Data Science Capstone Project: Suicide Rates Overview

Orestis Bouras

November 25, 2020

Contents

1 Introduction

This report analyzes worldwide Suicide Rates based on the dataset named "Suicide Rates Overview 1985 to 2016", found in the website "kaggle".In the first part of the report, a thorough analysis and visualization of the data is included.In the second part, a machine learning algorithm is developed using multiple methods such as lda, knn and random forest, in order to predict the Suicide Rate based on several of the provided features.

The main target of this prediction algorithm is to get some useful insights on the possible factors that lead to the suicide action. Furthermore, it could enable the prediction of the suicide rate for countries that do not have an official rate measurement or for future forecasts.

Death by suicide is an extremely complex issue that causes pain to hundreds of thousands of people every year around the world. Every year close to 800.000 people take their own life and there are many more people who attempt suicide. Every suicide is a tragedy that affects families, communities and entire countries and has long-lasting effects on the people left behind. Some of the key facts of Suicide, as stated by the "World Health Organisation" are:

- Close to 800 000 people die due to suicide every year
- Suicide is the third leading cause of death in 15-19-year-olds
- 79% of global suicides occur in low- and middle-income countries
- Ingestion of pesticide, hanging and firearms are among the most common methods of suicide globally

The respective dataset is uploaded in my personal Github account in order to provide public access and it is automatically downloaded in the respective R code. The original dataset was initially cleaned and then divided into two main subsets: "train" and "test". The training of the prediction algorithm takes place based on the "train" set while the evaluation is based on the "test" set. The accuracy is calculated based on this "test" set for each applied prediction method. Accuracy is defined as 'the degree to which the result of a measurement conforms to the correct value or a standard' and it essentially refers to how close a measurement is to it's agreed value.

To sum up, the report is structured as follows. Chapter 1 includes the introduction of the report. Chapter 2 describes the dataset and the data cleaning. Chapter 3 is divided into two sectors. The first sector presents the data exploration and visualization through multiple graphs that provide useful insights for the Suicide Rates dataset. The second sector includes the multiple developed models with the respective accuracy results and discussion on each model's performance. Chapter 4 summarizes the results for each model while Chapter 5 concludes with a brief summary of the report, possible limitations and future work.

2 Dataset

2.1 Data download

The dataset is downloaded from the "kaggle" website under the name "Suicide Rates Overview 1985 to 2016" by "Rusty". The dataset was downloaded in a .csv format and was uploaded in my personal Github account with open public access. The file can be found and downloaded from the following link:

"https://raw.githubusercontent.com/Bouraso/Suicide-Rates/main/Suicide%20Rates%20Overview%201985%20to%202016.csv"

For this report, the dataset is automatically downloaded based on the code presented below through the read.csv function. The code also includes the automatic installation of the required R packages with if(!require) statements and the loading of the respective libraries.

```
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(lubridate)) install.packages("lubridate", repos = "http://cran.us.r-project.org")
if(!require(tidyr)) install.packages("tidyr", repos = "http://cran.us.r-project.org")
if(!require(gridExtra)) install.packages("gridExtra", repos = "http://cran.us.r-project.org")
if(!require(rpart)) install.packages("rpart", repos = "http://cran.us.r-project.org")

library(tidyverse)
library(caret)
library(data.table)
library(lubridate)
library(tidyr)
library(gridExtra)
library(rpart)

suiciderates <- read.csv(file="https://raw.githubusercontent.com/Bouraso/Suicide-Rates/main/Suicide%20Ra
                         header = TRUE, sep = ",",fileEncoding="UTF-8-BOM")
```

2.2 Data cleaning

The original dataset that is read from the csv file is named as "suiciderates". In order to understand the structure of the dataset, the first rows of the raw data are investigated as below:

```
##   country year    sex         age suicides_no population suicides.100k.pop
## 1 Albania 1987   male 15-24 years          21     312900              6.71
## 2 Albania 1987   male 35-54 years          16     308000              5.19
## 3 Albania 1987 female 15-24 years          14     289700              4.83
## 4 Albania 1987   male   75+ years           1      21800              4.59
## 5 Albania 1987   male 25-34 years           9     274300              3.28
## 6 Albania 1987 female   75+ years           1      35600              2.81
##   country.year HDI.for.year gdp_for_year.... gdp_per_capita....      generation
## 1  Albania1987           NA    2,156,624,900               796    Generation X
## 2  Albania1987           NA    2,156,624,900               796          Silent
## 3  Albania1987           NA    2,156,624,900               796    Generation X
## 4  Albania1987           NA    2,156,624,900               796 G.I. Generation
## 5  Albania1987           NA    2,156,624,900               796         Boomers
## 6  Albania1987           NA    2,156,624,900               796 G.I. Generation
```

After the first data visualization, data screening is required. The feature "HDI.for.year" includes mainly NA values, thus it will be removed. Some of the features will be renamed for simplicity reasons,for example, the four dots at the end of the "gdp_for_year" and "gdp_per_capita" will be removed. Finally, the rows including additional NA values will be deleted.

```
suiciderates <- suiciderates%>%rename(gdp_per_capita=gdp_per_capita....,gdp_for_year=gdp_for_year....)
```

```
suiciderates <- suiciderates[-9]
suiciderates <- na.omit(suiciderates)
```

The new structure of the "suiciderates" dataset is represented below:

```
##   country year    sex         age suicides_no population suicides.100k.pop
## 1 Albania 1987   male 15-24 years          21     312900              6.71
## 2 Albania 1987   male 35-54 years          16     308000              5.19
## 3 Albania 1987 female 15-24 years          14     289700              4.83
## 4 Albania 1987   male   75+ years           1      21800              4.59
## 5 Albania 1987   male 25-34 years           9     274300              3.28
## 6 Albania 1987 female   75+ years           1      35600              2.81
##   country.year  gdp_for_year gdp_per_capita      generation
## 1  Albania1987 2,156,624,900            796    Generation X
## 2  Albania1987 2,156,624,900            796          Silent
## 3  Albania1987 2,156,624,900            796    Generation X
## 4  Albania1987 2,156,624,900            796 G.I. Generation
## 5  Albania1987 2,156,624,900            796         Boomers
## 6  Albania1987 2,156,624,900            796 G.I. Generation
```

while the summary of the dataset confirms that there are no missing values:

```
##    country              year          sex                age
##  Length:27820       Min.   :1985   Length:27820       Length:27820
##  Class :character   1st Qu.:1995   Class :character   Class :character
##  Mode  :character   Median :2002   Mode  :character   Mode  :character
##                     Mean   :2001
##                     3rd Qu.:2008
##                     Max.   :2016
##   suicides_no        population       suicides.100k.pop country.year
##  Min.   :    0.0   Min.   :     278   Min.   :  0.00    Length:27820
##  1st Qu.:    3.0   1st Qu.:   97498   1st Qu.:  0.92    Class :character
##  Median :   25.0   Median :  430150   Median :  5.99    Mode  :character
##  Mean   :  242.6   Mean   : 1844794   Mean   : 12.82
##  3rd Qu.:  131.0   3rd Qu.: 1486143   3rd Qu.: 16.62
##  Max.   :22338.0   Max.   :43805214   Max.   :224.97
##  gdp_for_year       gdp_per_capita    generation
##  Length:27820       Min.   :   251   Length:27820
##  Class :character   1st Qu.:  3447   Class :character
##  Mode  :character   Median :  9372   Mode  :character
##                     Mean   : 16866
##                     3rd Qu.: 24874
##                     Max.   :126352
```

Note that each row/observation does not correspond to the suicide rate of one country. Each country is characterized by multiple rows for different age, year, and generation.

3 Methods and Analysis

3.1 Data Analysis

In the "Data Analysis" chapter, several graphs and summary statistics are created in order to understand how each feature can impact the outcome.The conclusions obtained during the analysis will help to build the machine learning model.

As presented in the previous Chapter, the "suiciderates" set contains several features, being the most imporant: "country", "year", "sex", "age", "suicides_no", "population", "suicides.100k.pop", "gdp_per_capita" and "generation". Each row represents a single suicide rate.

Note that the column "suicides.100k.pop" includes the Suicide rate per 100.000 population and it is the dominant feature that will be analyzed the most.

To begin with, the total number of the unique values for the abovementioned important features can be visualized:

```
suiciderates %>%
  summarize(n_country = n_distinct(country), n_year = n_distinct(year), n_sex = n_distinct(sex),
            n_age = n_distinct(age), n_suiciderate = n_distinct(suicides.100k.pop), n_gdp_capita = n_di
```

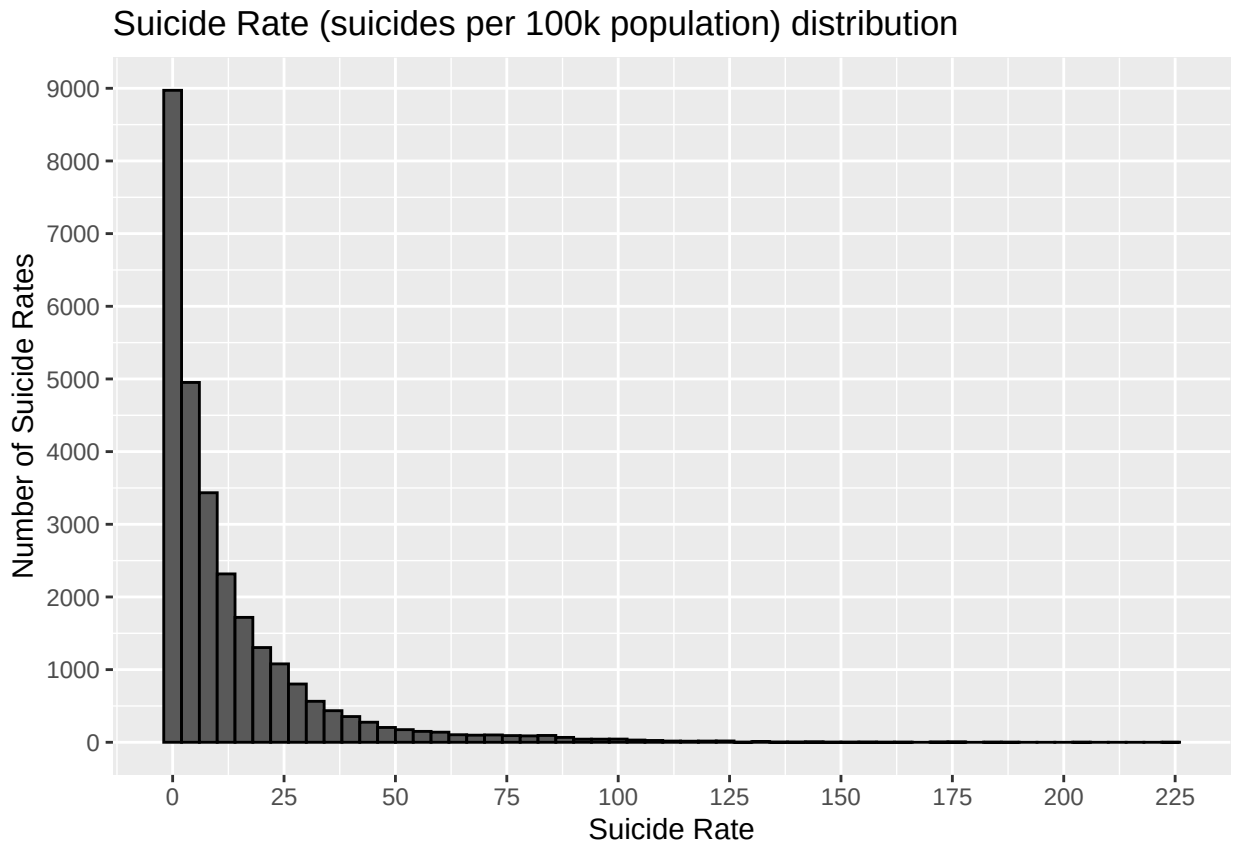
```
##   n_country n_year n_sex n_age n_suiciderate n_gdp_capita n_generation
## 1       101     32     2     6          5298         2233            6
```

The countries investigated in this dataset are 101, thus there are multiple countries that do not have an officialy measured Suicide rate (SR). The studied years are from 1985 to 2016 making 32 years in total, while the "age" factor is divided into 6 different categories that will be represented in a following graph.The "generation" factor is also divided into 6 different categories.

