Data Analysis

2022-04-16

```
library(knitr)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'

## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```
library(dplyr)
library(vtable)
library(qwraps2)
library(modest)
library(ggplot2)
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
theme_set(theme_classic())
```

```r
data <- read.csv("Gdphealth.csv", header = TRUE, stringsAsFactors = TRUE)
head(data)
```

```
##       Country    GDP GDP.pc Growth Health.exp
## 1     Ireland  3,564 25,200    9.0       6.90
## 2 New Zealand  7,449 18,000    1.6       7.12
## 3      Poland 28,888  8,100    5.3       5.80
## 4  Luxembourg 37,247 39,300    5.1      13.00
## 5 Switzerland 38,044 27,500    1.4      10.90
## 6     Iceland 41,651 27,300    5.5       7.15
```

Remove comma from GDP and GDP.pc variables and convert them to numeric

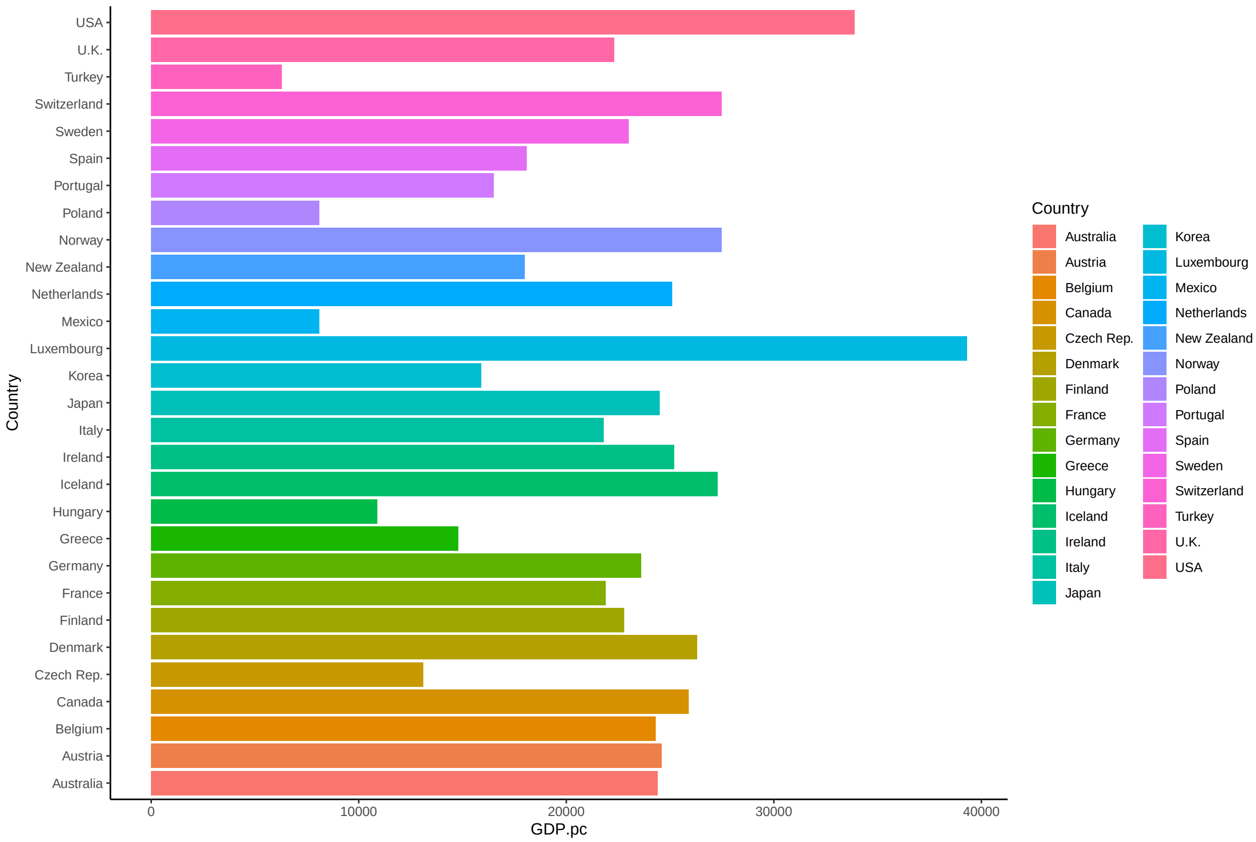
```r
data <- data %>%
    mutate_each(funs(as.character(.)), GDP:GDP.pc) %>%
    mutate_each(funs(gsub(",", "", .)), GDP:GDP.pc) %>%
    mutate_each(funs(as.numeric(.)), GDP:GDP.pc)
```

```
## Warning: `funs()` was deprecated in dplyr 0.8.0.
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with `tibble::lst()`:
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.

## Warning: `mutate_each_()` was deprecated in dplyr 0.7.0.
## Please use `across()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```

```r
head(data,3)
```

```
##       Country   GDP GDP.pc Growth Health.exp
## 1     Ireland  3564  25200    9.0       6.90
## 2 New Zealand  7449  18000    1.6       7.12
## 3      Poland 28888   8100    5.3       5.80
```

```r
p <- ggplot(data, aes(x=Country, y=GDP.pc, fill = Country))+
  geom_bar(stat="identity")
p + coord_flip()
```



GDP are in local currencies, so convert to usd.

