World Happiness Data

Malay Raj

2023-05-13

## WORLD HAPPINESS DATA:

This analysis is based on the World Happiness case study “‘Sophisticated, Clear, and Polished’: World Happiness data and Data Visualization”.

## Installing required packages

Installing and loading common packages and libraries.

library(tidyverse)

## Warning: package 'tidyverse' was built under R version 4.2.3

## Warning: package 'ggplot2' was built under R version 4.2.3

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.0 ✔ readr 2.1.4  
## ✔ forcats 1.0.0 ✔ stringr 1.5.0  
## ✔ ggplot2 3.4.2 ✔ tibble 3.1.8  
## ✔ lubridate 1.9.2 ✔ tidyr 1.3.0  
## ✔ purrr 1.0.1   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the ]8;;http://conflicted.r-lib.org/conflicted package]8;; to force all conflicts to become errors

library(lubridate)  
library(ggplot2)

library(readr)  
library(readxl)  
library(dplyr)  
library(plyr)

## ------------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.  
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:  
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##   
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

## The following object is masked from 'package:purrr':  
##   
## compact

library(skimr)

## Warning: package 'skimr' was built under R version 4.2.3

library(tidyr)  
library(reshape2)

## Warning: package 'reshape2' was built under R version 4.2.3

##   
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':  
##   
## smiths

library(stringr)  
library(e1071)

## Warning: package 'e1071' was built under R version 4.2.3

library(pROC)

## Warning: package 'pROC' was built under R version 4.2.3

## Type 'citation("pROC")' for a citation.

##   
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':  
##   
## cov, smooth, var

library(corrplot)

## Warning: package 'corrplot' was built under R version 4.2.3

## corrplot 0.92 loaded

library(GGally)

## Warning: package 'GGally' was built under R version 4.2.3

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

## Loading the data

The data which is being loaded here are World Happiness Data from the year 2015-2019.

q1\_2015 = read.csv ("C:/Users/KIIT/OneDrive/Desktop/Happiness Data/2015.csv")  
q2\_2016 = read.csv ("C:/Users/KIIT/OneDrive/Desktop/Happiness Data/2016.csv")  
q3\_2017 = read.csv ("C:/Users/KIIT/OneDrive/Desktop/Happiness Data/2017.csv")  
q4\_2018 = read.csv ("C:/Users/KIIT/OneDrive/Desktop/Happiness Data/2018.csv")  
q5\_2019 = read.csv ("C:/Users/KIIT/OneDrive/Desktop/Happiness Data/2019.csv")

## Looking at the data of the year 2019.

head(q5\_2019)

## Overall.rank Country.or.region Score GDP.per.capita Social.support  
## 1 1 Finland 7.769 1.340 1.587  
## 2 2 Denmark 7.600 1.383 1.573  
## 3 3 Norway 7.554 1.488 1.582  
## 4 4 Iceland 7.494 1.380 1.624  
## 5 5 Netherlands 7.488 1.396 1.522  
## 6 6 Switzerland 7.480 1.452 1.526  
## Healthy.life.expectancy Freedom.to.make.life.choices Generosity  
## 1 0.986 0.596 0.153  
## 2 0.996 0.592 0.252  
## 3 1.028 0.603 0.271  
## 4 1.026 0.591 0.354  
## 5 0.999 0.557 0.322  
## 6 1.052 0.572 0.263  
## Perceptions.of.corruption  
## 1 0.393  
## 2 0.410  
## 3 0.341  
## 4 0.118  
## 5 0.298  
## 6 0.343

## Renaming columns:

Renaming few columns of year 2018, 2019 according to the columns of year 2015, 2016 and 2017.

q4\_2018=plyr::rename(q4\_2018, replace = c( "Country.or.region"="Country",   
 "Overall.rank"="Happiness.Rank" ,  
 "GDP.per.capita"="Economy..GDP.per.Capita.",  
 "Healthy.life.expectancy"="Health..Life.Expectancy.",  
 "Freedom.to.make.life.choices"="Freedom",  
 "Perceptions.of.corruption"="Trust..Government.Corruption.",  
 "Social.support"="Family",  
 "Score"="Happiness.Score"))

q5\_2019=plyr::rename(q5\_2019, replace = c( "Country.or.region"="Country",   
 "Overall.rank"="Happiness.Rank" ,  
 "GDP.per.capita"="Economy..GDP.per.Capita.",  
 "Healthy.life.expectancy"="Health..Life.Expectancy.",  
 "Freedom.to.make.life.choices"="Freedom",  
 "Perceptions.of.corruption"="Trust..Government.Corruption.",  
 "Social.support"="Family",  
 "Score"="Happiness.Score"))

Viewing the column names of all the years.

colnames(q1\_2015)

