GYZR7-Report

Table of contents

1 Introduction

1.1 Description of Dataset

We will be using the data set detailing the Vaccine Coverage and Disease Burden statistics provided and compiled by WHO in 2017. The data set has 14 variables and 7818 rows. The data details the amount of immunizations as a percentage of the total population, deaths by illness and number of cases of infection over 42 years from 1974 to 2015. The variables contained in this data are either Nominal, Ratio scale or Interval data .I will further detail the variables and characteristics in Table 1 below:

1.1.1 Table 1

Variable	Description	Levels/Values	NA Values
Entity	Country name, Nominal Data	201 Levels	0
Year	Year of data recorded, 1974-2015	42 Levels	0
BCG.immunization coverage among 1 year olds	Percentage of population immunized		2324
DTP3 immunization coverage among 1 year olds	Percentage of population immunized		1200
Polio (Pol3) immunization coverage among 1 year olds	Percentage of population immunized		1194
Measles (MCV) immunization coverage among one year olds	Percentage of population immunized		1328
Number of Confirmed Tetanus Cases	Number of population infected		1186
Number of confirmed Polio cases	Number of population infected		6834
Number of confirmed Pertussis cases	Number of population infected		1062
Number of confirmed Measles cases	Number of population infected		518
Number of confirmed Diphtheria cases	Number of population infected		1020
Estimated Deaths due to Tuberculosis per 100,000 population	Infection rate per 100,000 population (treat as percentage)		4625
Estimated deaths due to Tuberculosis excluding HIV	Percentage of population that perished to Tuberculosis		4625

2 Descriptive Statistics of Data & Data Wrangling: Part 1A

```
Vaccine.1<-read.csv("Vaccine Coverage and Disease Burden - WHO (2017).csv")
if(!require("pacman")) {
  install.packages("pacman")
}
```

```
Loading required package: pacman
```

```
pacman::p_load(
  tidyverse,
  glue,
  gapminder,
  ggplot2,
  dplyr,
  purrr

)
```

Once I loaded in the data set I now begin to clean and format the dataset. I factor both the columns 'Year' and 'Entity' with this code

```
Vaccine.1$Entity<-factor(Vaccine.1$Entity)
l1<-levels(Vaccine.1$Entity)
Vaccine.1$Year<-factor(Vaccine.1$Year)
l2<-levels(Vaccine.1$Year)
```

I then figure out the number of *NA* values in each column to filter out the *NA* data later. Then I also use the functions glimpse and summary to find out and evaluate the mean, standard deviations, min, max and the data types of variable. I further detail the data types and variables in Table 2 below.

```
table(colSums(is.na(Vaccine.1)))
glimpse(Vaccine.1)
summary(Vaccine.1)
```

2.0.1 Table 2

Variable Name	Max	Mean	Median	Min
Entity	NA	NA	NA	NA
Year	NA	NA	NA	NA
BCG immunization coverage among 1 year olds	99	82	91	1
Hepatitis B immunization coverage among 1 year olds	99	81	91	1
DTP3 immunization coverage among 1 year olds	99	77	87	1

Variable Name	Max	Mean	Median	Min
Polio (Pol3) immunization coverage among 1 year olds	99	78	88	1
Measles (MCV) immunization coverage among one year olds	99	76	85	1
Number of Confirmed Tetanus Cases	115791	786.3	10	0
Number of confirmed Polio cases	461	4.912	0	0
Number of confirmed Pertussis cases	1982355	9758	107	0
Number of confirmed Measles cases	4430074	23675	382	0
Number of confirmed Diphtheria cases	97511	454.7	0	0
Estimated Deaths due to Tuberculosis per 100,000 population, excluding HIV	161	17.02	5.8	0
Estimated number of deaths due to Tuberculosis excluding HIV	900000	15541	450	0

I then subset **Vaccine.1** and transform it into Vaccine.2, by dropping all NA values. This leaves the dataset having only 595 observations of 14 variables.

```
Vaccine.2<-drop_na(Vaccine.1)
Vaccine.2
glimpse(Vaccine.2)
colSums(is.na(Vaccine.2))
summary(Vaccine.2)
```

I acknowledge that the reduction of the sample is a drawback of the statistical inferences I have made, however, I feel that the dropping of NA values makes the calculations more robust and the code more reproducible as it is less prone to errors.

