Análise exploratória e Regressão na base do rank de Felicidade por País de '2018' e '2019' com o R

Céasar Lemos - B.Sc Matemática

19/01/2020

Contents

1 Introdução

Regressão é uma técnica que permite entender e inferir a relação de uma variável dependente (y) com outras variáveis chamadas variáveis explicativas (x_1, x_2,... x_n). Entendendo a relação entre estas variávies, é possível estimar e/ou prever a variável y retornando o valor médio. Este termo foi criado por Francis Galton em seu experimento que mensurava a altura dos pais e seus respectivos filhos. Observando os dados coletados, Galton reparou que a estatura média das crianças nascidas de pais com uma certa altura tendia a "regredir" para a altura média da população. Esta lei foi confirmada por Karl Pearson que, com mais de mil observações coletadas, concluiu que a altura média de filhos de pais altos era menor do que a de seus genitores e os filhos de pais mais baixos era maior que seus pais. Ou seja, filhos de pais altos e de pais baixos "regrediam" igualmente para a altura média.

Uma forma comum de avaliar se uma variável pode inferir em outra é traçando um gráfico de dispersão. Nela, visualmente vemos se há alguma correlação, podemos ter noção do grau de correlação e se ela é linear ou não linear.

O objetivo deste artigo é realizar uma análise de regressão para verificar a inferência de indicadores econômicos e sociais sobre o indicador de felicidade mensurado por país em 2018 e projetar 2019.

2 Pacotes e Importação dos dados

2.1 Pacotes

Os pacotes utilizados para replicar o experimento estão listados abaixo. Caso não tenha nenhum instalado, pode usar o comando *install.packages*("nome do pacote entre aspas").

```
library(readr)
library(caret)
library(forecast)
library(xtable)
library(tidyverse)
library(scales)
library(GGally)
library(gridExtra)
library(lmtest)
```

2.2 Dados

Os dados foram coletados no site do Kaggle no seguinte link: https://www.kaggle.com/unsdsn/world-happiness#2019.csv

Os arquivos de 2018 e 2019 estão no formato *.csv* e contém os seguintes dados por país:

- Score: índice que mede a felicidade da população
- GDP per capita: Produto Interno Bruto (PIB) per capita
- Social Support: Suporte social fornecido pelo governo
- Healthy life expectancy: Expectativa de vida saudável
- Freedom to make life choices: Liberdade de fazer escolhas para a sua vida
- Generosity: Generosidade da população
- Perceptions of corruption: Percepção de Corrupção no país

2.3 Importando os dados

```
# Importando 2018
dt2018 <- read_csv(choose.files(), skip = 1,
                   col_names = c("rank","country","score","gdp","social_support",
                                 "life_expect","freedom_life","generosity","corruption"))

# Importando 2019
dt2019 <- read_csv(choose.files(), skip = 1,
                   col_names = c("rank","country","score","gdp","social_support",
                                 "life_expect","freedom_life","generosity","corruption"))
```

3 Análise exploratória dos dados

Vamos olhar os dados de 2018. Ele será a base que fornecerá a função para projetar 2019.

```
# Verificando o formato dos dados
str(dt2018[,3:9])
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    156 obs. of  7 variables:
##  $ score         : num  7.63 7.59 7.55 7.5 7.49 ...
##  $ gdp           : num  1.3 1.46 1.35 1.34 1.42 ...
##  $ social_support: num  1.59 1.58 1.59 1.64 1.55 ...
##  $ life_expect   : num  0.874 0.861 0.868 0.914 0.927 0.878 0.896 0.876 0.913 0.91 ...
##  $ freedom_life  : 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 ...
##  $ corruption    : chr  "0.393" "0.340" "0.408" "0.138" ...
```

O indicador *Perceptions of corruption* está como character(). Vamos ajustar para *double* antes de prosseguirmos com a análise.

```
# Convertendo de Character para Double
options(digits = 3)
dt2018$corruption <- as.double(dt2018$corruption)

class(dt2018$corruption)
```

[1] "numeric"

Agora o formato do dado de 2018 está correto. Vamos fazer isto para 2019 também.

```
# Convertendo de Character para Double
options(digits = 3)
dt2019$corruption <- as.double(dt2018$corruption)

class(dt2019$corruption)
```

[1] "numeric"

Vamos dar continuidade verificando se existe miss value.

```r
# Verificando a existência de miss value

result <- as.data.frame(sapply(dt2018, function(x) sum(is.na(x))))
colnames(result) <- c("Miss Value")
print(xtable(result))
```

% latex table generated in R 3.6.1 by xtable 1.8-4 package % Mon Jan 20 01:59:59 2020

	Miss Value
rank	0
country	0
score	0
gdp	0
social_support	0
life_expect	0
freedom_life	0
generosity	0
corruption	1

Existe 1 miss value no indicador Perceptions of corruption. Como se trata de apenas uma observação, podemos usar a função *view(data2018)* para visualizar toda a tabela e verificar a posição do miss value. No modelo de regressão podemos ignorar valores *n.a*, logo isso não será um problema. Uma técnica usada para lidar com situações assim é a interpolação.

