.r-project.org")
if(!require(data.table)) install.packages("data.table",
                                     repos = "http://cran.us.r-project.org")
if(!require(stringr)) install.packages("stringr",
                                     repos = "http://cran.us.r-project.org")
if(!require(readr)) install.packages("readr",
                                     repos = "http://cran.us.r-project.org")
if(!require(rpart)) install.packages("rpart",
                                     repos = "http://cran.us.r-project.org")
if(!require(rpart.plot)) install.packages("rpart.plot",
                                     repos = "http://cran.us.r-project.org")

# Load the libraries
library(dplyr)
library(tidyverse)
library(caret)
library(dslabs)
library(lubridate)
library(ggplot2)
library(data.table)
library(stringr)
library(readr)
library(rpart)
library(rpart.plot)

# Extract data
covid<-read_csv("https://raw.githubusercontent.com/davidsp-github/Covid19/master/WHO-COVID-19-g
                skip = 0,col_names = TRUE)
life_expectancy<-read_csv("https://raw.githubusercontent.com/davidsp-github/Covid19/master/Life
                          skip = 0,col_names = TRUE)
population<-read_csv("https://raw.githubusercontent.com/davidsp-github/Covid19/master/density.c
                     skip = 0,col_names = TRUE)

# Combine and mutate data
population<- population %>% mutate(Population=Population*1000, Area_km2=Area_km2*10)
combine <- covid %>% inner_join(life_expectancy)
```

```
covid_dataset <- combine %>% inner_join(population)
rm(combine)
covid_dataset <- covid_dataset %>% mutate(ratio=Cumulative_deaths/Cumulative_cases)
covid_dataset$Country_code[is.na(covid_dataset$Country_code)]<-"NAM"
```

The csv files used to generate the data have been already cleaned. For example, the names of the countries have been modified to match those used by the WHO. Also, the code above generates concise datasets, so there would be no need to apply **data cleaning**.

Once we have the data we must split edx data into training and test sets, in order to validate the models. 80% of the edx data will make up the training set while remaining 20% will form the test set.

```
# Create test and training sets
# Test set will be 20% of Covid data
set.seed(1, sample.kind="Rounding")
test_index <- createDataPartition(y = covid_dataset$Country,
                                  times = 1, p = 0.2, list = FALSE)
train_set <- covid_dataset[-test_index,]
temp <- covid_dataset[test_index,]

# Make sure Countries in test set are also in train set
test_set <- temp %>%
  semi_join(train_set, by = "Country")
rm(temp)
```

1.2 Data Exploration and Visualization

With RStudio we can see the dimensions and the different variables that are in the **covid_dataset**. However, we can explore the data as follows:

- *covid_dataset* fields:

```
head(covid_dataset)
```

```
## # A tibble: 6 x 16
##   Date_reported       Country_code Country WHO_region New_cases Cumulative_cases
##   <dttm>              <chr>        <chr>   <chr>          <dbl>             <dbl>
## 1 2020-02-24 00:00:00 AF           Afghan~ EMRO               1                 1
## 2 2020-02-25 00:00:00 AF           Afghan~ EMRO               0                 1
## 3 2020-02-26 00:00:00 AF           Afghan~ EMRO               0                 1
## 4 2020-02-27 00:00:00 AF           Afghan~ EMRO               0                 1
## 5 2020-02-28 00:00:00 AF           Afghan~ EMRO               0                 1
## 6 2020-02-29 00:00:00 AF           Afghan~ EMRO               0                 1
## # ... with 10 more variables: New_deaths <dbl>, Cumulative_deaths <dbl>,
## #   Life_Expectancy <dbl>, Female_LE <dbl>, Male_LE <dbl>, Density_km <dbl>,
## #   Density_Mi <dbl>, Population <dbl>, Area_km2 <dbl>, ratio <dbl>
```

Therefore this data set contains the following information:
- Date_reported: date of register
- Country_code: country acronym
- Country: name of the country
- WHO_region: world region
- New_cases: new cases registered
- Cumulative_cases: accumulated cases
- New_deaths: new deaths registered
- Cumulative_deaths: accumulated deaths
- Life_Expectancy: average life expectancy
- Female_LE: female life expectancy
- Male_LE: male life expectancy
- Density_km: population density in square kilometers
- Density_Mi: population density in square miles
- Area_km2: country area in square kilometers
- ratio: cumulative deaths per Cumulative cases

- Number of data inputs and fields in *covid_dataset*:

```
dim(covid_dataset)
```

```
## [1] 17094    16
```

- Different countries in *covid_dataset*:

```
unique(covid_dataset$Country)
```

```
##   [1] "Afghanistan"                          "Albania"
##   [3] "Algeria"                              "Andorra"
##   [5] "Angola"                               "Antigua and Barbuda"
##   [7] "Argentina"                            "Armenia"
##   [9] "Aruba"                                "Australia"
##  [11] "Austria"                              "Azerbaijan"
##  [13] "Bahamas"                              "Bahrain"
##  [15] "Bangladesh"                           "Barbados"
##  [17] "Belarus"                              "Belgium"
##  [19] "Belize"                               "Benin"
##  [21] "Bermuda"                              "Bhutan"
##  [23] "Bolivia (Plurinational State of)"     "Bosnia and Herzegovina"
##  [25] "Botswana"                             "Brazil"
##  [27] "British Virgin Islands"               "Brunei Darussalam"
##  [29] "Bulgaria"                             "Burkina Faso"
##  [31] "Burundi"                              "Cabo Verde"
##  [33] "Cambodia"                             "Cameroon"
##  [35] "Canada"                               "Central African Republic"
```

```
##  [37] "Chad"                                 "Chile"
##  [39] "China"                                "Colombia"
##  [41] "Comoros"                              "Congo"
##  [43] "Costa Rica"                           "Croatia"
##  [45] "Cuba"                                 "Curacao"
##  [47] "Cyprus"                               "Czechia"
##  [49] "Democratic Republic of the Congo"     "Denmark"
##  [51] "Djibouti"                             "Dominica"
##  [53] "Dominican Republic"                   "Ecuador"
##  [55] "Egypt"                                "El Salvador"
##  [57] "Equatorial Guinea"                    "Eritrea"
##  [59] "Estonia"                              "Eswatini"
##  [61] "Ethiopia"                             "Fiji"
##  [63] "Finland"                              "France"
##  [65] "French Guiana"                        "French Polynesia"
##  [67] "Gabon"                                "Gambia"
##  [69] "Georgia"                              "Germany"
##  [71] "Ghana"                                "Greece"
##  [73] "Grenada"                              "Guadeloupe"
##  [75] "Guam"                                 "Guatemala"
##  [77] "Guinea"                               "Guinea-Bissau"
##  [79] "Guyana"                               "Haiti"
##  [81] "Honduras"                             "Hungary"
##  [83] "Iceland"                              "India"
##  [85] "Indonesia"                            "Iran (Islamic Republic of)"
##  [87] "Iraq"                                 "Ireland"
##  [89] "Israel"                               "Italy"
##  [91] "Jamaica"                              "Japan"
##  [93] "Jordan"                               "Kazakhstan"
##  [95] "Kenya"                                "Kuwait"
##  [97] "Kyrgyzstan"                           "Lao People's Democratic Republic"
##  [99] "Latvia"                               "Lebanon"
## [101] "Lesotho"                              "Liberia"
## [103] "Libya"                                "Lithuania"
## [105] "Luxembourg"                           "Madagascar"
## [107] "Malawi"                               "Malaysia"
## [109] "Maldives"                             "Mali"
## [111] "Malta"                                "Martinique"
## [113] "Mauritania"                           "Mauritius"
## [115] "Mayotte"                              "Mexico"
## [117] "Mongolia"                             "Montenegro"
## [119] "Morocco"                              "Mozambique"
## [121] "Myanmar"                              "Namibia"
## [123] "Nepal"                                "Netherlands"
## [125] "New Caledonia"                        "New Zealand"
## [127] "Nicaragua"                            "Niger"
## [129] "Nigeria"                              "North Macedonia"
## [131] "Norway"                               "Palestine"
```

```
## [133] "Oman"                                "Pakistan"
## [135] "Panama"                              "Papua New Guinea"
## [137] "Paraguay"                            "Peru"
## [139] "Philippines"                         "Poland"
## [141] "Portugal"                            "Puerto Rico"
## [143] "Qatar"                               "Republic of Korea"
## [145] "R<e9>union"                          "Romania"
## [147] "Russian Federation"                  "Rwanda"
## [149] "Saint Lucia"                         "Saint Vincent and the Grenadines"
## [151] "Sao Tome and Principe"               "Saudi Arabia"
## [153] "Senegal"                             "Serbia"
## [155] "Seychelles"                          "Sierra Leone"
## [157] "Singapore"                           "Slovakia"
## [159] "Slovenia"                            "Somalia"
## [161] "South Africa"                        "South Sudan"
## [163] "Spain"                               "Sri Lanka"
## [165] "Sudan"                               "Suriname"
## [167] "Sweden"                              "Switzerland"
## [169] "Syrian Arab Republic"                "Tajikistan"
## [171] "Thailand"                            "The United Kingdom"
## [173] "Timor-Leste"                         "Togo"
## [175] "Trinidad and Tobago"                 "Tunisia"
## [177] "Turkey"                              "Uganda"
## [179] "Ukraine"                             "United Arab Emirates"
## [181] "United Republic of Tanzania"         "United States of America"
## [183] "United States Virgin Islands"        "Uruguay"
## [185] "Uzbekistan"                          "Venezuela (Bolivarian Republic of)"
## [187] "Viet Nam"                            "Yemen"
## [189] "Zambia"                              "Zimbabwe"
```

- Top 10 countries with the higher and the lower life expectancy respectively:

```
top_n(life_expectancy,10,Life_Expectancy) %>% arrange(-Life_Expectancy)
```

```
## # A tibble: 10 x 4
##    Country           Life_Expectancy Female_LE Male_LE
##    <chr>                       <dbl>     <dbl>   <dbl>
##  1 Japan                        85.0      88.1    81.9
##  2 Macao                        84.7      87.6    81.7
##  3 Switzerland                  84.2      86.0    82.4
##  4 Singapore                    84.1      86.2    82.1
##  5 Italy                        84.0      86.0    81.9
##  6 Spain                        84.0      86.7    81.3
##  7 Australia                    83.9      85.8    82.1
##  8 Iceland                      83.5      84.9    82.2
##  9 Republic of Korea            83.5      86.4    80.5
## 10 Israel                       83.5      84.9    82.0
```

```
top_n(life_expectancy,-10,Life_Expectancy) %>% arrange(Life_Expectancy)
```

```
## # A tibble: 10 x 4
##    Country                     Life_Expectancy Female_LE Male_LE
##    <chr>                                 <dbl>     <dbl>   <dbl>
##  1 "Central African Republic"             54.4      56.6    52.2
##  2 "Chad"                                 55.2      56.6    53.7
##  3 "Lesotho"                              55.6      58.9    52.5
##  4 "Nigeria"                              55.8      56.8    54.8
##  5 "Sierra Leone"                         55.9      56.8    55.0
##  6 "Somalia"                              58.3      60.1    56.6
##  7 "South Sudan"                          58.7      60.3    57.2
##  8 "C\xf4te d'Ivoire"                     58.8      60.1    57.5
##  9 "Guinea-Bissau"                        59.4      61.3    57.3
## 10 "Equatorial Guinea"                    59.8      61.1    58.8
```

- Top 10 countries with the higher and the lower population density respectively:

```
top_n(population,10,Density_km) %>% arrange(-Density_km)
```

```
## # A tibble: 10 x 5
##    Country      Density_km Density_Mi Population Area_km2
##    <chr>             <dbl>      <dbl>      <dbl>    <dbl>
##  1 Macao            21644.     56059.     649335     30
##  2 Monaco           19427.     50315.      39242     20.2
##  3 Singapore         8240.     21341.    5850342    710
##  4 Gibraltar         5615.     14543.      33691     60
##  5 Bahrain           2224.      5761.    1701575    760
##  6 Maldives          1802.      4667.     540544    300
##  7 Malta             1397.      3619.     441543    310
##  8 Sint Maarten      1261.      3266.      42876     30
##  9 Bermuda           1153.      2987.      62278     50
## 10 Bangladesh        1116.      2890.  164689383 147570
```

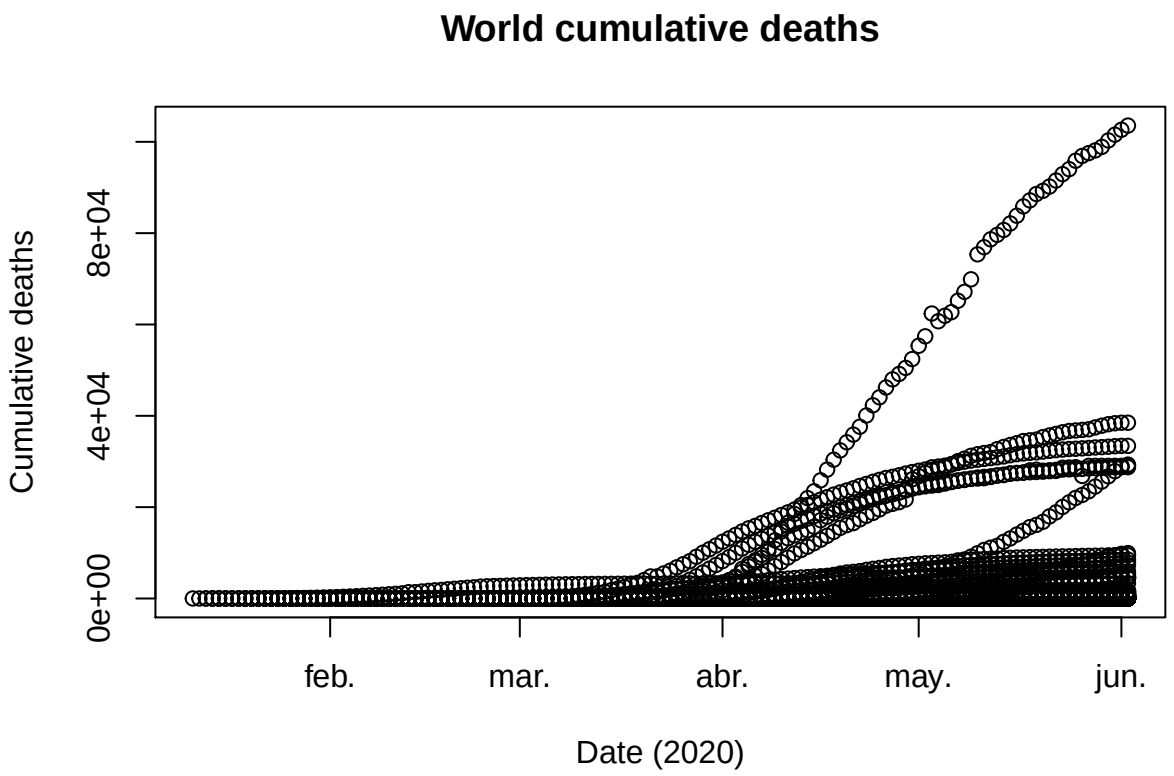
```
top_n(population,-10,Density_km) %>% arrange(Density_km)
```

