SuicideRateRMD

Baraa Jallad

10/12/2021

Introduction

This is my project for the capstone section in the Data Science program which is held in the EDX platform, after passing 8 courses which we learned how to use R language and how to install the tools we need and how to use the math concepts in R, Visualization, Probability, Inference and Modeling, Productivity Tools, Wrangling, Linear Regression and Machine Learning. In the last course we have to use a data we chose from the sites to study it analysis it and apply what we learned in the previous courses on it.

The data is called Suicide Rates, it contains more than 27K record about the Suicide Rates which is overview 1985 to 2016. We have to Develop our algorithm using to predict Suicide Rates.

Downloading Data and Extract it

We are going to download the Packages we need and load the librares required.

```
##########################################################################
# The final project of the 9th course 'Capstone' in the Data Science Program.
# Suicide Rates Overview 1985 to 2016 ####################################
##########################################################################


# Note: this process could take a couple of minutes

#installing the packges needed in the solution
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(ggplot2)) install.packages("ggplot2", repos = "http://cran.us.r-project.org")
if(!require(lubridate)) install.packages("lubridate", repos = "http://cran.us.r-project.org")
if(!require(magrittr)) install.packages("magrittr", repos = "http://cran.us.r-project.org")
if(!require(dplyr)) install.packages("dplyr", repos = "http://cran.us.r-project.org")

#Load the libraries required
library(tidyverse)
library(caret)
library(data.table)
library(ggplot2)
library(lubridate)
library(magrittr) # needs to be run every time you start R and want to use %>%
library(dplyr)    # alternatively, this also loads %>%
```

I downloaded the file before and load it to the tempfile to process it.

```
# Set path to the directory that contains the dataset
#https://github.com/BaraJallad/SuicideRates/raw/main/master.csv.zip
path <- "."
filename <- "master.csv"
fullpath <- file.path(path, filename)
Suicide_Rates <- read.csv(fullpath)
```

Now we are check the structure of the data set and rename the columns.

```
# Lets find the structure of the data frame
str (Suicide_Rates)
```

```
## 'data.frame':    27820 obs. of  12 variables:
##  $ ï..country        : chr  "Albania" "Albania" "Albania" "Albania" ...
##  $ year              : int  1987 1987 1987 1987 1987 1987 1987 1987 1987 1987 ...
##  $ sex               : chr  "male" "male" "female" "male" ...
##  $ age               : chr  "15-24 years" "35-54 years" "15-24 years" "75+ years" ...
##  $ suicides_no       : int  21 16 14 1 9 1 6 4 1 0 ...
##  $ population        : int  312900 308000 289700 21800 274300 35600 278800 257200 137500 3110
00 ...
##  $ suicides.100k.pop : num  6.71 5.19 4.83 4.59 3.28 2.81 2.15 1.56 0.73 0 ...
##  $ country.year      : chr  "Albania1987" "Albania1987" "Albania1987" "Albania1987" ...
##  $ HDI.for.year      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gdp_for_year....  : chr  "2,156,624,900" "2,156,624,900" "2,156,624,900" "2,156,624,900"
...
##  $ gdp_per_capita....: int  796 796 796 796 796 796 796 796 796 796 ...
##  $ generation        : chr  "Generation X" "Silent" "Generation X" "G.I. Generation" ...
```

```
# Rename the column
Suicide_Rates <- rename (Suicide_Rates, "rate" = "suicides.100k.pop",
                         "country" = "ï..country"                   )


# Lets find the header and the contant of the data frame
head (Suicide_Rates)
```

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

```
nrow(Suicide_Rates)
```

```
## [1] 27820
```

Find the number of unique variables.

```
n_distinct(Suicide_Rates$country )
```

```
## [1] 101
```

```
n_distinct(Suicide_Rates$year    )
```

```
## [1] 32
```

```
n_distinct(Suicide_Rates$generation    )
```

```
## [1] 6
```

```
n_distinct(Suicide_Rates$age     )
```

```
## [1] 6
```

Replace the NA values

```
#Replace the NA values with 0's using replace() in R
Suicide_Rates[is.na(Suicide_Rates)]<-0
```

