Assignment 1 – Data Analysis using R Programming

2024-02-08

Group 3
Yuen Ting Kung
Ceci Lau
Christopher Sta. Romana
Giwoon Yang
Alex Sydfrey Ygusguiza
Man Chun, Yuen
Lawrence Wu

```
library(ggplot2)
library(readxl)
covid_19_data <- read_excel("covid_19_data.xlsx")
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

- **Print the structure of your dataset**

```
str(covid_19_data)
```

```
## tibble [205,957 x 8] (S3: tbl_df/tbl/data.frame)
##  $ SNo            : num [1:205957] 1 1 2 2 3 3 4 5 6 7 ...
##  $ ObservationDate: chr [1:205957] "01/22/2020" "01/22/2020" "01/22/2020" "01/22/2020" ...
##  $ Province/State : chr [1:205957] "Anhui" "Anhui" "Beijing" "Beijing" ...
##  $ Country/Region : chr [1:205957] "Mainland China" "Mainland China" "Mainland China" "Mainland China
##  $ Last Update    : chr [1:205957] "1/22/2020 17:00" "1/22/2020 17:00" "1/22/2020 17:00" "1/22/2020
##  $ Confirmed      : num [1:205957] 1 1 14 14 6 6 1 0 26 2 ...
##  $ Deaths         : num [1:205957] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Recovered      : num [1:205957] 0 0 0 0 0 0 0 0 0 0 ...
```

- **List the variables in your dataset**

```
names(covid_19_data)
```

```
## [1] "SNo"             "ObservationDate" "Province/State"  "Country/Region"
## [5] "Last Update"     "Confirmed"       "Deaths"          "Recovered"
```

- **Print the top 15 rows of your dataset**

```
head(covid_19_data, n = 15)
```

```
## # A tibble: 15 x 8
##      SNo ObservationDate `Province/State` `Country/Region` `Last Update`
##    <dbl> <chr>           <chr>            <chr>            <chr>
##  1     1 01/22/2020      Anhui            Mainland China   1/22/2020 17:00
##  2     1 01/22/2020      Anhui            Mainland China   1/22/2020 17:00
##  3     2 01/22/2020      Beijing          Mainland China   1/22/2020 17:00
##  4     2 01/22/2020      Beijing          Mainland China   1/22/2020 17:00
##  5     3 01/22/2020      Chongqing        Mainland China   1/22/2020 17:00
##  6     3 01/22/2020      Chongqing        Mainland China   1/22/2020 17:00
##  7     4 01/22/2020      Fujian           Mainland China   1/22/2020 17:00
##  8     5 01/22/2020      Gansu            Mainland China   1/22/2020 17:00
##  9     6 01/22/2020      Guangdong        Mainland China   1/22/2020 17:00
## 10     7 01/22/2020      Guangxi          Mainland China   1/22/2020 17:00
## 11     8 01/22/2020      Guizhou          Mainland China   1/22/2020 17:00
## 12     9 01/22/2020      Hainan           Mainland China   1/22/2020 17:00
## 13    10 01/22/2020      Hebei            Mainland China   1/22/2020 17:00
## 14    11 01/22/2020      Heilongjiang     Mainland China   1/22/2020 17:00
## 15    12 01/22/2020      Henan            Mainland China   1/22/2020 17:00
## # i 3 more variables: Confirmed <dbl>, Deaths <dbl>, Recovered <dbl>
```

- **Write a user defined function using any of the variables from the data set**

```
# Function that calculates the average recovered covid cases
avg_recovered <- function(data) {
  return(mean(covid_19_data$Recovered))
}

# Example:
avg_recovered(covid_19_data)
```

```
## [1] 27662.68
```

- **Use data manipulation techniques and filter rows based on any logical criteria that exist in your dataset**

```
country_reg = as.data.frame(covid_19_data %>% filter(covid_19_data$`Country/Region` == "Taiwan"))
head(country_reg)
```

```
##   SNo ObservationDate Province/State Country/Region     Last Update Confirmed
## 1  29      01/22/2020         Taiwan         Taiwan 1/22/2020 17:00         1
## 2  67      01/23/2020         Taiwan         Taiwan   1/23/20 17:00         1
## 3 108      01/24/2020         Taiwan         Taiwan   1/24/20 17:00         3
## 4 155      01/25/2020         Taiwan         Taiwan   1/25/20 17:00         3
## 5 200      01/26/2020         Taiwan         Taiwan   1/26/20 16:00         4
## 6 248      01/27/2020         Taiwan         Taiwan   1/27/20 23:59         5
##   Deaths Recovered
## 1      0         0
## 2      0         0
## 3      0         0
## 4      0         0
## 5      0         0
## 6      0         0
```

- **Identify the dependent & independent variables and use reshaping techniques and create a new data frame by joining those variables from your dataset**