## [1] "Country" "Region"   
## [3] "Happiness.Rank" "Happiness.Score"   
## [5] "Standard.Error" "Economy..GDP.per.Capita."   
## [7] "Family" "Health..Life.Expectancy."   
## [9] "Freedom" "Trust..Government.Corruption."  
## [11] "Generosity" "Dystopia.Residual"

colnames(q2\_2016)

## [1] "Country" "Region"   
## [3] "Happiness.Rank" "Happiness.Score"   
## [5] "Lower.Confidence.Interval" "Upper.Confidence.Interval"   
## [7] "Economy..GDP.per.Capita." "Family"   
## [9] "Health..Life.Expectancy." "Freedom"   
## [11] "Trust..Government.Corruption." "Generosity"   
## [13] "Dystopia.Residual"

colnames(q3\_2017)

## [1] "Country" "Happiness.Rank"   
## [3] "Happiness.Score" "Whisker.high"   
## [5] "Whisker.low" "Economy..GDP.per.Capita."   
## [7] "Family" "Health..Life.Expectancy."   
## [9] "Freedom" "Generosity"   
## [11] "Trust..Government.Corruption." "Dystopia.Residual"

colnames(q4\_2018)

## [1] "Happiness.Rank" "Country"   
## [3] "Happiness.Score" "Economy..GDP.per.Capita."   
## [5] "Family" "Health..Life.Expectancy."   
## [7] "Freedom" "Generosity"   
## [9] "Trust..Government.Corruption."

colnames(q5\_2019)

## [1] "Happiness.Rank" "Country"   
## [3] "Happiness.Score" "Economy..GDP.per.Capita."   
## [5] "Family" "Health..Life.Expectancy."   
## [7] "Freedom" "Generosity"   
## [9] "Trust..Government.Corruption."

## Inspect the dataframes and look for inconguencies.

str(q1\_2015)

## 'data.frame': 158 obs. of 12 variables:  
## $ Country : chr "Switzerland" "Iceland" "Denmark" "Norway" ...  
## $ Region : chr "Western Europe" "Western Europe" "Western Europe" "Western Europe" ...  
## $ Happiness.Rank : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ Happiness.Score : num 7.59 7.56 7.53 7.52 7.43 ...  
## $ Standard.Error : num 0.0341 0.0488 0.0333 0.0388 0.0355 ...  
## $ Economy..GDP.per.Capita. : num 1.4 1.3 1.33 1.46 1.33 ...  
## $ Family : num 1.35 1.4 1.36 1.33 1.32 ...  
## $ Health..Life.Expectancy. : num 0.941 0.948 0.875 0.885 0.906 ...  
## $ Freedom : num 0.666 0.629 0.649 0.67 0.633 ...  
## $ Trust..Government.Corruption.: num 0.42 0.141 0.484 0.365 0.33 ...  
## $ Generosity : num 0.297 0.436 0.341 0.347 0.458 ...  
## $ Dystopia.Residual : num 2.52 2.7 2.49 2.47 2.45 ...

str(q2\_2016)

## 'data.frame': 157 obs. of 13 variables:  
## $ Country : chr "Denmark" "Switzerland" "Iceland" "Norway" ...  
## $ Region : chr "Western Europe" "Western Europe" "Western Europe" "Western Europe" ...  
## $ Happiness.Rank : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ Happiness.Score : num 7.53 7.51 7.5 7.5 7.41 ...  
## $ Lower.Confidence.Interval : num 7.46 7.43 7.33 7.42 7.35 ...  
## $ Upper.Confidence.Interval : num 7.59 7.59 7.67 7.58 7.47 ...  
## $ Economy..GDP.per.Capita. : num 1.44 1.53 1.43 1.58 1.41 ...  
## $ Family : num 1.16 1.15 1.18 1.13 1.13 ...  
## $ Health..Life.Expectancy. : num 0.795 0.863 0.867 0.796 0.811 ...  
## $ Freedom : num 0.579 0.586 0.566 0.596 0.571 ...  
## $ Trust..Government.Corruption.: num 0.445 0.412 0.15 0.358 0.41 ...  
## $ Generosity : num 0.362 0.281 0.477 0.379 0.255 ...  
## $ Dystopia.Residual : num 2.74 2.69 2.83 2.66 2.83 ...

str(q3\_2017)

## 'data.frame': 155 obs. of 12 variables:  
## $ Country : chr "Norway" "Denmark" "Iceland" "Switzerland" ...  
## $ Happiness.Rank : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ Happiness.Score : num 7.54 7.52 7.5 7.49 7.47 ...  
## $ Whisker.high : num 7.59 7.58 7.62 7.56 7.53 ...  
## $ Whisker.low : num 7.48 7.46 7.39 7.43 7.41 ...  
## $ Economy..GDP.per.Capita. : num 1.62 1.48 1.48 1.56 1.44 ...  
## $ Family : num 1.53 1.55 1.61 1.52 1.54 ...  
## $ Health..Life.Expectancy. : num 0.797 0.793 0.834 0.858 0.809 ...  
## $ Freedom : num 0.635 0.626 0.627 0.62 0.618 ...  
## $ Generosity : num 0.362 0.355 0.476 0.291 0.245 ...  
## $ Trust..Government.Corruption.: num 0.316 0.401 0.154 0.367 0.383 ...  
## $ Dystopia.Residual : num 2.28 2.31 2.32 2.28 2.43 ...