There are 5298 unique values of the suicide rate feature ("suicides.100k.pop"). As it will be presented later on, due to this high number of unique values, a new column will be added in the dataset where the suicide rate will be rounded and then categorized based on a specific criterion.

In order to obtain a better view of the provided rates, their distribution can be visualized as shown below:

```
suiciderates %>%
  ggplot(aes(suicides.100k.pop)) +
  geom_histogram(binwidth = 4, color = "black") +
  xlab("Suicide Rate")+
  scale_x_continuous(breaks = c(seq(0, 250, 25))) +
  ylab("Number of Suicide Rates")+
  scale_y_continuous(breaks = c(seq(0, 10000, 1000))) +
  ggtitle("Suicide Rate (suicides per 100k population) distribution")
```



It appears that the majority of the suicide rates is less than 5, meaning less than 5-per-100.000 people while there are several rows/observations that give a suicide rate more than 30.

The following code summarizes the Suicide rate dataset by calculating the complete set's average rate per 100k population and the maximum rate as well as where and when it appeared:

```
avg_suiciderate <- mean(suiciderates$suicides.100k.pop)
message("The average Suicide rate per 100k population is ", round(avg_suiciderate,digits = 1))
```

```
## The average Suicide rate per 100k population is 12.8
```

```
sd_suiciderate <- sd(suiciderates$suicides.100k.pop)
max_suicide_rate <- max(suiciderates$suicides.100k.pop)
message("The maximum Suicide rate is ", round(max_suicide_rate,digits = 1),
        " and appears in ", suiciderates$country[which.max(suiciderates$suicides.100k.pop)], " in the y
```

```
## The maximum Suicide rate is 225 and appears in Aruba in the year 1995
```

It is also interesting to investigate the countries that present the highest and the lowest average SR. In order to represent this, the average rate per country is calculated:

```
suiciderates%>%group_by(country)%>%summarize(avg_SR=mean(suicides.100k.pop))%>%
  arrange(-avg_SR)%>%head(10)
```

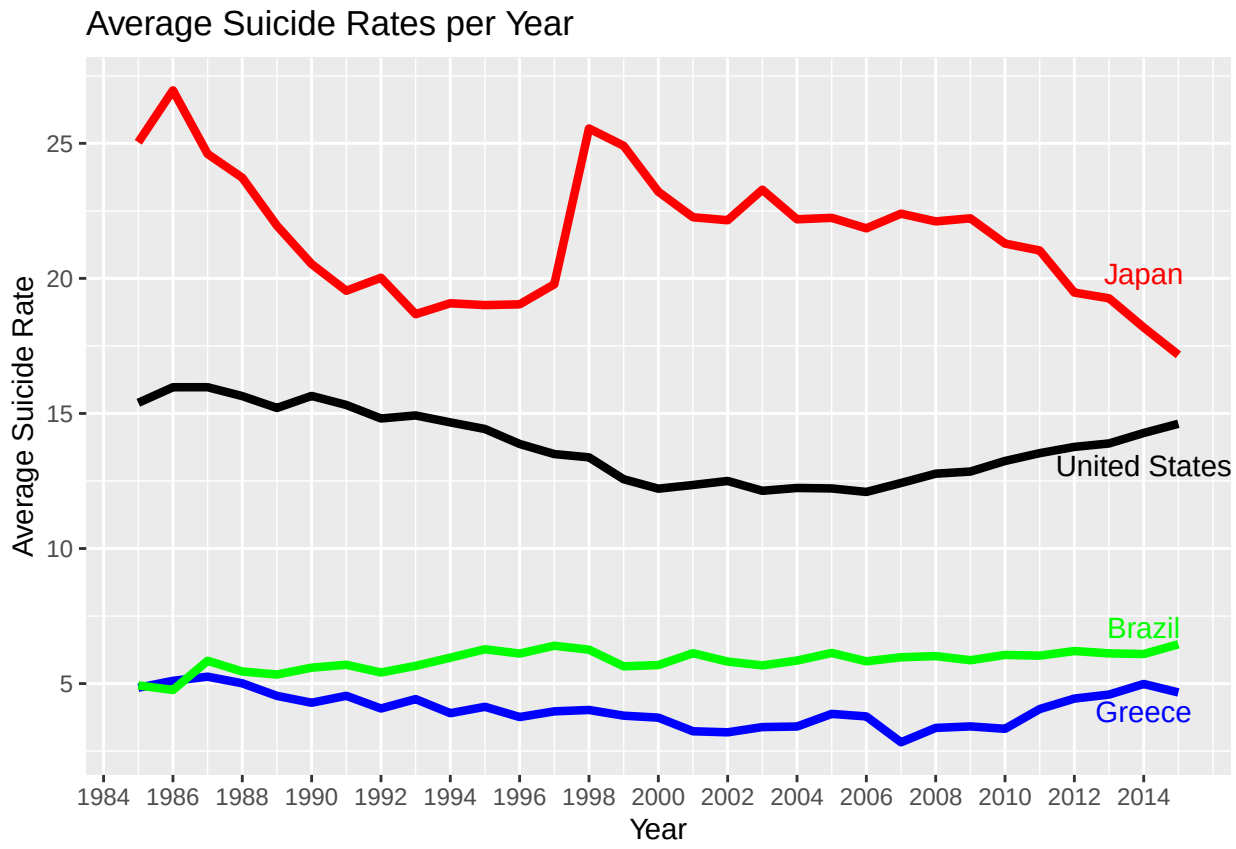
```
## # A tibble: 10 x 2
##    country            avg_SR
##    <chr>               <dbl>
##  1 Lithuania            40.4
##  2 Sri Lanka            35.3
##  3 Russian Federation   34.9
##  4 Hungary              32.8
##  5 Belarus              31.1
##  6 Kazakhstan           30.5
##  7 Latvia               29.3
##  8 Slovenia             27.8
##  9 Estonia              27.3
## 10 Ukraine              26.6
```

```
suiciderates%>%group_by(country)%>%summarize(avg_SR=mean(suicides.100k.pop))%>%
  arrange(avg_SR)%>%head(10)
```

```
## # A tibble: 10 x 2
##    country               avg_SR
##    <chr>                  <dbl>
##  1 Dominica               0
##  2 Saint Kitts and Nevis  0
##  3 Jamaica                0.522
##  4 Antigua and Barbuda    0.553
##  5 Oman                   0.736
##  6 South Africa           0.965
##  7 Kuwait                 1.19
##  8 Bahamas                1.25
##  9 United Arab Emirates   1.32
## 10 Maldives               1.37
```

It appears that Lithuania presents the highest average SR while several Eastern Europeans countries, such as Russia and Hungary are also included in the highest rates. On the other side, many Caribbean countries such Dominica and Saint Kitts and Nevis, present the lowest SR with a value less than 2 per 100k population. Note that all the countries with the lowest Suicide rates are coastal while most of the countries with a high rate belong to the North hemisphere of Earth.

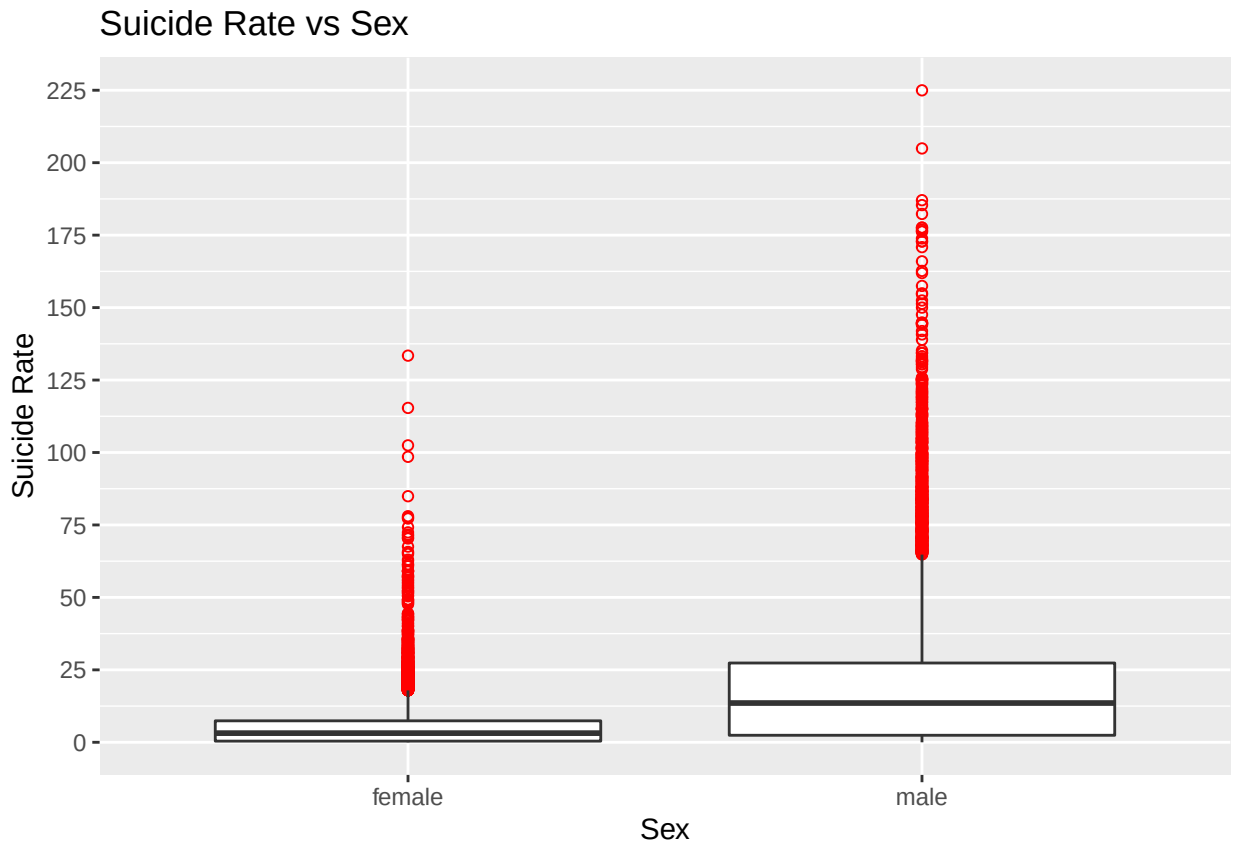
The following graph is summarizing the average SR per 100k population for 4 countries, each one from a different continent, through the years:



Japan presents a signficantly higher average rate than the rest of the countries with big deviations through the years. United States' rate is almost constant through the years while Greece presents a permanently low SR, confirming the abovementioned conclusion referring to coastal countries.