3 Data Processing: Part 1-B

Now, I begin preparing the data for visualization, by creating Data frames, I will split this section into three parts„ as I did separate tasks for each plot I would create.

3.1 Data Plot 1

For this plot I create a new column in **Vaccine.2** and create new variable named *'Total Population'*. I calculate this by taking the total number of Tuberculosis deaths and multiplying the average number of Tuberculosis deaths per 100 people while dividing it by hundred.

```
Vaccine.2[,15]<-Vaccine.2[,14]*(100/Vaccine.2[,13])
colnames(Vaccine.2)[15]<-"Total Population"
```

Then I create another data frame- **Vaccine.3**. I subset **Vaccine.2** and select the variables relating to measles, Country names and Years. I also create a new variable column called *Total Number of Children Immunized (MCV)* , with the raw number of children immunized against Measles, by taking the percentage value given and multiplying it by the total population. This is sound, as the values for Immunization are given as a percentage of the total population according to the handbook of the dataset.

```
Vaccine.3<- Vaccine.2 %>% select(Entity,Year,
                           Measles..MCV..immunization.coverage.among.1.year.olds..WHO.2017.
                           Number.of.confirmed.measles.cases..WHO.2017.,'Total Population')
Vaccine.3[,6]<-Vaccine.3[,5]*(Vaccine.3[,3]/100)
colnames(Vaccine.3)[6]<-"Total Number of Children Immunized (MCV)"
Vaccine.3
```

I notice that the column names for the variables are very large, so I create a function to change all the column names at once. I wrote the function, so that the list input would be indexed as the new column names using colnames function from base package and return only the data input.

```
rename_columns <- function(data, new_names) {
  colnames(data) <- new_names
  return(data)
}
```

I then renamed the columns.

```
new_colnames<- c("Country","Years","Total Percent of Children Immunized (MCV)",
                 "Number of Measles Cases",
                 "Total Population",
                 "Total Number of Children Immunized (MCV)")
Vaccine.3<-rename_columns(Vaccine.3,new_colnames)
```

I parsed the data and noticed that after NA's and subsetting had been accomplished, the data only had Year values from 2010-2014. I wanted countries that had a larger number of observations to ensure that the graphs would render well. So, I then went on to create a function to check the number of objects corresponding to each level value. I wanted to know which countries had 5 objects, which was the maximum. I then filtered to only Georgia and Afghanistan, that had 5 data points and created a new dataframe **Vaccine.GA**.

```r
count_levels <- function(data, variable_name) {
  result <- table(data[[variable_name]])
  return(result)
}
count_levels(Vaccine.3,"Country")
Vaccine.GA<- subset.data.frame(Vaccine.3,Country %in% c("Georgia","Afghanistan"))
levels(Vaccine.GA$Country)
factor(Vaccine.GA$Country)
```

The ***count_levels*** function, works to create a table of counts for the *variable name* object inputted into the function, essentially counting the occurrences of a factored variable.

3.2 Data Plot 2

For this Data processing, I wanted to later create a bar graph across different types of diseases and compare the rates. So, I subsetted to the Year 2014 and only included variables containing confirmed cases of diseases. I also filtered country values encompassing large regions of the world, then used the function *rename_columns* I created.

```r
Vaccine.2014<-subset(Vaccine.2[1:12], Vaccine.2$Year=="2014" & Entity %in%
                       c("Europe","South-East Asia","Africa"))
Vaccine.2014<-Vaccine.2014[,-c(3:7)]
new_colnames_2<-c("Entity","Year","Tetanus Cases",
                  "Polio Cases","Pertussis Cases",
                  "Measles Cases",
                  "Diphtheria Cases")
Vaccine.2014<-rename_columns(Vaccine.2014,new_colnames_2)
Vaccine.2014
```