Vamos continuar analisando os dados.

```r
# Verificando os 10 primeiros países no rank
print(xtable(head(dt2018[,-1], 10)))
```

% latex table generated in R 3.6.1 by xtable 1.8-4 package % Mon Jan 20 01:59:59 2020

	country	score	gdp	social_support	life_expect	freedom_life	generosity	corruption
1	Finland	7.63	1.30	1.59	0.87	0.68	0.20	0.39
2	Norway	7.59	1.46	1.58	0.86	0.69	0.29	0.34
3	Denmark	7.55	1.35	1.59	0.87	0.68	0.28	0.41
4	Iceland	7.50	1.34	1.64	0.91	0.68	0.35	0.14
5	Switzerland	7.49	1.42	1.55	0.93	0.66	0.26	0.36
6	Netherlands	7.44	1.36	1.49	0.88	0.64	0.33	0.29
7	Canada	7.33	1.33	1.53	0.90	0.65	0.32	0.29
8	New Zealand	7.32	1.27	1.60	0.88	0.67	0.36	0.39
9	Sweden	7.31	1.35	1.50	0.91	0.66	0.28	0.38
10	Australia	7.27	1.34	1.57	0.91	0.65	0.36	0.30

```r
# Verificando os 10 últimos países no rank
print(xtable(tail(dt2018[,-1], 10)))
```

% latex table generated in R 3.6.1 by xtable 1.8-4 package % Mon Jan 20 01:59:59 2020

	country	score	gdp	social_support	life_expect	freedom_life	generosity	corruption
1	Malawi	3.59	0.19	0.54	0.31	0.53	0.21	0.08
2	Haiti	3.58	0.32	0.71	0.29	0.02	0.39	0.10
3	Liberia	3.50	0.08	0.86	0.27	0.42	0.21	0.03
4	Syria	3.46	0.69	0.38	0.54	0.09	0.38	0.14
5	Rwanda	3.41	0.33	0.90	0.40	0.64	0.20	0.44
6	Yemen	3.35	0.44	1.07	0.34	0.24	0.08	0.06
7	Tanzania	3.30	0.46	0.99	0.38	0.48	0.27	0.10
8	South Sudan	3.25	0.34	0.61	0.18	0.11	0.22	0.11
9	Central African Republic	3.08	0.02	0.00	0.01	0.30	0.22	0.04
10	Burundi	2.90	0.09	0.63	0.14	0.06	0.15	0.08

```r
# Verificando as principais estatísticas dos dados
print(xtable(summary(dt2018[,3:9])))
```

% latex table generated in R 3.6.1 by xtable 1.8-4 package % Mon Jan 20 01:59:59 2020

	score	gdp	social_support	life_expect	freedom_life	generosity	corruption
X	Min. :2.90	Min. :0.000	Min. :0.00	Min. :0.000	Min. :0.000	Min. :0.000	Min. :0.000
X.1	1st Qu.:4.45	1st Qu.:0.616	1st Qu.:1.07	1st Qu.:0.422	1st Qu.:0.356	1st Qu.:0.110	1st Qu.:0.051
X.2	Median :5.38	Median :0.950	Median :1.25	Median :0.644	Median :0.487	Median :0.174	Median :0.082
X.3	Mean :5.38	Mean :0.891	Mean :1.21	Mean :0.597	Mean :0.455	Mean :0.181	Mean :0.112
X.4	3rd Qu.:6.17	3rd Qu.:1.198	3rd Qu.:1.46	3rd Qu.:0.777	3rd Qu.:0.578	3rd Qu.:0.239	3rd Qu.:0.137
X.5	Max. :7.63	Max. :2.096	Max. :1.64	Max. :1.030	Max. :0.724	Max. :0.598	Max. :0.457
X.6							NA's :1