str(q4\_2018)

## 'data.frame': 156 obs. of 9 variables:  
## $ Happiness.Rank : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ Country : chr "Finland" "Norway" "Denmark" "Iceland" ...  
## $ Happiness.Score : num 7.63 7.59 7.55 7.5 7.49 ...  
## $ Economy..GDP.per.Capita. : num 1.3 1.46 1.35 1.34 1.42 ...  
## $ Family : num 1.59 1.58 1.59 1.64 1.55 ...  
## $ Health..Life.Expectancy. : num 0.874 0.861 0.868 0.914 0.927 0.878 0.896 0.876 0.913 0.91 ...  
## $ Freedom : num 0.681 0.686 0.683 0.677 0.66 0.638 0.653 0.669 0.659 0.647 ...  
## $ Generosity : num 0.202 0.286 0.284 0.353 0.256 0.333 0.321 0.365 0.285 0.361 ...  
## $ Trust..Government.Corruption.: chr "0.393" "0.340" "0.408" "0.138" ...

str(q5\_2019)

## 'data.frame': 156 obs. of 9 variables:  
## $ Happiness.Rank : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ Country : chr "Finland" "Denmark" "Norway" "Iceland" ...  
## $ Happiness.Score : num 7.77 7.6 7.55 7.49 7.49 ...  
## $ Economy..GDP.per.Capita. : num 1.34 1.38 1.49 1.38 1.4 ...  
## $ Family : num 1.59 1.57 1.58 1.62 1.52 ...  
## $ Health..Life.Expectancy. : num 0.986 0.996 1.028 1.026 0.999 ...  
## $ Freedom : num 0.596 0.592 0.603 0.591 0.557 0.572 0.574 0.585 0.584 0.532 ...  
## $ Generosity : num 0.153 0.252 0.271 0.354 0.322 0.263 0.267 0.33 0.285 0.244 ...  
## $ Trust..Government.Corruption.: num 0.393 0.41 0.341 0.118 0.298 0.343 0.373 0.38 0.308 0.226 ...

## Inserting ‘Year’ column at 0th index.

q1\_2015<-cbind(Year=2015,q1\_2015)  
q2\_2016<-cbind(Year=2016,q2\_2016)  
q3\_2017<-cbind(Year=2017,q3\_2017)  
q4\_2018<-cbind(Year=2018,q4\_2018)  
q5\_2019<-cbind(Year=2019,q5\_2019)

## Changing datatype of Trust..Government.Corruption. to numeric and inspecting it.

q4\_2018$Trust..Government.Corruption. = as.numeric(q4\_2018$Trust..Government.Corruption.)

## Warning: NAs introduced by coercion

str(q4\_2018)

## 'data.frame': 156 obs. of 10 variables:  
## $ Year : num 2018 2018 2018 2018 2018 ...  
## $ Happiness.Rank : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ Country : chr "Finland" "Norway" "Denmark" "Iceland" ...  
## $ Happiness.Score : num 7.63 7.59 7.55 7.5 7.49 ...  
## $ Economy..GDP.per.Capita. : num 1.3 1.46 1.35 1.34 1.42 ...  
## $ Family : num 1.59 1.58 1.59 1.64 1.55 ...  
## $ Health..Life.Expectancy. : num 0.874 0.861 0.868 0.914 0.927 0.878 0.896 0.876 0.913 0.91 ...  
## $ Freedom : num 0.681 0.686 0.683 0.677 0.66 0.638 0.653 0.669 0.659 0.647 ...  
## $ Generosity : num 0.202 0.286 0.284 0.353 0.256 0.333 0.321 0.365 0.285 0.361 ...  
## $ Trust..Government.Corruption.: num 0.393 0.34 0.408 0.138 0.357 0.295 0.291 0.389 0.383 0.302 ...