```
# Add currecy conversion rate column
To_usd <- c(1.08,0.68,0.23,1.08,1.06,0.0077,1.08,0.74,1.31,1.08,
0.79,0.11,0.15,0.044,0.1,0.05,1.08,1.08,0.0029,1.08,
1.08,1,1.08,1.08,1.08,0.00081,0.0079,1.08,1.068)

#Multiply rates by GDP.pc and view 2 rows to confirm the operation
data['GDp'] <- To_usd*data$GDP.pc
head(data,2)
```

```
##       Country  GDP GDP.pc Growth Health.exp   GDp
## 1     Ireland 3564  25200    9.0       6.90 27216
## 2 New Zealand 7449  18000    1.6       7.12 12240
```

Select GDp, Growth and Health expenditures

```
mydf <- select(data, -c(2,3))
head(mydf,5)
```

```
##       Country Growth Health.exp   GDp
## 1     Ireland    9.0       6.90 27216
## 2 New Zealand    1.6       7.12 12240
## 3      Poland    5.3       5.80  1863
## 4  Luxembourg    5.1      13.00 42444
## 5 Switzerland    1.4      10.90 29150
```

Table 1: Mean and Median

Growth	Health.exp	GDp	MCT
3.275862	8.448621	15950.04	Mean
3.200000	7.800000	18056.00	Median

Descriptive Statistics: Measures of location.

Mean

```
Mean <- mydf[2:4] %>% summarise_all(list(mean))
Median <- mydf[2:4] %>% summarise_all(list(median))
M <- rbind(Mean, Median)
MM <- cbind(M, MCT=c("Mean", "Median"))
knitr::kable(MM, caption = "Mean and Median")
```

mode

```
x <- table(mydf$Growth)
print("Mode of Growth is")
```

```
## [1] "Mode of Growth is"
```

```
names(x)[which(x==max(x))]
```

```
## [1] "2.5" "3.4"
```

```
y <- table(mydf$Health.exp)
print("Mode of Health.exp is")
```

```
## [1] "Mode of Health.exp is"
```

```
names(y) [which(y==max(y))]
```

```
## [1] "7.12"
```

```
z <- table(mydf$GDp)
print("Mode of GDP is")
```

```
## [1] "Mode of GDP is"
```

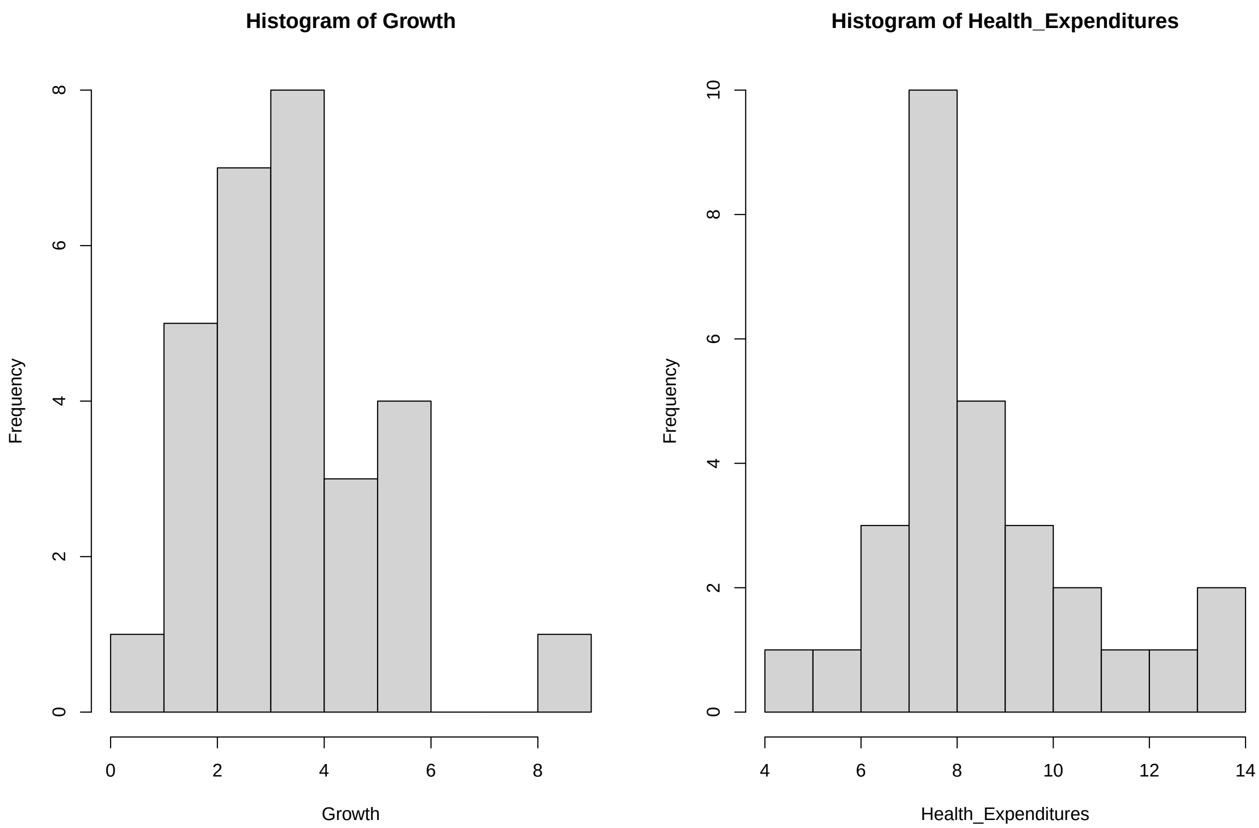
```
names(z) [which(z==max(z), TRUE)]
```

```
##  [1] "12.879" "31.61"  "193.55" "210.21" "405"    "576.4"  "1863"   "2300"
##  [9] "3025"   "3945"   "6728.4" "12240"  "15984"  "17820"  "18056"  "19548"
## [17] "20461"  "23544"  "23652"  "24624"  "25488"  "26244"  "26568"  "27108"
## [25] "27216"  "29150"  "29213"  "33900"  "42444"
```

There is no Mode

Use histogram to check the distributions of Growth and Health expenditures