3.3 Data Plot 3

For this Data set I wanted to create a bubble graph. So I subsetted by all of the Tuberculosis data from the dataset Vaccine.2 and also by 5 countries that had data for 5 years, I also included the Year variable. Thus, I made the dataframe **Vaccine.TB**. I then created a new variable column called *Total Number of Children Immunized (BCG)* by multiplying the immunization percentages by the total population. I also re-factored and checked the levels of the **Year** variable as my plot had rendering issues.

```r
Vaccine.TB<- subset(Vaccine.2[,-c(4:12)], Vaccine.2$Entity %in% c("Iran","Ireland",
"Romania","India","Gambia","Costa Rica","Russia"))
```

```
Vaccine.TB[,6]<-Vaccine.TB[,5]*(100/Vaccine.TB[,4])
colnames(Vaccine.TB)[6]<-"Total Population"
Vaccine.TB[,7]<-Vaccine.TB[,6]*(Vaccine.TB[,3]/100)
new_colnames_3<- c("Country","Years","BCG vaccine among 1 year olds",
                   "Deaths due to TB per hundred people",
                   "Estimated TB deaths","Total Population")
Vaccine.TB<-rename_columns(Vaccine.TB,new_colnames_3)
colnames(Vaccine.TB)[7]<-"Total Number of Children Immunized (BCG)"

Vaccine.TB$Year<-`levels<-`(Vaccine.TB$Year,
                            c(2010,2011,2012,2013,2014))
droplevels.factor(Vaccine.TB$Year)
```

4 Data Visualization: Part 2A

4.1 Plot 1

```
##Re-Loading Dataframes for Rendering##
Vaccine.1<-read.csv("Vaccine Coverage and Disease Burden - WHO (2017).csv")
Vaccine.2<-drop_na(Vaccine.1)
Vaccine.2[,15]<-Vaccine.2[,14]*(100/Vaccine.2[,13])
colnames(Vaccine.2)[15]<-"Total Population"
Vaccine.3<- Vaccine.2 %>% select(Entity,Year,
                                 Measles..MCV..immunization.coverage.among.1.year.olds..WH
                                 'Total Population')
Vaccine.3[,6]<-Vaccine.3[,5]*(Vaccine.3[,3]/100)
colnames(Vaccine.3)[6]<-"Total Number of Children Immunized (MCV)"
rename_columns <- function(data, new_names) {
  colnames(data) <- new_names
  return(data)
}
new_colnames<- c("Country","Years","Total Percent of Children Immunized (MCV)","Number of
Vaccine.3<-rename_columns(Vaccine.3,new_colnames)

Vaccine.GA <- subset(Vaccine.3, Country %in% c("Georgia", "Afghanistan"))

##Plot 1##

library(ggplot2)
```

```
Vaccine.GA<- Vaccine.GA %>% arrange(Years)
ggplot(Vaccine.GA, aes(x =`Total Number of Children Immunized (MCV)`,
                       y =`Number of Measles Cases`, color = Country)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Immunization rates and Measles Rate contrast for Afghanistan and Georgia",
       x = "Immunization for 1 year olds 2010-14",
       y = "Measles Infection rate 2010-14",
       color = "Country") +
  scale_x_continuous(breaks = seq(0,
  max(Vaccine.GA$`Total Number of Children Immunized (MCV)`), by = 2000)) +
  scale_y_continuous(breaks = seq(0,
                    max(Vaccine.GA$`Number of Measles Cases`), by = 2000)) +
  theme_grey()+
  theme(plot.title=element_text(hjust=0.5))
```