Another important correlation of the suicide rate is the influence of gender:

```
suiciderates%>%ggplot(aes(sex,suicides.100k.pop))+
  geom_boxplot(outlier.colour = "red", outlier.shape = 1)+
  xlab('Sex') +
  ylab('Suicide Rate') +
  scale_y_continuous(breaks = c(seq(0, 250, 25))) +
  ggtitle("Suicide Rate vs Sex")
```




```
mean_SR_male <- suiciderates%>%group_by(sex)%>%summarize(mean(suicides.100k.pop))%>%.[2,2]
mean_SR_male
```

```
## # A tibble: 1 x 1
##   `mean(suicides.100k.pop)`
##                       <dbl>
## 1                      20.2
```

```
mean_SR_female <- suiciderates%>%group_by(sex)%>%summarize(mean(suicides.100k.pop))%>%.[1,2]
mean_SR_female
```

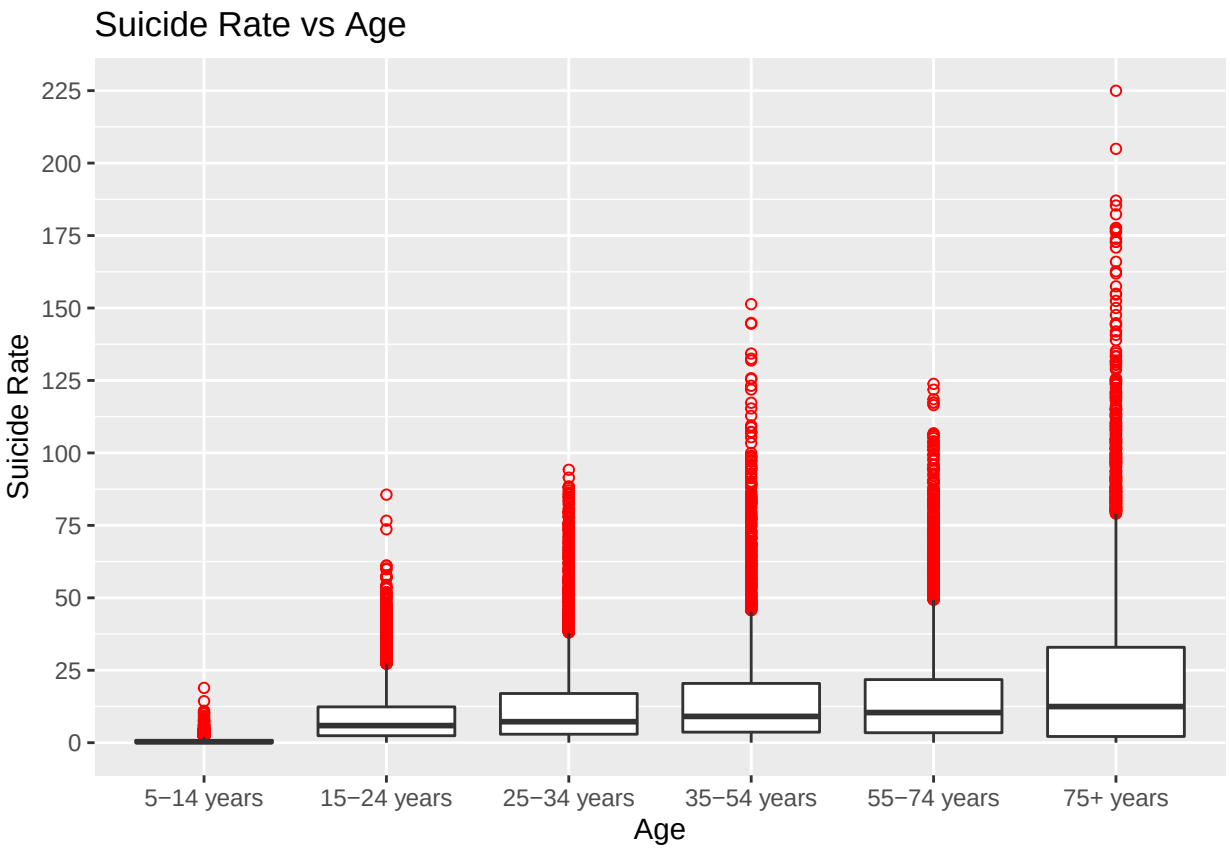
```
## # A tibble: 1 x 1
##   `mean(suicides.100k.pop)`
##                       <dbl>
## 1                      5.39
```

As presented above, the males show a significantly higher average SR (~20) than the females (~5).

Moreover, the correlation of SR vs Age can be depicted:

```
suiciderates$age <- factor(suiciderates$age,c("5-14 years","15-24 years","25-34 years",
                                              "35-54 years","55-74 years","75+ years"))
suiciderates%>%ggplot(aes(age,suicides.100k.pop))+
  geom_boxplot(outlier.colour = "red", outlier.shape = 1)+
```

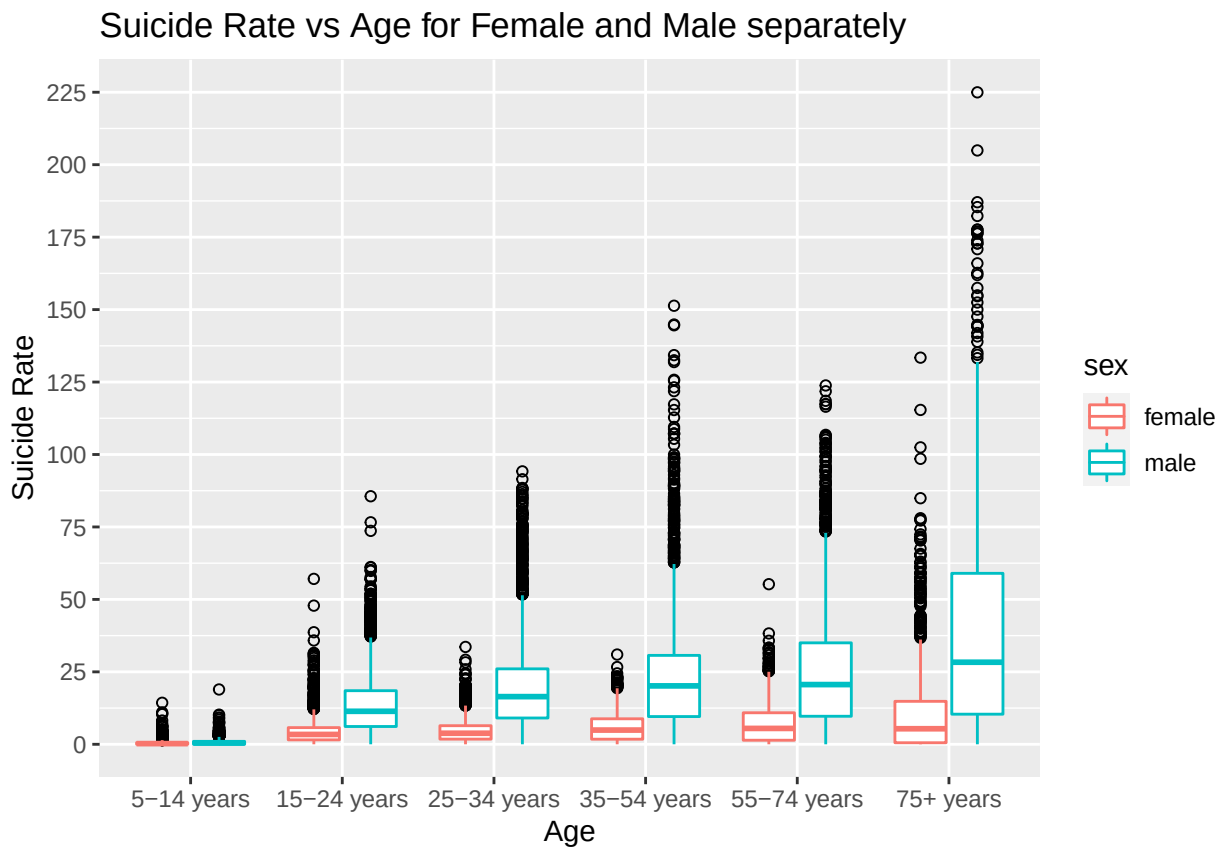
```
  xlab('Age') +
  ylab('Suicide Rate') +
  scale_y_continuous(breaks = c(seq(0, 250, 25))) +
  ggtitle("Suicide Rate vs Age")
```



The feature of "Age" is divided into 6 categories from 5 to 75+ years old. As expected, young populations present an almost zero SR while the oldest (in the category of 75 years and above) present the highest average SR. Loneliness and/or serious health issues could be key factors motivating suicide in the aging years.

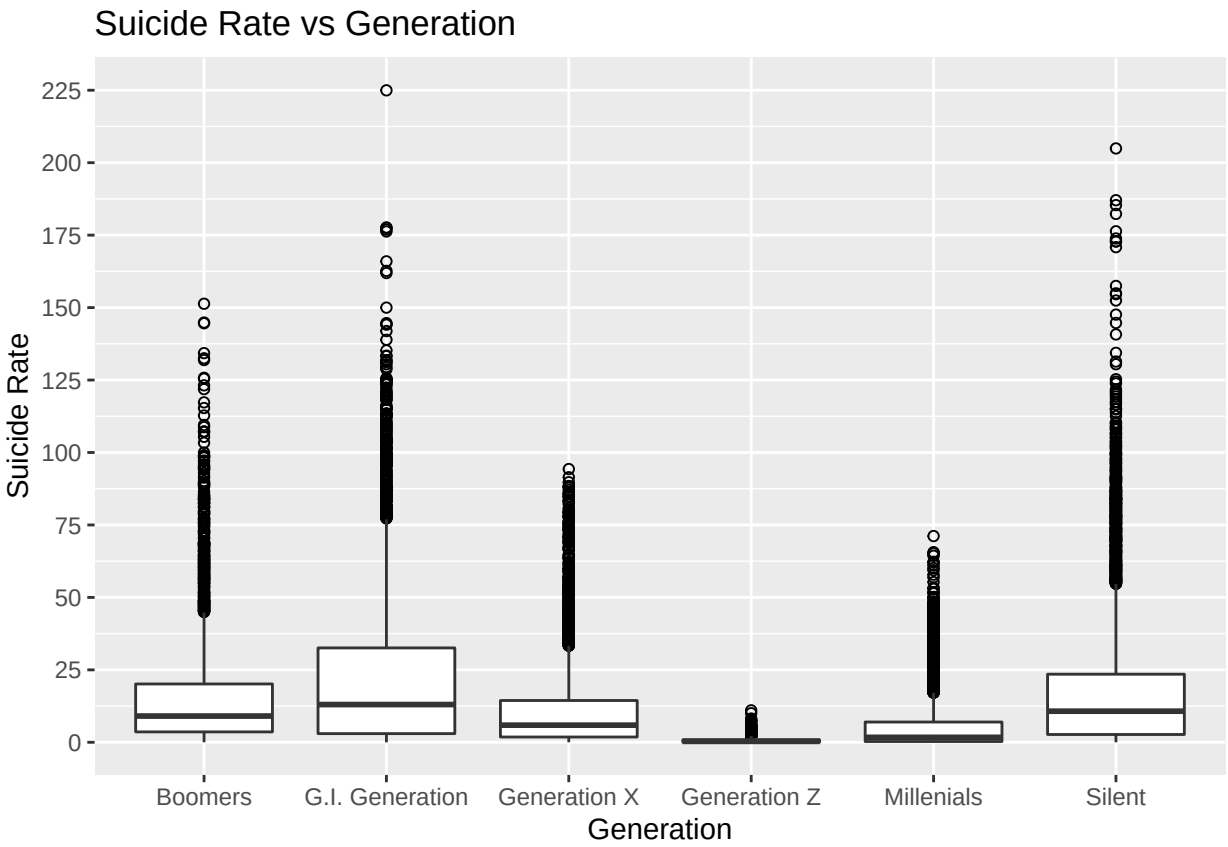
The following graph combines the factors of "Age" and "Sex" and represents the Suicide rate for each age category for females and males respectively. It appears that the males have a significantly higher average SR than the females at all ages.

```
suiciderates%>%ggplot(aes(age,suicides.100k.pop,col=sex))+
  geom_boxplot(outlier.colour = "black", outlier.shape = 1)+
  xlab('Age') +
  ylab('Suicide Rate') +
  scale_y_continuous(breaks = c(seq(0, 250, 25))) +
  ggtitle("Suicide Rate vs Age for Female and Male separately")
```