```
## # A tibble: 10 x 5
##    Country                          Density_km Density_Mi Population Area_km2
##    <chr>                                 <dbl>      <dbl>      <dbl>    <dbl>
##  1 Greenland                            0.0262     0.0679      56770  2166080
##  2 Falkland Islands (Malvinas) [2]      0.286      0.740        3480    12170
##  3 Mongolia                             2.10       5.43      3278290  1564110
##  4 Namibia                              3.08       7.97      2540905   825610
##  5 Iceland                              3.31       8.58       341243   103000
##  6 Australia                            3.32       8.59     25499884  7692020
```

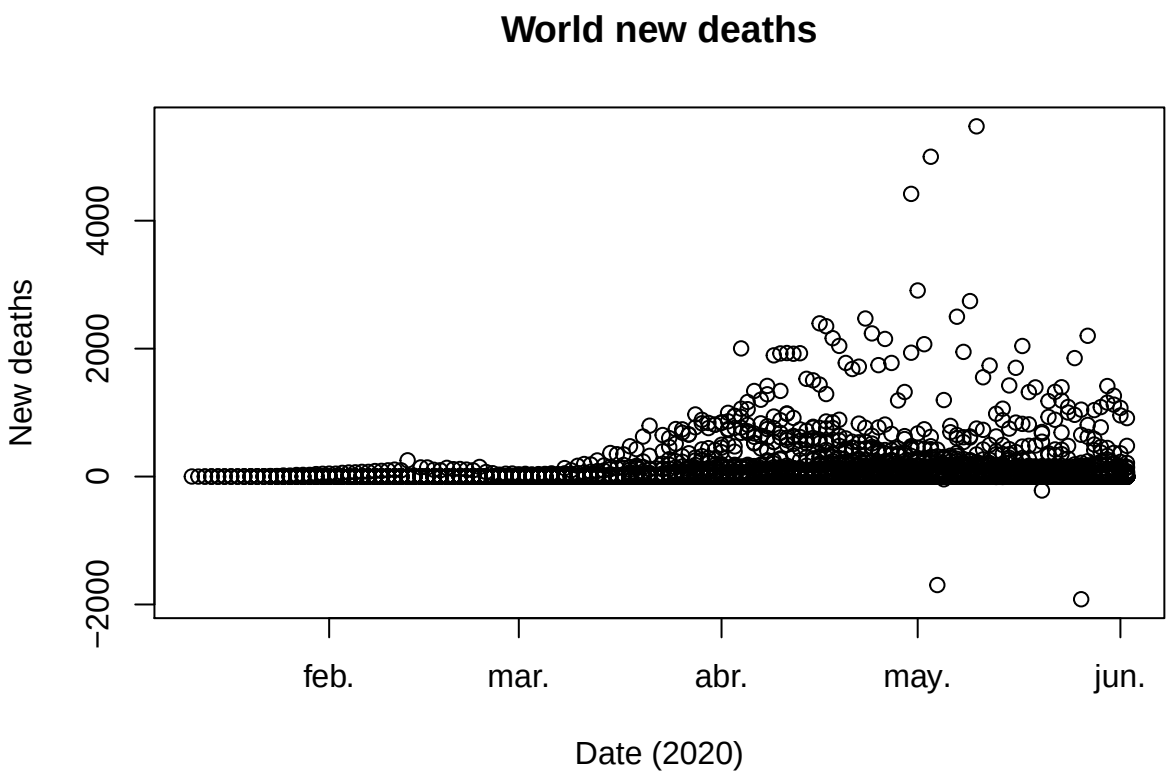
```
##  7 French Guiana                3.58       9.26       298682    83530
##  8 Suriname                     3.58       9.27       586632   163820
##  9 Guyana                       3.66       9.48       786552   214960
## 10 Canada                       3.78       9.79     37742154  9984670
```

- Plots of cumulative/new deaths and cases respectively:

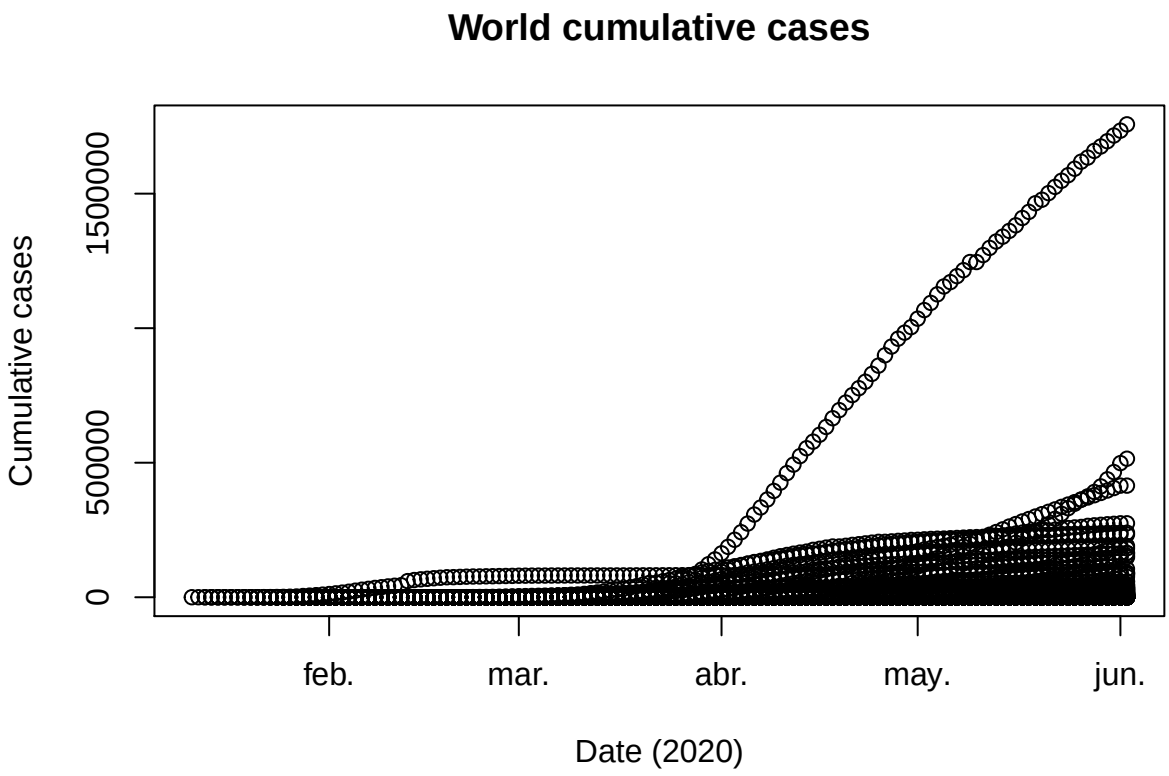
```
# Plot new deaths & cumulative deaths
plot(covid_dataset$Date_reported,covid_dataset$Cumulative_deaths,main="World cumulative deaths"
     xlab="Date (2020)", ylab="Cumulative deaths")
```



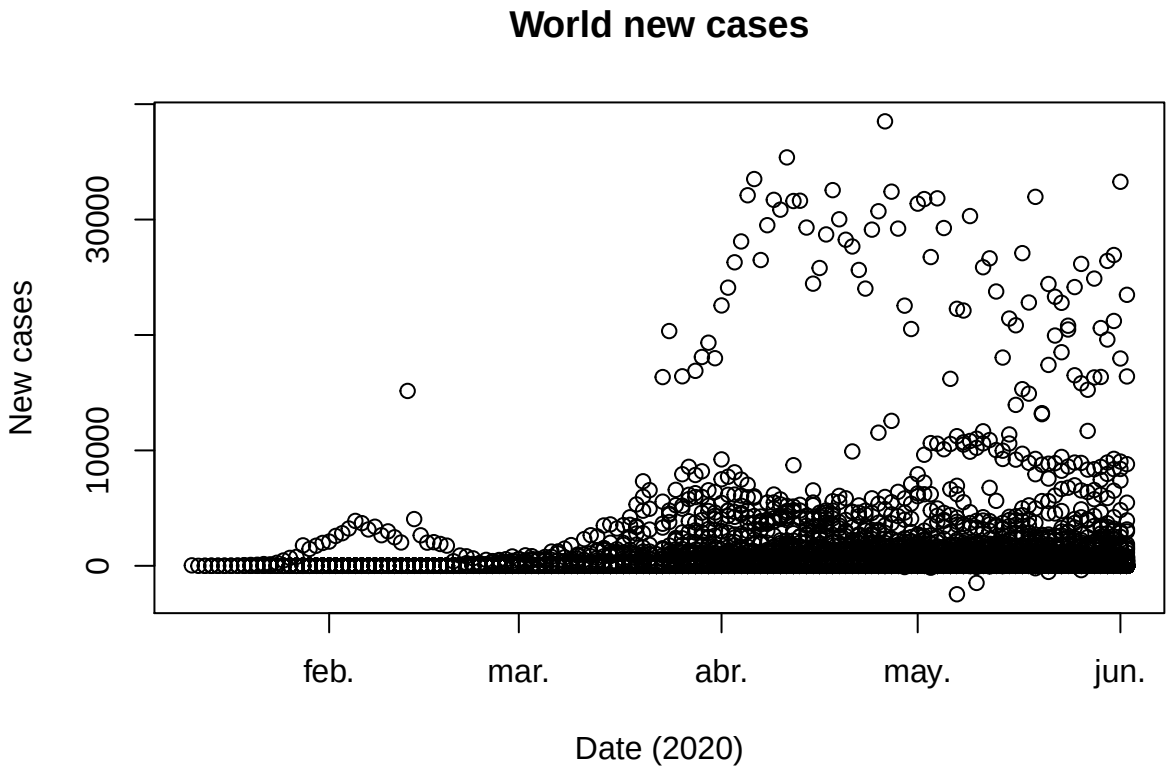

```
plot(covid_dataset$Date_reported,covid_dataset$New_deaths,main="World new deaths",
     xlab="Date (2020)", ylab="New deaths")
```




```
plot(covid_dataset$Date_reported,covid_dataset$Cumulative_cases,main="World cumulative cases",
     xlab="Date (2020)", ylab="Cumulative cases")
```




```
plot(covid_dataset$Date_reported,covid_dataset$New_cases,main="World new cases",
     xlab="Date (2020)", ylab="New cases")
```

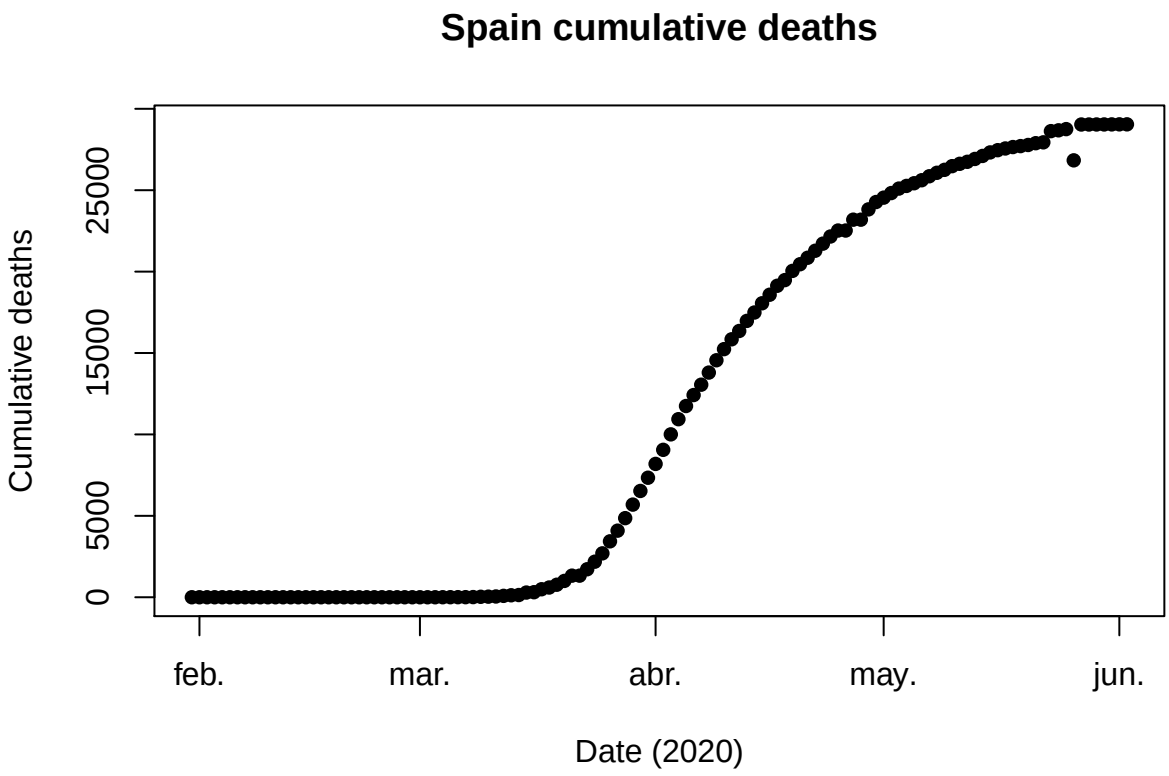


As we can see in these plots, cases and deaths follow similar patterns. We can also see how some countries follow a positive trend nowadays while others have stabilized the contagion/death curves.

Another aspect that can attract our attention is the negative data in new cases and new deaths. It is very likely that this is an adjustment made by the WHO to standardize the measurement system or to balance incorrect data.

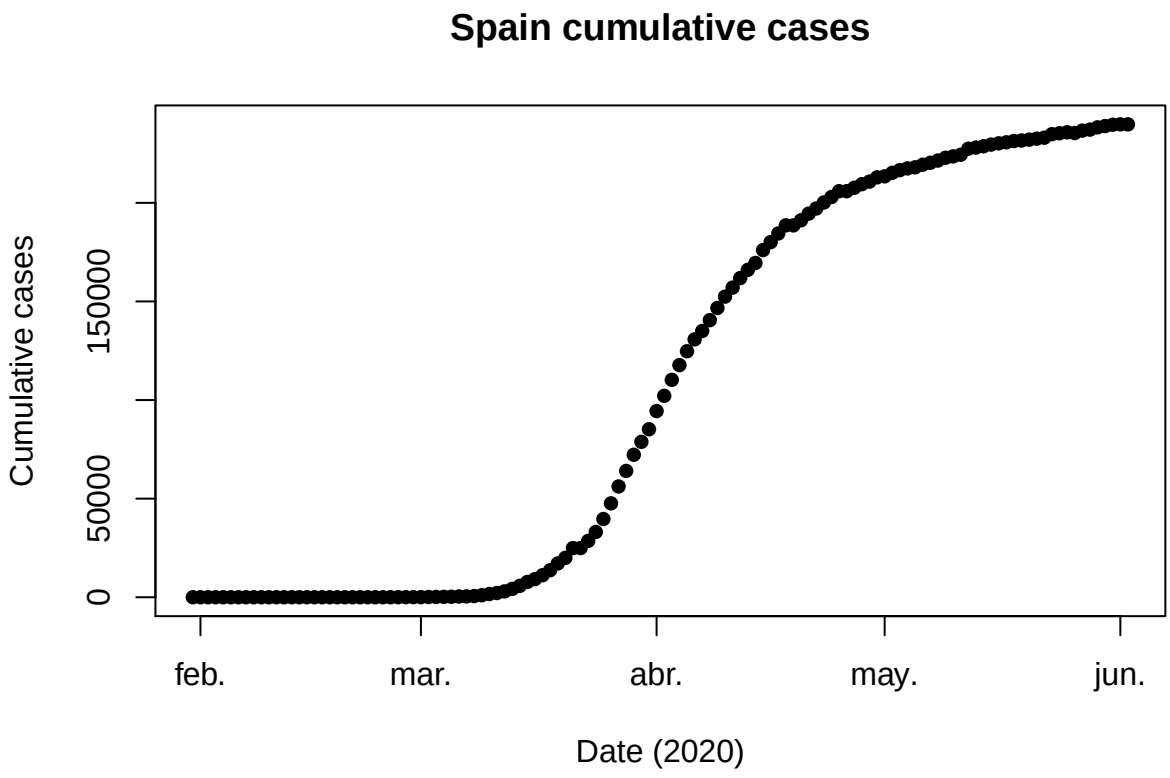
- Example plots for 3 countries (Spain, China and USA):