```
Growth=mydf$Growth
Health_Expenditures=mydf$Health.exp

par(mfrow=c(1,2))
hist(Growth, breaks = 10)
hist(Health_Expenditures, breaks = 10)
```

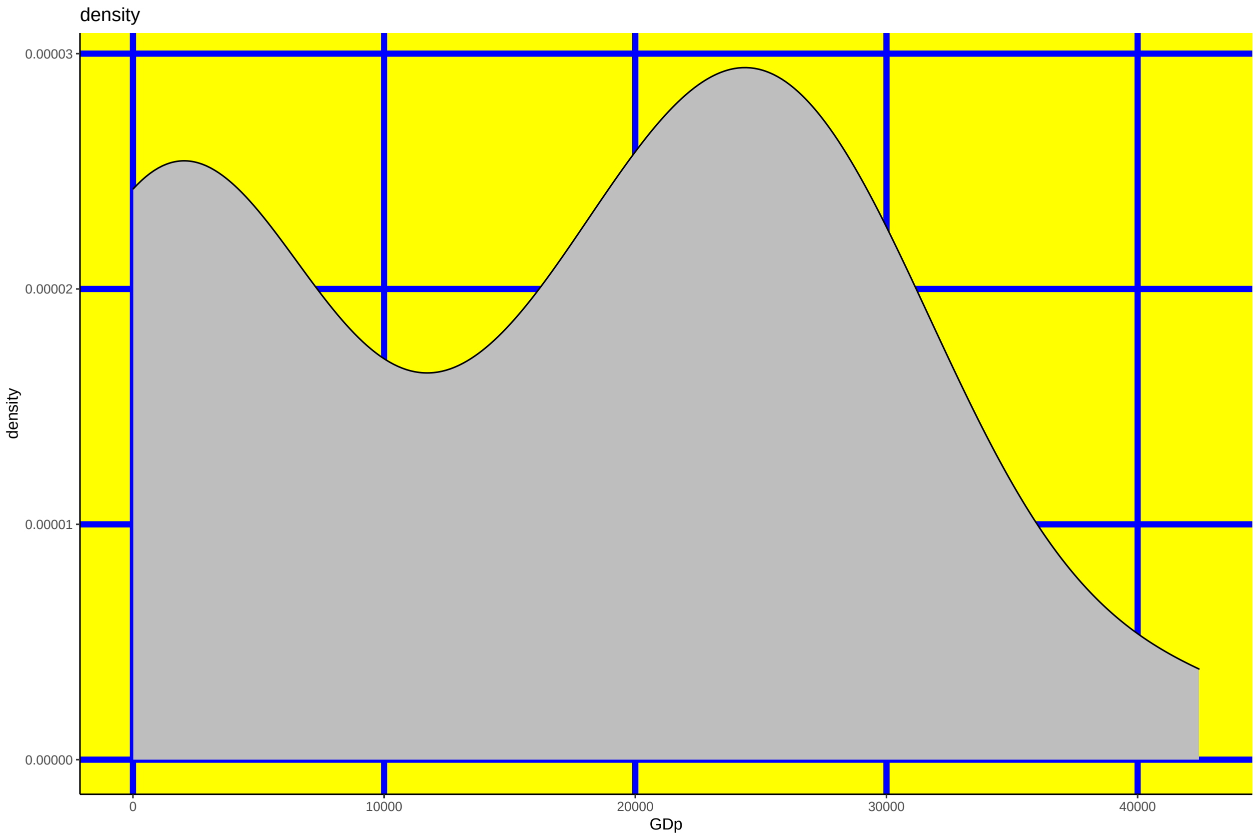


AS can be seen, the distributions appears to right-skewed, and also we have an outlier in the Growth variable with growth index grater than 8. Hence we cannot repport mean as the measure of central tendency because mean is not centrally located, but rather the median is preferred.

```
par(mfrow=c(1,2))
ggplot(mydf, aes(x=GDp)) + geom_density(fill="grey") +
  labs(title="density") +
  theme(panel.background=element_rect(fill="yellow"),
        panel.grid.major=element_line(color="blue", size=2))
```

Table 2: summary statistics

	Growth	Health.exp	GDp
	Min. :0.300	Min. : 4.130	Min. : 12.88
	1st Qu.:2.200	1st Qu.: 7.110	1st Qu.: 2300.00
	Median :3.200	Median : 7.800	Median :18056.00
	Mean :3.276	Mean : 8.449	Mean :15950.04
	3rd Qu.:4.100	3rd Qu.: 9.700	3rd Qu.:26244.00
	Max. :9.000	Max. :14.000	Max. :42444.00



Interestingly, we have two peaks, hence the GDP is bimodal. It is therefore worth digging deeper to find out why this is the case.

Measures of dispersion

The quartieles, Maximum and minimum values can be taken from the summary table below

```
knitr::kable(summary(mydf[2:4]), caption = "summary statistics")
```

Range

```
# Range : maximum - mimimum
knitr::kable(mydf[2:4] %>% summarise_all(list(range)),
             caption = "Range")
```

Table 3: Range

Growth	Health.exp	GDp
0.3	4.13	12.879
9.0	14.00	42444.000

Table 4: Measures of Dispersion

Growth	Health.exp	GDp	Measure
I_Range	2.5900000	23944.0000000	I_Range
Variance	5.4971409	158749112.9031927	Variance
STDeviation	2.3445982	12599.5679649	STDeviation
Skewness	0.8247195	0.0607562	Skewness
Kurtosis	3.1397007	1.8067584	Kurtosis

Interquartile, Variance, Standard deviation, Skewness, and Kurtosis.