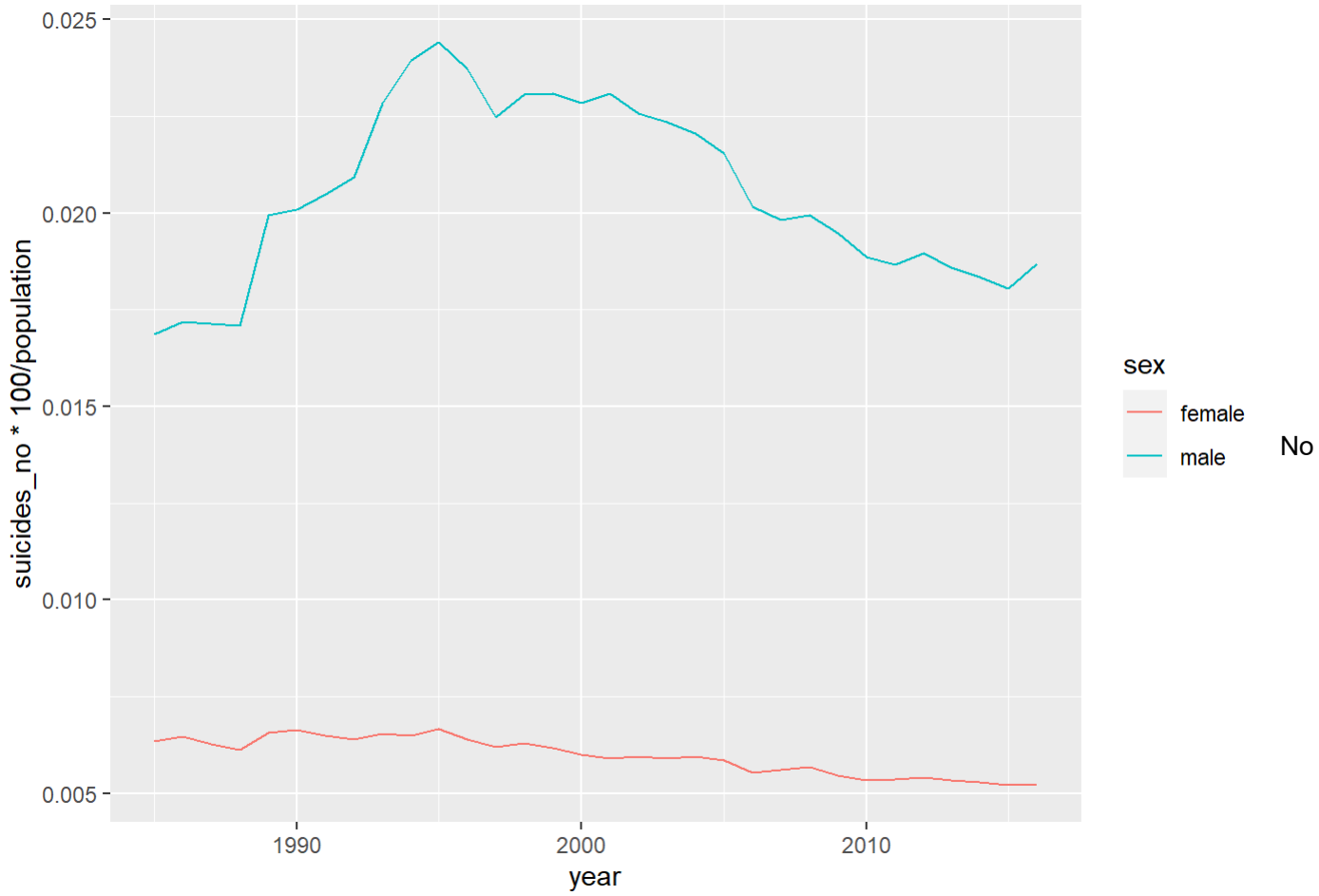
Analyst Data

#Plot between years and suicides no (sex).

```
df_sex <- Suicide_Rates %>%  group_by(year, sex) %>% summarise(suicides_no = sum (suicides_no),
population=sum(population))
```

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

```
df_sex %>%
  ggplot(aes(year,suicides_no*100/population, col = sex)) +
  geom_line()
```