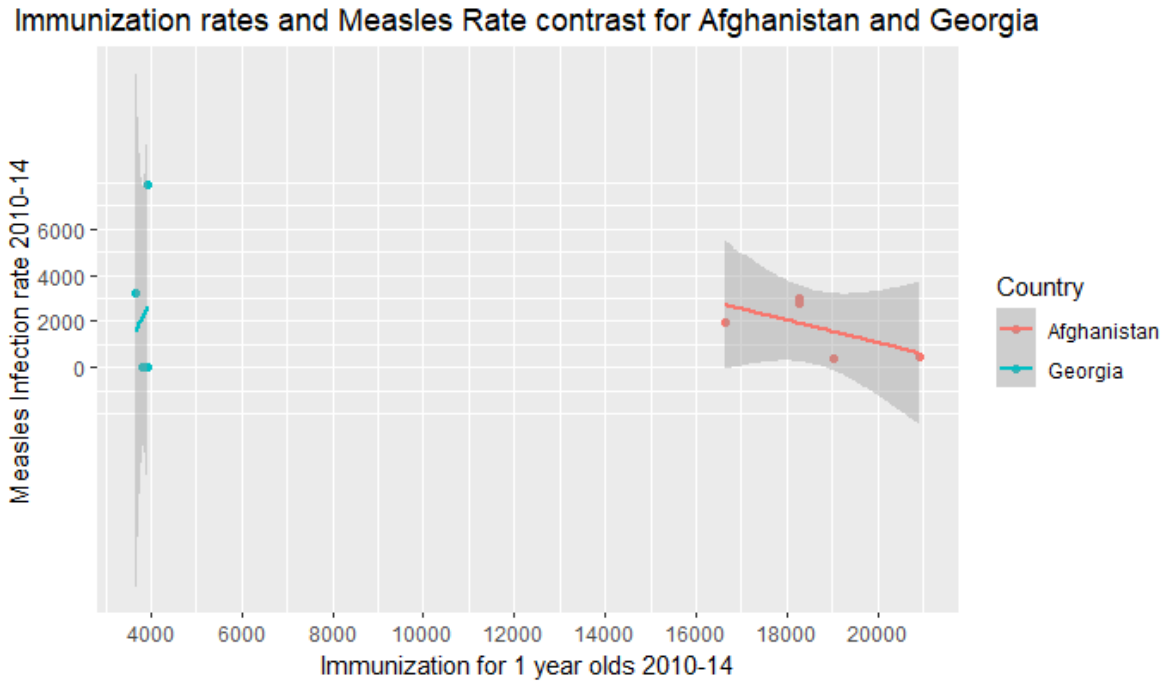

Figure 1: Plot 1: Scatter with AB line

The plot is a simple scatterplot, and I opted to forgo labels as they overlapped when loaded into the report. The plot points are in order of year and show an clear upward trend of measles infection in Georgia over 5 years, and a gentle downward trend in Afghanistan. However, Georgia has generally low rates of Measles, with values being 0 for two years with an outlier in

one year. While, Afghanistan hovers between 1,000 and 3,000 cases over all years. This also makes sense with immunization rates being low in Georgia as their normal rates of infection are low, while Afghanistan would try to increase immunization since their rates of infection are so high. I chose to do a scatter-plot as I felt it was the best way to include the two variables of Country and Year, which are descriptive variables for the two dependent ones that were Immunization rates and Measles infection rates. I felt that adding a trend line and shape would help with comprehending the data more clearly. It also gives insight into how Immunization rates and Infection rates (For Measles) are related.

4.2 Plot 2

```
##Re-Loading Dataframes for Rendering##
Vaccine.1<-read.csv("Vaccine Coverage and Disease Burden - WHO (2017).csv")
Vaccine.2<-drop_na(Vaccine.1)
Vaccine.2[,15]<-Vaccine.2[,14]*(100/Vaccine.2[,13])
colnames(Vaccine.2)[15]<-"Total Population"
Vaccine.2014<-subset(Vaccine.2[1:12], Vaccine.2$Year=="2014" & Entity %in% c("Europe","Sou
Vaccine.2014<-Vaccine.2014[,-c(3:7)]
new_colnames_2<-c("Entity","Year","Tetanus Cases","Polio Cases","Pertussis Cases","Measles
Vaccine.2014<-rename_columns(Vaccine.2014,new_colnames_2)


##Plot 2##

create_bar_g <- function(data, x_col, y_cols) {
  plots <- list() ##Empty List##
  for (y_col in y_cols) {
    p <- ggplot(data, aes(x = {{x_col}}, y = .data[[y_col]])) +
      geom_bar(stat = "identity", position = "dodge") +
      geom_text(aes(label = .data[[y_col]]), vjust = -0.5)+
      labs(title = glue("Bar Graph for {.data[[x_col]][1]} ({y_col})",
                        .data) ,
           x = deparse(substitute("Parts of the World 2014")),
           y = y_col)
    plots[[y_col]] <- p
  }
  return(plots)
}

y_cols <- colnames(Vaccine.2014)[3:7]
```