## Binding all the dataframes from the year 2015-2019 into a single dataframe for further analysis.

q15\_16<-dplyr::bind\_rows(q1\_2015,q2\_2016)  
q15\_16\_17<-dplyr::bind\_rows(q15\_16,q3\_2017)  
q18\_19<-dplyr::bind\_rows(q4\_2018,q5\_2019)  
q<-dplyr::bind\_rows(q18\_19,q15\_16\_17)

Viewing the new dataframe ‘q’.

head(q)

## Year Happiness.Rank Country Happiness.Score Economy..GDP.per.Capita.  
## 1 2018 1 Finland 7.632 1.305  
## 2 2018 2 Norway 7.594 1.456  
## 3 2018 3 Denmark 7.555 1.351  
## 4 2018 4 Iceland 7.495 1.343  
## 5 2018 5 Switzerland 7.487 1.420  
## 6 2018 6 Netherlands 7.441 1.361  
## Family Health..Life.Expectancy. Freedom Generosity  
## 1 1.592 0.874 0.681 0.202  
## 2 1.582 0.861 0.686 0.286  
## 3 1.590 0.868 0.683 0.284  
## 4 1.644 0.914 0.677 0.353  
## 5 1.549 0.927 0.660 0.256  
## 6 1.488 0.878 0.638 0.333  
## Trust..Government.Corruption. Region Standard.Error Dystopia.Residual  
## 1 0.393 <NA> NA NA  
## 2 0.340 <NA> NA NA  
## 3 0.408 <NA> NA NA  
## 4 0.138 <NA> NA NA  
## 5 0.357 <NA> NA NA  
## 6 0.295 <NA> NA NA  
## Lower.Confidence.Interval Upper.Confidence.Interval Whisker.high Whisker.low  
## 1 NA NA NA NA  
## 2 NA NA NA NA  
## 3 NA NA NA NA  
## 4 NA NA NA NA  
## 5 NA NA NA NA  
## 6 NA NA NA NA

## Changing datatype of Happiness.Rank into numeric and then inspecting the dataframe.

q$Happiness.Rank = as.numeric(q$Happiness.Rank )  
  
str(q)

## 'data.frame': 782 obs. of 17 variables:  
## $ Year : num 2018 2018 2018 2018 2018 ...  
## $ Happiness.Rank : num 1 2 3 4 5 6 7 8 9 10 ...  
## $ Country : chr "Finland" "Norway" "Denmark" "Iceland" ...  
## $ Happiness.Score : num 7.63 7.59 7.55 7.5 7.49 ...  
## $ Economy..GDP.per.Capita. : num 1.3 1.46 1.35 1.34 1.42 ...  
## $ Family : num 1.59 1.58 1.59 1.64 1.55 ...  
## $ Health..Life.Expectancy. : num 0.874 0.861 0.868 0.914 0.927 0.878 0.896 0.876 0.913 0.91 ...  
## $ Freedom : num 0.681 0.686 0.683 0.677 0.66 0.638 0.653 0.669 0.659 0.647 ...  
## $ Generosity : num 0.202 0.286 0.284 0.353 0.256 0.333 0.321 0.365 0.285 0.361 ...  
## $ Trust..Government.Corruption.: num 0.393 0.34 0.408 0.138 0.357 0.295 0.291 0.389 0.383 0.302 ...  
## $ Region : chr NA NA NA NA ...  
## $ Standard.Error : num NA NA NA NA NA NA NA NA NA NA ...  
## $ Dystopia.Residual : num NA NA NA NA NA NA NA NA NA NA ...  
## $ Lower.Confidence.Interval : num NA NA NA NA NA NA NA NA NA NA ...  
## $ Upper.Confidence.Interval : num NA NA NA NA NA NA NA NA NA NA ...  
## $ Whisker.high : num NA NA NA NA NA NA NA NA NA NA ...  
## $ Whisker.low : num NA NA NA NA NA NA NA NA NA NA ...

## Counting the number of NULL values in all the columns of q dataframe.

colSums(is.na(q))

## Year Happiness.Rank   
## 0 0   
## Country Happiness.Score   
## 0 0   
## Economy..GDP.per.Capita. Family   
## 0 0   
## Health..Life.Expectancy. Freedom   
## 0 0   
## Generosity Trust..Government.Corruption.   
## 0 1   
## Region Standard.Error   
## 467 624   
## Dystopia.Residual Lower.Confidence.Interval   
## 312 625   
## Upper.Confidence.Interval Whisker.high   
## 625 627   
## Whisker.low   
## 627

## Removing the unnecessary columns.

q = subset(q, select = -c(Lower.Confidence.Interval,Upper.Confidence.Interval,Dystopia.Residual,Standard.Error,Whisker.high,Whisker.low))

Again counting the number of NULL values in columns after removing the unncessary ones.

colSums(is.na(q))

## Year Happiness.Rank   
## 0 0   
## Country Happiness.Score   
## 0 0   
## Economy..GDP.per.Capita. Family   
## 0 0   
## Health..Life.Expectancy. Freedom   
## 0 0   
## Generosity Trust..Government.Corruption.   
## 0 1   
## Region   
## 467

## Impute with mean or median values for numerical columns and then checking for the NULL values of all the columns.

q$Trust..Government.Corruption.[is.na(q$Trust..Government.Corruption.)] <- median(q$Trust..Government.Corruption., na.rm = T)  
  
colSums(is.na(q))

## Year Happiness.Rank   
## 0 0   
## Country Happiness.Score   
## 0 0   
## Economy..GDP.per.Capita. Family   
## 0 0   
## Health..Life.Expectancy. Freedom   
## 0 0   
## Generosity Trust..Government.Corruption.   
## 0 0   
## Region   
## 467

## Counting number of countries of every years.

aggregate(q$Country, by=list(q$Year), FUN=length)

## Group.1 x  
## 1 2015 158  
## 2 2016 157  
## 3 2017 155  
## 4 2018 156  
## 5 2019 156

From the table shown as above, the number of countries involved in this dataset for different year is different. Therefore, it is necessary to make an intersection of them to get the most common country list.