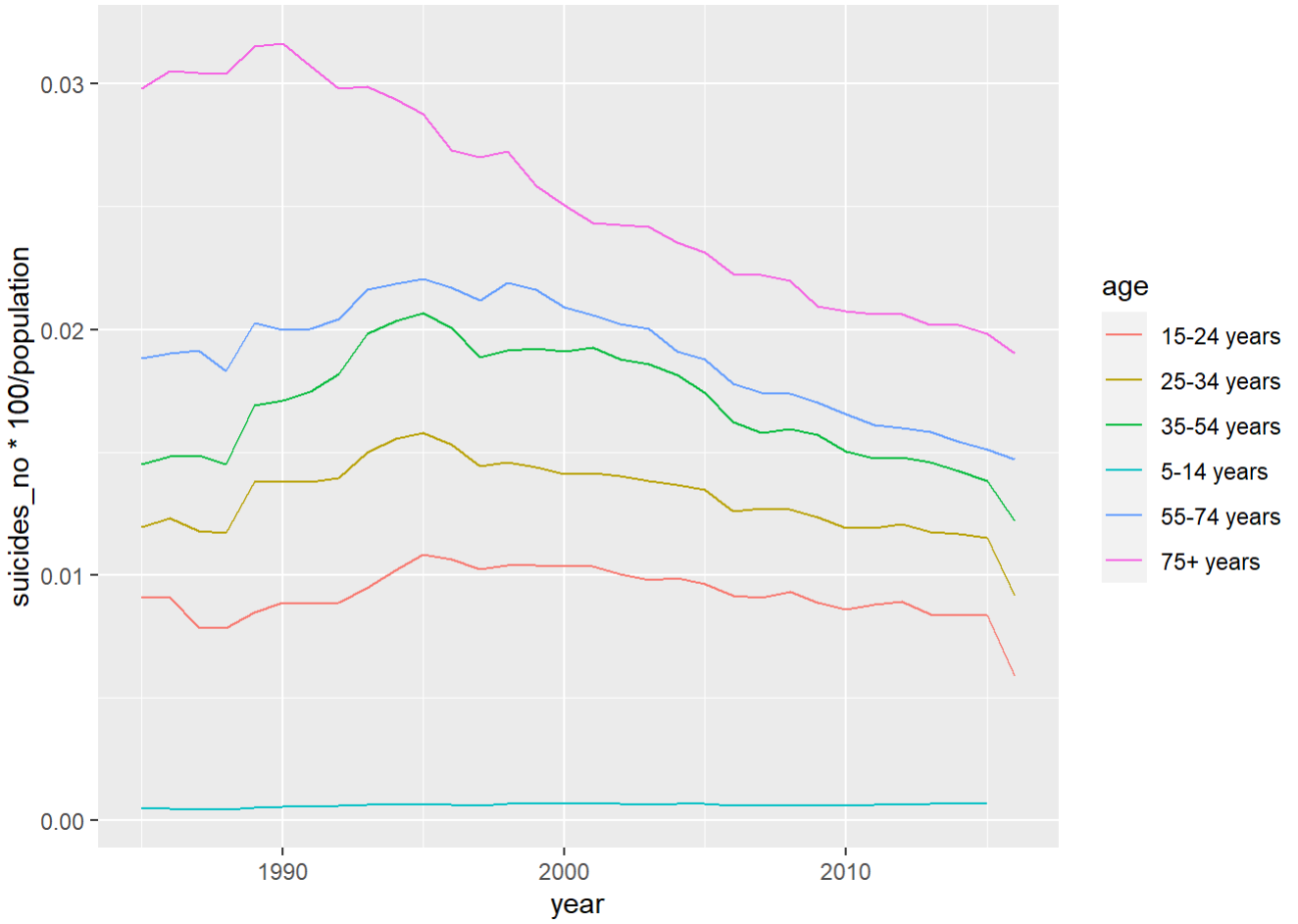
matter which year it is, the suicides number of male are about three times higher than of female.