```
# Plot new deaths & cumulative deaths for different countries
Spain<-filter(covid_dataset, Country=="Spain")
Spain$Colour="black"
Spain$Colour[Spain$New_deaths>=1000]="red"
Spain$Colour[Spain$New_deaths<0]="purple"
plot(Spain$Date_reported,Spain$Cumulative_deaths,main="Spain cumulative deaths",
     xlab="Date (2020)", ylab="Cumulative deaths",pch = 16)
```

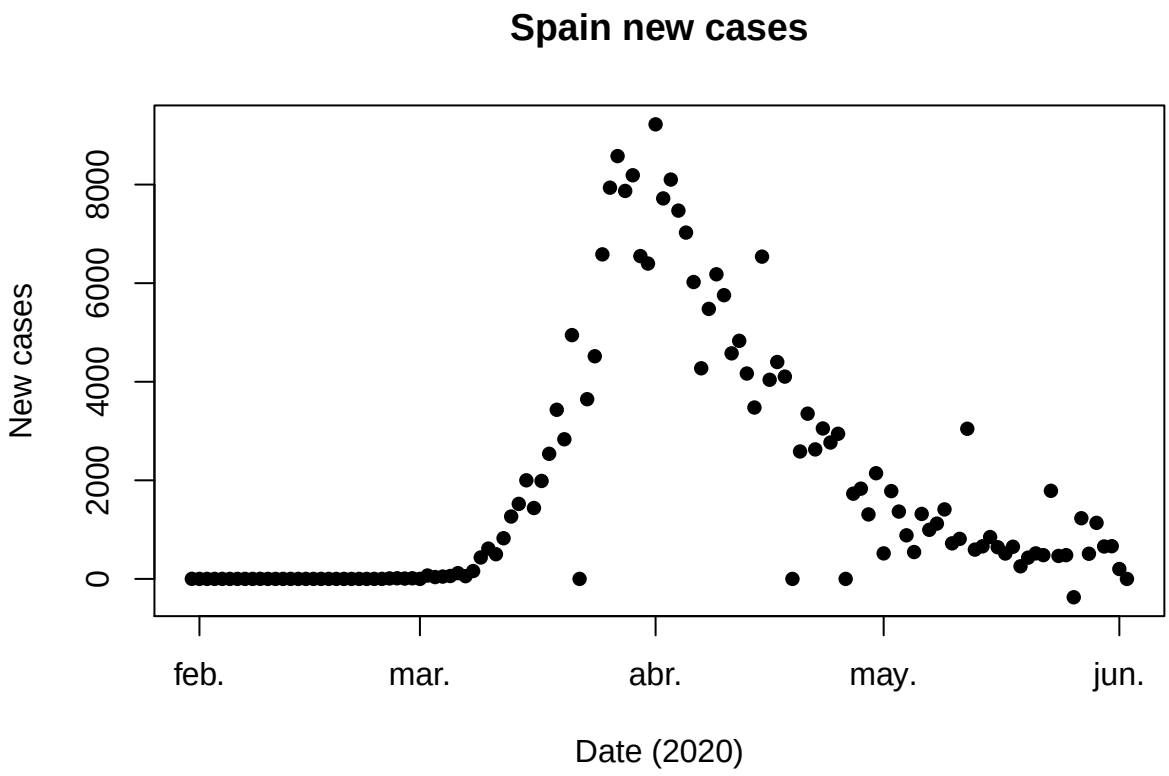



```
plot(Spain$Date_reported,Spain$New_deaths,main="Spain new deaths",
     xlab="Date (2020)", ylab="New deaths",col=Spain$Colour,pch = 16)
```

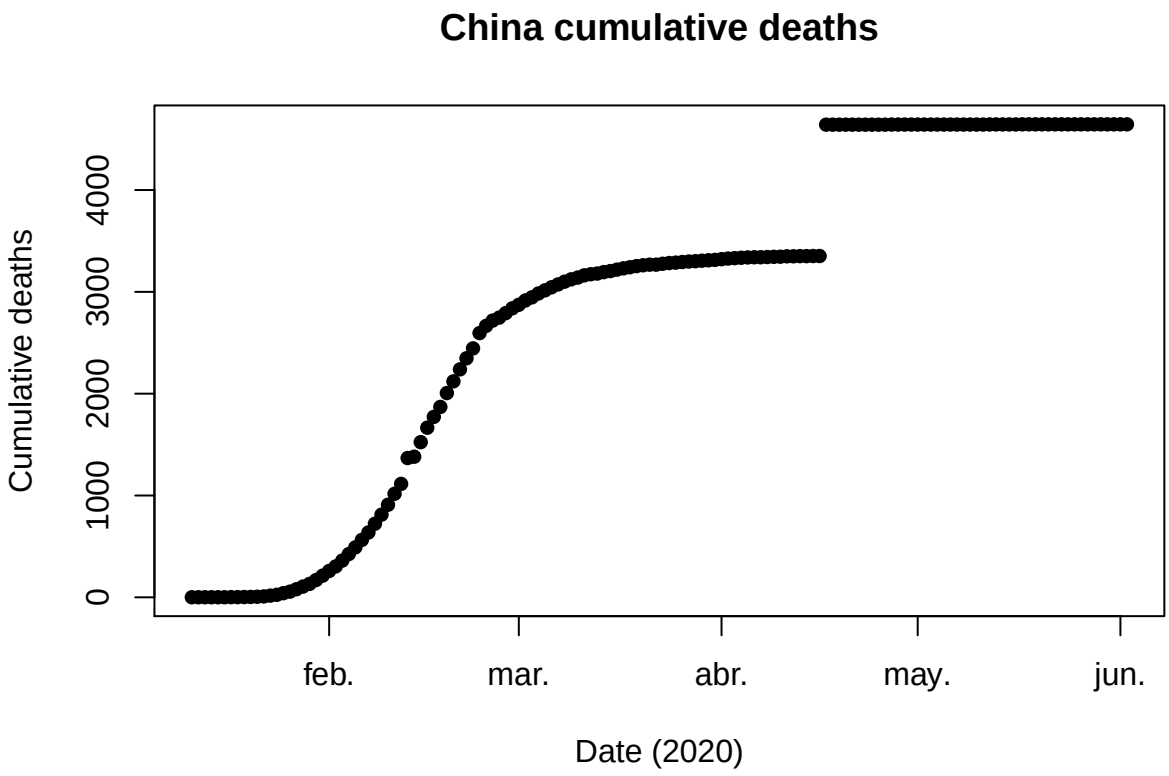

```
plot(Spain$Date_reported,Spain$Cumulative_cases,main="Spain cumulative cases",
     xlab="Date (2020)", ylab="Cumulative cases",pch = 16)
```



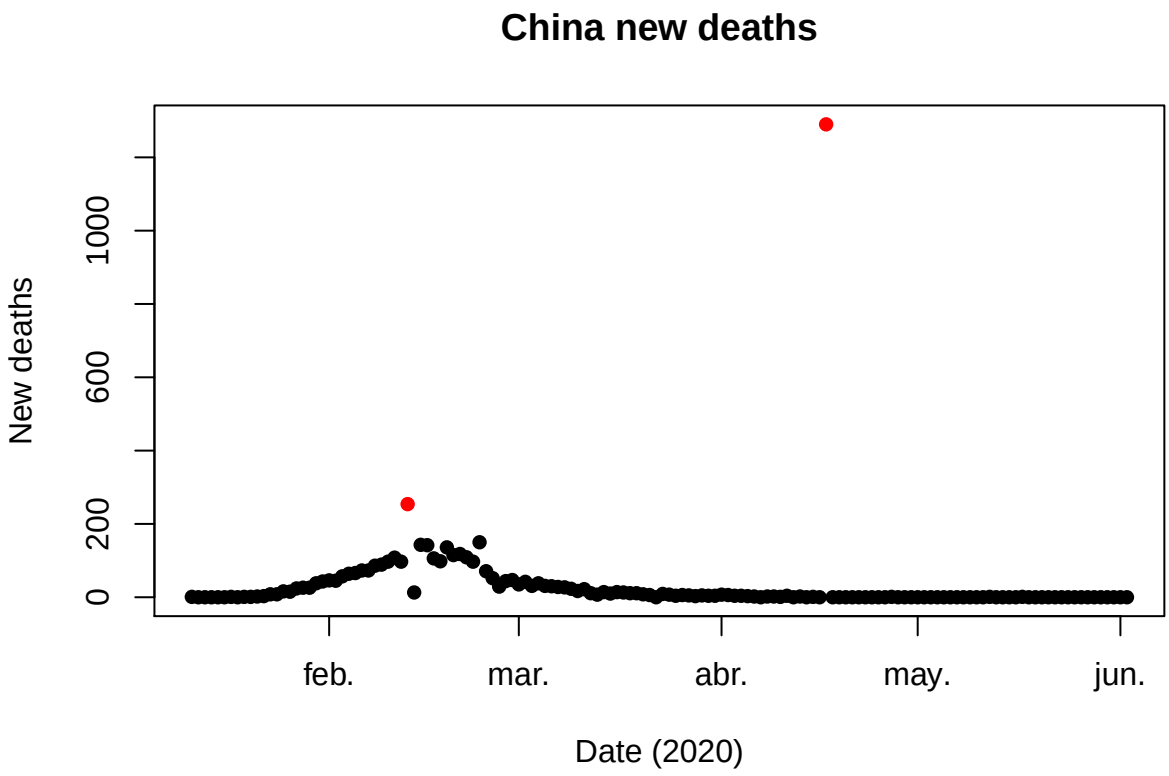

```
plot(Spain$Date_reported,Spain$New_cases,main="Spain new cases",
     xlab="Date (2020)", ylab="New cases",pch = 16)
```



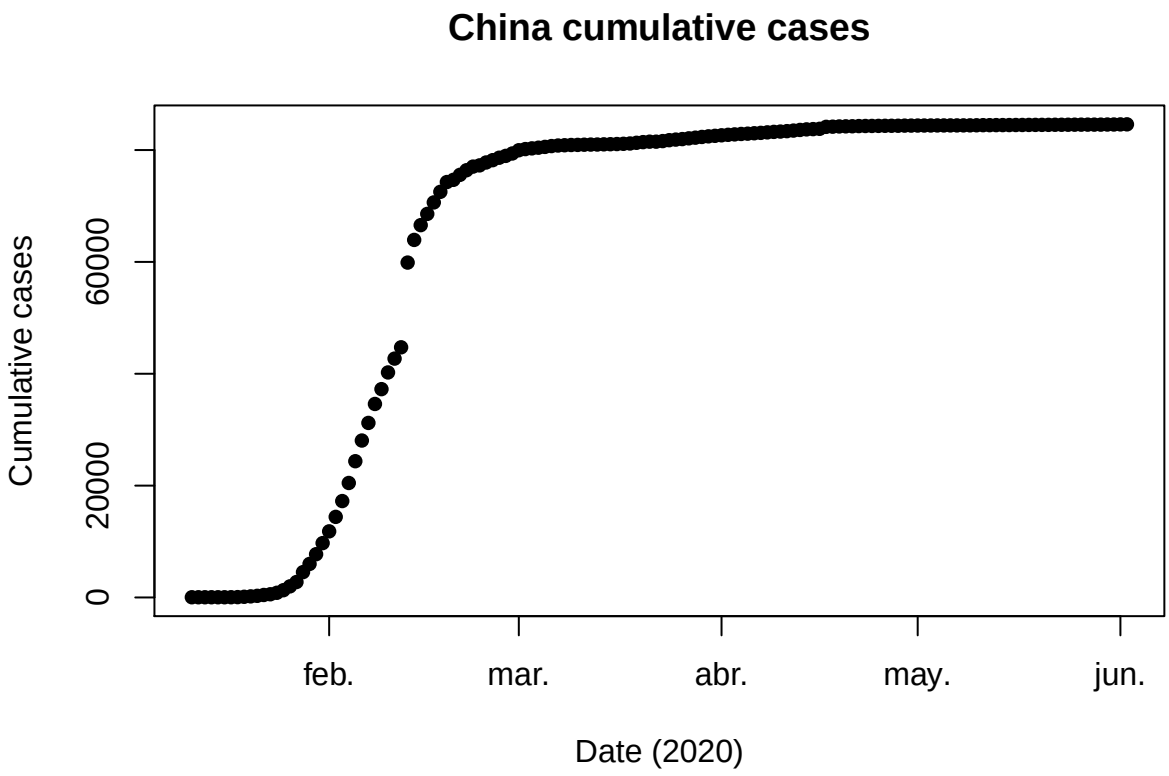

```
China<-filter(covid_dataset, Country=="China")
China$Colour="black"
China$Colour[China$New_deaths>=1000]="red"
China$Colour[China$New_cases>=10000]="red"
plot(China$Date_reported,China$Cumulative_deaths,main="China cumulative deaths",
     xlab="Date (2020)", ylab="Cumulative deaths",pch = 16)
```



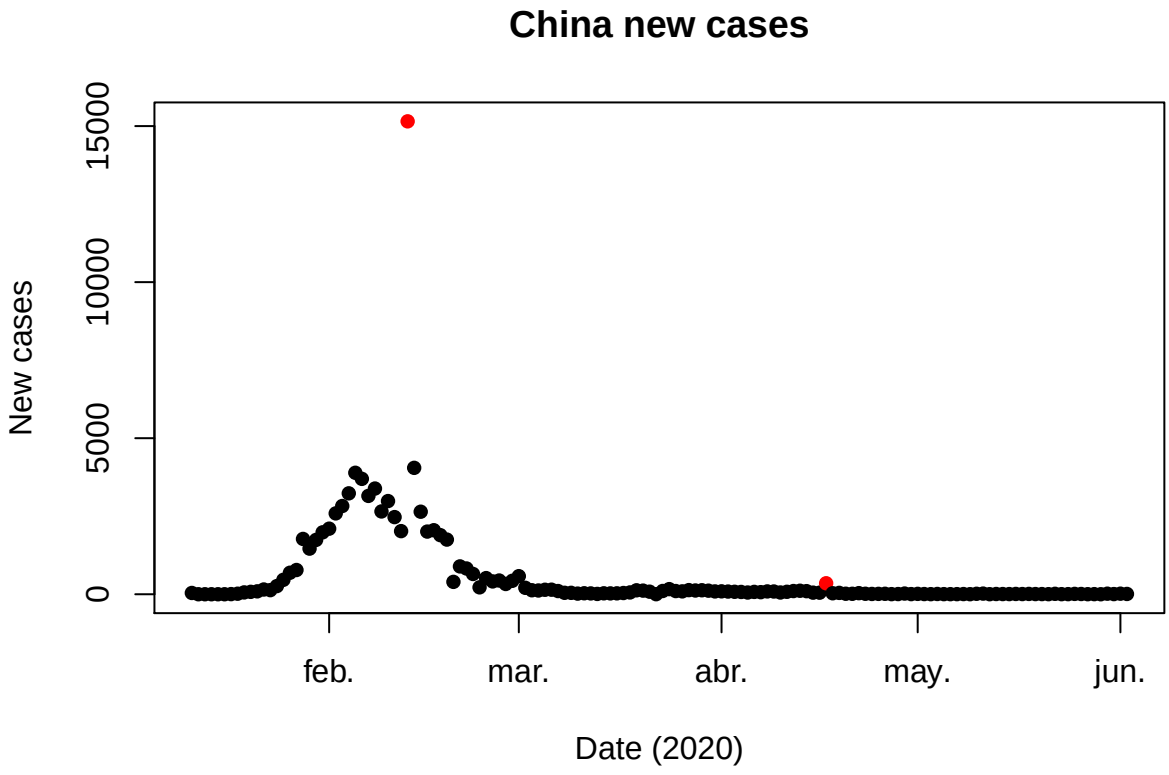