```
library(moments);
I_QR <- mydf[2:4] %>% summarise_all(list(IQR))
Var <- mydf[2:4] %>% summarise_all(list(var))
Stdv <- mydf[2:4] %>% summarise_all(list(sd))
Skness <- mydf[2:4] %>% summarise_all(list(skewness))
Kurtsis <- mydf[2:4] %>% summarise_all(list(kurtosis))
A = rbind(I_QR, Var)
B = rbind(A, Stdv)
C = rbind(B, Skness)
D = rbind(C, Kurtsis)
MOD <- cbind(D, Measure = c("I_Range","Variance","STDeviation", "Skewness","Kurtosis"))
MOD[1] = MOD$Measure
knitr::kable(MOD,
             caption = "Measures of Dispersion")
```

Covariance and Correlation

```
# covariance matrix
mydf1 <- mydf[2:4]
knitr::kable(cov(mydf1), caption = "covariance matrix")
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

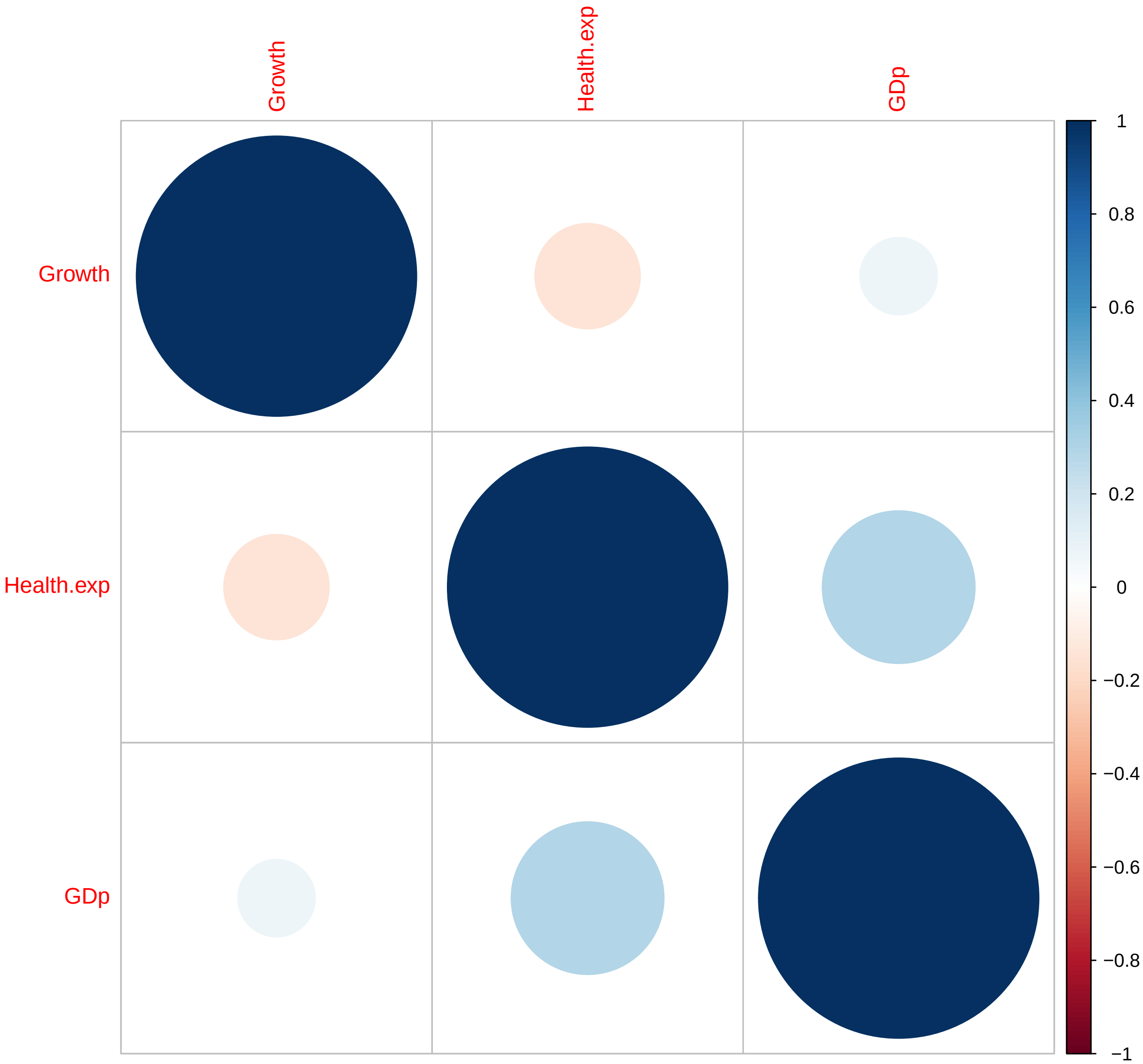
Table 5: covariance matrix

	Growth	Health.exp	GDp
Growth	3.0411823	-0.5767845	1675.148
Health.exp	-0.5767845	5.4971409	8758.778
GDp	1675.1478222	8758.7779999	158749112.903

Table 6: correlation

	Growth	Health.exp	GDp
Growth	1.0000000	-0.1410665	0.0762388
Health.exp	-0.1410665	1.0000000	0.2964964
GDp	0.0762388	0.2964964	1.0000000