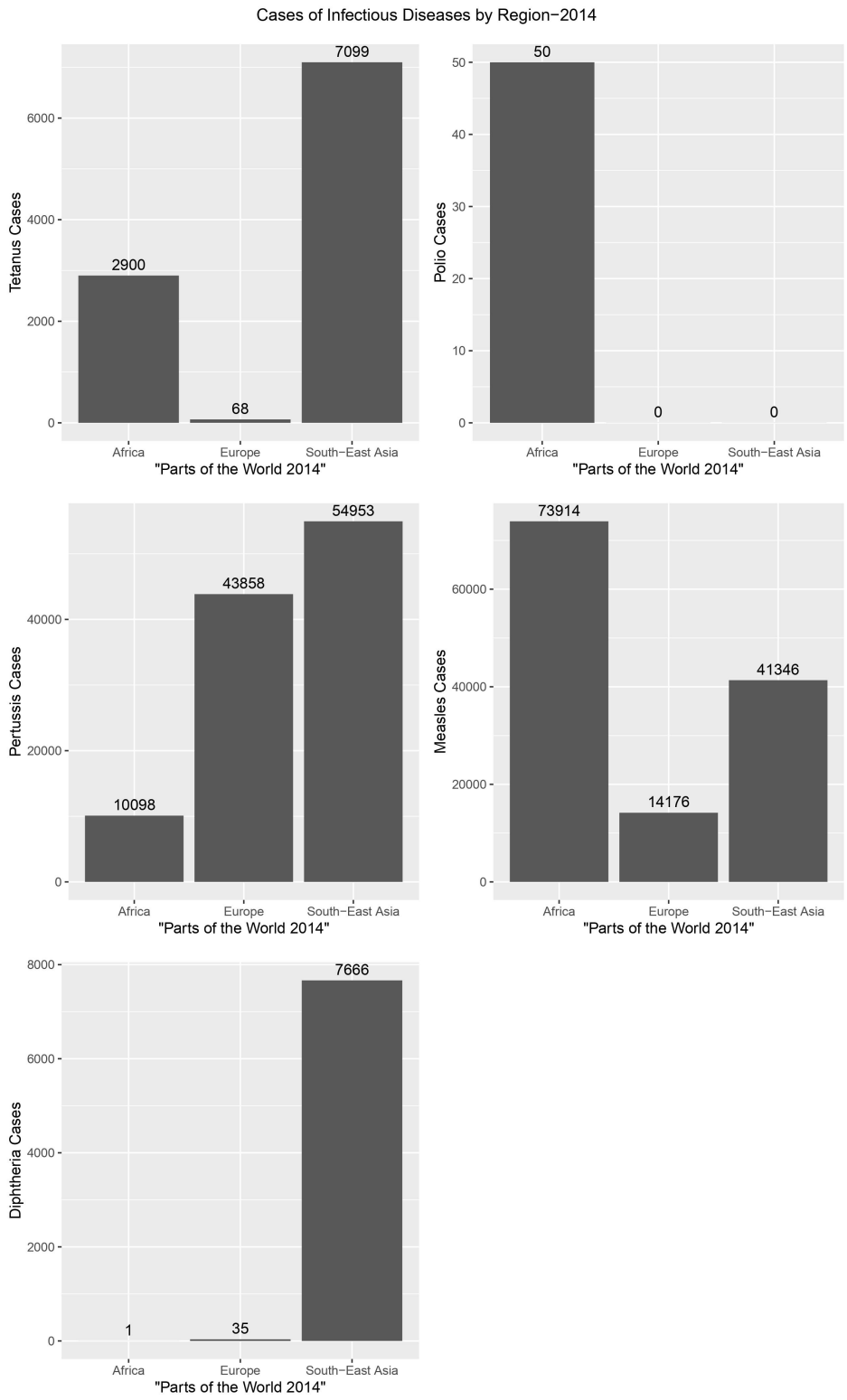

Figure 2: Plot 2: Bar Graphs

```
bar_graphs <- create_bar_g(Vaccine.2014,
                           x_col = Vaccine.2014$Entity, y_cols = y_cols)
bar_graphs <- map(y_cols, ~ create_bar_g(Vaccine.2014,
                                         x_col = Vaccine.2014$Entity, y_col = .x))
walk(bar_graphs, print)

library(gridExtra)

b_list<-c(bar_graphs[[1]],
          bar_graphs[[2]],
          bar_graphs[[3]],
          bar_graphs[[4]],bar_graphs[[5]])
n <- length(b_list)
nCol <- floor(sqrt(n))
do.call("grid.arrange", c(b_list, ncol=nCol,
                          top= "Cases of Infectious Diseases by Region-2014"))
```

Firstly, I created the function **create_bar_g**. This function was used to create a set of Bar Graphs. I then used *map* to apply the function to each variable column . The function first sets out a data source, an x column to pull data from and multiple y columns (ycols) it will iterate through to produce different bar_graphs. The function produces an empty list and the code ***for ycol in ycols*** makes the function go through each y-column specified. Then I create a plot object **p** and use glue function to create labels in the bar graph, that reflect the current **y-co**l data. I then use grid.arrange to arrange all the plots into a grid. The bar plots show that Southeast Asia region has the highest number of Infectious disease cases when compared to the other two regions, with Africa being second highest. However, this data may be skewed as dropping the NA's in the dataset did change the standard deviation and mean of the variables in the data, and decreased the amount of entries. We further cut down the entries and only look at data from 2014. In conclusion, we cannot observe longitudinal trends with this visualization, but we can compare rates between countries at a certain moment in time.