```
plot(China$Date_reported,China$New_deaths,main="China new deaths",
     xlab="Date (2020)", ylab="New deaths",col=China$Colour,pch = 16)
```



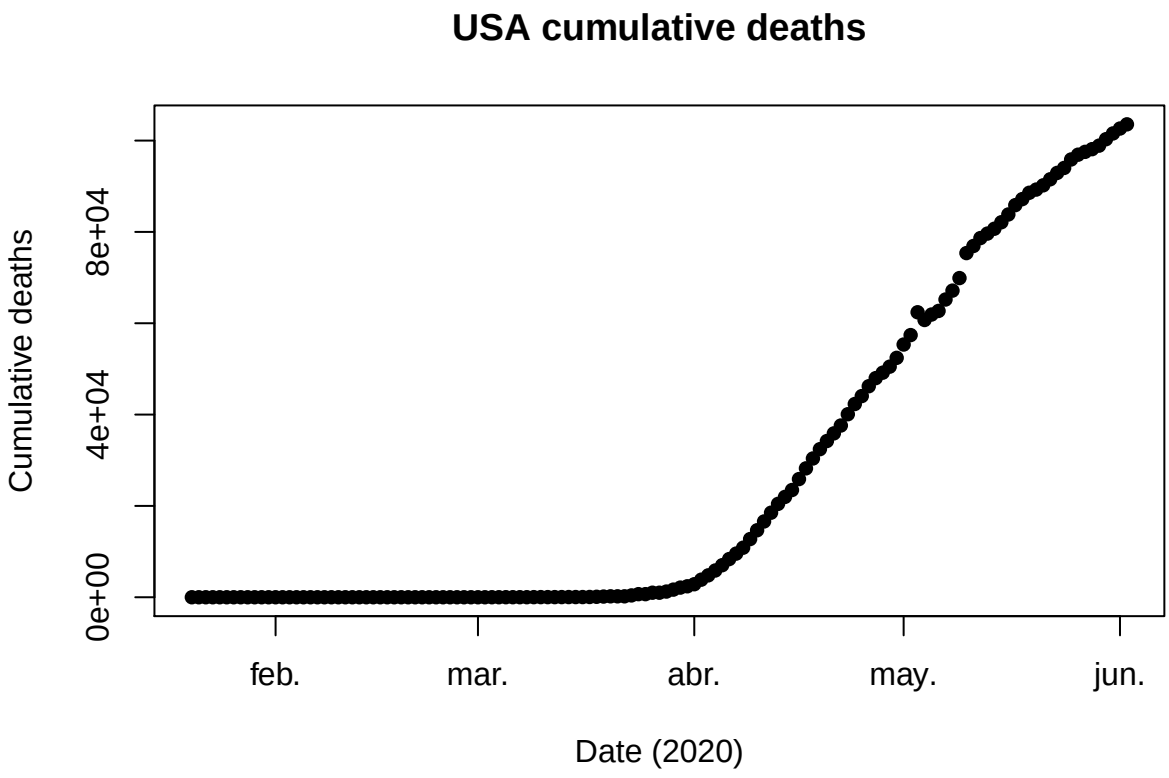

```
plot(China$Date_reported,China$Cumulative_cases,main="China cumulative cases",
     xlab="Date (2020)", ylab="Cumulative cases",pch = 16)
```



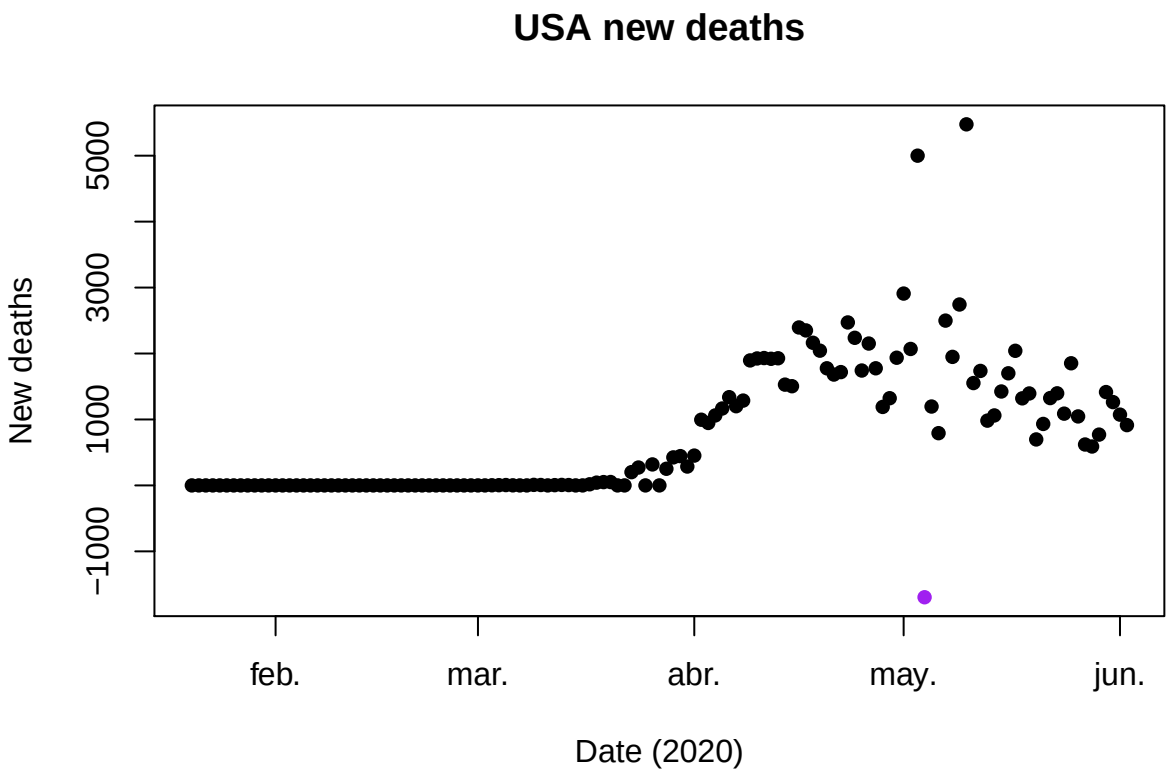

```
plot(China$Date_reported,China$New_cases,main="China new cases",
     xlab="Date (2020)", ylab="New cases",col=China$Colour,pch = 16)
```



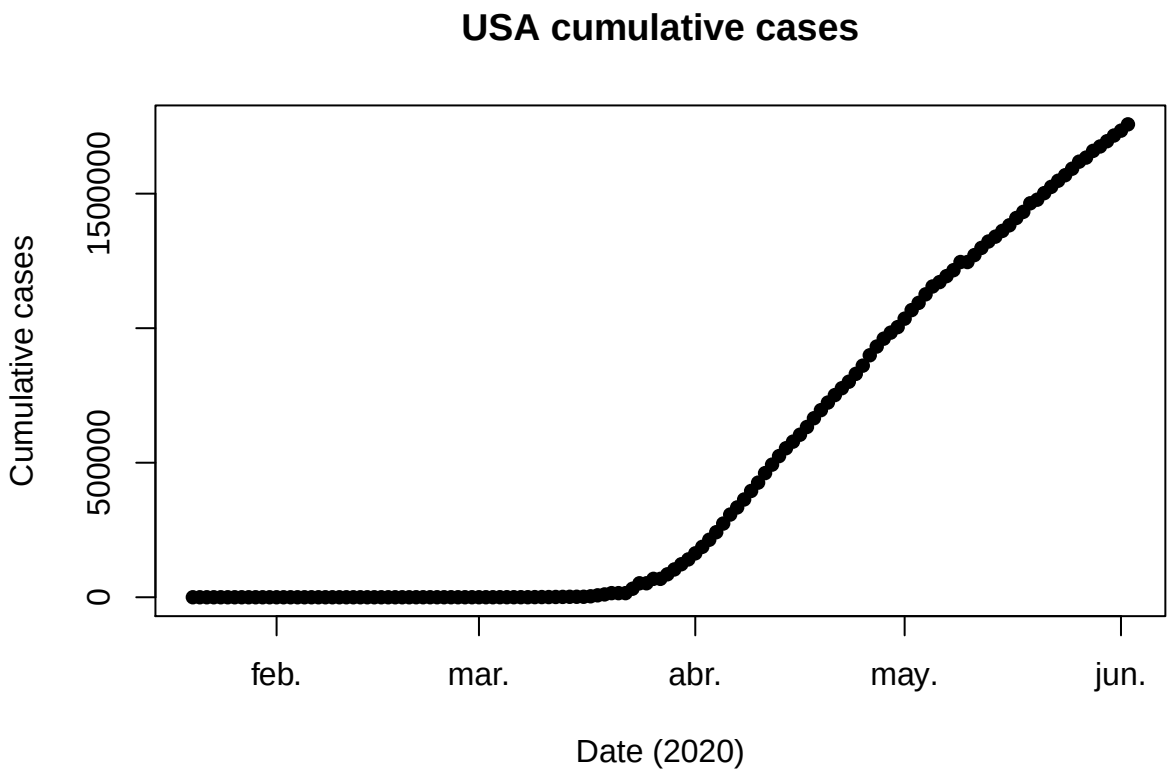

```
USA<-filter(covid_dataset, Country=="United States of America")
USA$Colour="black"
USA$Colour[USA$New_deaths<0]="purple"
plot(USA$Date_reported,USA$Cumulative_deaths,main="USA cumulative deaths",
     xlab="Date (2020)", ylab="Cumulative deaths",pch = 16)
```



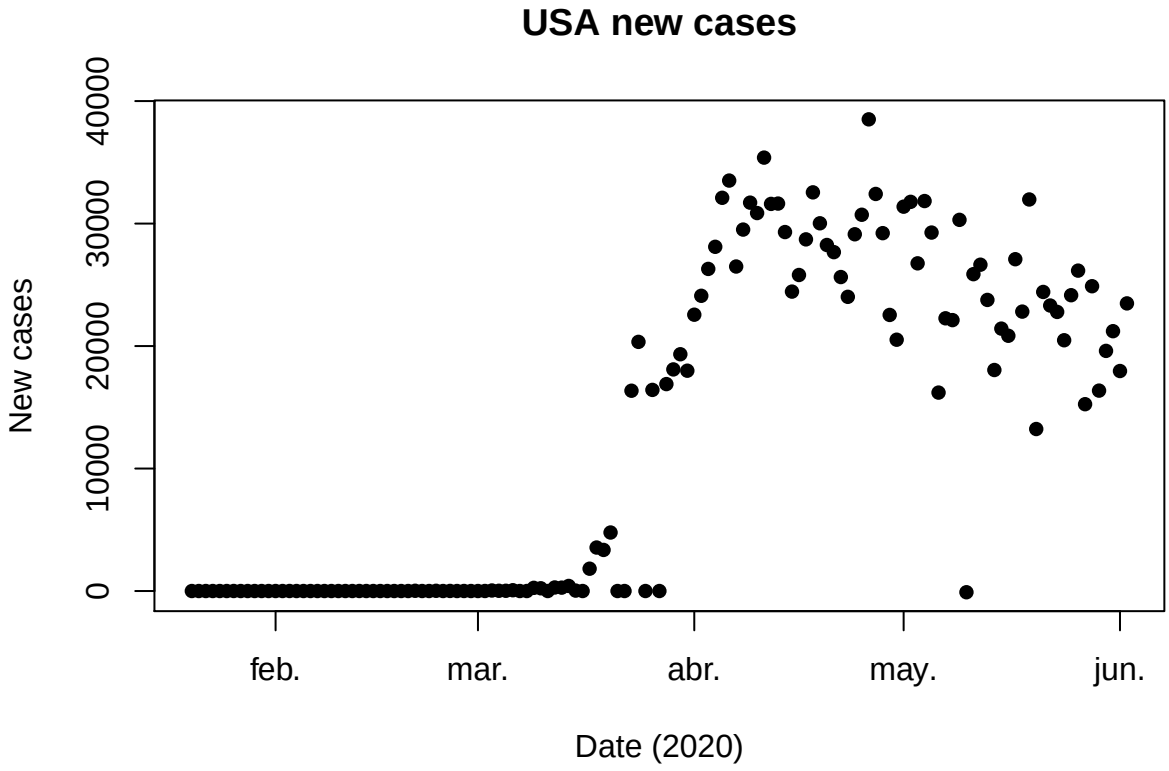

```
plot(USA$Date_reported,USA$New_deaths,main="USA new deaths",
     xlab="Date (2020)", ylab="New deaths",col=USA$Colour,pch = 16)
```




```
plot(USA$Date_reported,USA$Cumulative_cases,main="USA cumulative cases",
     xlab="Date (2020)", ylab="Cumulative cases",pch = 16)
```