Regarding the "generation" correlation, it can be depicted as below:

```
suiciderates%>%ggplot(aes(generation,suicides.100k.pop))+
  geom_boxplot(outlier.colour = "black", outlier.shape = 1)+
  xlab('Generation') +
  ylab('Suicide Rate') +
  scale_y_continuous(breaks = c(seq(0, 250, 25))) +
  ggtitle("Suicide Rate vs Generation")
```

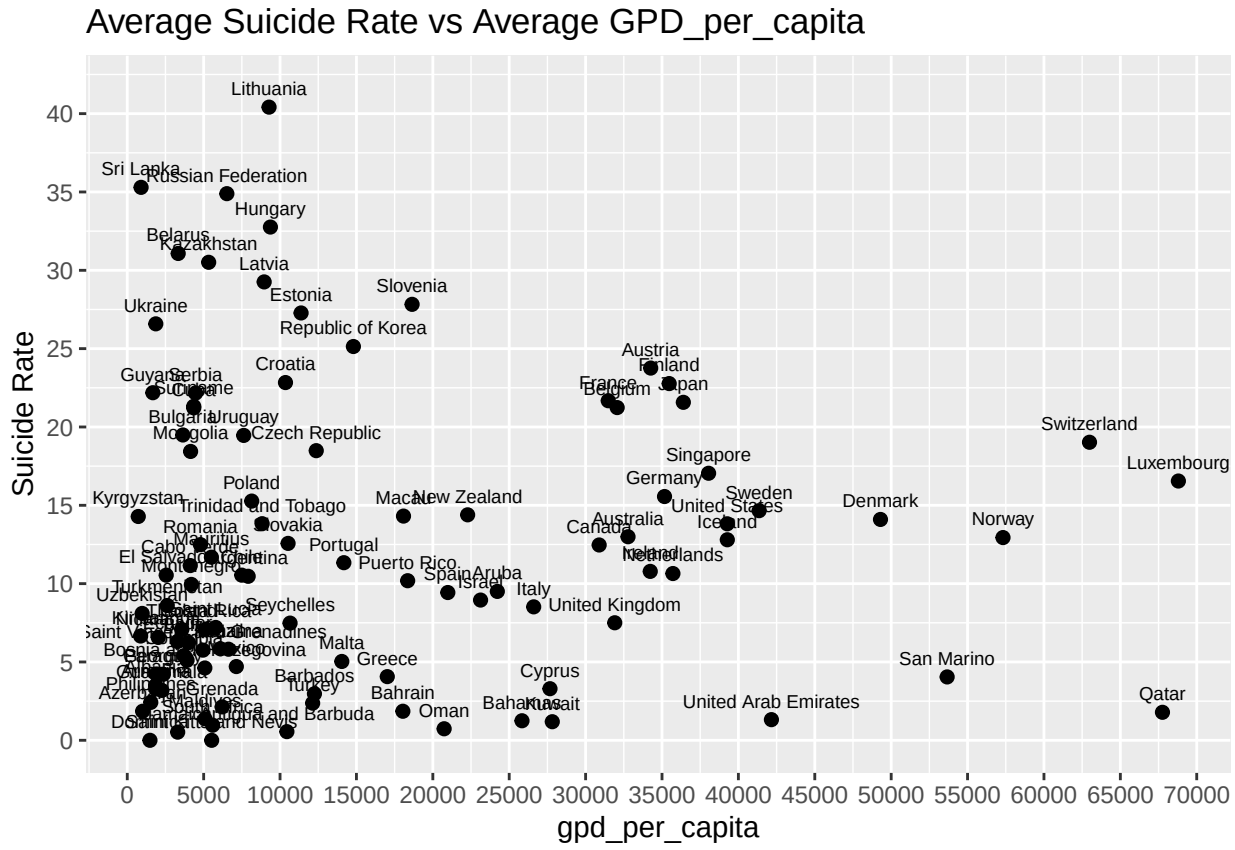


The Greatest Generation, also known as the G.I. Generation and the World War II generation, represent the highest average SR. The people of this generation were shaped by the Great Depression and were the primary participants in World War II, thus explaining the indicative high SR.

Finally, the correlation between the average SR and the gpd_per_capita is shown in the graph below where each point represents a country. A zoomed graph is also included in the area of low gpd_per_capita in order to be easier for the reader to identify the rate for a specific country.

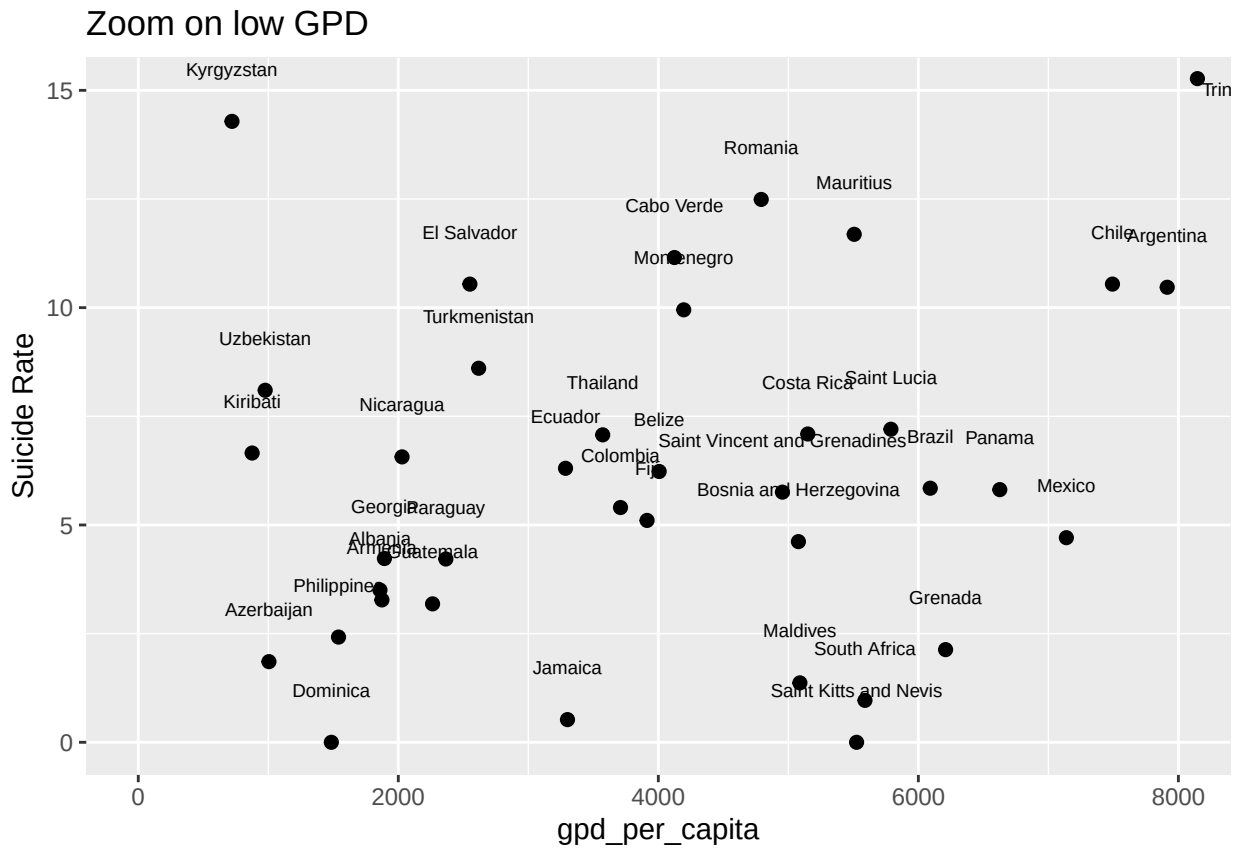
```
SR_GPD_plot <- suiciderates%>%group_by(country)%>%
  summarize(avg_SR=mean(suicides.100k.pop),avg_gdp=mean(gdp_per_capita))%>%
  ggplot(aes(avg_gdp,avg_SR))+
  geom_point(size=2)+
  geom_text(aes(avg_gdp,avg_SR, label=country),size=2.4,position = position_nudge(y = 1.2))+
  xlab('gpd_per_capita') +
  scale_x_continuous(breaks = c(seq(0, 70000, 5000))) +
  ylab('Suicide Rate') +
  scale_y_continuous(breaks = c(seq(0, 50, 5))) +
  ggtitle("Average Suicide Rate vs Average GPD_per_capita")

SR_GPD_plot
```




```
SR_GPD_plot_zoom <- SR_GPD_plot+
  scale_x_continuous(breaks = c(seq(0, 8000, 2000)))+
  ggtitle("Zoom on low GPD")+
  coord_cartesian(xlim=c(0,8000),ylim=c(0,15))

SR_GPD_plot_zoom
```



It is observed that most of the countries included in the dataset have a relatively low gpd_per_capita. The maximum suicide rates are also observed for countries with low gdp, such as Sri Lanka, Belarus, Kazakhstan and Ukraine. However, even countries with very high gdp_per_capita, such as Switzerland and Luxembourg, have a noticeable high Suicide rate, higher than the total average of the dataset (~12.8 per 100k population).