Country\_2015 = subset(q, Year == 2015)$Country  
Country\_2016 = subset(q, Year == 2016)$Country  
Country\_2017 = subset(q, Year == 2017)$Country  
Country\_2018 = subset(q, Year == 2018)$Country  
Country\_2019 = subset(q, Year == 2019)$Country  
  
common\_country =intersect(intersect(intersect(intersect(Country\_2015,Country\_2016),Country\_2017),Country\_2018),Country\_2019)  
length(common\_country)

## [1] 141

Therefore, there are 141 countries’ data existing across from 2015-2019 in this dataset.Then we need to filter the original dataset by this common\_country list.

q1 = subset(q,Country %in% common\_country)  
print(paste("The amount of rows in the dataset is: ",dim(q1)[1]))

## [1] "The amount of rows in the dataset is: 705"

print(paste("The amount of columns in the dataset is: ",dim(q1)[2]))

## [1] "The amount of columns in the dataset is: 11"

## Creating a new dataset for storing common region and country.

common\_region <- unique(subset(q1, Region!="NA", c(Country, Region)))  
  
head(common\_country)

## [1] "Switzerland" "Iceland" "Denmark" "Norway" "Canada"   
## [6] "Finland"

## Filling relate region to missing value of region column.

assign\_region <- function(x){  
 Region <- common\_region$Region[common\_region$Country == x]  
}  
  
for(country in common\_country)  
 q1$Region[q1$Country == country] <- assign\_region(country)

## Saving the clean Dataset for further Data Visualization in Tableau.

library(readr)  
write\_csv(q1, path = "World Happiness Data 2015\_\_2019\_cleaned.csv")

## Warning: The `path` argument of `write\_csv()` is deprecated as of readr 1.4.0.  
## ℹ Please use the `file` argument instead.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_lifecycle\_warnings()` to see where this warning was  
## generated.

## Viewing the statistics of the dataset.

skimr::skim\_without\_charts(q1)

Data summary

|  |  |
| --- | --- |
| Name | q1 |
| Number of rows | 705 |
| Number of columns | 11 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| character | 2 |
| numeric | 9 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | None |

**Variable type: character**

| skim\_variable | n\_missing | complete\_rate | min | max | empty | n\_unique | whitespace |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Country | 0 | 1 | 4 | 23 | 0 | 141 | 0 |
| Region | 0 | 1 | 12 | 31 | 0 | 10 | 0 |

**Variable type: numeric**

| skim\_variable | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Year | 0 | 1 | 2017.00 | 1.42 | 2015.00 | 2016.00 | 2017.00 | 2018.00 | 2019.00 |
| Happiness.Rank | 0 | 1 | 76.85 | 45.28 | 1.00 | 37.00 | 77.00 | 116.00 | 158.00 |
| Happiness.Score | 0 | 1 | 5.43 | 1.13 | 2.84 | 4.52 | 5.39 | 6.29 | 7.77 |
| Economy..GDP.per.Capita. | 0 | 1 | 0.93 | 0.40 | 0.00 | 0.64 | 1.00 | 1.24 | 2.10 |
| Family | 0 | 1 | 1.09 | 0.32 | 0.00 | 0.88 | 1.14 | 1.35 | 1.64 |
| Health..Life.Expectancy. | 0 | 1 | 0.63 | 0.23 | 0.00 | 0.49 | 0.66 | 0.81 | 1.14 |
| Freedom | 0 | 1 | 0.41 | 0.15 | 0.00 | 0.31 | 0.43 | 0.53 | 0.72 |
| Generosity | 0 | 1 | 0.22 | 0.13 | 0.00 | 0.13 | 0.20 | 0.28 | 0.84 |
| Trust..Government.Corruption. | 0 | 1 | 0.12 | 0.11 | 0.00 | 0.05 | 0.09 | 0.15 | 0.55 |

## Checking the number of rows, columns and all the column names.

print(paste("The amount of rows in the dataset is: ",dim(q)[1]))

## [1] "The amount of rows in the dataset is: 782"

print(paste("The amount of columns in the dataset is: ",dim(q)[2]))