Plot between years and suicides no (age)

```
df_sum <- Suicide_Rates %>%  group_by(year, age) %>% summarise(suicides_no = sum (suicides_no), population=sum(population))
```

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

```
df_sum %>%
  ggplot(aes(year,suicides_no*100/population, col = age)) +
  geom_line()
```



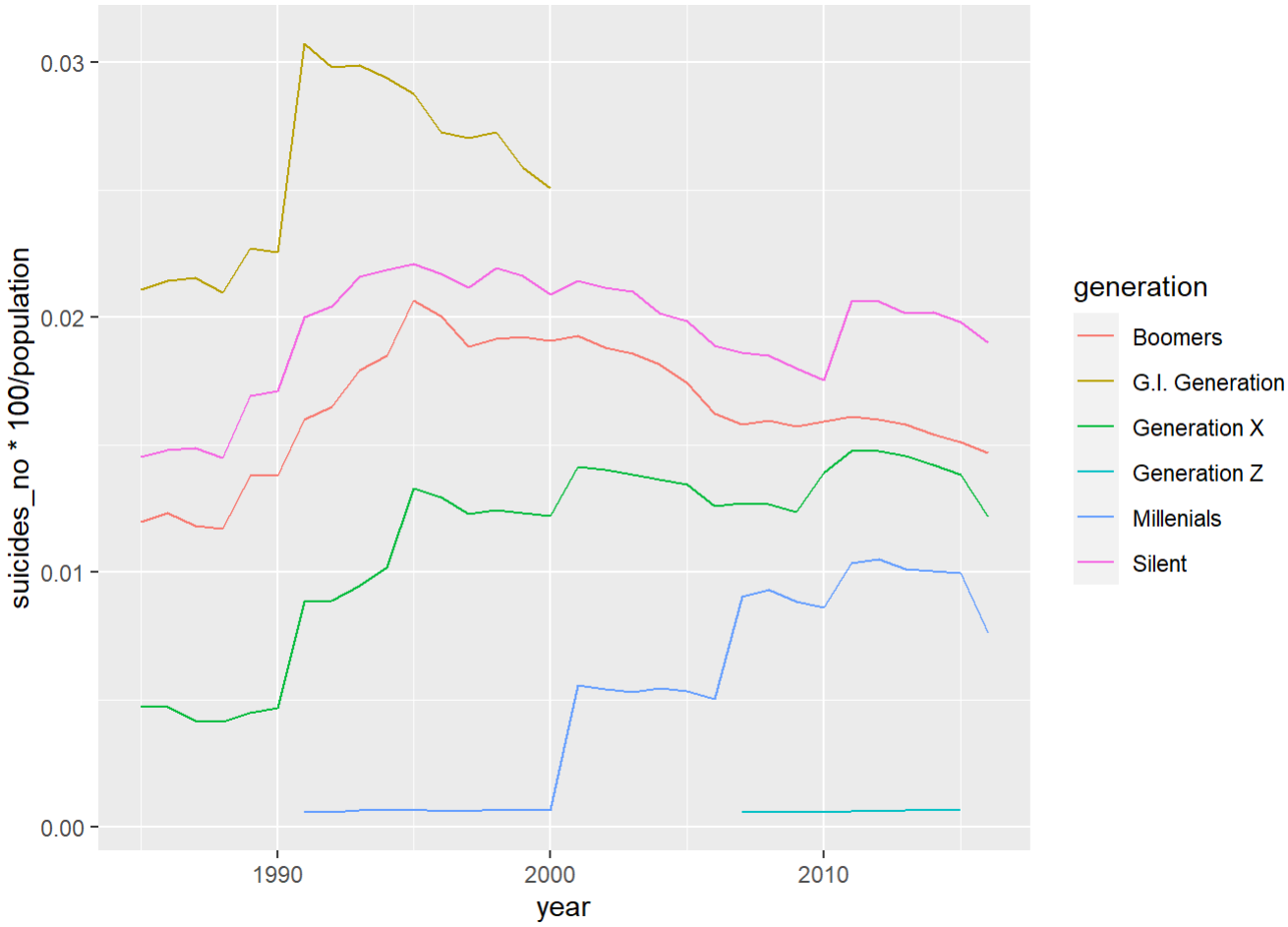
Obviously, the suicide rate is getting higher when the age is higher. That is, age is a factor of suicide.