4.3 Plot 3

I then began plotting my bubble plot.

```
##Re-Loading Dataframes for Rendering##

Vaccine.1<-read.csv("Vaccine Coverage and Disease Burden - WHO (2017).csv")
Vaccine.2<-drop_na(Vaccine.1)
Vaccine.2[,15]<-Vaccine.2[,14]*(100/Vaccine.2[,13])
```

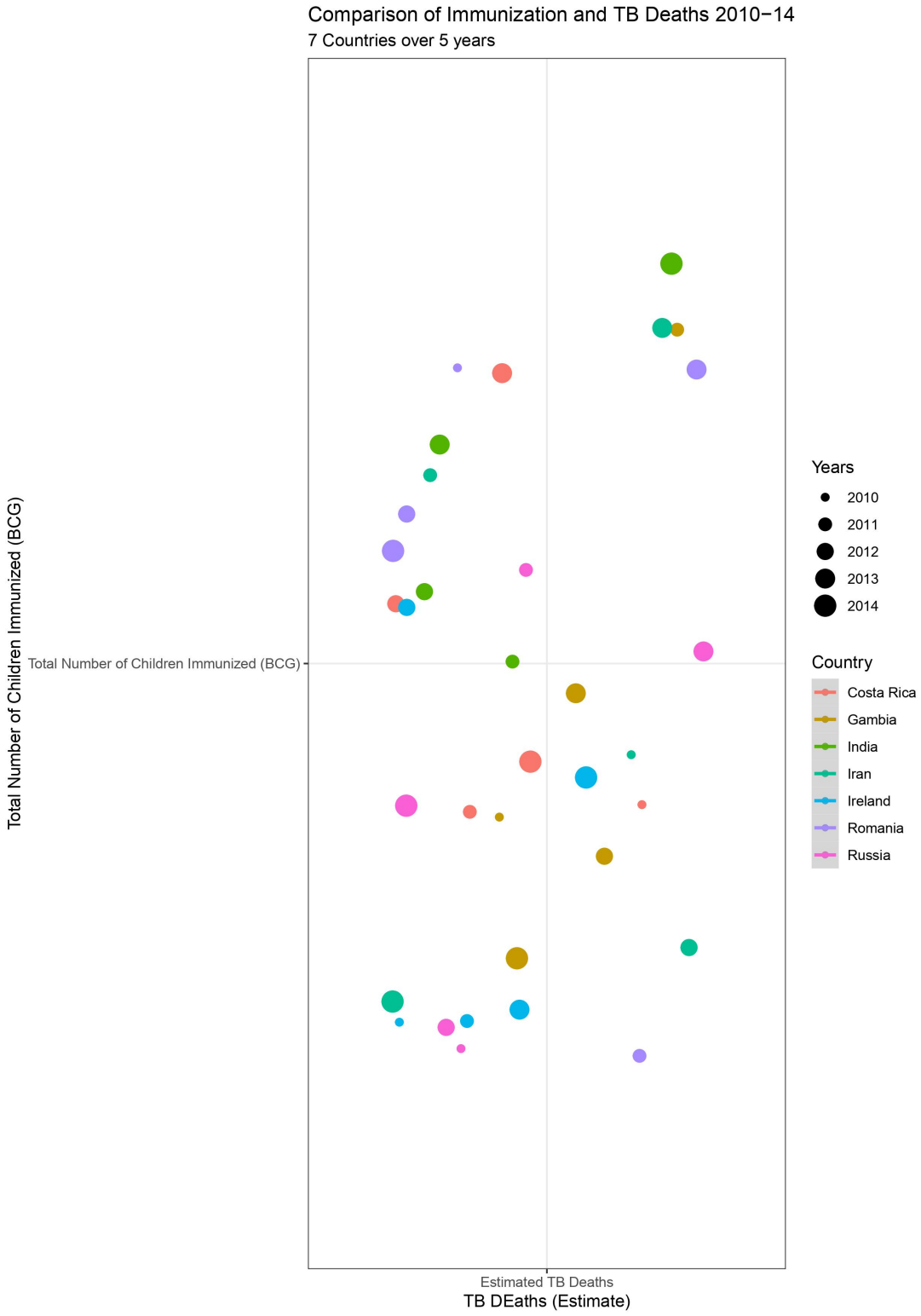

Figure 3: Plot 3: Bubble Plot

```
colnames(Vaccine.2)[15]<-"Total Population"
Vaccine.TB<- subset(Vaccine.2[,-c(4:12)], Vaccine.2$Entity %in% c("Iran","Ireland","Romani
Vaccine.TB[,6]<-Vaccine.TB[,5]*(100/Vaccine.TB[,4])
colnames(Vaccine.TB)[6]<-"Total Population"
Vaccine.TB[,7]<-Vaccine.TB[,6]*(Vaccine.TB[,3]/100)
new_colnames_3<- c("Country","Years","BCG vaccine among 1 year olds","Deaths due to TB per
Vaccine.TB<-rename_columns(Vaccine.TB,new_colnames_3)
colnames(Vaccine.TB)[7]<-"Total Number of Children Immunized (BCG)"
Vaccine.TB$Year<-`levels<-`(Vaccine.TB$Year, c(2010,2011,2012,2013,2014))

##Plot 3##

Vaccine.TB$Years<-as.factor(Vaccine.TB$Years)

g <- ggplot(Vaccine.TB, aes('Estimated TB Deaths',
                            'Total Number of Children Immunized (BCG)' )) +
  labs(subtitle="7 Countries over 5 years",
       title="Comparison of Immunization and TB Deaths 2010-14",
       size = "Years",
       color = 'Country',x= "TB DEaths (Estimate)" ,
       y= "Total Number of Children Immunized (BCG)")+
  geom_jitter(aes(col=Vaccine.TB$Country, size= Vaccine.TB$Years)) +
  geom_smooth(aes(col=Vaccine.TB$`Country`), method="lm")

print(g)
```

The bubble plot maps out the Estimated TB deaths against the Total number of Children Immunized with BCG. The size of the bubble denotes the year while the colour denotes the country. We have data for 5 years from 7 countries . We can look closely at Year data or Country data, depending on what we want ot observe. For example, we can see that India has an upward trend of immunization and a generally lower trend of TB deaths . We can also observe that Ireland, India and Costa Rica all had similar rates in 2012, as they are clustered together in the graph. The bubble plot gives insight into the trends between countries, between different Years and also insight on the correlation between TB immunization and infection rates.