## [1] "The amount of columns in the dataset is: 11"

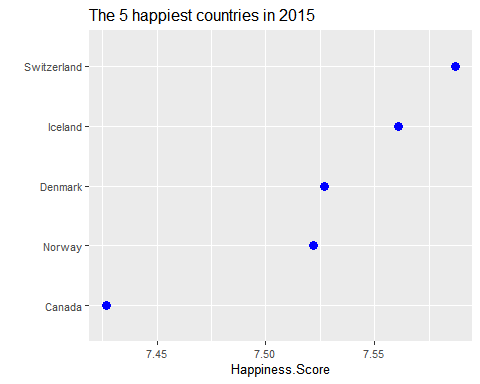
print(paste("the column names in this dataset are:", paste(shQuote(colnames(q)), collapse=", ")))

## [1] "the column names in this dataset are: \"Year\", \"Happiness.Rank\", \"Country\", \"Happiness.Score\", \"Economy..GDP.per.Capita.\", \"Family\", \"Health..Life.Expectancy.\", \"Freedom\", \"Generosity\", \"Trust..Government.Corruption.\", \"Region\""

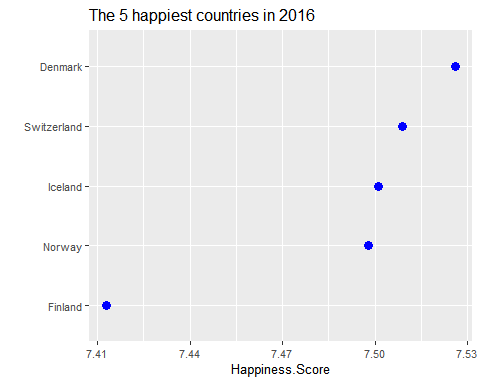
## MOVING ON FURTHER WITH DATA VISUALIZATION:

Checking the top 5 countries of the year 2015-2019 in terms of their happiness score.

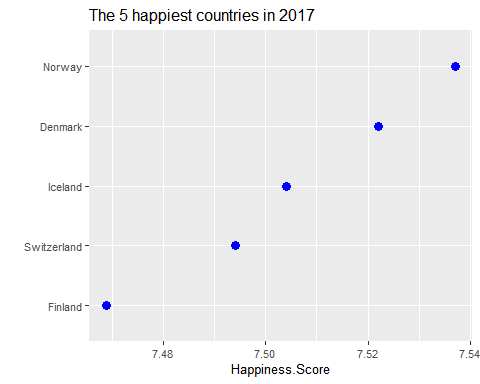
q1 %>%  
 filter(Year == 2015) %>%  
 arrange(-Happiness.Score) %>%  
 slice\_head(n=5) %>%  
 ggplot(aes(reorder(Country, Happiness.Score), Happiness.Score)) +  
 geom\_point(colour = "blue", size = 3) +  
 theme(text=element\_text(size=10)) +   
 coord\_flip() +  
 labs(title = "The 5 happiest countries in 2015", x = "")



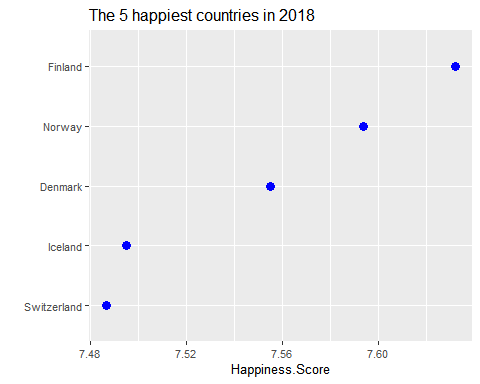
q1 %>%  
 filter(Year == 2016) %>%  
 arrange(-Happiness.Score) %>%  
 slice\_head(n=5) %>%  
 ggplot(aes(reorder(Country, Happiness.Score), Happiness.Score)) +  
 geom\_point(colour = "blue", size = 3) +  
 theme(text=element\_text(size=10)) +   
 coord\_flip() +  
 labs(title = "The 5 happiest countries in 2016", x = "")



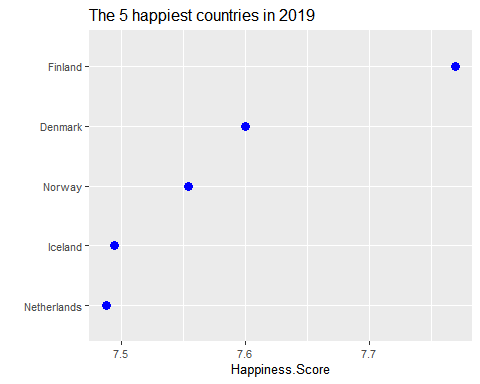
q1 %>%  
 filter(Year == 2017) %>%  
 arrange(-Happiness.Score) %>%  
 slice\_head(n=5) %>%  
 ggplot(aes(reorder(Country, Happiness.Score), Happiness.Score)) +  
 geom\_point(colour = "blue", size = 3) +  
 theme(text=element\_text(size=10)) +   
 coord\_flip() +  
 labs(title = "The 5 happiest countries in 2017", x = "")



q1 %>%  
 filter(Year == 2018) %>%  
 arrange(-Happiness.Score) %>%  
 slice\_head(n=5) %>%  
 ggplot(aes(reorder(Country, Happiness.Score), Happiness.Score)) +  
 geom\_point(colour = "blue", size = 3) +  
 theme(text=element\_text(size=10)) +   
 coord\_flip() +  
 labs(title = "The 5 happiest countries in 2018", x = "")