```
plot(USA$Date_reported,USA$New_cases,main="USA new cases",
     xlab="Date (2020)", ylab="New cases",pch = 16)
```



As there are many countries, it has simply been decided to analyze the plots of Spain, China and the USA as an example. The following conclusions can be drawn:

Spain:
- Peak of infections and deaths in April
- First infections registered at the end of February
- First deaths in mid-March
- Contagion / death curve stabilized in late May - early June
- 2 data of new deaths that do not correspond to reality can be seen (one value is negative). This migth be a WHO adjustment

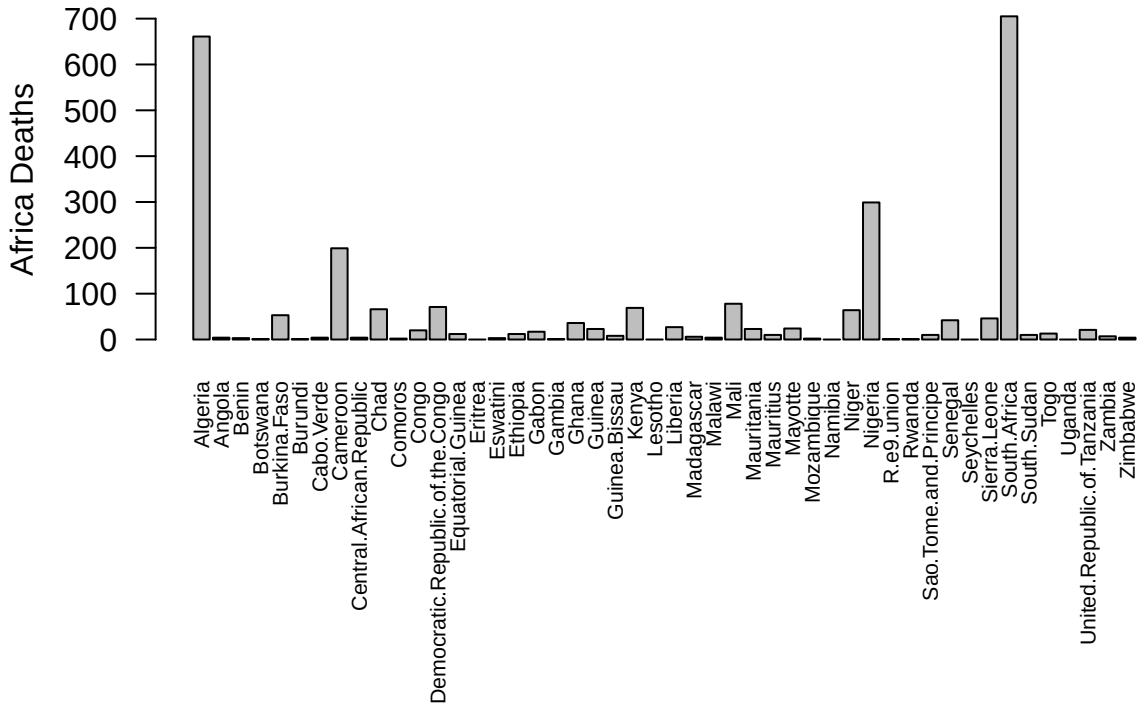
China:
- Peak of infections and deaths in February
- First infections registered in mid-January
- First deaths in late January
- Contagion / death curve stabilized between March and April
- You can see data on new deaths and new cases that do not correspond to reality. This will be a WHO adjustment
- The step in the cumulative deaths curve may be due to new contagion measurement systems or deaths from covid-19 not classified as such

USA:
- Peak of infections and deaths still to be determined

- First infections registered at the end of March
- First deaths in late March
- Contagion / death curve not stabilized

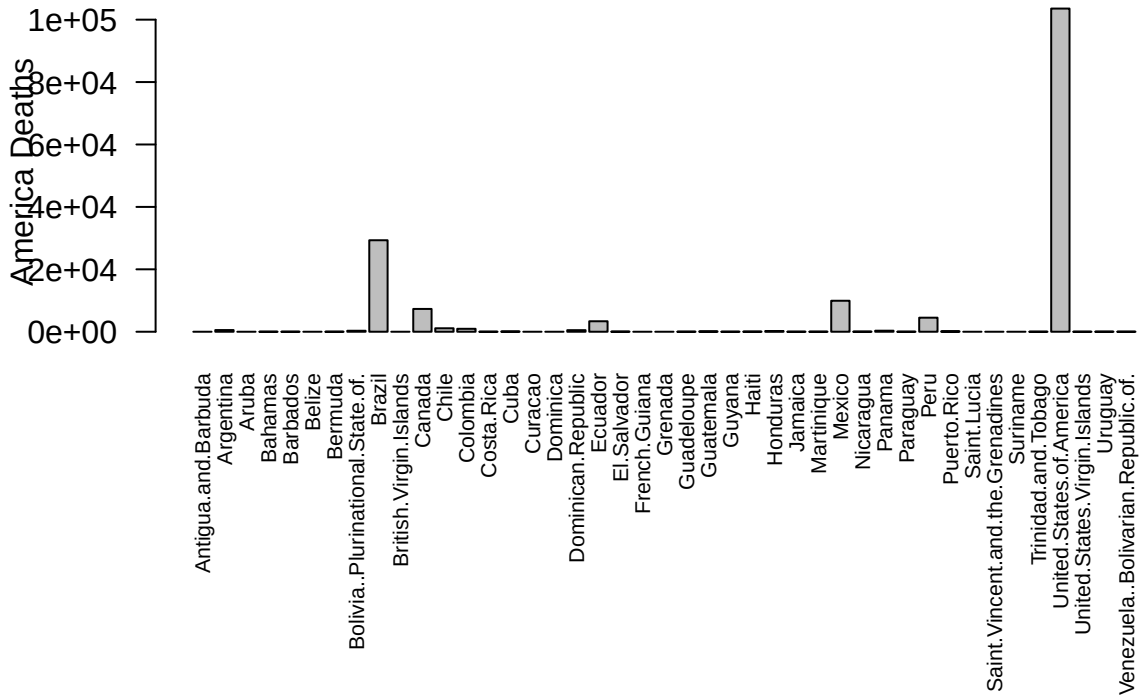
- Distribution of deaths by region:

```
#Group by region, mean of deaths
Africa_region<-covid_dataset %>% filter(WHO_region=="AFRO") %>%
  group_by(Country) %>% summarise(Deaths=max(Cumulative_deaths), Ratio=max(ratio), LE=mean(Life
Africa_region$Country<-make.names(Africa_region$Country)
op <- par(mar=c(10,4,4,2))
barplot(Africa_region$Deaths,
        ylab="Africa Deaths",
        axisnames = TRUE,
        names.arg=Africa_region$Country,
        cex.names=0.7,
        las=2)
```



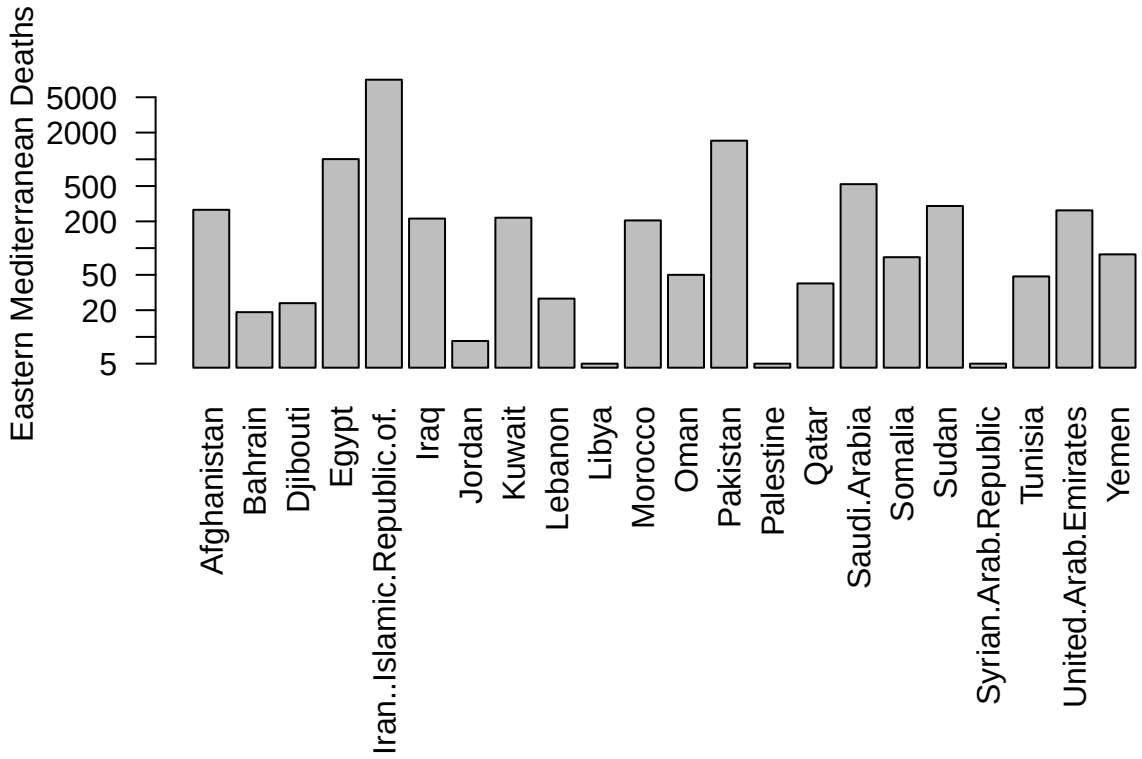

```
rm(op)
region_deaths <- data_frame(Region = "Africa",
                            Deaths = sum(Africa_region$Deaths))
```

```
America_region<-covid_dataset %>% filter(WHO_region=="AMRO") %>%
  group_by(Country) %>% summarise(Deaths=max(Cumulative_deaths), Ratio=max(ratio), LE=mean(Life
America_region$Country<-make.names(America_region$Country)
op <- par(mar=c(10,4,4,2))
barplot(America_region$Deaths,
        ylab="America Deaths",
        axisnames = TRUE,
        names.arg=America_region$Country,
        cex.names=0.7,
        las=2)
```

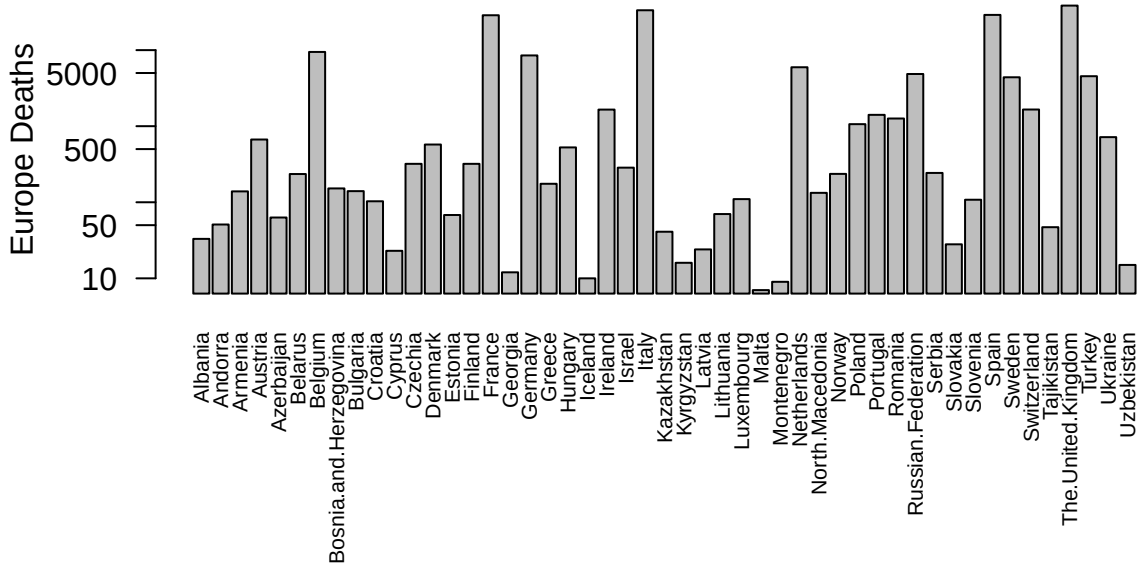



```
rm(op)
region_deaths <- bind_rows(region_deaths,
                           data_frame(Region = "America",
                                      Deaths = sum(America_region$Deaths)))

Eastern_Mediterranean_region<-covid_dataset %>% filter(WHO_region=="EMRO") %>%
  group_by(Country) %>% summarise(Deaths=max(Cumulative_deaths), Ratio=max(ratio), LE=mean(Life
Eastern_Mediterranean_region$Country<-make.names(Eastern_Mediterranean_region$Country)
op <- par(mar=c(11,4,4,2))
barplot(Eastern_Mediterranean_region$Deaths,
```

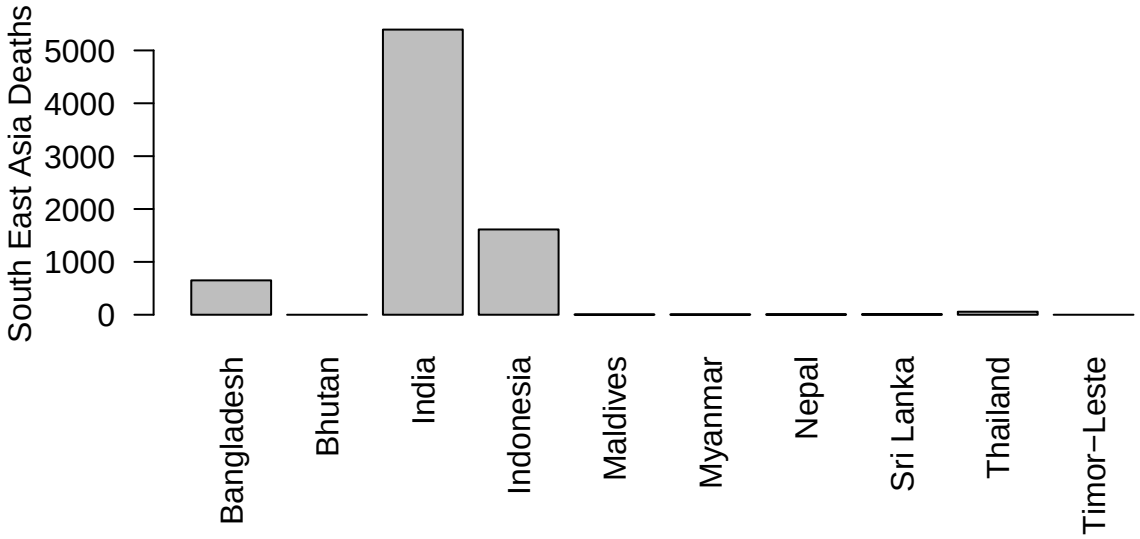
```
        log="y",
        ylab="Eastern Mediterranean Deaths",
        axisnames = TRUE,
        names.arg=Eastern_Mediterranean_region$Country,
        las=2)
```



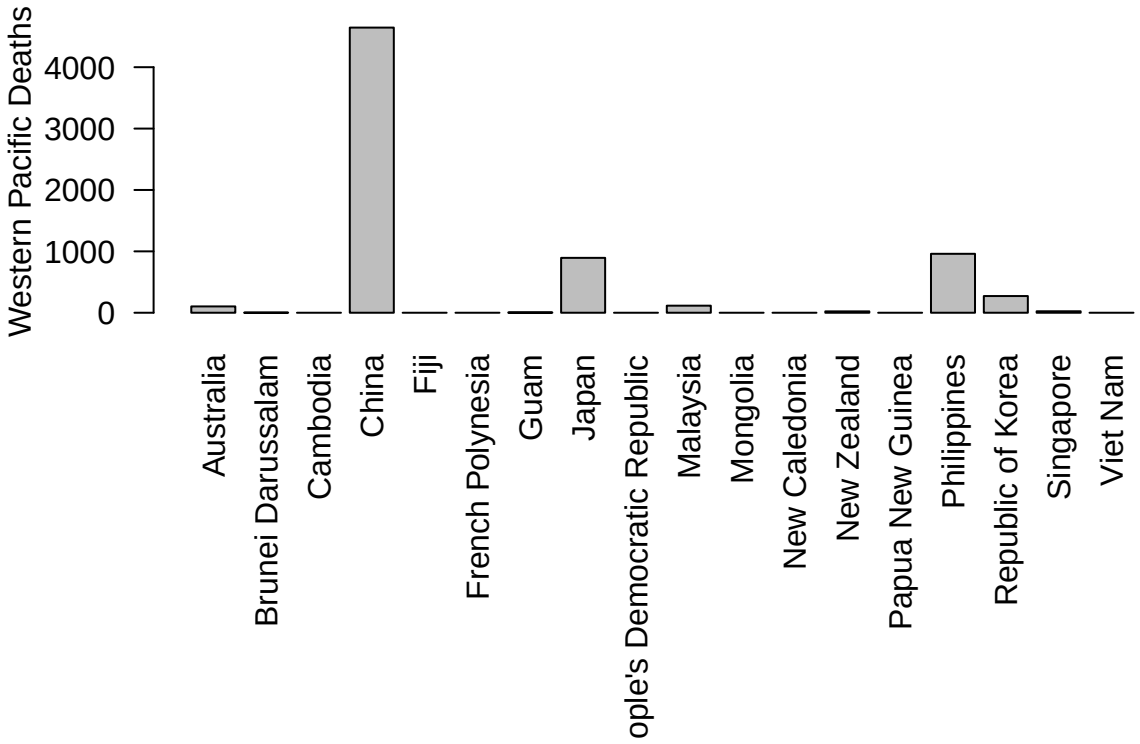

```
rm(op)
region_deaths <- bind_rows(region_deaths,
                           data_frame(Region = "East Mediterranean",
                                      Deaths = sum(Eastern_Mediterranean_region$Deaths)))