Plot between years and suicides no (generation)

```
df_generation <- Suicide_Rates %>%  group_by(year, generation) %>% summarise(suicides_no = sum
(suicides_no), population=sum(population))
```

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

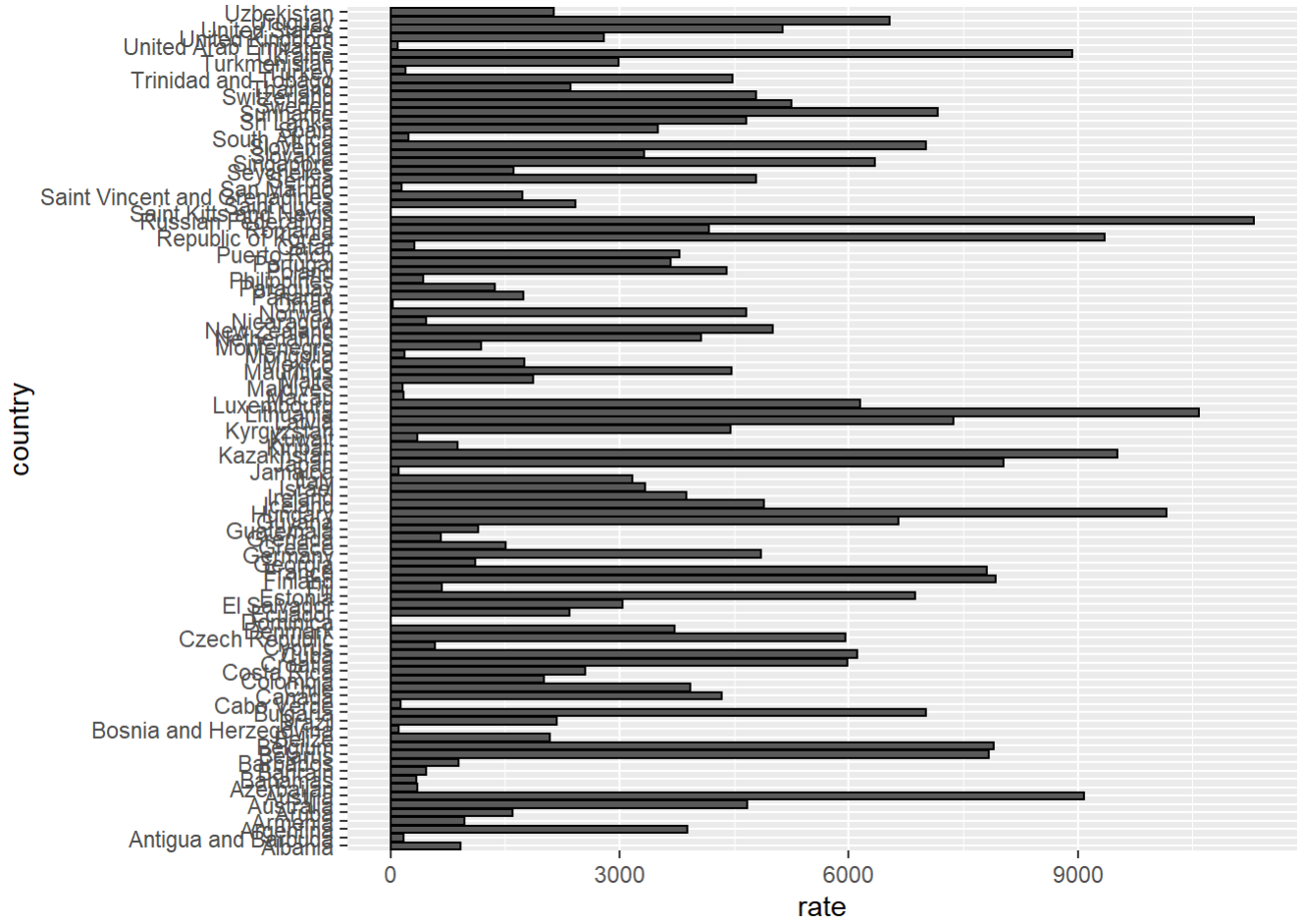
```
df_generation %>%
  ggplot(aes(year,suicides_no*100/population, col = generation)) +
  geom_line()
```