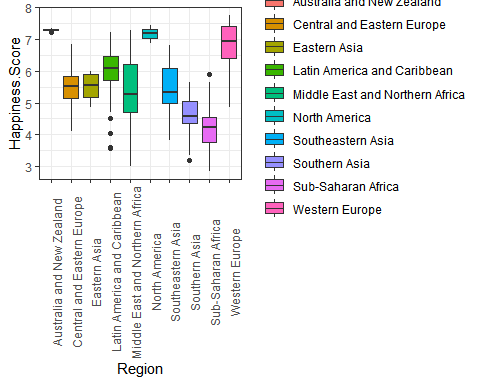


q1 %>%  
 filter(Year == 2019) %>%  
 arrange(-Happiness.Score) %>%  
 slice\_head(n=5) %>%  
 ggplot(aes(reorder(Country, Happiness.Score), Happiness.Score)) +  
 geom\_point(colour = "blue", size = 3) +  
 theme(text=element\_text(size=10)) +   
 coord\_flip() +  
 labs(title = "The 5 happiest countries in 2019", x = "")



## Checking Mean Happiness Score of all the Regions.

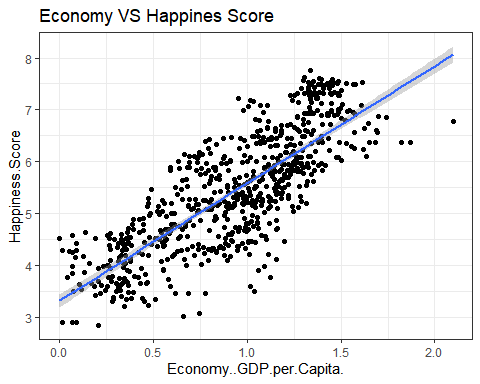
ggplot(q1 , aes(x = Region, y = Happiness.Score)) +  
 geom\_boxplot(aes(fill=Region)) + theme\_bw() +  
 theme(axis.text.x = element\_text (angle = 90))



## Using Scatterplot to find the relationship between Happiness Score and Economy.

ggplot(q1, aes(x = Economy..GDP.per.Capita., y = Happiness.Score)) +   
 geom\_point() +   
 geom\_smooth(method = "lm", fullrange = TRUE) +  
 theme\_bw() + labs(title = "Economy VS Happines Score")

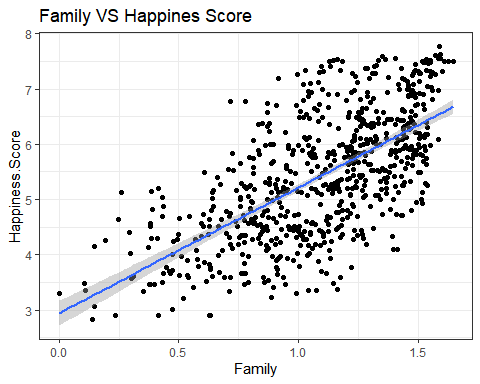
## `geom\_smooth()` using formula = 'y ~ x'



## Using Scatterplot to find the relationship between Happiness Score and Family.

ggplot(q1, aes(x = Family, y = Happiness.Score)) +   
 geom\_point() +   
 geom\_smooth(method = "lm", fullrange = TRUE) +  
 theme\_bw() + labs(title = "Family VS Happines Score")

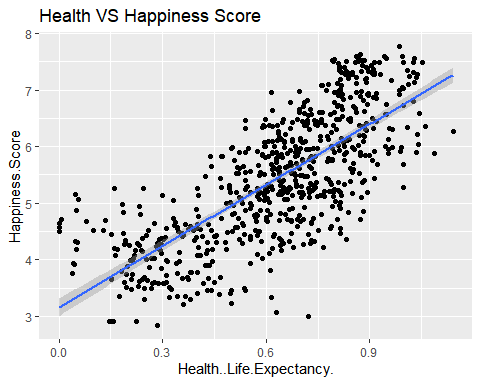
## `geom\_smooth()` using formula = 'y ~ x'



## Using Scatterplot to find the relationship between Happiness Score and Health.

ggplot(q1, aes(x = Health..Life.Expectancy., y = Happiness.Score)) +   
 geom\_point() +   
 geom\_smooth(method = "lm") +  
 labs(title = "Health VS Happiness Score")