```
cov19_new = as.data.frame((covid_19_data %>% select(3,6:8)))
head(cov19_new)
```

```
##   Province/State Confirmed Deaths Recovered
## 1          Anhui         1      0         0
## 2          Anhui         1      0         0
## 3        Beijing        14      0         0
## 4        Beijing        14      0         0
## 5      Chongqing         6      0         0
## 6      Chongqing         6      0         0
```

- **Remove missing values in your dataset**

```
na.omit(covid_19_data)
```

```
## # A tibble: 150,580 x 8
##      SNo ObservationDate `Province/State` `Country/Region` `Last Update`
##    <dbl> <chr>           <chr>            <chr>            <chr>
##  1     1 01/22/2020      Anhui            Mainland China   1/22/2020 17:00
##  2     1 01/22/2020      Anhui            Mainland China   1/22/2020 17:00
##  3     2 01/22/2020      Beijing          Mainland China   1/22/2020 17:00
##  4     2 01/22/2020      Beijing          Mainland China   1/22/2020 17:00
##  5     3 01/22/2020      Chongqing        Mainland China   1/22/2020 17:00
##  6     3 01/22/2020      Chongqing        Mainland China   1/22/2020 17:00
##  7     4 01/22/2020      Fujian           Mainland China   1/22/2020 17:00
##  8     5 01/22/2020      Gansu            Mainland China   1/22/2020 17:00
```

```
##  9     6 01/22/2020      Guangdong        Mainland China   1/22/2020 17:00
## 10     7 01/22/2020      Guangxi          Mainland China   1/22/2020 17:00
## # i 150,570 more rows
## # i 3 more variables: Confirmed <dbl>, Deaths <dbl>, Recovered <dbl>
```

- **Identify and remove duplicated data in your dataset**

```
covid_19_data_test <- read_excel("covid_19_data_test.xlsx")

# Identifty duplicated data
duplicated(covid_19_data_test)
```

```
##   [1] FALSE  TRUE FALSE  TRUE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
##  [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [25]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE
##  [61] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [121] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE FALSE
## [133] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [145] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [157] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [169] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [181] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
## [193] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [205] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE
## [217]  TRUE  TRUE
```

```
# Remove duplicated data
cov19_unique = unique(covid_19_data_test)

# Run duplicated operation again, expected output: all entry is FALSE
duplicated(cov19_unique)
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [121] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [133] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [145] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
## [157] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [169] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [181] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [193] FALSE
```

- **Reorder multiple rows in descending order**

```
covid_19_data %>% arrange(desc(SNo))
```

```
## # A tibble: 205,957 x 8
##       SNo ObservationDate `Province/State`   `Country/Region` `Last Update`
##     <dbl> <chr>           <chr>              <chr>            <chr>
##  1 205951 01/19/2021      Zuid-Holland       Netherlands      44216.223541666666
##  2 205950 01/19/2021      Zhytomyr Oblast    Ukraine          44216.223541666666
##  3 205949 01/19/2021      Zhejiang           Mainland China   44216.223541666666
##  4 205948 01/19/2021      Zeeland            Netherlands      44216.223541666666
##  5 205947 01/19/2021      Zaporizhia Oblast  Ukraine          44216.223541666666
##  6 205946 01/19/2021      Zakarpattia Oblast Ukraine          44216.223541666666
##  7 205945 01/19/2021      Zacatecas          Mexico           44216.223541666666
##  8 205944 01/19/2021      Zabaykalsky Krai   Russia           44216.223541666666
##  9 205943 01/19/2021      Yunnan             Mainland China   44216.223541666666
## 10 205942 01/19/2021      Yukon              Canada           44216.223541666666
## # i 205,947 more rows
## # i 3 more variables: Confirmed <dbl>, Deaths <dbl>, Recovered <dbl>
```

- **Rename some of the column names in your dataset**

```
covid_19_data %>% rename(Observation_Date = ObservationDate, Province_State =`Province/State`)
```