As mentioned in this Chapter, there are 5298 unique values of the suicide rate feature ("suicides.100k.pop"). In order to reduce this number and be able to predict more accurately the suicide rate, this feature will be rounded to its integer. The new column is named as "SR":

```
suiciderates <- suiciderates%>%mutate(SR=round(suicides.100k.pop,digits = 0))
```

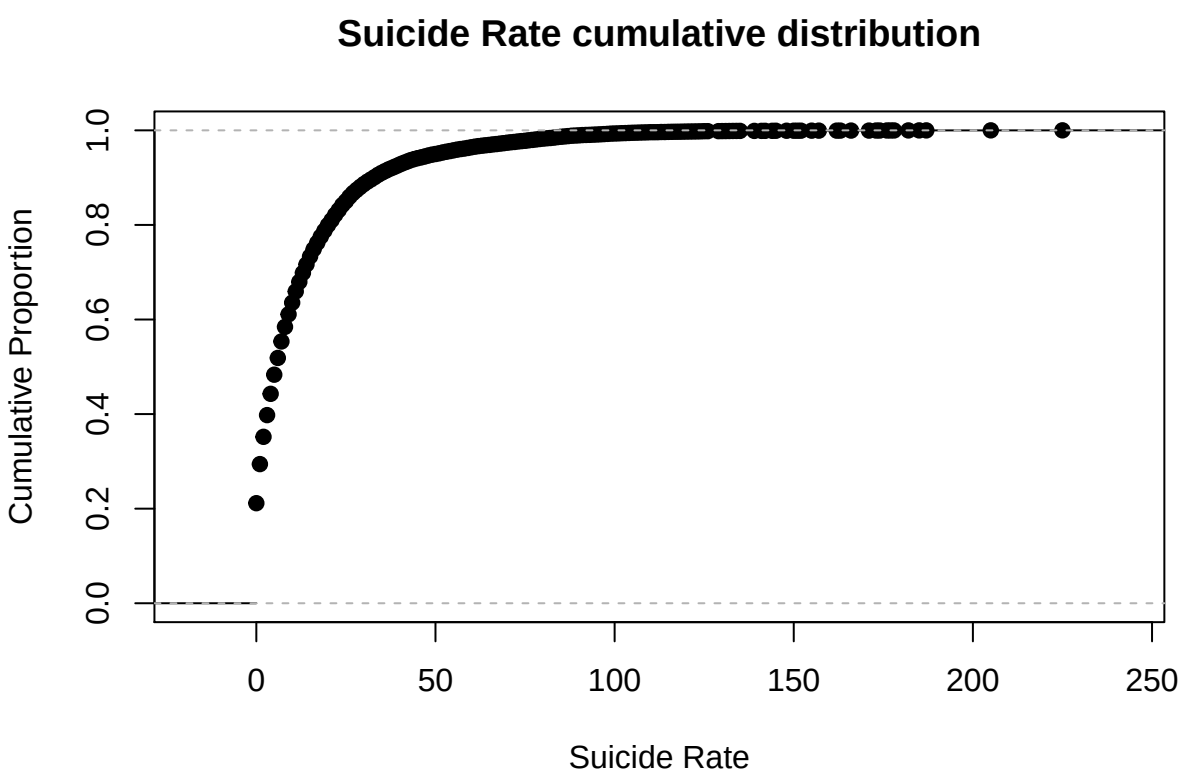
The unique values of the rounded suicide rate is now 159, which is significantly lower.

```
suiciderates %>%
  summarize( n_SR = n_distinct(SR))
```

```
##   n_SR
## 1  159
```

In order to visualize the distribution of the suicide rate through the 101 different countries, the cumulative distribution function is plotted as below:

```
plot(ecdf(suiciderates[,"SR"]),
     xlab="Suicide Rate",
     ylab="Cumulative Proportion",
     main="Suicide Rate cumulative distribution")
```



The calculation of the five main quantiles (0, 25, 50, 75 and 100%) of the "SR" feature takes place as:

```
p <- seq(0,1,0.25)
quantile(suiciderates$SR,p)
```

```
##   0%  25%  50%  75% 100%
##    0    1    6   17  225
```

It seems that almost half of the reported observations present a Suicide rate less than 6 while 75% of the observations have a SR less than 17. From the cumulative distribution plot, it can be seen that a very small percentage (~10%) present an "extreme" suicide rate value with more than 100 suicides per 100k population.

Based on these facts and the conclusions gained from the data visualization plots, a new column will be added in the original dataset. Each observation/row is categorized based on the rounded Suicide rate per 100k population ("suicides.100k.pop"). The categorization takes place as following:

- SR<=5 –> "Low"
- SR>5 and SR<=15 –> "Medium"
- SR>15 and SR<=30 –> "High"
- SR>30 –> "Very High"

and it is executed through the following code. Note that the name of the new columd is "SR_cat":

```
suiciderates <- suiciderates%>%mutate(SR_cat=
                                        ifelse(SR<=5,"Low",
                                               ifelse(SR>5 & SR<=15, "Medium",
                                                      ifelse(SR>15 & SR<=30, "High", "Very High"))))
```

The new structure of the "suiciderates" dataset is represented below:

```
##   country year    sex         age suicides_no population suicides.100k.pop
## 1 Albania 1987   male 15-24 years          21     312900              6.71
## 2 Albania 1987   male 35-54 years          16     308000              5.19
## 3 Albania 1987 female 15-24 years          14     289700              4.83
## 4 Albania 1987   male   75+ years           1      21800              4.59
## 5 Albania 1987   male 25-34 years           9     274300              3.28
## 6 Albania 1987 female   75+ years           1      35600              2.81
##   country.year  gdp_for_year gdp_per_capita      generation SR SR_cat
## 1  Albania1987 2,156,624,900            796    Generation X  7 Medium
## 2  Albania1987 2,156,624,900            796          Silent  5    Low
## 3  Albania1987 2,156,624,900            796    Generation X  5    Low
## 4  Albania1987 2,156,624,900            796 G.I. Generation  5    Low
## 5  Albania1987 2,156,624,900            796         Boomers  3    Low
## 6  Albania1987 2,156,624,900            796 G.I. Generation  3    Low
```

3.2 Modeling approach

Based on the insights gained through the dataset investigation, a prediction model can now be developed. The main prediction target is the Suicide rate category ("SR_cat") and multiple learning-methods that were thoroughly examined during the Data Schience course will be applied. The methods belong to the Classification algorithms.

A potential benefit of this learning algorithm could be the prediciton of Suicide rates for countries that do not have an officially measured rate or to forecast suicide rates for upcoming years. Furthermore, based on the performed investigation and categorization, useful information can be taken into consideration in order to check the severity of the situation for each country and also examine in detail the critical factors that can cause these high Suicide rates.

Regarding the accuracy of the model, several features can be used to build a more complete meachine learning algorithm. However, the higher the number of predictors, the higher the model's complexity. In the current report, a simplified model is initially examined with only one predictor, while the complexity will keep increasing as more complex methods and predictors are introduced. The evaluation of the different methods is performed through the calculation of the respective "Accuracy" for each method. The comparison will be summarized in a common table.

The steps/models that are followed are summarized below. Each step mentions the specific predictors that are used (eg: Sex, Age, Country) and the learning-method (eg: LDA, KNN, Classification tree).

1) Baseline prediction by guessing the outcome
2) Predicting SR_cat by Sex
3) Predicting SR_cat by Age
4) Predicting SR_cat by Sex and Age
5) Predicting SR_cat by GDP_per_capita / LDA
6) Predicting SR_cat by GDP_per_capita / GLM
7) Predicting SR_cat by Sex, Age, GDP, Generation, Country / GLM
8) Predicting SR_cat by Sex, Age, GDP, Generation, Country / KNN
9) Predicting SR_cat by Sex, Age, GDP, Generation, Country / KNN & 10-fold cross validation
10) Predicting SR_cat by Sex, Age, GDP, Generation, Country / Classification tree
11) Predicting SR_cat by Sex, Age, GDP, Generation, Country / Random forest

As it will be explained later on, for the steps 8 to 11, an optimization process is followed, thus the running time of the respective codes is high.

In order to reduce the size of the original dataset, only the useful predictors will be kept while the rest of them will be filtered based on the code:

```
suiciderates_clean<-suiciderates%>%select(SR_cat,country,year,sex,age,gdp_per_capita,generation,SR)
```

Before starting with the modeling process, the "suiciderates_clean" subset is divided into two subsets: "train" and "test". The train-set is used for the training of each model, while the test-set is used for the evaluation of each method.

The division of the "suiciderates_clean" subset takes place through the following code. Note that test-set is set equal to 10% of the original dataset.

```
# Test set will be 10% of Suicide Rates data
set.seed(1, sample.kind="Rounding") # if using R 3.5 or earlier, use `set.seed(1)`
test_index <- createDataPartition(y = suiciderates_clean$SR_cat, times = 1, p = 0.1, list = FALSE)
train_set<-suiciderates_clean[-test_index,]
test_set<-suiciderates_clean[test_index,]
```

3.2.1 1. Baseline prediction by guessing the outcome

The first simplified model is built based on guessing the outcome (Suicide Rate category) with equal probabilities.The Accuracy of the simple guessing is calculated in comparison to the test-set values.

```
guess <- sample(c("Low","Medium","High","Very High"), nrow(test_set), replace = TRUE)
accuracy_guessing<-mean(guess == test_set$SR_cat)
```

As more than one Accuracy value is calculated through the report, it is advisable to create a table and save each one with the corresponding name and method.

```
Accuracy_sum <- data_frame(method = "Simple Guessing", Accuracy = accuracy_guessing)
```

```
## Warning: `data_frame()` is deprecated as of tibble 1.1.0.
## Please use `tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

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

method	Accuracy
Simple Guessing	0.248204

The accuracy of the "Simple Guessing" is equal to 0.248, a value that is very low. In order to do better than simply guessing, the abovementioned insights of Chapter 2 regarding the Sex and Age correlation with the SR, will be taken into account.

3.2.2 2. Predicting SR_cat by Sex

As a second step, the prediction will be performed based only on the gender. For this reason, the average SR of the females (always corresponding to the train-set) and the males is calculated as below:

```
train_set %>%
    group_by(sex) %>%
    summarize(SR = mean(SR)) %>%
    filter(sex == "female") %>%
    pull(SR)
```

```
## [1] 5.410727
```

```
# >5 thus Medium
```

```
train_set %>%
    group_by(sex) %>%
    summarize(SR = mean(SR)) %>%
    filter(sex == "male") %>%
    pull(SR)
```

```
## [1] 20.12985
```

```
# >15 and <30 thus High
```

It appears that females present an average SR equal to 5.4, thus categorized as "Medium" (>5) while the males present an average SR equal to 20.1, categorized as "High" (>15 and <30). As a result, the new prediction based only on the gender will be:

```
sex_model <- ifelse(test_set$sex == "female", "Medium", "High")
# predict "Medium" if female, "0"High" if male
accuracy_by_sex <- mean(sex_model == test_set$SR_cat)
```

meaning that the model will predict "Medium" category if the sex is "female" and "High" if the sex is "male".

```
Accuracy_sum <- bind_rows(Accuracy_sum,data_frame(method="Predicting SR_cat by Sex",Accuracy = accuracy
Accuracy_sum %>% knitr::kable()
```

method	Accuracy
Simple Guessing	0.2482040
Predicting SR_cat by Sex	0.2582615

A minimum improvement of the Accuracy is observed (0.258). As a next step, a different option is investigated by predicting only based on the "Age".

3.2.3 3. Predicting SR_cat by Age

Similar as before, the average SR for each Age category (always for the train-set) is calculated:

```
train_set %>%
    group_by(age) %>%
    summarize(SR = mean(SR))
```

```
## # A tibble: 6 x 2
##   age             SR
##   <fct>        <dbl>
## 1 5-14 years   0.572
## 2 15-24 years  9.02
## 3 25-34 years 12.2
## 4 35-54 years 14.9
## 5 55-74 years 16.1
## 6 75+ years   23.7
```

The new prediction based on the "Age" can be calculated as below:

```
age_model <- ifelse(test_set$age == "5-14 years", "Low", ifelse(test_set$age == "15-24 years","Medium",
                                                                ifelse(test_set$age == "25-34 years","M
accuracy_by_age <- mean(age_model == test_set$SR_cat)
```

Similar to the previous step, based on the calculated average SR for each age-group, the corresponding SR_cat is predicted. For example, the ages from 5 to 14 years show an average SR equal to 0.57, thus all the observations that belong to this age-group will be predicted as "Low" (<5).