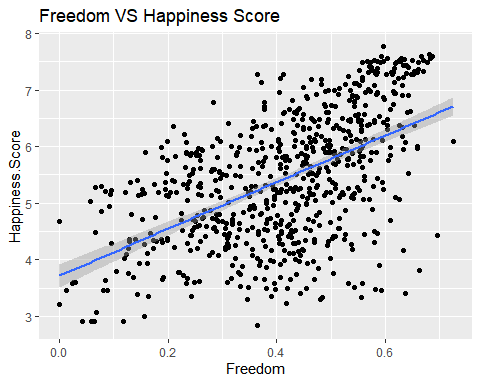
## `geom\_smooth()` using formula = 'y ~ x'



## Using Scatterplot to find the relationship between Happiness Score and Freedom.

ggplot(q1, aes(x = Freedom, y = Happiness.Score)) +   
 geom\_point() +   
 geom\_smooth(method = "lm") +  
 labs(title = "Freedom VS Happiness Score")

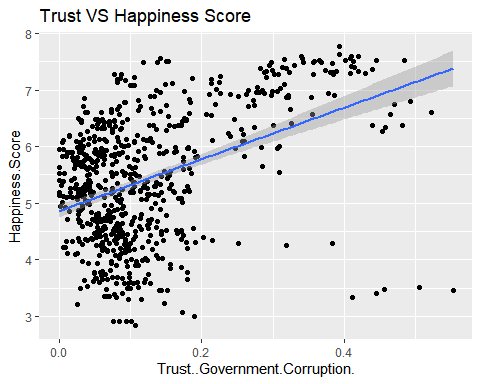
## `geom\_smooth()` using formula = 'y ~ x'



## Using Scatterplot to find the relationship between Happiness Score and Trust.

ggplot(q1, aes(x = Trust..Government.Corruption., y = Happiness.Score)) +   
 geom\_point() +   
 geom\_smooth(method = "lm") +  
 labs(title = "Trust VS Happiness Score")

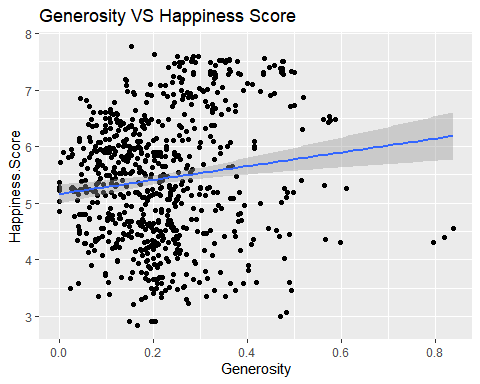
## `geom\_smooth()` using formula = 'y ~ x'



## Using Scatterplot to find the relationship between Happiness Score and Generosity.

ggplot(q1, aes(x = Generosity, y = Happiness.Score)) +   
 geom\_point() +   
 geom\_smooth(method = "lm") +  
 labs(title = "Generosity VS Happiness Score")

## `geom\_smooth()` using formula = 'y ~ x'



## Droping columns based on Heatmap Correlation.

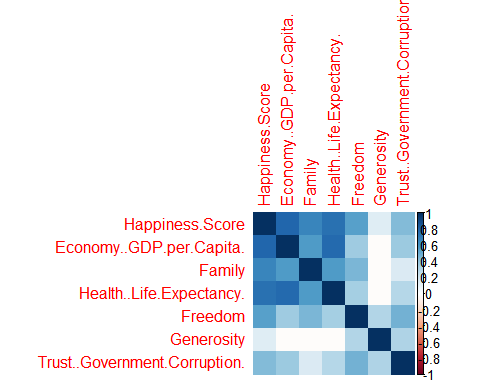
We are removing Year, Country, Happiness.Rank, Region column before making heatmap and storing it in a new dataframe called ‘df’.

df = select(q1,-c("Year","Country","Happiness.Rank","Region"))  
  
head(df)

## Happiness.Score Economy..GDP.per.Capita. Family Health..Life.Expectancy.  
## 1 7.632 1.305 1.592 0.874  
## 2 7.594 1.456 1.582 0.861  
## 3 7.555 1.351 1.590 0.868  
## 4 7.495 1.343 1.644 0.914  
## 5 7.487 1.420 1.549 0.927  
## 6 7.441 1.361 1.488 0.878  
## Freedom Generosity Trust..Government.Corruption.  
## 1 0.681 0.202 0.393  
## 2 0.686 0.286 0.340  
## 3 0.683 0.284 0.408  
## 4 0.677 0.353 0.138  
## 5 0.660 0.256 0.357  
## 6 0.638 0.333 0.295

## Computing the heatmap.

Num.cols <- sapply(df, is.numeric)  
Cor.data <- cor(df[, Num.cols])  
  
corrplot(Cor.data, method = 'color')



## Computing Correlation heatmap.

ggcorr(df, label = TRUE, label\_round = 2, label\_size = 3.5, size = 2, hjust = .85) + ggtitle("Correlation Heatmap") + theme(plot.title = element\_text(hjust = 0.5))