Europe_region<-covid_dataset %>% filter(WHO_region=="EURO") %>%
  group_by(Country) %>% summarise(Deaths=max(Cumulative_deaths), Ratio=max(ratio), LE=mean(Life
Europe_region$Country<-make.names(Europe_region$Country)
op <- par(mar=c(11,4,4,2))
barplot(Europe_region$Deaths,
        log="y",
        ylab="Europe Deaths",
        axisnames = TRUE,
        names.arg=Europe_region$Country,
        cex.names=0.7,
        las=2)
```




```
rm(op)
region_deaths <- bind_rows(region_deaths,
                           data_frame(Region = "Europe",
                                      Deaths = sum(Europe_region$Deaths)))

South_East_Asia_region<-covid_dataset %>% filter(WHO_region=="SEARO") %>%
  group_by(Country) %>% summarise(Deaths=max(Cumulative_deaths), Ratio=max(ratio), LE=mean(Life
op <- par(mar=c(11,4,4,2))
barplot(South_East_Asia_region$Deaths,
        ylab="South East Asia Deaths",
        axisnames = TRUE,
        names.arg=South_East_Asia_region$Country,
        las=2)
```




```
rm(op)
region_deaths <- bind_rows(region_deaths,
                           data_frame(Region = "South East Asia",
                                      Deaths = sum(South_East_Asia_region$Deaths)))

Western_Pacific_region<-covid_dataset %>% filter(WHO_region=="WPRO") %>%
  group_by(Country) %>% summarise(Deaths=max(Cumulative_deaths), Ratio=max(ratio), LE=mean(Life
op <- par(mar=c(11,4,4,2))
barplot(Western_Pacific_region$Deaths,
        ylab="Western Pacific Deaths",
        axisnames = TRUE,
        names.arg=Western_Pacific_region$Country,
        las=2)
```




```
rm(op)
region_deaths <- bind_rows(region_deaths,
                           data_frame(Region = "West Pacific",
                                      Deaths = sum(Western_Pacific_region$Deaths)))

region_deaths <- bind_rows(region_deaths,
                           data_frame(Region = "TOTAL",
                                      Deaths = sum(region_deaths$Deaths)))
region_deaths %>% knitr::kable()
```

Region	Deaths
Africa	2667
America	162534
East Mediterranean	12899
Europe	180165
South East Asia	7743
West Pacific	7042
TOTAL	373050

In the summary table it can be seen that most of the deaths correspond to the areas of Europe and America, although it must be emphasized that these deaths are not distributed equally in the

different countries that make up that regions.

2 Methods and Analysis

This section presents the different analysis methods that will be used. To see how these methods are used in R, you should look in the next section "Results", which will include the modeling codes for each of the methods.

We are going to try six different methods:
- Linear discriminant analysis (LDA)
- Quadratic discriminant analysis (QDA)
- Logistic regression (GLM)
- K-nearest neighbors (KNN)
- Decision trees
- Regularization

LDA, QDA, GLM and KNN are generative models. These are Methods that model the joint distribution of y and the predictors x.

2.1 Linear discriminant analysis (LDA)

Forcing the assumption that all predictors share the same standard deviations and correlations, the boundary will be a line. For this reason, we call the method linear discriminant analysis (LDA). Follows the next boundary:

$$y = ax + b$$

This method analyzes if there are new cases of Covid-19 through the variable cases_new (1 if there are new cases and 0 if not). The accuracy of this method can be measured by doing the mean of well predicted cases.

2.2 Quadratic discriminant analysis (QDA)

is a version of Naive Bayes in which we assume that the conditional probabilities for the predictors are multivariate normal. Becomes harder to use as the number of predictors increases. The boundary is:

$$y = ax^2 + bx + c$$

This method analyzes if there are new cases of Covid-19 through the variable cases_new (1 if there are new cases and 0 if not). The accuracy of this method can be measured by doing the mean of well predicted cases.

2.3 Logistic regression (GLM)

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In statistics, the logistic model is used to model the probability of a certain class or event existing such as pass/fail or win/lose. The boundary will be a line, just as with linear discriminant analysis:

$$y = ax + b$$

This method analyzes if there are new cases of Covid-19 through the variable cases_new (1 if there are new cases and 0 if not). The accuracy of this method can be measured by doing the mean of well predicted cases.

2.4 K-nearest neighbors (KNN)

K-nearest neighbors (kNN) estimates the conditional probabilities in a similar way to bin smoothing. However, kNN is easier to adapt to multiple dimensions. Using kNN, for any point for which we want an estimate of it, we look for the k nearest points to the reference point and take an average of the 0s and 1s associated with these points. We refer to the set of points used to compute the average as the neighborhood. Larger values of k result in smoother estimates, while smaller values of k result in more flexible and more wiggly estimates.

This method analyzes if there are new cases of Covid-19 through the variable cases_new (1 if there are new cases and 0 if not). The accuracy of this method can be measured by doing the mean of well predicted cases.

2.5 Decision trees

Decision trees ,or Classification trees, are used in prediction problems where the outcome is categorical.Decision trees form predictions by calculating which class is the most common among the training set observations within the partition, rather than taking the average in each partition.

This method analyzes if there are new cases of Covid-19 through the variable cases_new (1 if there are new cases and 0 if not). The accuracy of this method can be measured by doing the mean of well predicted cases.

2.6 Regularization

This method has been studied in a previous course and provides a good approximation because it takes into account the predictors effects.

It also improves results constrainning the total variability of the effect sizes by penalizing large estimates that come from small sample sizes. In other words, we penalize countries and dates that have few ratios. This is what we call **regularization**. This method compares predicted ratios with real ratios.

The formula of the predicted ratios using this approach is:

$$\hat{y}_{c,d} = \hat{\mu} + \hat{b}_c + \hat{b}_d$$

Where:

$$\hat{b}_c = \frac{1}{n_c + \lambda} \sum_{c=1}^{n_c} (y_{c,d} - \hat{\mu})$$

$$\hat{b}_d = \frac{1}{n_d + \lambda} \sum_{d=1}^{n_d} (y_{c,d} - \hat{b}_c - \hat{\mu})$$

And being:
$\hat{\mu}$ = average of the training set ratios
n_c = number of ratios of the country c
n_d = number of ratios on the day d
$\hat{b}_c$ = country ratio
$\hat{b}_d$ = date ratio
λ = regularization tuning parameter that penalizes countries and dates that have few ratios

To measure the precision of this method, we compute the Root Mean Square Error (RMSE). RMSE is the standard deviation of the residuals and follows the next formula:

$$RMSE = \sqrt{\frac{1}{N} \sum_{c,d} (y_{c,d} - \hat{y}_{c,d})^2}$$

Being:
N = number of ratios
c = country c
d = date d
$\hat{y}_{c,d}$ = predicted ratio for country c and date d
$y_{c,d}$ = real ratio for country c and date d

Here is the code:

```
#Residual Means Squared Errorfunction
RMSE<-function(true_ratings, predicted_ratings){

  sqrt(mean((true_ratings - predicted_ratings)^2))

}
```

3 Results

This section, like the previous one, is made up of the six methods. Here we can find not only the results but the code.

First we prepare the training set and the test set. We include the categorical variable cases_new (1 if there are new cases and 0 if not) and then we remove those varibles that are related to others.

```
train_set<-train_set%>%mutate(cases_new=1*(New_cases>0))
train_set$cases_new<-as.factor(train_set$cases_new)
train_set_m<-train_set%>%select(-New_cases)%>%select(-Female_LE)%>%
  select(-Male_LE)%>%select(-Density_Mi)%>%select(-Population)%>%select(-Area_km2)%>%
  select(-ratio)

test_set<-test_set%>%mutate(cases_new=1*(New_cases>0))
test_set$cases_new<-as.factor(test_set$cases_new)
test_set_m<-test_set%>%select(-New_cases)%>%select(-Female_LE)%>%
  select(-Male_LE)%>%select(-Density_Mi)%>%select(-Population)%>%select(-Area_km2)%>%
  select(-ratio)
```

3.1 LDA results

We use all the predictors in the LDA method.

```
set.seed(1, sample.kind = "Rounding")
train_lda <- train(cases_new ~ ., method = "lda", data = train_set_m)
lda_preds <- predict(train_lda, test_set_m)
lda_ac<-mean(lda_preds == test_set_m$cases_new)
results <- data_frame(Method = "LDA",
                            Accuracy = lda_ac)
```

The accuracy of this method is **0.77746**

3.2 QDA results

We use only the life expectancy as a predictor because trying other predictors we get a lower accuracy.

```
set.seed(1, sample.kind = "Rounding")
train_qda <- train(cases_new ~ Life_Expectancy, method = "qda", data = train_set_m)
qda_preds <- predict(train_qda, test_set_m)
qda_ac<-mean(qda_preds == test_set_m$cases_new)
results <- bind_rows(results,
                            data_frame(Method = "QDA",
                                       Accuracy = qda_ac))
```

The accuracy of this method is **0.6641876**

3.3 GLM results

We use only the life expectancy as a predictor because trying other predictors we get a lower accuracy.

```r
set.seed(1, sample.kind = "Rounding")
train_glm <- train(cases_new ~ Life_Expectancy, method = "glm", data = train_set_m)
glm_preds <- predict(train_glm, test_set_m)
glm_ac<-mean(glm_preds == test_set_m$cases_new)
results <- bind_rows(results,
                           data_frame(Method = "GLM",
                                      Accuracy = glm_ac))
```

The accuracy of this method is **0.6707666**

3.4 KNN results

We use only the population density as a predictor because trying other predictors we get a lower accuracy.

```r
set.seed(1, sample.kind = "Rounding")
train_knn <- train(cases_new ~ Density_km,
                   method = "knn",
                   data = train_set_m,
                   tuneGrid = data.frame(k = seq(3, 51, 2)))
train_knn$bestTune
```

```
##   k
## 1 3
```

```r
max(train_knn$results$Accuracy)
```

```
## [1] 0.7707367
```

```r
knn_preds <- predict(train_knn, test_set_m)
knn_ac<-mean(knn_preds == test_set_m$cases_new)
results <- bind_rows(results,
                           data_frame(Method = "KNN",
                                      Accuracy = knn_ac))
```

The accuracy of this method is **0.7725973**

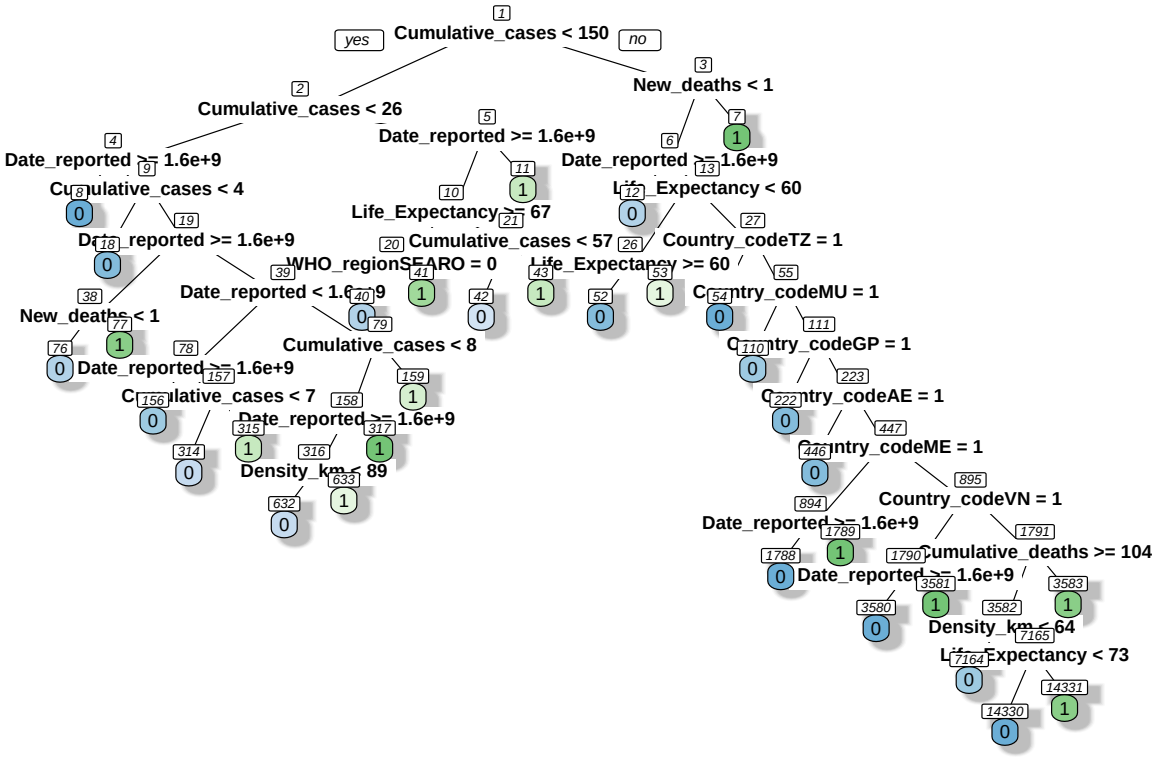
3.5 Decision trees results

We use all the predictors in the decision trees method.

```
set.seed(1, sample.kind = "Rounding")
train_rpart <- train(cases_new ~ .,
                     method = "rpart",
                     tuneGrid = data.frame(cp = seq(0, 0.05, 0.002)),
                     data = train_set_m)
train_rpart$bestTune
```

```
##      cp
## 2 0.002
```

```
rpart_preds <- predict(train_rpart, test_set_m)
rpart_ac<-mean(rpart_preds == test_set_m$cases_new)
results <- bind_rows(results,
                           data_frame(Method = "Decision trees",
                                      Accuracy = rpart_ac))
rpart.plot(train_rpart$finalModel, type=0,fallen.leaves=FALSE,
           clip.facs=TRUE, shadow.col="gray", nn=TRUE, cex=0.5, extra=0,tweak=1.2)
```



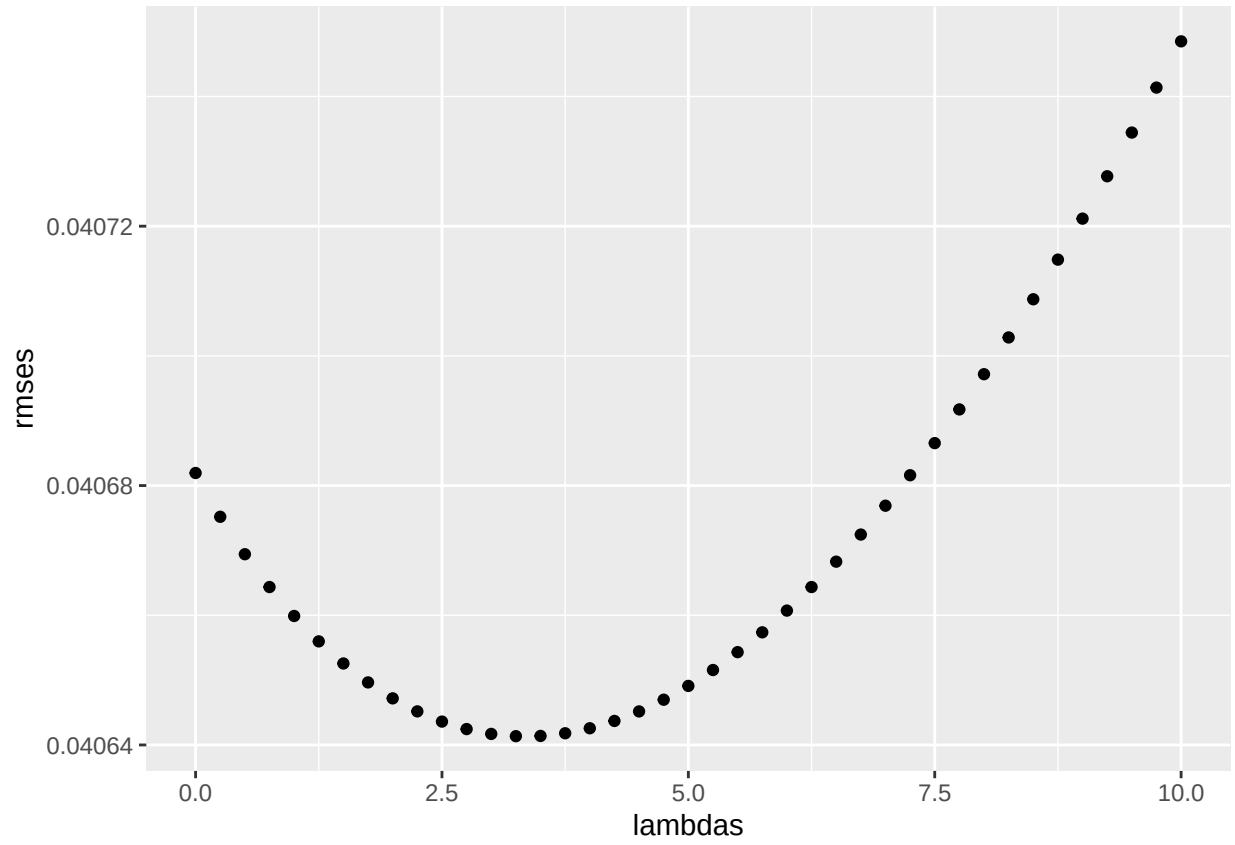
The accuracy of this method is **0.8223684**

Advice: the image corresponding to the decision tree can be seen more clearly if you run the code from Rstudio and choose the zoom option in the plots tab.

3.6 Regularization results

This code runs the regularization model, tuning lambda to optimize the RMSE.