```
Accuracy_sum <- bind_rows(Accuracy_sum,data_frame(method="Predicting SR_cat by Age",
                                                  Accuracy = accuracy_by_age))
Accuracy_sum %>% knitr::kable()
```

method	Accuracy
Simple Guessing	0.2482040
Predicting SR_cat by Sex	0.2582615
Predicting SR_cat by Age	0.3864943

The use of "Age" as a predictor instead of "Sex" presented a higher accuracy than before, equal to 0.386. This was expected, as the "Age" predictor consists of 6 different groups and there is higher variability than the "Sex" predictor. In order to achieve a further improvement, the combination of the previous two steps is considered.

3.2.4 4. Predicting SR_cat by Sex and Age

Based on the conclusions from the previous two steps and the relevant graph in Chapter 2, a combined prediction is developed for this step. As a prediction example, when the the age-group is equal to "35-54 years" and the sex is "female", then the prediction will be "Medium" while if the sex is equal to "male", the prediction will be "High". The new predictions and accuracy of the combined "Sex" and "Age" model are calculated as follows:

```
sex_age_model <- ifelse(test_set$age == "5-14 years", "Low",
                        ifelse(test_set$age == "15-24 years","Medium",
                               ifelse(test_set$age == "25-34 years" & test_set$sex == "female","Medium"
                                      ifelse(test_set$age == "25-34 years" & test_set$sex == "male","Hig
                                             ifelse(test_set$age == "35-54 years" & test_set$sex == "fe
                                                    ifelse(test_set$age == "35-54 years" & test_set$sex
                                                           ifelse(test_set$age == "55-74 years" & test_s
                                                                  ifelse(test_set$age == "55-74 years" &
```

```
                                                                    ifelse(test_set$age == "75+ yea
accuracy_by_sex_age <- mean(sex_age_model == test_set$SR_cat)
```

```
Accuracy_sum <- bind_rows(Accuracy_sum,data_frame(method="Predicting SR_cat by Sex and Age",
                                                  Accuracy = accuracy_by_sex_age))
Accuracy_sum %>% knitr::kable()
```

method	Accuracy
Simple Guessing	0.2482040
Predicting SR_cat by Sex	0.2582615
Predicting SR_cat by Age	0.3864943
Predicting SR_cat by Sex and Age	0.4346264

The combined model ended up with an imporved accuracy equal to 0.434. As a next step, a new predictor will be used, the "gdp_per_capita".

3.2.5 5. Predicting SR_cat by GDP_per_capita - LDA

In the fifth step of the learning algorithm, the LDA method is examined. In this case, only one predictor is used, the "gdp_per_capita". The training takes place based only on the train-set, while the accuracy is based on the test-set. The new predictions and accuracy by LDA method, are calculated in the following code:

```
train_lda <- train(SR_cat ~ gdp_per_capita, method = "lda", data = train_set)
lda_preds <- predict(train_lda, test_set)
accuracy_by_gdp_LDA <- mean(lda_preds == test_set$SR_cat)
```

By adding the new accuracy in the summary table:

```
Accuracy_sum <- bind_rows(Accuracy_sum,data_frame(method="Predicting SR_cat by GDP_per_capita - LDA",
                                                  Accuracy = accuracy_by_gdp_LDA))
Accuracy_sum %>% knitr::kable()
```

method	Accuracy
Simple Guessing	0.2482040
Predicting SR_cat by Sex	0.2582615
Predicting SR_cat by Age	0.3864943
Predicting SR_cat by Sex and Age	0.4346264
Predicting SR_cat by GDP_per_capita - LDA	0.4791667

A higher accuracy is achieved with the LDA method, even if only one predictor is applied. The new accuracy is equal to 0.479. Note that a similar accuracy is also calculated for the QDA method, thus the respective calculations are not included in the report.

3.2.6 6. Predicting SR_cat by GDP_per_capita - Generalized Linear Model

As the number of predictors increase, the LDA and QDA methods cannot provide a higher accuracy. Thus, in the sixth step, the Generalized Linear model (GLM) is examined. Initially, only one predictor (gdp_per_capita) will be considered for the development of the model.

```
train_gdp_glm <- train(SR ~ gdp_per_capita, method = "glm", data = train_set)
glm_gdp_preds <- predict(train_gdp_glm, test_set)
glm_gdp_preds <- as.data.frame(glm_gdp_preds)
glm_gdp_preds <- glm_gdp_preds%>%mutate(SR_cat=
                                        ifelse(glm_gdp_preds<=5,"Low",
                                               ifelse(glm_gdp_preds>5 & glm_gdp_preds<=15, "Medium",
                                                      ifelse(glm_gdp_preds>15 & glm_gdp_preds<=30,
                                                             "High", "Very High"))))
accuracy_by_gdp_glm <- mean(glm_gdp_preds$SR_cat == test_set$SR_cat)
```

```
Accuracy_sum <- bind_rows(Accuracy_sum,data_frame(method="Predicting SR_cat by GDP_per_capita - GLM",Ac
Accuracy_sum %>% knitr::kable()
```

method	Accuracy
Simple Guessing	0.2482040
Predicting SR_cat by Sex	0.2582615
Predicting SR_cat by Age	0.3864943
Predicting SR_cat by Sex and Age	0.4346264
Predicting SR_cat by GDP_per_capita - LDA	0.4791667
Predicting SR_cat by GDP_per_capita - GLM	0.2500000

It seems that the accuracy has dropped significantly (0.25). As a result, more predictors will be included in the GLM model.

3.2.7 7. Predicting SR_cat by Sex, Age, GDP, Generation, Country - GLM

The new model is more complex as five features are being used to improve its predictivity. The features are: "Age", "Sex", "GDP_per_capita", "Generation" and "Country". The following code executes the new prediction and calculates the accuracy:

```
train_glm <- train(SR ~ sex+age+gdp_per_capita+generation+country, method = "glm", data = train_set)
glm_preds <- predict(train_glm, test_set)
glm_preds <- as.data.frame(glm_preds)
glm_preds <- glm_preds%>%mutate(SR_cat= ifelse(glm_preds<=5,"Low",
                                               ifelse(glm_preds>5 & glm_preds<=15, "Medium",
                                                      ifelse(glm_preds>15 & glm_preds<=30,
                                                             "High", "Very High"))))
accuracy_by_glm <- mean(glm_preds$SR_cat == test_set$SR_cat)
```

```
Accuracy_sum <- bind_rows(Accuracy_sum,data_frame(method="Predicting SR_cat by Sex, Age, GDP, Generatio
                                                  Accuracy = accuracy_by_glm))
Accuracy_sum %>% knitr::kable()
```

method	Accuracy
Simple Guessing	0.2482040
Predicting SR_cat by Sex	0.2582615
Predicting SR_cat by Age	0.3864943
Predicting SR_cat by Sex and Age	0.4346264
Predicting SR_cat by GDP_per_capita - LDA	0.4791667
Predicting SR_cat by GDP_per_capita - GLM	0.2500000
Predicting SR_cat by Sex, Age, GDP, Generation, Country - GLM	0.5876437

The updated accuracy has reached its maximum value in comparison to the previous steps as it is equal to 0.587. This value though is still relatively low, so the "k-nearest neighbors" algorithm is examined as a next step.

3.2.8 8. Predicting SR_cat by Sex, Age, GDP, Generation, Country - KNN

The knn method was throroughly explained during the Machine Learning lesson of the Data Science course. KNN works by finding the distances between a query and all the examples in the data, selecting the specified number examples (K) closest to the query and then votes for the most frequent label, in the case of classification. The argument K is an optimiseable parameter. Theoretically, a larger K can lead to a smoother estimate while a smaller K value results to a more flexible but noisy estimate.

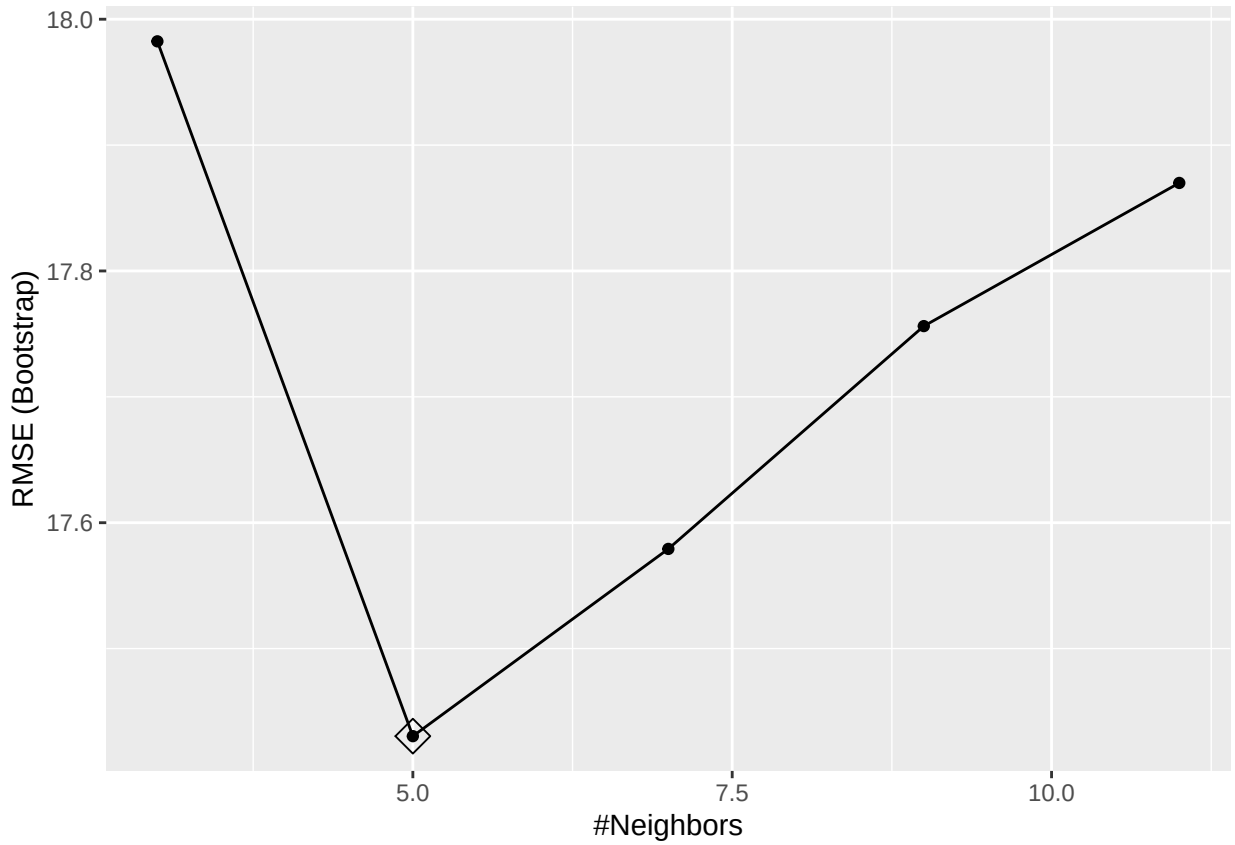
In order to estimate the value of the K that could provide the highest model's accuracy, the "tuneGrid" argument of the "train" function is activated. In this way, the code will run multiple times, once for each K value. The provided range of the K is 3 to 11 with a step of 2. The following code performs the knn model and the respective optimization of the k value. The training of the algorithm is based only on the train-set and the optimum K value is calculated based only on this. The test-set is used for the calculation of the final accuracy of the knn model with the optimized k value.

Note that the specific process takes a considerable amount of time to run.