Before 2000, we can see that the highest suicide rate is G.I. generation, and this generation is also known as WW2 generation. They suffered from the worldwide great depression before WW2, at this time, the income, profit, taxes are decreased seriously, so this generation experienced economic and social turmoil.

```
df_new_country <- Suicide_Rates %>%  group_by( country) %>% summarise(rate = sum(rate))

df_new_country %>%
  ggplot(aes(rate,country),  width = 8, height = 300, res=36) +
  geom_bar(stat="identity", color = "black")
```



The country which have the high rate of suicide, and the low one.

```
which.max(df_new_country$rate)
```

```
## [1] 76
```

```
df_new_country[76,]
```

```
## # A tibble: 1 x 2
##   country              rate
##   <chr>               <dbl>
## 1 Russian Federation 11305.
```

```
which.min(df_new_country$rate)
```

```
## [1] 28
```

```
df_new_country[28,]
```

```
## # A tibble: 1 x 2
##   country   rate
##   <chr>    <dbl>
## 1 Dominica     0
```

The validation data should ONLY be used for evaluating the RMSE. The edx data will be used in the developed algorithm and to predict movie ratings.

```
# Validation 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)`

ds <- Suicide_Rates %>% select (country,year,sex,age,rate,generation)
set.seed(1)
test_index <- createDataPartition(y = ds$rate, times = 1, p = 0.1, list = FALSE)
edx <- ds[-test_index,]
temp <- ds[test_index,]

# Make sure country,sex and age in validation set are also in edx set

validation <- temp %>%
  semi_join(edx, by = "country") %>%
  semi_join(edx, by = "age") %>%
  semi_join(edx, by = "sex")


# Add rows removed from validation set back into edx set
removed <- anti_join(temp, validation)
```

```
## Joining, by = c("country", "year", "sex", "age", "rate", "generation")
```

```
edx <- rbind(edx, removed)



#rm is used to delete the unnecessary data to focus on what we are working on
rm( test_index, temp, ds, removed, plot1)
```

```
## Warning in rm(test_index, temp, ds, removed, plot1): object 'plot1' not found
```

To find the RMSE which stand for (Residual Mean Squared Error), and we can calculate it mathematical on a test set, and We are going to create a datafram to compare the different results on it.

```
#create function that computes the RMSE for vectors
RMSE <- function(true_ratings, predicted_ratings){
  sqrt(mean((true_ratings - predicted_ratings)^2))
}

#create data frame to save the results on it
the_final_results <- tibble ()
#find the mean of raiting
mu <- mean(edx$rate)
mu
```

```
## [1] 12.81683
```

The function to find the RMSE is ready to use.

#Apply Models Now we are going to use the models to find the best way to predict the new data set.

Model 1 - Average

```
##### Model 1 - Average #####
#Simplest possible model

# calculate the average rating
mu_hat <- mean(edx$rate)
mu_hat
```

```
## [1] 12.81683
```

```
# calculate rmse for model
average_rmse <- RMSE(validation$rate, mu_hat)
average_rmse
```

```
## [1] 19.3761
```

```
predictions <- rep(12, nrow(validation))
RMSE(validation$rate, predictions)
```

```
## [1] 19.393
```

```
# create a table to display all the calculated rmses
the_final_results <- tibble(method = "Just the average", RMSE = average_rmse)
```

This is the simplest way.

Model 2 - Country Effect

```
##### Model 2 - Country Effect #####
#we are going to group the data by country to find  c_m for each country with the equation mean
(rate - average)
country_effect <- edx %>%
  group_by(country) %>%
  summarize(c_m = mean(rate - mu_hat))

#take test dataset and calculate predicted rating
country_pred_rate <- validation %>%
  left_join(country_effect, by ="country") %>%
  mutate(predicted_rating = mu_hat + c_m) %>%
  pull(predicted_rating)
# calculate rmse for model
rmse_country_effect <- RMSE(validation$rate, country_pred_rate)
#Add the rmse results to the data frame
the_final_results <- bind_rows(the_final_results,
                               tibble(method = "Average + Country Effect", RMSE = rmse_country_e
ffect))
rmse_country_effect
```

```
## [1] 17.3207
```

Lets try another model.