```
#Calculte RMSE with best lambda
lambdas <- seq(0, 10, 0.25)
rmses <- sapply(lambdas, function(l){
  mu <- mean(train_set$ratio)
  b_c <- train_set %>%
    group_by(Country) %>%
    summarize(b_c = sum(ratio - mu)/(n()+l))
  b_d <- train_set %>%
    left_join(b_c, by="Country") %>%
    group_by(Date_reported) %>%
    summarize(b_d = sum(ratio - b_c - mu)/(n()+l))
  predicted_data <-
    test_set %>%
    left_join(b_c, by = "Country") %>%
    left_join(b_d, by = "Date_reported") %>%
    mutate(pred = mu + b_c + b_d)
  return(RMSE(predicted_data$pred, test_set$ratio))
})
qplot(lambdas, rmses)
```



```
lambda <- lambdas[which.min(rmses)]
lambda
```

```
## [1] 3.25
```

```
model_rmse<-min(rmses)

mu <- mean(train_set$ratio)
b_c <- train_set %>%
  group_by(Country) %>%
  summarize(b_c = sum(ratio - mu)/(n()+lambda))
b_d <- train_set %>%
  left_join(b_c, by="Country") %>%
  group_by(Date_reported) %>%
  summarize(b_d = sum(ratio - b_c - mu)/(n()+lambda))
predicted_data <-
  test_set %>%
  left_join(b_c, by = "Country") %>%
  left_join(b_d, by = "Date_reported") %>%
  mutate(pred = mu + b_c + b_d)
RMSE(predicted_data$pred, test_set$ratio)
```

```
## [1] 0.04064136
```

```
results <- bind_rows(results,
                           data_frame(Method = "Regularization",
                                      RMSE = model_rmse))
```

The RMES using regularization is **0.0406414**.

3.7 Results summary

Here is the comparison of the results. The behavior of all methods is measured in precision except for regularization, which is measured in RMSE:

```
results %>% knitr::kable()
```

Method	Accuracy	RMSE
LDA	0.7774600	NA
QDA	0.6641876	NA
GLM	0.6707666	NA
KNN	0.7725973	NA
Decision trees	0.8223684	NA
Regularization	NA	0.0406414

4 Conclusions

After testing all the generative models, we can conclude that the model that best fits and therefore the most optimal is the decision tree method.

However, we cannot assume that this model has a better precision than the regularization method, since they do not predict the same variable and its precision is measured in different ways. The regularization method also behaves efficiently, since its RMSE does not exceed the standard deviation of the variable, which is **0.0516522**.

4.1 Limitations

There are several limitations regarding this project. On the one hand, the data set used is static and is not automatically updated daily, making it obsolete. The difficulty of finding reliable and complete data on cases of Covid-19 infections and deaths also comes into play, as the specific data of, for example, ages, previous pathologies and city of residence of the victims of Covid-19 are not available.

4.2 Other possible models / Future work

Other models that could have been studied in this project will simply be mentioned in this section.

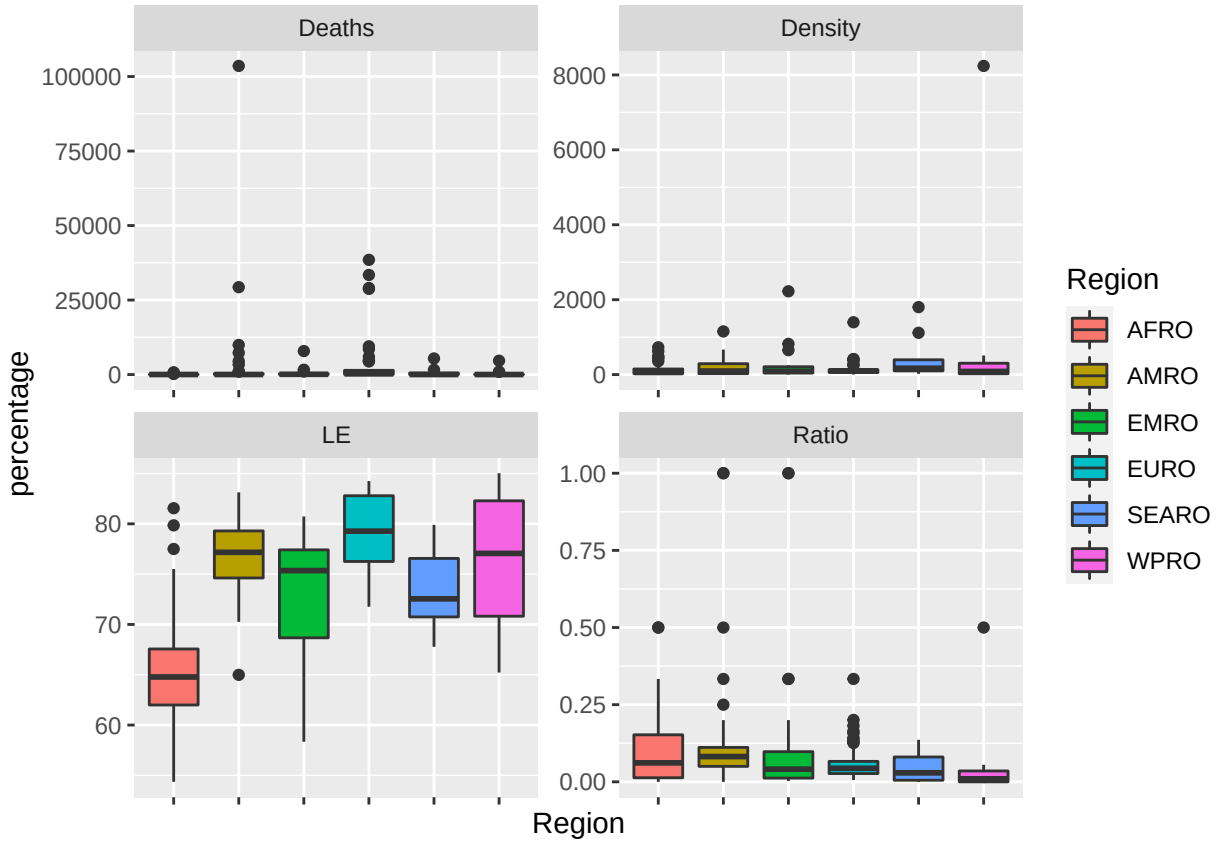
- Region study:

To improve the knn method, a study by regions of the world can be carried out in order to find relationships between them. Due to the uneven spread of the virus across countries, it is difficult to find a relationship and further analysis would have to be carried out.

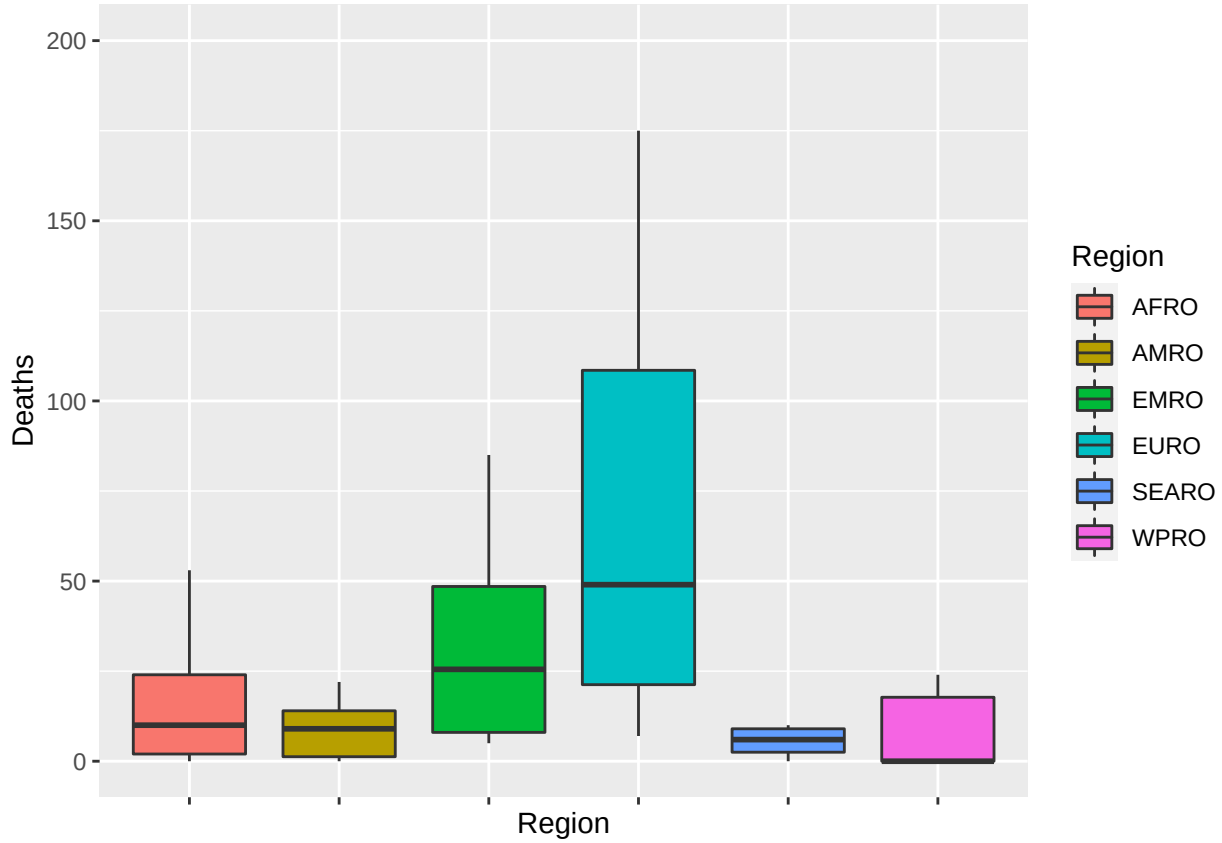
Below are some box diagrams showing the different variables depending on the region. While many of the diagrams contain outliers, some relationship could be drawn if you drill down.

Finally, a scatter chart is shown over its entire range and zooming to see how complex it would be to find any relationship between the different regions.

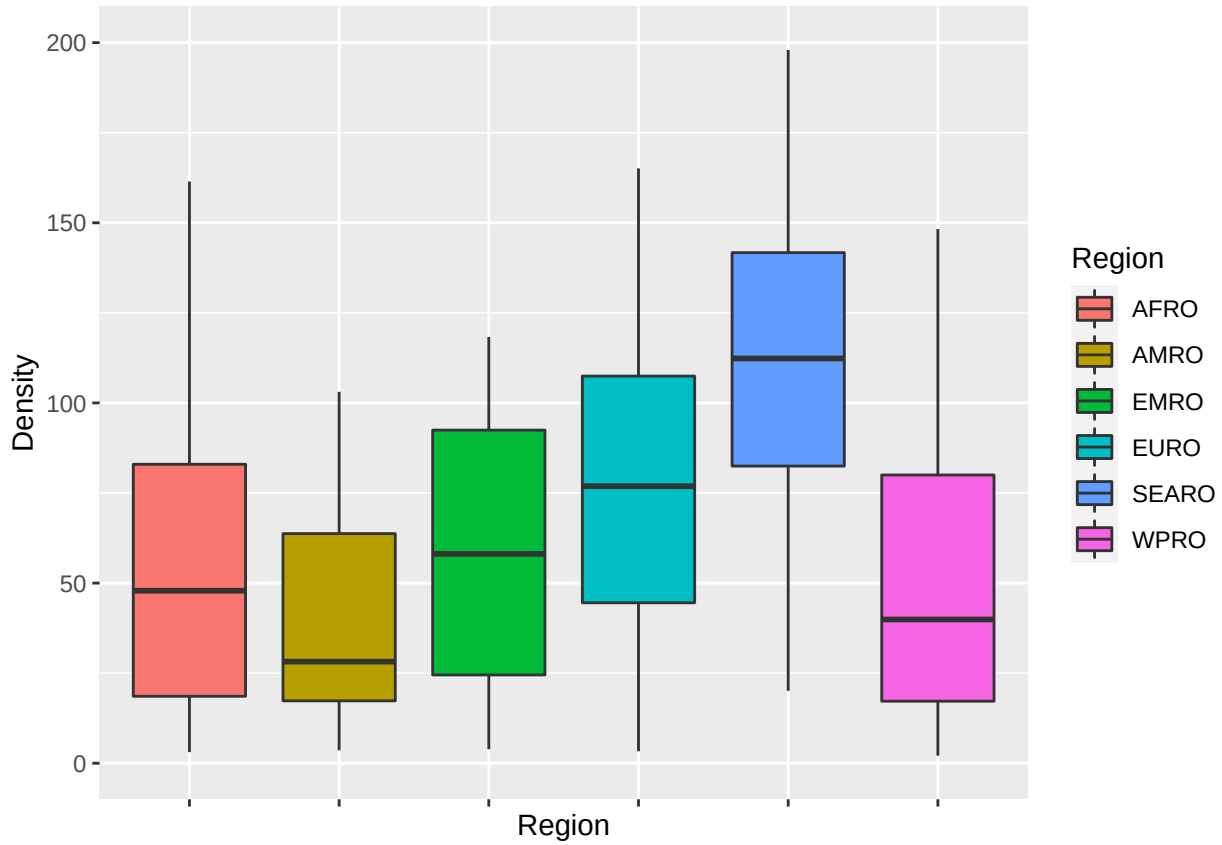
```
covid_summary<-Africa_region %>%
  full_join(America_region) %>%
  full_join(Eastern_Mediterranean_region) %>%
  full_join(Europe_region) %>%
  full_join(South_East_Asia_region) %>%
  full_join(Western_Pacific_region)