```
train_knn <- train(SR ~ sex+age+gdp_per_capita+generation+country, method = "knn",
                   data = train_set, tuneGrid = data.frame(k = seq(3, 11, 2)))
```

The following plot represents the calculated RMSE (root-mean-square deviation) for each k value. The optimal k is the one that minimizes the RMSEs.

```
ggplot(train_knn, highlight=TRUE)
```



The optimum k is equal to 5:

```
train_knn$bestTune
```

```
##   k
## 2 5
```

Based on this, the new predictions and accuracy are calculated:

```
knn_preds <- predict(train_knn, test_set)
knn_preds <- as.data.frame(knn_preds)
knn_preds <- knn_preds%>%mutate(SR_cat= ifelse(knn_preds<=5,"Low",
                                               ifelse(knn_preds>5 & knn_preds<=15, "Medium",
                                                      ifelse(knn_preds>15 & knn_preds<=30,
                                                             "High", "Very High"))))
accuracy_by_knn <- mean(knn_preds$SR_cat == test_set$SR_cat)
```

```
Accuracy_sum <- bind_rows(Accuracy_sum,data_frame(method="Predicting SR_cat by Sex, Age, GDP, Generatio
                                                  Accuracy = accuracy_by_knn))
Accuracy_sum %>% knitr::kable()
```

method	Accuracy
Simple Guessing	0.2482040
Predicting SR_cat by Sex	0.2582615

method	Accuracy
Predicting SR_cat by Age	0.3864943
Predicting SR_cat by Sex and Age	0.4346264
Predicting SR_cat by GDP_per_capita - LDA	0.4791667
Predicting SR_cat by GDP_per_capita - GLM	0.2500000
Predicting SR_cat by Sex, Age, GDP, Generation, Country - GLM	0.5876437
Predicting SR_cat by Sex, Age, GDP, Generation, Country - KNN	0.4910201

The knn method did not manage to further improve the model's accuracy (equal to 0.491) in comparison to the glm method, with the same five predictors. As a next step, the knn with 10-fold cross validation is examined.

3.2.9 9. Predicting SR_cat by Sex, Age, GDP, Generation, Country - KNN & 10-fold cross validation

With the 10-fold cross validation procedure, the original train-set will randomly be split into 10 different groups. For each unique group, the model will keep it as a test-set and will use the remaining groups to fit the model.The evaluation will then take place based on the group that was kept as test-set. The evaluation score of each case is retained and the respective models are discarded. Finally, the skill of the model is summarized by using the evaluation scores.

The most important factor for the 10-fold cross validation is that each observation in the data sample is assigned to an individual group and stays in that group for the duration of the procedure. This means that each sample is given the opportunity to be used in the hold out set 1 time and used to train the model k-1 times (9 in this case).

The following code performs the training of the new model, the calculation of the new predictions and the respective accuracy. The knn method is again optimized based on several k values (the range is 3 to 15 with a step of 2) for the train dataset only. Note that the code takes a significant amount of time to run due to the optimization process.

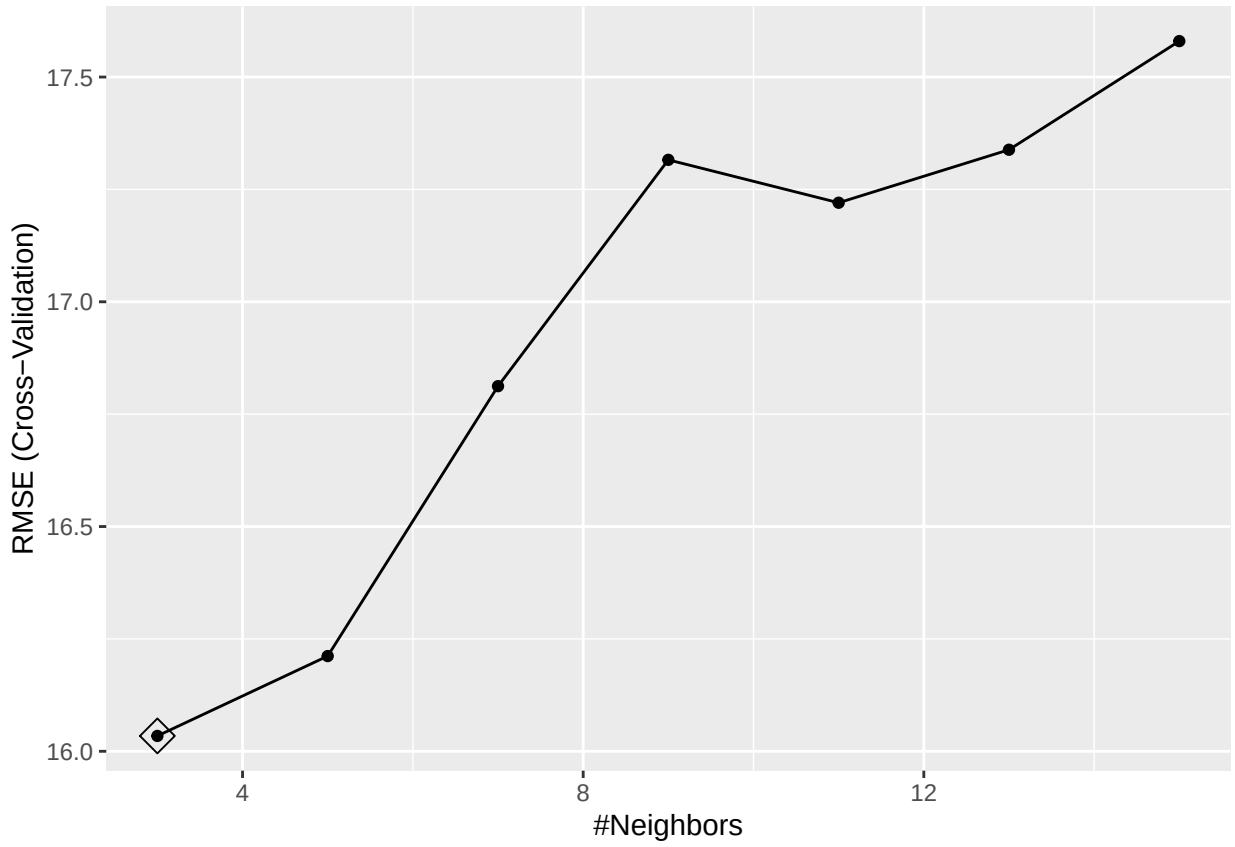
```
train_knn_cv <- train(SR ~ sex+age+gdp_per_capita+generation+country,
                      method = "knn",
                      data = train_set,
                      tuneGrid = data.frame(k = seq(3, 15, 2)),
                      trControl = trainControl(method = "cv", number = 10, p = 0.9))
knn_cv_preds <- predict(train_knn_cv, test_set)
knn_cv_preds <- as.data.frame(knn_cv_preds)
knn_cv_preds <- knn_cv_preds%>%mutate(SR_cat= ifelse(knn_cv_preds<=5,"Low",
                                                     ifelse(knn_cv_preds>5 & knn_cv_preds<=15, "Medium",
                                                            ifelse(knn_cv_preds>15 & knn_cv_preds<=30,
                                                                   "High", "Very High"))))
accuracy_by_knn_cv <- mean(knn_cv_preds$SR_cat == test_set$SR_cat)

train_knn_cv$bestTune
```

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

Similar to the previous step, the optimal k value is decided based on the following plot:

```
ggplot(train_knn_cv, highlight=TRUE)
```



As observed above, it is equal to 3. The new accuracy is addded to the summary table:

```
Accuracy_sum <- bind_rows(Accuracy_sum,data_frame(method="Predicting SR_cat by Sex, Age, GDP, Generatio
                                                  Accuracy = accuracy_by_knn_cv))
Accuracy_sum %>% knitr::kable()
```

method	Accuracy
Simple Guessing	0.2482040
Predicting SR_cat by Sex	0.2582615
Predicting SR_cat by Age	0.3864943
Predicting SR_cat by Sex and Age	0.4346264
Predicting SR_cat by GDP_per_capita - LDA	0.4791667
Predicting SR_cat by GDP_per_capita - GLM	0.2500000
Predicting SR_cat by Sex, Age, GDP, Generation, Country - GLM	0.5876437
Predicting SR_cat by Sex, Age, GDP, Generation, Country - KNN	0.4910201
Predicting SR_cat by Sex, Age, GDP, Generation, Country - KNN & 10-fold cross validation	0.4770115

The achieved accuracy is even lower than the simple knn method (0.477). As a next step, the classification tree is examined in order to improve the accuracy.

3.2.10 10. Predicting SR_cat by Sex, Age, GDP, Generation, Country - Classification tree

Classification or decision tree is a predictive model that is based on an iterative process of splitting the data into partitions and then keep splitting them up further on multiple branches (specific observation values). The process continues until no more useful splits can be found. Classification trees are very powerful algorithms, capable of fitting complex datasets. Besides, decision trees are fundamental components of random forests, which are among the most potent Machine Learning algorithms available today.

The heart of the algorithm is the rule that determines the initial split rule. To choose the best splitter at a node, the algorithm considers each input field in turn. Every possible split is tried and considered and the best split is the one that produces the largest decrease in diversity of the classification label within each partition. This is repeated for all fields and the winner is chosen as the best splitter for that node. The process is continued at subsequent nodes until a full tree is generated.

The classification tree can be optimized based on the "complexity parameter" (cp). This parameter is used to control the size of the decision tree and to select the optimal tree size. If the cost of adding another variable to the decision tree from the current node is above the value of cp, then tree building does not continue. This can also be interpreted as that the tree construction does not continue unless it would decrease the overall lack of fit by a factor of cp.

The following code is used to train the algorithm based on the classification tree method. Similar to the last steps, the same five predicors are used and the training and optimization is only based on the train-set. Once again, note that due to the optimization process of the "cp", the specific code takes time to run.