```
## # A tibble: 205,957 x 8
##      SNo Observation_Date Province_State `Country/Region` `Last Update`
##    <dbl> <chr>            <chr>          <chr>            <chr>
##  1     1 01/22/2020       Anhui          Mainland China   1/22/2020 17:00
##  2     1 01/22/2020       Anhui          Mainland China   1/22/2020 17:00
##  3     2 01/22/2020       Beijing        Mainland China   1/22/2020 17:00
##  4     2 01/22/2020       Beijing        Mainland China   1/22/2020 17:00
##  5     3 01/22/2020       Chongqing      Mainland China   1/22/2020 17:00
##  6     3 01/22/2020       Chongqing      Mainland China   1/22/2020 17:00
##  7     4 01/22/2020       Fujian         Mainland China   1/22/2020 17:00
##  8     5 01/22/2020       Gansu          Mainland China   1/22/2020 17:00
##  9     6 01/22/2020       Guangdong      Mainland China   1/22/2020 17:00
## 10     7 01/22/2020       Guangxi        Mainland China   1/22/2020 17:00
## # i 205,947 more rows
## # i 3 more variables: Confirmed <dbl>, Deaths <dbl>, Recovered <dbl>
```

• **Add new variables in your data frame by using a mathematical function (for e.g. – multiply an existing column by 2 and add it as a new variable to your data frame)**

```
covid_19_data = as.data.frame(covid_19_data %>% mutate(New.Status = covid_19_data$Confirmed -
                                                       covid_19_data$Recovered))
head(covid_19_data)
```

```
##   SNo ObservationDate Province/State Country/Region     Last Update Confirmed
## 1   1      01/22/2020          Anhui Mainland China 1/22/2020 17:00         1
## 2   1      01/22/2020          Anhui Mainland China 1/22/2020 17:00         1
## 3   2      01/22/2020        Beijing Mainland China 1/22/2020 17:00        14
## 4   2      01/22/2020        Beijing Mainland China 1/22/2020 17:00        14
## 5   3      01/22/2020      Chongqing Mainland China 1/22/2020 17:00         6
## 6   3      01/22/2020      Chongqing Mainland China 1/22/2020 17:00         6
##   Deaths Recovered New.Status
## 1      0         0          1
## 2      0         0          1
## 3      0         0         14
## 4      0         0         14
## 5      0         0          6
## 6      0         0          6
```

• **Create a training set using random number generator engine**

```
cov19_training = cbind(covid_19_data$`Province/State`, covid_19_data$`Country/Region`,
                       covid_19_data$Confirmed, covid_19_data$Recovered)
cov19_training = as.data.frame(cov19_training)
set.seed(1234)

cov19_training = as.data.frame(cov19_training %>% sample_frac(0.8, replace = FALSE))
head(cov19_training)
```

```
##            V1              V2    V3    V4
## 1       Nuble           Chile  5229  4612
## 2     Guainia        Colombia     7     0
## 3  Chandigarh           India 14528 13708
## 4   Los Lagos           Chile   894     0
## 5        <NA> Papua New Guinea   541   527
## 6 La Libertad            Peru 28643     0
```

```
# Run generator again to show difference
cov19_training = as.data.frame(cov19_training %>% sample_frac(0.8, replace = FALSE))
head(cov19_training)
```

```
##                V1       V2    V3    V4
## 1         Atacama    Chile  7905  7725
## 2 Belgorod Oblast   Russia  8008  7389
## 3           Piaui   Brazil 74096 71690
```

```
## 4         Abruzzo    Italy  3237  2063
## 5 Valle del Cauca Colombia 43598 31195
## 6         Jalisco   Mexico 17587 13661
```

- **Print the summary statistics of your dataset**

```
summary(covid_19_data)
```

```
##       SNo          ObservationDate    Province/State     Country/Region
##  Min.   :     1   Length:205957      Length:205957      Length:205957
##  1st Qu.: 51484   Class :character   Class :character   Class :character
##  Median :102973   Mode  :character   Mode  :character   Mode  :character
##  Mean   :102973
##  3rd Qu.:154462
##  Max.   :205951
##  Last Update          Confirmed           Deaths        Recovered
##  Length:205957      Min.   :-302844   Min.   : -178   Min.   :-854405
##  Class :character   1st Qu.:    596   1st Qu.:    7   1st Qu.:     11
##  Mode  :character   Median :   5361   Median :  101   Median :   1028
##                     Mean   :  45400   Mean   : 1263   Mean   :  27663
##                     3rd Qu.:  26184   3rd Qu.:  713   3rd Qu.:   9879
##                     Max.   :3049037   Max.   :80143   Max.   :6399531
##    New.Status
##  Min.   :-6399531
##  1st Qu.:     101
##  Median :    1407
##  Mean   :   17737
##  3rd Qu.:    8934
##  Max.   : 3049037
```

- **Use any of the numerical variables from the dataset and perform the following statistical functions**