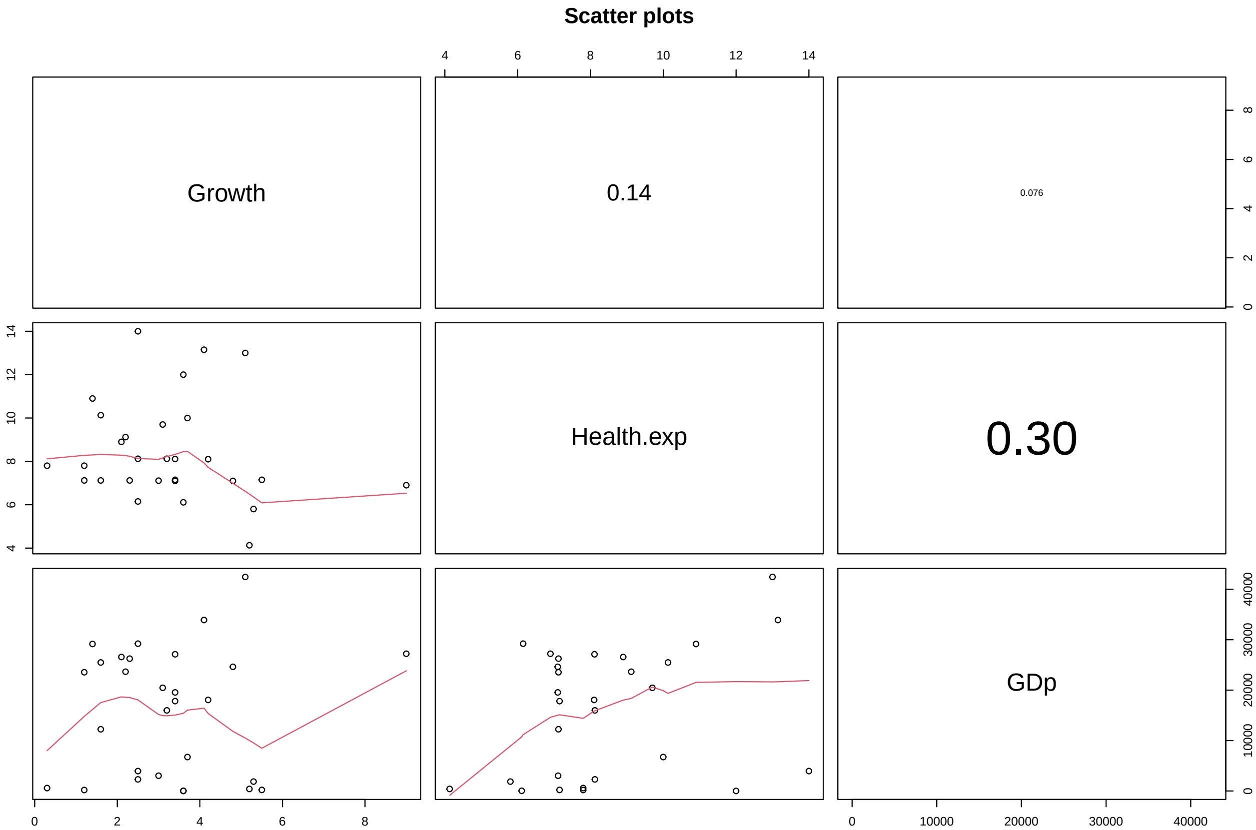
```
corrplot(cor(mydf1))
```




```
knitr::kable(cor(mydf1), caption = "correlation")
```

```
panel.cor <- function(x, y, digits = 2, prefix = "", cex.cor, ...)
      {
          usr <- par("usr"); on.exit(par(usr))
          par(usr = c(0, 1, 0, 1))
          r <- abs(cor(x, y))
```

```
        txt <- format(c(r, 0.123456789), digits = digits)[1]
        txt <- paste0(prefix, txt)
        if(missing(cex.cor)) cex.cor <- 0.8/strwidth(txt)
        text(0.5, 0.5, txt, cex = cex.cor * r)
    }

pairs(mydf1,
      upper.panel = panel.cor,    # Correlation panel
      lower.panel = panel.smooth,
      main="Scatter plots") # Smoothed regression lines
```