cov <- subset(covid_summary, select = -Country )

cov %>% gather(Variables,percentage, -Region) %>%
  ggplot(aes(Region, percentage, fill = Region)) +
  geom_boxplot() +
  facet_wrap(~Variables, scales = "free") +
  theme(axis.text.x = element_blank())
```



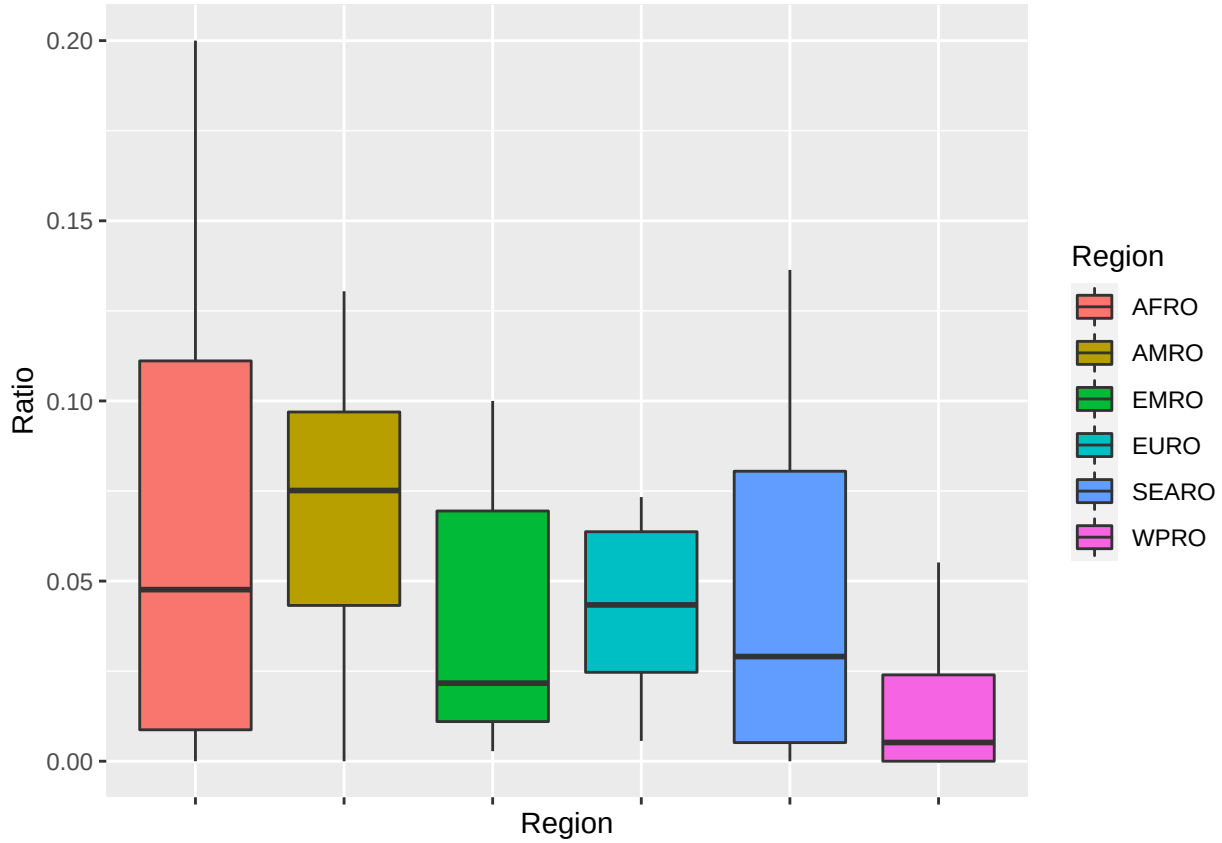

```
cov %>%
  ggplot(aes(Region, Deaths, fill = Region)) +
  geom_boxplot(outlier.shape = NA) +
  scale_y_continuous(limits = c(0, 200)) +
  theme(axis.text.x = element_blank())
```



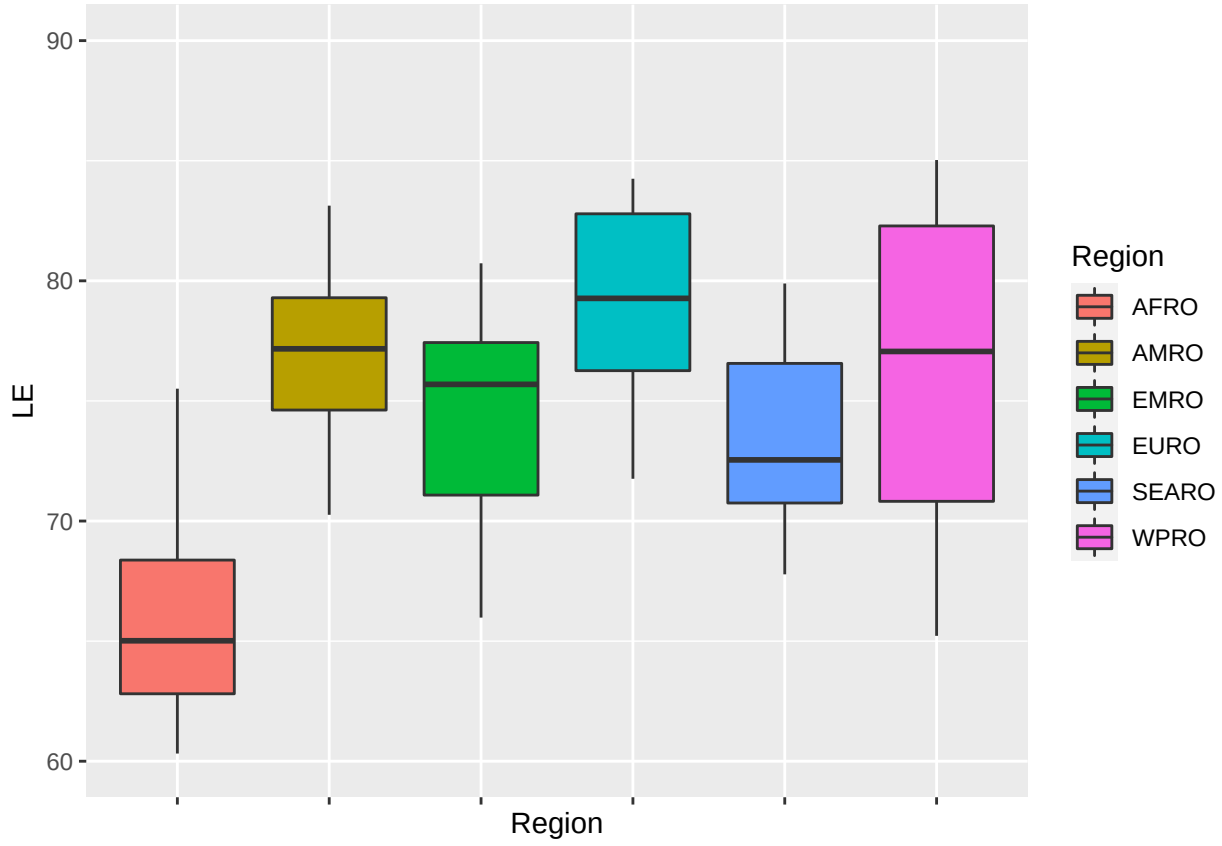

```
cov %>%
  ggplot(aes(Region, Density, fill = Region)) +
  geom_boxplot(outlier.shape = NA) +
  scale_y_continuous(limits = c(0, 200)) +
  theme(axis.text.x = element_blank())
```



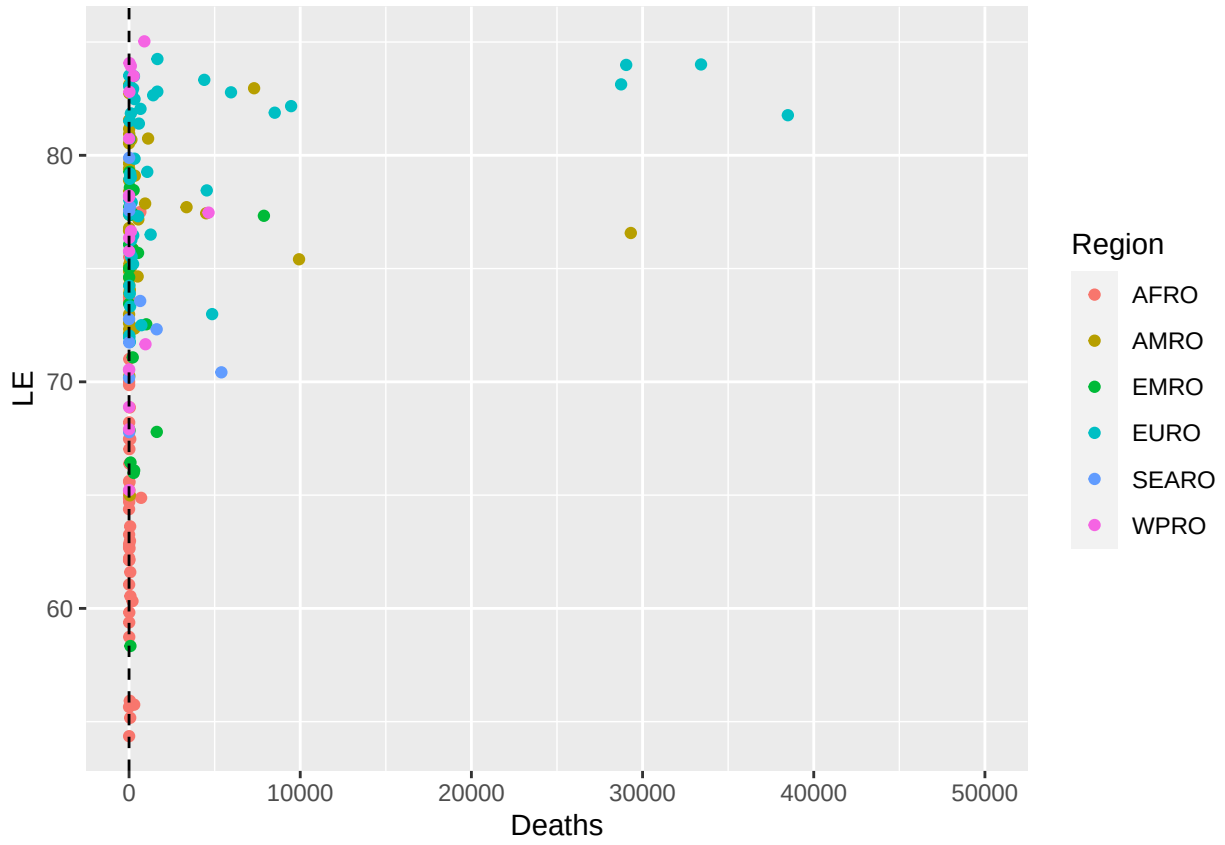

```
cov %>%
  ggplot(aes(Region, Ratio, fill = Region)) +
  geom_boxplot(outlier.shape = NA) +
  scale_y_continuous(limits = c(0, 0.2)) +
  theme(axis.text.x = element_blank())
```



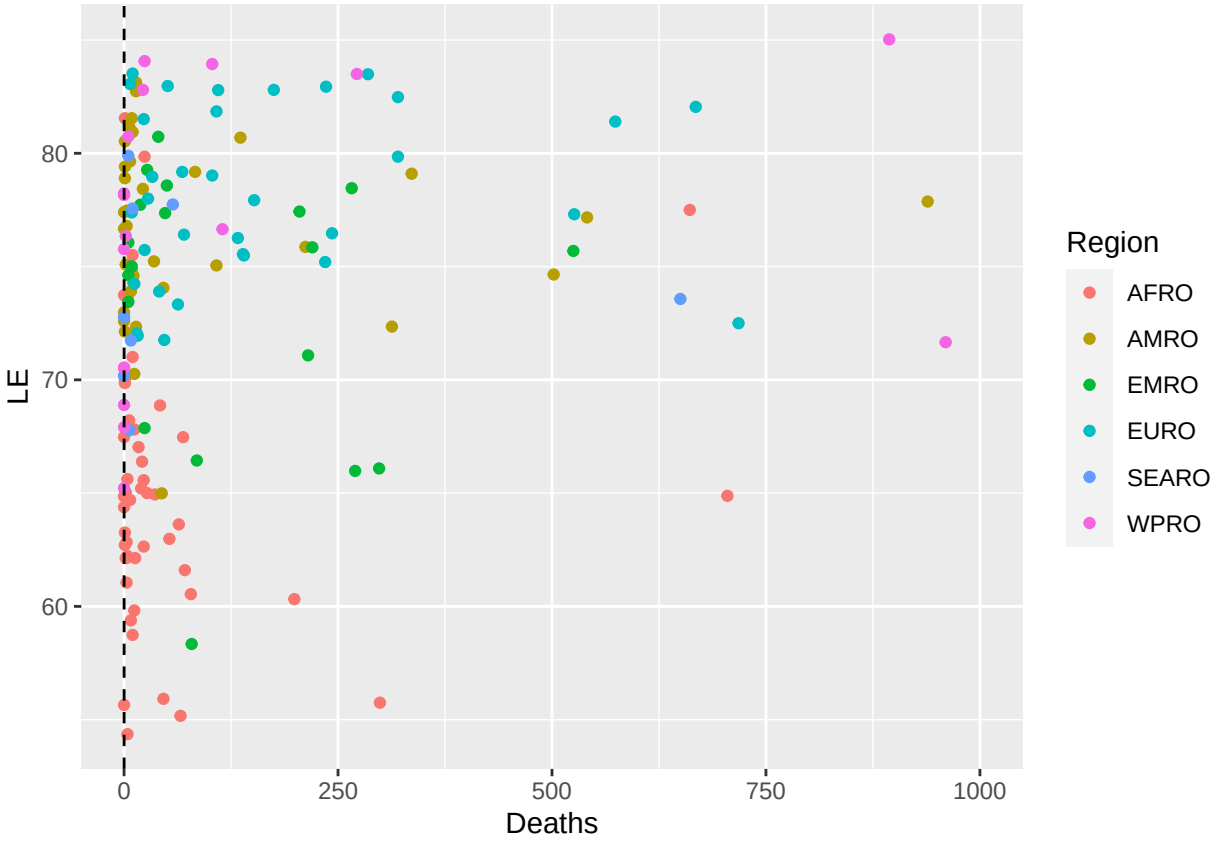

```
cov %>%
  ggplot(aes(Region, LE, fill = Region)) +
  geom_boxplot(outlier.shape = NA) +
  scale_y_continuous(limits = c(60, 90)) +
  theme(axis.text.x = element_blank())
```




```
cov %>%
  ggplot(aes(Deaths,LE, color = Region)) +
  geom_point()+
  geom_vline(xintercept = 0.065, lty = 2) +
  geom_segment(x = -0.2, y = 10.54, xend = 0.065, yend = 10.54, color = "black", lty = 2)+
  scale_x_continuous(limits = c(0, 50000))
```




```
cov %>%
  ggplot(aes(Deaths,LE, color = Region)) +
  geom_point()+
  geom_vline(xintercept = 0.065, lty = 2) +
  geom_segment(x = -0.2, y = 10.54, xend = 0.065, yend = 10.54, color = "black", lty = 2)+
  scale_x_continuous(limits = c(0, 1000))
```



- More complex data:

As mentioned previously, access to more complete and complex data would expand the range of possibilities when carrying out different studies. Having the age of the infected or deceased, as well as possible previous medical pathologies would help to predict new cases or simulate a more realistic curve of the pandemic.

References

All references are included in the report through hyperlinks.