```
# Mean of confirmed cases
conf_mean <- c(covid_19_data$Confirmed)
result.mean <- mean(conf_mean)
print(result.mean)
```

```
## [1] 45400.16
```

```
# Median of deaths
death_median <- c(covid_19_data$Deaths)
result.median <- mean(death_median)
print(result.median)
```

```
## [1] 1262.597
```

```r
# Mode of range of Country/Region
reg_mode <- c(covid_19_data$`Country/Region`)
result.mode <- mode(reg_mode)
print(result.mode)
```

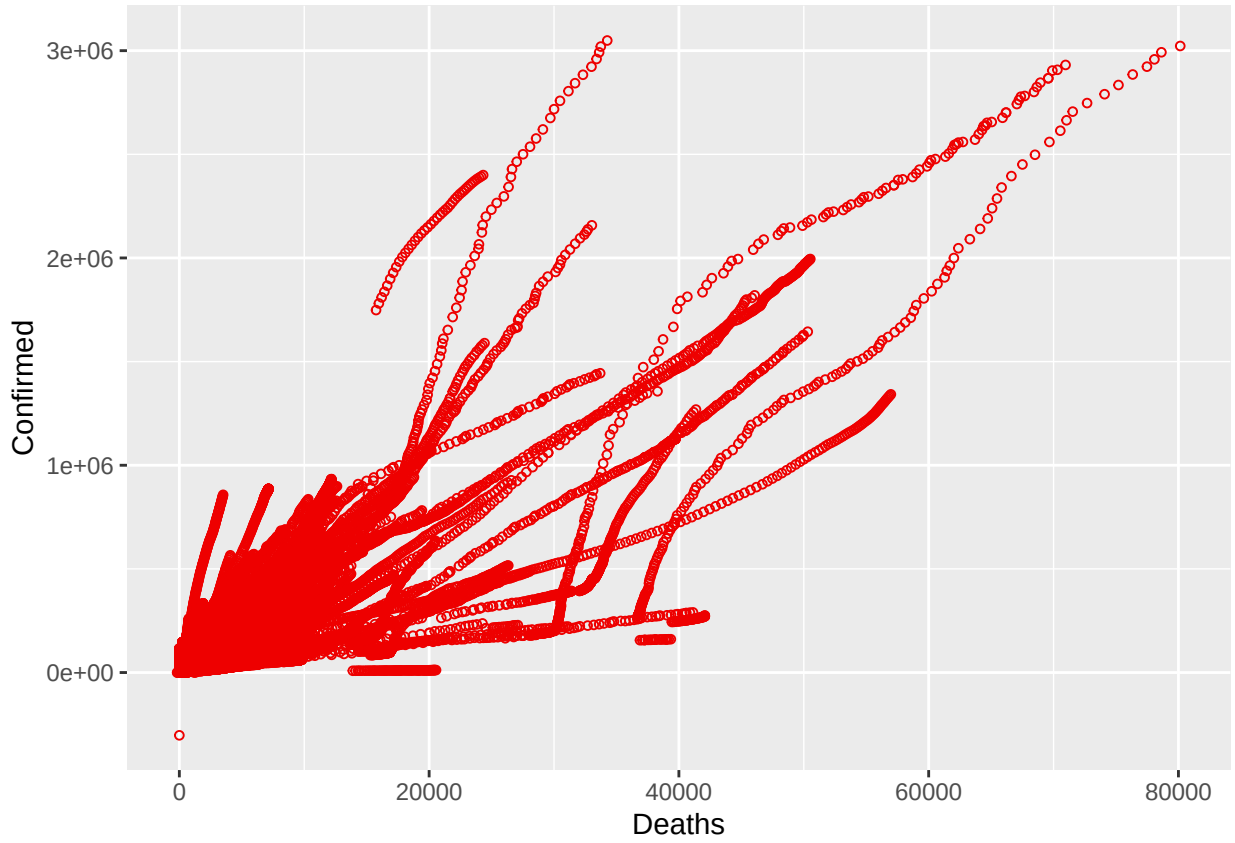
```
## [1] "character"
```