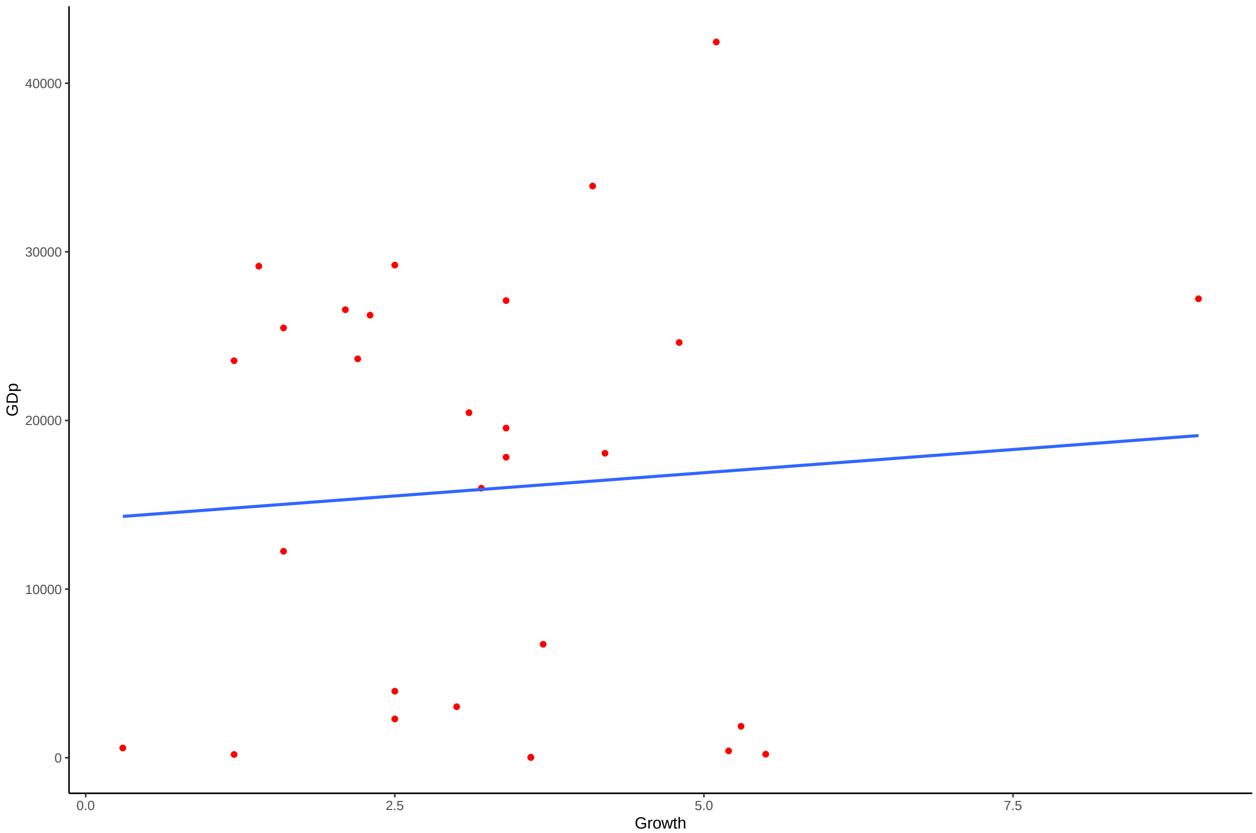


The OLS (Ordinary Least Squares) Method

GDP vs Growth

```
# OLS model
mydf1 %>%
 ggplot(aes(x = Growth, y = GDp)) +
 geom_point(colour = "red") +
 geom_smooth(method = "lm", fill = NA)
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
# Correlation Coefficient
cor(mydf1$Growth, mydf1$GDp)
```

```
## [1] 0.07623884
```

```
ols1 <- lm(GDp ~ Growth, data = mydf1)
# model summary
summary(ols1)
```

```
##
## Call:
## lm(formula = GDp ~ Growth, data = mydf1)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -16965 -13223   1802  10461  25489
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  14145.6     5125.5   2.760   0.0103 *
## Growth         550.8     1386.4   0.397   0.6943
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12790 on 27 degrees of freedom
## Multiple R-squared:  0.005812,   Adjusted R-squared:  -0.03101
## F-statistic: 0.1579 on 1 and 27 DF,  p-value: 0.6943
```

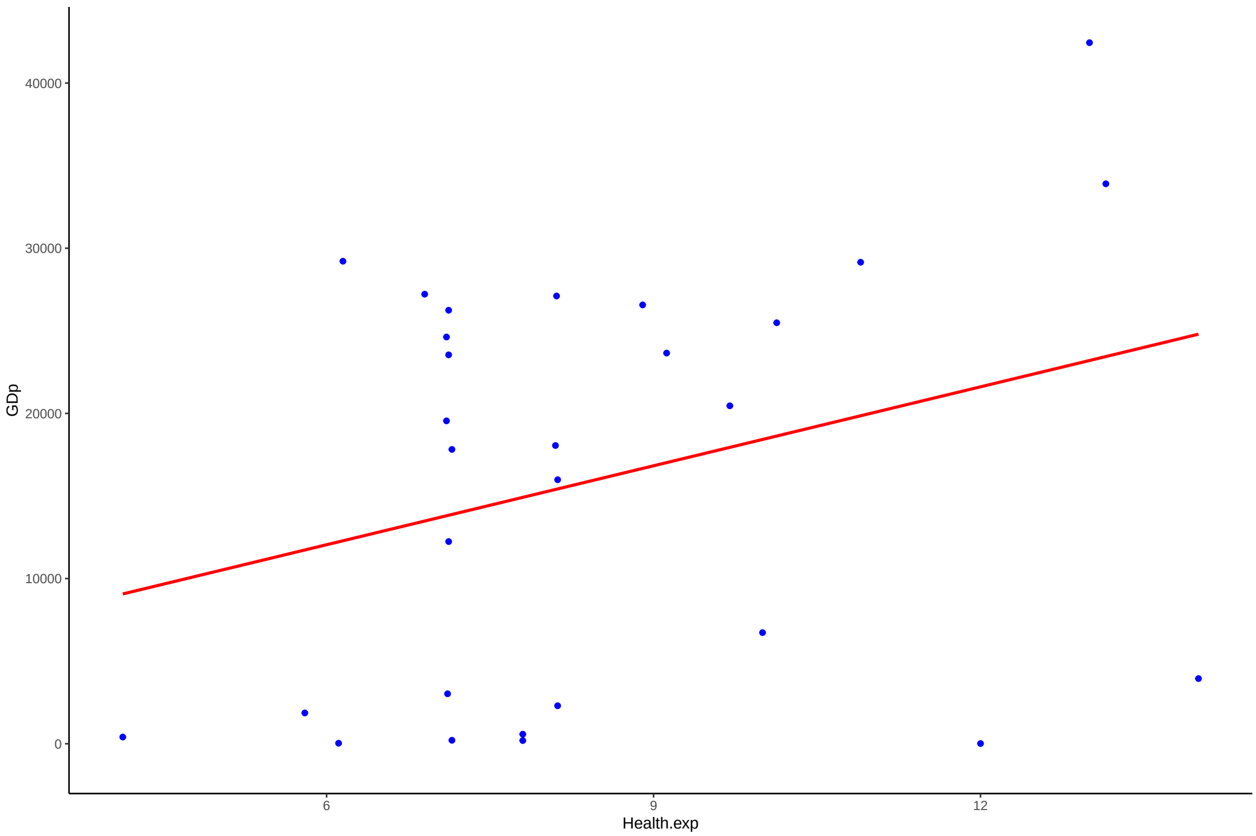
```r
# Confidence Interval
confint(ols1)
```

```
##                  2.5 %    97.5 %
## (Intercept)  3629.052 24662.192
## Growth      -2293.825  3395.467
```