5 AI Engagement:Part 2-C

I used the Chat GPT 3.5 A.I tool to polish code and understand errors.

Firstly, I used A.I to polish function codes that I was creating. For example:

I originally did this to create the function

```
count_levels <- function(data, variable_name) {
  result <- table(data[[variable_name]])
  return(result)
}
```

I later asked A.I to polish it and it told me to do this:

```
count_levels <- function(data, variable_name) {
  table(data[[variable_name]])
}
```

However, this code did not give me the required result and I stuck with my iteration. A second example:

```
library(gridExtra)
```

```
library(plyr)
```

```
grobs<-lapply(bar_graphs,ggplotGrob)
```

```
grid.arrange(grobs= bar_graphs,ncols= 1, top= "Cases of Infectious Diseases
by Region-2014")
```

To create a grid of graphs, however I was encountering error messages. I understood the problem after pasting the error message into GPT, and I was able to come up with a longer method which was again refined by GPT into this:

```
b_list<-c(bar_graphs[[1]], bar_graphs[[2]],bar_graphs[[3]],bar_graphs[[4]],bar_graphs[[5]])
n <- length(b_list) nCol <- floor(sqrt(n)) do.call("grid.arrange", c(b_list,
ncol=nCol, top= "Cases of Infectious Diseases by Region-2014"))
```

Thirdly, I used GPT to fix my graph indents for this report. I pasted my graph and asked GPT to indent it properly, as the text was not indented coherently. I also used A.I to understand a rendering Error on Quarto and GPT advised me on how to arrange my code chunks so that data frames are seen as existing in the IDE environment. In Conclusion, A.I helped ease my workflow and also helped me work through bugs that would have otherwise taken me much longer.