Model 3 - age Effect

```
##### Model 3 - age Effect #####
#we are going to group the data by age to find  a_m for each age with the equation mean(rate - a
verage)
age_effect <- edx %>%
  group_by(age) %>%
  summarize(a_m = mean(rate - mu_hat))

#take test dataset and calculate predicted rating
age_pred_rate <- validation %>%
  left_join(age_effect, by ="age") %>%
  mutate(predicted_rating = mu_hat + a_m) %>%
  pull(predicted_rating)
# calculate rmse for model
rmse_age_effect <- RMSE(validation$rate, age_pred_rate)
#Add the rmse results to the data frame
the_final_results <- bind_rows(the_final_results,
                               tibble(method = "Average + Age Effect", RMSE = rmse_age_effect))
rmse_age_effect
```

```
## [1] 17.98061
```

Model 4 - year Effect

```
##### Model 4 - year Effect #####
#we are going to group the data by year to find  y_m for each year with the equation mean(rate - average)
year_effect <- edx %>%
  group_by(year) %>%
  summarize(y_m = mean(rate - mu_hat))

#take test dataset and calculate predicted rating
year_pred_rate <- validation %>%
  left_join(year_effect, by ="year") %>%
  mutate(predicted_rating = mu_hat + y_m) %>%
  pull(predicted_rating)
# calculate rmse for model
rmse_year_effect <- RMSE(validation$rate, year_pred_rate)
#Add the rmse results to the data frame
the_final_results <- bind_rows(the_final_results,
                               tibble(method = "Average + Year Effect", RMSE = rmse_year_effec
t))
rmse_year_effect
```

```
## [1] 19.35271
```

Model 5 - Country Regularization

```
##### Model 5 - Country Regularization #####

lambdas <- seq(0, 10, 0.25) # define a set of lambdas to test


#calculate rmses for all defined lambdas by creating a function that predicte the rating and ret
urn rmses for each lambda
reg_country_rmses <- sapply(lambdas, function(l){
  e_c <- edx %>%
    group_by(country) %>%
    summarize(e_c = sum(rate - mu_hat) / (n() + l))
  predicted_ratings <- validation %>%
    left_join(e_c, by = "country") %>%
    mutate(pred = mu_hat + e_c) %>%
    pull(pred)
  return(RMSE(validation$rate, predicted_ratings))
})

# return minimum rmse
rmse_reg_country_effect <- min(reg_country_rmses)
# add calculated rmse to rmse table
the_final_results <- bind_rows(the_final_results,
                               tibble(method = "Average + Country Effect + Regularization", RMSE
= rmse_reg_country_effect))
rmse_reg_country_effect
```

```
## [1] 17.31911
```

Model 6 - generation Effect

```
##### Model 6 - generation Effect #####
#we are going to group the data by generation to find  g_m for generation year with the equation
mean(rate - average)
generation_effect <- edx %>%
  group_by(generation) %>%
  summarize(g_m = mean(rate - mu_hat))

#take test dataset and calculate predicted rating
generation_pred_rate <- validation %>%
  left_join(generation_effect, by ="generation") %>%
  mutate(predicted_rating = mu_hat + g_m) %>%
  pull(predicted_rating)
# calculate rmse for model
rmse_generation_effect <- RMSE(validation$rate, generation_pred_rate)
#Add the rmse results to the data frame
the_final_results <- bind_rows(the_final_results,
                               tibble(method = "Average + Generation Effect", RMSE = rmse_genera
tion_effect))
rmse_generation_effect
```

```
## [1] 18.23881
```

Results

```
the_final_results
```

```
## # A tibble: 6 x 2
##   method                                    RMSE
##   <chr>                                    <dbl>
## 1 Just the average                          19.4
## 2 Average + Country Effect                  17.3
## 3 Average + Age Effect                      18.0
## 4 Average + Year Effect                     19.4
## 5 Average + Country Effect + Regularization  17.3
## 6 Average + Generation Effect               18.2
```

Here we can export the predicted rate data to csv file.

```
#export the data as csv
write.csv(validation %>% select(country, year,sex,age,  rate,      generation) %>% mutate(rating
= generation_pred_rate),
          "Predicted.csv", na = "", row.names=FALSE)
```

The Predicted.csv is exported through the code.

Conclusion

In general, the suicide rate is related to several factors like age, sex, country, and the years.Every factor will affect different than the other like the age we can see the suicide people are old more than the young, and so on, so we can find more factor and study it deeply.