GDP vs Health expenditure

```r
# OLS model2
mydf1 %>%
 ggplot(aes(x = Health.exp, y = GDp)) +
 geom_point(colour = "blue") +
 geom_smooth(method = "lm", fill = NA, colour = "red")
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```r
ols2 <- lm(GDp ~ Health.exp, data = mydf1)
```

```r
# Correlation coefficient
cor(mydf1$Health.exp, mydf1$GDp)
```

```
## [1] 0.2964964
```

$R^2 = 0.3$, which shows a ly positive linear relationship between GDP and Health Expenditures.

```r
# Model2 summary
summary(ols2)
```

```
##
## Call:
## lm(formula = GDp ~ Health.exp, data = mydf1)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -21596 -11694   2661   9899  19242
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2488.6     8649.4   0.288    0.776
## Health.exp    1593.3      987.7   1.613    0.118
##
## Residual standard error: 12250 on 27 degrees of freedom
## Multiple R-squared:  0.08791,    Adjusted R-squared:  0.05413
## F-statistic: 2.602 on 1 and 27 DF,  p-value: 0.1183
```

```
# Confidence Interval
confint(ols2)
```

```
##                   2.5 %    97.5 %
## (Intercept) -15258.4712 20235.605
## Health.exp    -433.2554  3619.922
```

They both have weakly positive correlation coefficients with GDP, meaning that there are very small variations within the GDP that are being explained by Growth and Health expenditures.

Non linear regression (OLS)

1. Prepare the data

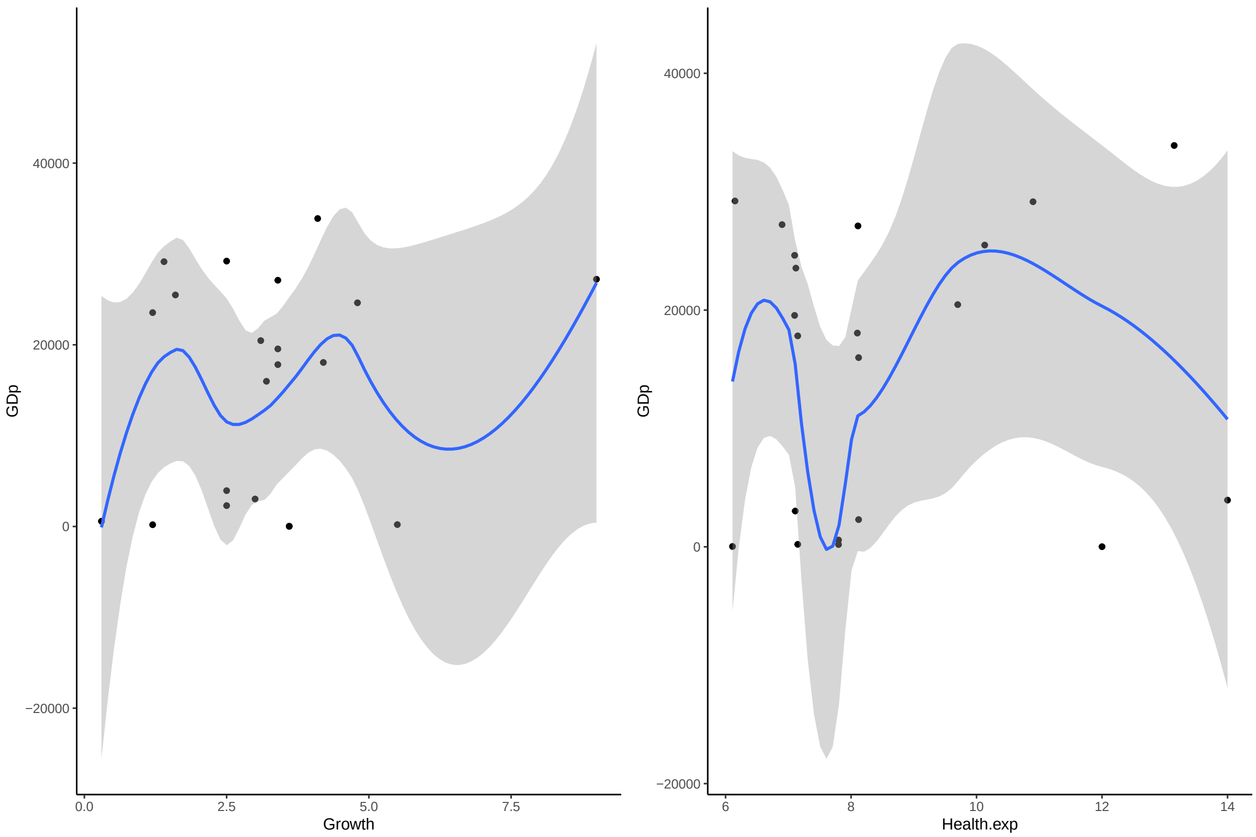
```
# Split the data into training and test set
set.seed(123)
training.samples <- mydf1$GDp %>%
  createDataPartition(p = 0.65, list = FALSE)
train.Gdp  <- mydf1[training.samples, ]
test.Gdp <- mydf1[-training.samples, ]

# Build the model
model <- lm(GDp ~ Growth, data = mydf1)
# Make predictions
predictions <- model %>% predict(test.Gdp)
# Model performance
data.frame(
  RMSE = RMSE(predictions, test.Gdp$GDp),
  R2 = R2(predictions, test.Gdp$GDp)
)
```

```
##       RMSE         R2
## 1 13994.31 0.06330223
```

```
# Lets Visualize and see how they look like
p1 <- ggplot(train.Gdp, aes(Growth, GDp) ) +
  geom_point() +
  stat_smooth()
```

```
p2 <- ggplot(train.Gdp, aes(Health.exp, GDp) ) +
  geom_point() +
  stat_smooth()
grid.arrange(p1,p2, ncol=2)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



GDP Vs. Growth