```r
# Range of recovered cases
print(range(covid_19_data$Recovered))
```

```
## [1] -854405 6399531
```

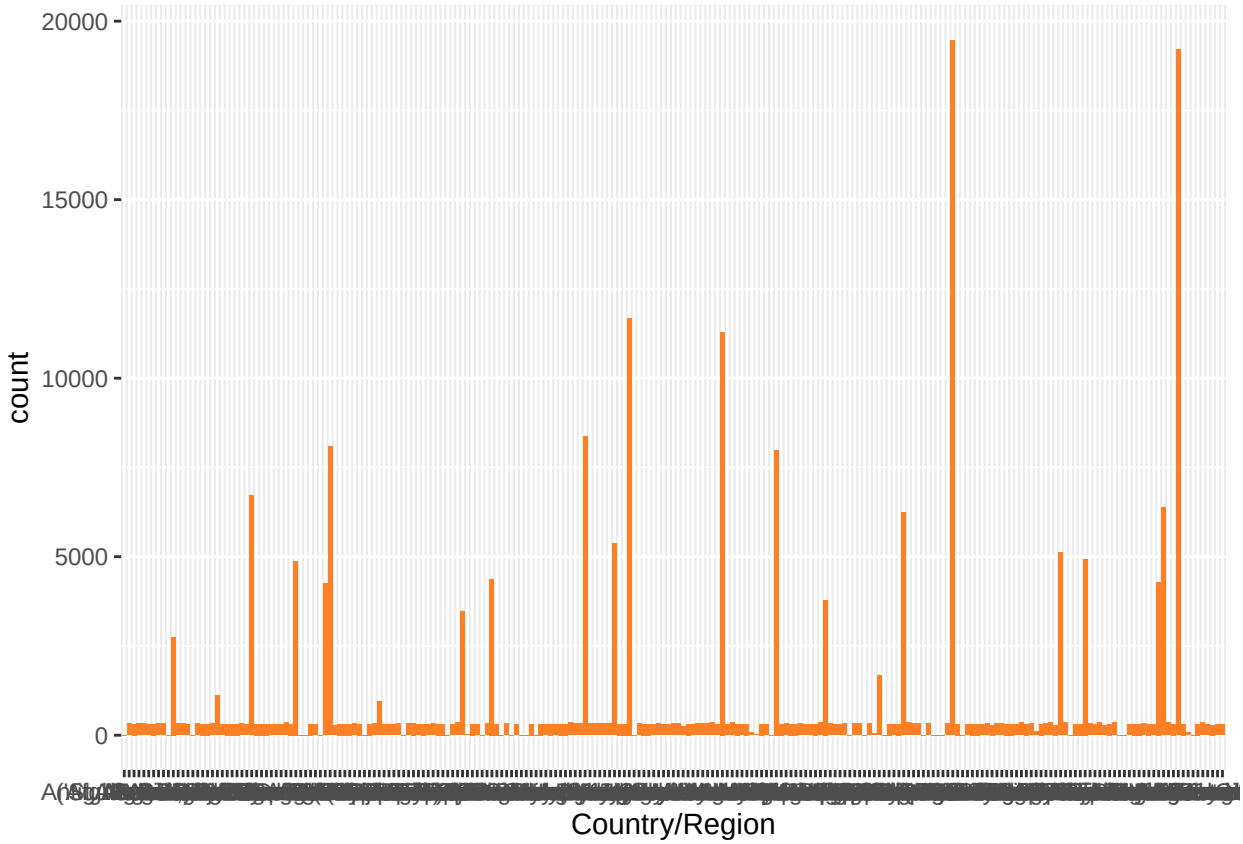
- **Plot a scatter plot for any 2 variables in your dataset**

```r
# Scatter plot of covid deaths against confirmed cases
ggplot(covid_19_data, aes(x = Deaths, y = Confirmed)) + geom_point(size = 1.2, color = "red2",
                                                                   shape = 21)
```



- **Plot a bar plot for any 2 variables in your dataset**

```r
# Bar Plot of covid deaths per country
ggplot(covid_19_data, aes(x = `Country/Region`, fill = Deaths))+geom_bar(fill = "chocolate1")
```



- **Find the correlation between any 2 variables by applying least square linear regression model**

```r
# Correlation of confirmed covid cases againts recovered cases
cov19_corr = cor(covid_19_data$Confirmed, covid_19_data$Recovered, method = "pearson")
cov19_corr
```

```
## [1] 0.5094465
```

[1]

[1]The .RMD file, PDF report, and the dataset can be found on https://github.com/alexsydfrey/Data-Analysis-using-R