```
train_rpart <- train(SR ~ sex+age+gdp_per_capita+generation+country,
                     method = "rpart",
                     data = train_set,
                     tuneGrid = data.frame(cp = seq(0, 0.04, 0.002)))
```

The new predictions and accuracy are calculated:

```
rpart_preds <- predict(train_rpart, test_set)
rpart_preds <- as.data.frame(rpart_preds)
rpart_preds <- rpart_preds%>%mutate(SR_cat= ifelse(rpart_preds<=5,"Low",
                                                   ifelse(rpart_preds>5 & rpart_preds<=15, "Medium",
                                                          ifelse(rpart_preds>15 & rpart_preds<=30,
                                                                 "High", "Very High"))))
accuracy_by_rpart <- mean(rpart_preds$SR_cat == test_set$SR_cat)

train_rpart$bestTune
```

```
##   cp
## 1  0
```

It seems that the optimal value of the "cp" is equal to 0.

The detailed decision could be plotted but due to its size, this plot is not represented here. In the next and last step, the importance of several objects used in the random forest method will be presented.

Adding the calculated accuracy of the classification tree model:

```
Accuracy_sum <- bind_rows(Accuracy_sum,data_frame(method="Predicting SR_cat by Sex, Age, GDP, Generatio
                                                  Accuracy = accuracy_by_rpart))
Accuracy_sum %>% knitr::kable()
```

method	Accuracy
Simple Guessing	0.2482040
Predicting SR_cat by Sex	0.2582615
Predicting SR_cat by Age	0.3864943
Predicting SR_cat by Sex and Age	0.4346264
Predicting SR_cat by GDP_per_capita - LDA	0.4791667
Predicting SR_cat by GDP_per_capita - GLM	0.2500000
Predicting SR_cat by Sex, Age, GDP, Generation, Country - GLM	0.5876437
Predicting SR_cat by Sex, Age, GDP, Generation, Country - KNN	0.4910201
Predicting SR_cat by Sex, Age, GDP, Generation, Country - KNN & 10-fold cross validation	0.4770115
Predicting SR_cat by Sex, Age, GDP, Generation, Country - Classification tree	0.7489224

A significantly accuracy improvement is achieved with the powerful algorithm of the classification tree. The model's accuracy is now maximized by reaching the value of 0.748.

3.2.11 11. Predicting SR_cat by Sex, Age, GDP, Generation, Country - Random forest

As a final step, the Random Forest model is examined. The Random Forest consists of a large number of individual decision trees that operate as a total. Each individual tree in the random forest splits out a class prediction and the class with the most votes becomes the model's prediction. The fundamental concept behind random forest is a simple but powerful one — the wisdom of crowds. The Random Forest model requires even higher CPU effort due to the multiple decision trees, thus the running time of the code is high.

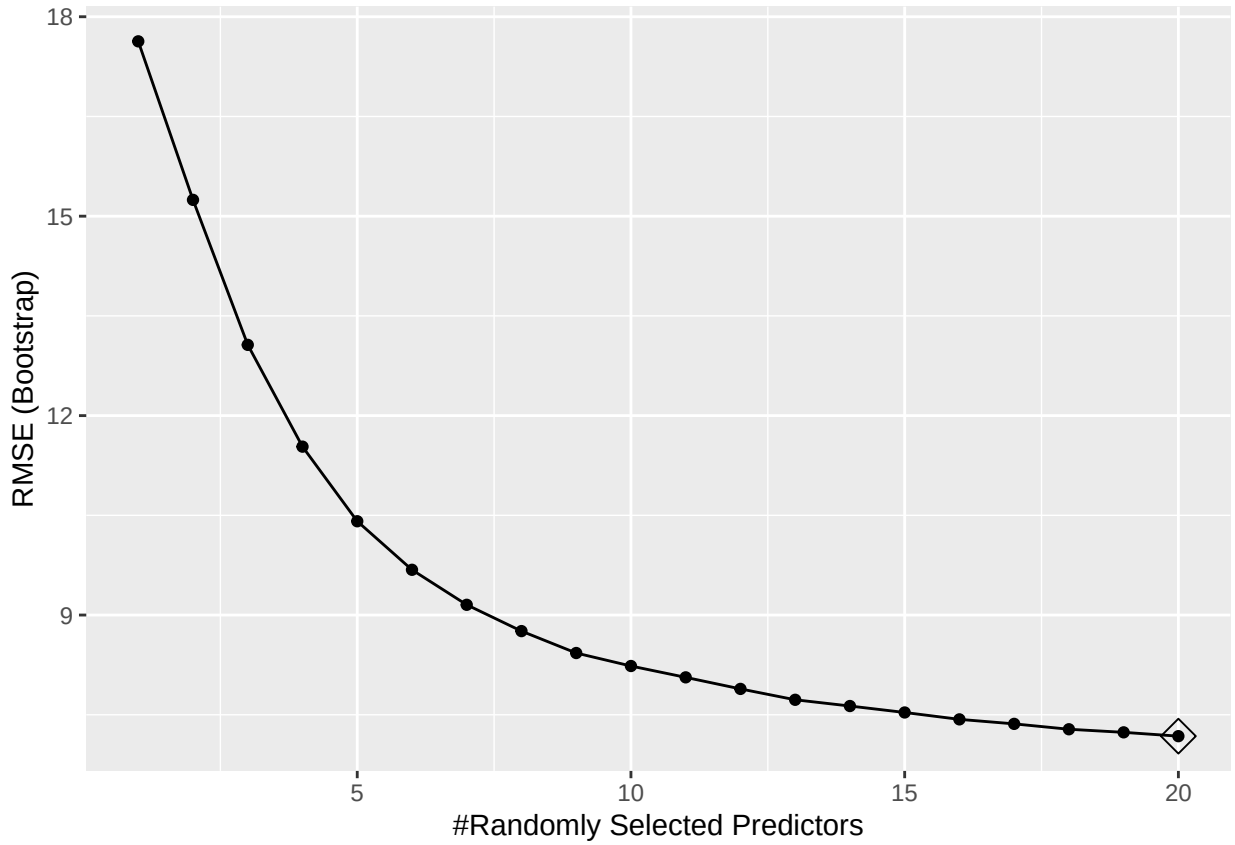
The code below is training the new fit based on the Random Forest model (using the same five predictors and the train-set) as well as calculating the new predictions and the accuracy. The parameter that is optimized is the "mtry". This parameter refers to the number of variables available for splitting at each tree node and it has a range of 1 to 20.

```
train_rf <- train(SR ~ sex+age+gdp_per_capita+generation+country,
                  data = train_set,
                  method = "rf",
                  ntree = 100,
                  tuneGrid = data.frame(mtry = seq(1:20)))

rf_preds <- predict(train_rf, test_set)
rf_preds <- as.data.frame(rf_preds)
rf_preds <- rf_preds%>%mutate(SR_cat= ifelse(rf_preds<=5,"Low",
                                             ifelse(rf_preds>5 & rf_preds<=15, "Medium",
                                                    ifelse(rf_preds>15 & rf_preds<=30,
                                                           "High", "Very High"))))
accuracy_by_rf <- mean(rf_preds$SR_cat == test_set$SR_cat)
```

The optimal "mtry" value is shown in the following plot:

```
ggplot(train_rf, highlight=TRUE)
```



and is equal to 20. Based on the plot, the RMSE is significanly reduced with a "mtry" value higher than 10. Until the value of 20, a decrease is indeed observed but with a slower pace. A "mtry" higher than 20 could lead to a further model's accuracy improvement, thus this improvement will be relatively small and can cost even higher CPU effort.

By using the "varImp" function, the variable importance of several objects can be represented. The top-10 objects with the highest importance are shown as below:

```
varImp(train_rf)
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 112)
##
##                            Overall
## sexmale                    100.000
## age75+ years                29.652
## gdp_per_capita              25.574
## countryLithuania            20.731
## countryHungary              19.942
## countryRussian Federation   17.458
## generationMillenials        15.923
## countryRepublic of Korea    11.670
## countryBelarus              11.631
## countryKazakhstan           11.575
## generationGeneration Z      11.090
```

```
## generationG.I. Generation  10.987
## countryLatvia              10.475
## countryUkraine             10.328
## generationSilent           10.219
## countrySlovenia             9.642
## countryAustria              9.535
## countryEstonia              8.417
## countrySri Lanka            7.691
## countryCroatia              7.593
```

As expected from the data visualization analysis in Chapter 2, the "male" sex is the important factor in predicting the suicide rate category, followed by the "75+ years" age-group. Indeed, these two objects indicate the strongest possibility for a high SR rate.

The following objects with high importance are the countries with a high average SR, Lithuania, Hungary and Russia. Finally the "Millenials" generation that proved to present the lowest average SR in comparison to the 6 different generations, seems to be imporant in predicting a lower SR rate, thus "Low" category.

At last, adding the new accuracy of the Random Forest model:

```
Accuracy_sum <- bind_rows(Accuracy_sum,data_frame(method="Predicting SR_cat by Sex, Age, GDP, Generatio
                                                  Accuracy = accuracy_by_rf))
Accuracy_sum %>% knitr::kable()
```

method	Accuracy
Simple Guessing	0.2482040
Predicting SR_cat by Sex	0.2582615
Predicting SR_cat by Age	0.3864943
Predicting SR_cat by Sex and Age	0.4346264
Predicting SR_cat by GDP_per_capita - LDA	0.4791667
Predicting SR_cat by GDP_per_capita - GLM	0.2500000
Predicting SR_cat by Sex, Age, GDP, Generation, Country - GLM	0.5876437
Predicting SR_cat by Sex, Age, GDP, Generation, Country - KNN	0.4910201
Predicting SR_cat by Sex, Age, GDP, Generation, Country - KNN & 10-fold cross validation	0.4770115
Predicting SR_cat by Sex, Age, GDP, Generation, Country - Classification tree	0.7489224
Predicting SR_cat by Sex, Age, GDP, Generation, Country - Random forest	0.8114224

A further improvement is introduced with an accuracy equal to 0.81, that is significantly higher than the rest of the methods. This accuracy is satisfactory and no further steps are performed for the current report.

4 Results

The summarized Accuracy values of all the presented models are depicted below:

method	Accuracy
Simple Guessing	0.2482040
Predicting SR_cat by Sex	0.2582615
Predicting SR_cat by Age	0.3864943
Predicting SR_cat by Sex and Age	0.4346264
Predicting SR_cat by GDP_per_capita - LDA	0.4791667
Predicting SR_cat by GDP_per_capita - GLM	0.2500000
Predicting SR_cat by Sex, Age, GDP, Generation, Country - GLM	0.5876437
Predicting SR_cat by Sex, Age, GDP, Generation, Country - KNN	0.4910201
Predicting SR_cat by Sex, Age, GDP, Generation, Country - KNN & 10-fold cross validation	0.4770115
Predicting SR_cat by Sex, Age, GDP, Generation, Country - Classification tree	0.7489224
Predicting SR_cat by Sex, Age, GDP, Generation, Country - Random forest	0.8114224

The highest accuracy on the prediction of the "Suicide rates Category" was achieved through the Random Forest model and was found equal to 0.81. This accuracy is satisfactory and is chosen as the optimal solution.

5 Conclusion

This report presents a thorough analysis on the worldwide Suicide Rates, based on the dataset of "Suicide Rates Overview 1985 to 2016" found in the website of "kaggle". Furthermore, a machine learning algorithm, able to predict the Suicide rate categorization from "Low" to "Very High" with an accuracy of 0.81 is succesfully developed. During the process, multiple predictive methods are discussed and tested in order to provide a complete picture around the algorithm's development.

The report starts with the preparation of the data, their exploration and visualization with descriptive graphs. Useful conclusions are derived around the important issue of Suicide, one of the worldwidely leading causes of death during the last years. As a next step, the report focuses on the development of a predictive model of the suicide rate based on the provided dataset and its respective features. Multiple predictive methods are examined and respective models are developed. More specifically, a simple guessing model is initially examined by using only one predictor. The complexity kept increasing as more methods, such as "lda" and "knn" required the introduction of more predictors. At the end of the modeling approach, powerful methods are introduced, including "decision trees" and "Random Forest". Each method is seprately optimized based on specific parameters. The evaluation of the different methods is performed through the calculation of the respective "Accuracy" for each method.

A further improvement of the accuracy could be achieved by taking into consideration more of the available features such as the year and perform further optimization of the random forest model with more than one parameter. However, the algorithm would become computationally expensive and a commercial laptop could not afford to train the model.

6 Appendix

```
print("Operating System:")
```

```
## [1] "Operating System:"
```

```
version
```

```
##                _
## platform       x86_64-w64-mingw32
## arch           x86_64
## os             mingw32
## system         x86_64, mingw32
## status
## major          4
## minor          0.0
## year           2020
## month          04
## day            24
## svn rev        78286
## language       R
## version.string R version 4.0.0 (2020-04-24)
## nickname       Arbor Day
```