```
# Now run the Regression models
model1 <- lm(GDp ~ Growth + I(Growth^2), data = train.Gdp)
summary(model1)
```

```
##
## Call:
## lm(formula = GDp ~ Growth + I(Growth^2), data = train.Gdp)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -17685 -12056   2586  10176  17875
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 12747.13    8957.31   1.423    0.172
## Growth        400.09    4450.18   0.090    0.929
## I(Growth^2)    97.43     476.08   0.205    0.840
```

```
##
## Residual standard error: 12480 on 18 degrees of freedom
## Multiple R-squared:  0.0387, Adjusted R-squared:  -0.06811
## F-statistic: 0.3624 on 2 and 18 DF,  p-value: 0.701
```

```
#Confidence Intervals
as.data.frame(confint(model1))
```

```
##                  2.5 %    97.5 %
## (Intercept) -6071.489 31565.740
## Growth      -8949.388  9749.565
## I(Growth^2)  -902.772  1097.625
```

$$model1 = a + bX + cX^2$$

which is the same as $\hat{y} = 16428.7 - 829.7X + 163.3X^2$

$R^2 = 0.024, p = 0.7675$ means that its a poor fit, so we need to try a polynomial of a higher degree.

After trying some values I found that only polynomial of 6th order has some almost significant values.

```
model_1 <- lm(GDp ~ poly(Growth, 6, raw = TRUE), data = train.Gdp)
summary(model_1)
```

```
##
## Call:
## lm(formula = GDp ~ poly(Growth, 6, raw = TRUE), data = train.Gdp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16969.9  -8975.3    523.3   6528.4  14144.0
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    3060.3    36553.5   0.084    0.934
## poly(Growth, 6, raw = TRUE)1 -31457.9   146054.1  -0.215    0.833
## poly(Growth, 6, raw = TRUE)2  88832.0   175058.2   0.507    0.620
## poly(Growth, 6, raw = TRUE)3 -63439.0    89763.5  -0.707    0.491
## poly(Growth, 6, raw = TRUE)4  19031.1    22172.4   0.858    0.405
## poly(Growth, 6, raw = TRUE)5  -2515.7     2580.0  -0.975    0.346
## poly(Growth, 6, raw = TRUE)6    118.6      111.8   1.061    0.307
##
## Residual standard error: 11890 on 14 degrees of freedom
## Multiple R-squared:  0.3213, Adjusted R-squared:  0.0305
## F-statistic: 1.105 on 6 and 14 DF,  p-value: 0.4072
```

Using such a high order polynomial would be a very huge abuse to linear regression so I considered the second order to make prediction

GDP Vs. Health exp

```
# Run the Regression models
model2 <- lm(GDp ~ Health.exp + I(Health.exp^2), data = train.Gdp)
summary(model2)
```

```
##
## Call:
```

```
## lm(formula = GDp ~ Health.exp + I(Health.exp^2), data = train.Gdp)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -17048 -12449   2999  10087  18091
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -11710.6    56733.1  -0.206    0.839
## Health.exp        5579.4    12143.5   0.459    0.651
## I(Health.exp^2)   -265.1      611.8  -0.433    0.670
##
## Residual standard error: 12640 on 18 degrees of freedom
## Multiple R-squared:  0.01439,    Adjusted R-squared:  -0.09512
## F-statistic: 0.1314 on 2 and 18 DF,  p-value: 0.8777
```

```
#Confidence Intervals
as.data.frame(confint(model2))
```

```
##                       2.5 %     97.5 %
## (Intercept)     -130902.431 107481.191
## Health.exp       -19933.250  31092.028
## I(Health.exp^2)   -1550.518   1020.228
```

Predictive Analysis

```
# Make predictions
new_data <- test.Gdp
predictions1 <- model1 %>% predict(new_data)
predictions2 <- model2 %>% predict(new_data)
# Models performance
grth <- data.frame(
  RMSE = RMSE(predictions1, new_data$Growth),
  R2 = R2(predictions1, new_data$Growth)
)

hlth <- data.frame(
  RMSE = RMSE(predictions2, new_data$Health.exp),
  R2 = R2(predictions2, new_data$Health.exp)
)
```

```
# model1 performance
grth
```

```
##       RMSE        R2
## 1 15566.37 0.9959716
```

```
# model2 performance
hlth
```

```
##       RMSE        R2
## 1 14798.57 0.6261322
```

```
g <- ggplot(new_data, aes(Growth, GDp) ) +
  geom_point() +
  stat_smooth(method = lm, formula = y ~ poly(x, 2, raw = TRUE))
```

```
h <- ggplot(new_data, aes(Health.exp, GDp) ) +
  geom_point() +
  stat_smooth(method = lm, formula = y ~ poly(x, 2, raw = TRUE))
grid.arrange(g, h, ncol=2)
```