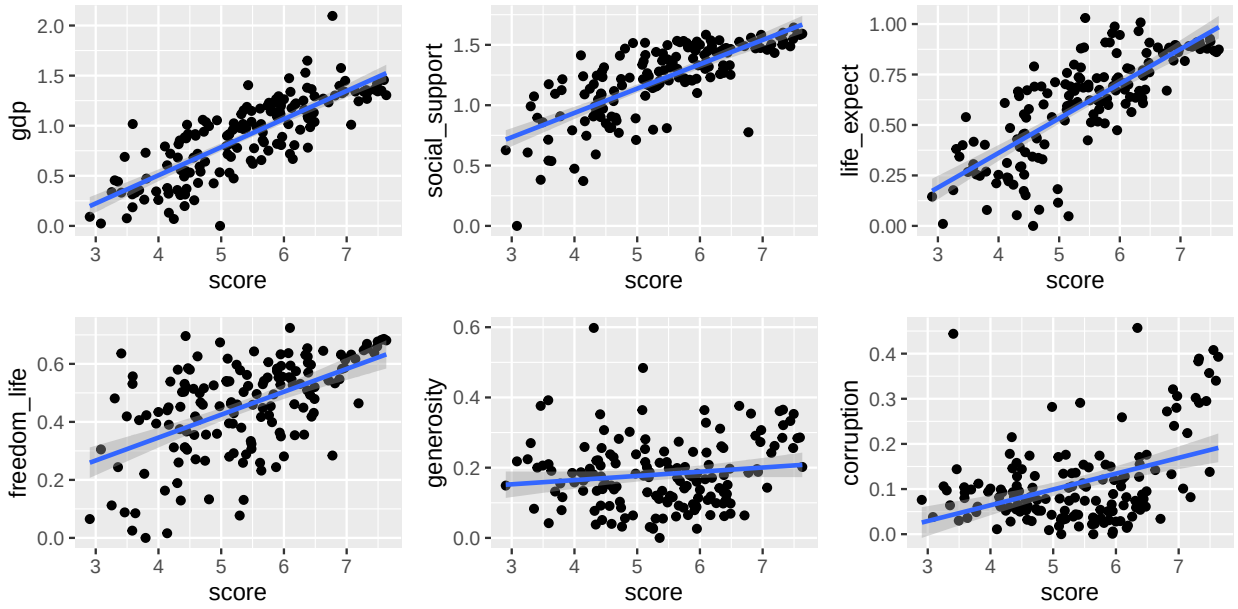
```r
# Analisando o desvio padrão das variáveis

desvpad <- as.data.frame(sapply(dt2018[,3:9], function(x) sd(x, na.rm = T)))
colnames(desvpad) <- c("Desvio Padrão")
print(xtable(desvpad))
```

% latex table generated in R 3.6.1 by xtable 1.8-4 package % Mon Jan 20 01:59:59 2020

	Desvio Padrão
score	1.12
gdp	0.39
social_support	0.30
life_expect	0.25
freedom_life	0.16
generosity	0.10
corruption	0.10

Vamos gerar os gráficos de correlação usando o pacote GGPlot2

```
# Plotando as Correlações
corr1 <- ggplot(dt2018) +
  geom_point(mapping = aes(x = score, y = gdp)) +
  geom_smooth(mapping = aes(x = score, y = gdp), method = lm)

corr2 <- ggplot(dt2018) +
  geom_point(mapping = aes(x = score, y = social_support)) +
  geom_smooth(mapping = aes(x = score, y = social_support), method = lm)

corr3 <- ggplot(dt2018) +
  geom_point(mapping = aes(x = score, y = life_expect)) +
  geom_smooth(mapping = aes(x = score, y = life_expect), method = lm)

corr4 <- ggplot(dt2018) +
  geom_point(mapping = aes(x = score, y = freedom_life)) +
  geom_smooth(mapping = aes(x = score, y = freedom_life), method = lm)

corr5 <- ggplot(dt2018) +
  geom_point(mapping = aes(x = score, y = generosity)) +
  geom_smooth(mapping = aes(x = score, y = generosity), method = lm)

corr6 <- ggplot(dt2018) +
  geom_point(mapping = aes(x = score, y = corruption)) +
  geom_smooth(mapping = aes(x = score, y = corruption), method = lm)

grid.arrange(corr1, corr2, corr3, corr4, corr5, corr6, ncol=3)
```



Analisando os gráficos, é perceptível que a correlação entre o score e a generosity e corruption não ocorre de forma linear. Com isso, podemos supor que estas variáveis terão um nível de significância ($p - value$) maior que $0,05$, aceitando a hipótese nula (H_0)

4 Fazendo o modelo de Regressão Linear

Para rodar o modelo de Regressão Linear, basta executar o seguinte código:

```
lm_model <- lm(score~., dt2018[,3:9])
summary(lm_model)
```

```
##
## Call:
## lm(formula = score ~ ., data = dt2018[, 3:9])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7307 -0.2864  0.0022  0.3590  1.0850
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.823      0.198    9.22  2.7e-16 ***
## gdp               0.902      0.242    3.72  0.00028 ***
## social_support    1.115      0.212    5.27  4.8e-07 ***
## life_expect       0.967      0.343    2.82  0.00540 **
## freedom_life      1.398      0.319    4.39  2.1e-05 ***
## generosity        0.524      0.472    1.11  0.26887
## corruption        0.728      0.528    1.38  0.17000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.522 on 148 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.791,  Adjusted R-squared:  0.782
## F-statistic: 93.1 on 6 and 148 DF,  p-value: <2e-16
```

Conforme visto no gráfico, as variáveis *generosity* e *corruptuon* não apresentam significância para explicar o score a um nível de confiança aceitável, portanto, vamos otimizar esta regressão usando a função *step()*. Esta função escolhe as melhores variáveis utilizando o *Critério de Informação de Akaike (AIC)*.

```
lm_model <- step(lm(score~., dt2018[,3:9]), direction = "both")
```

```
## Start:  AIC=-195
## score ~ gdp + social_support + life_expect + freedom_life + generosity +
##     corruption
##
##                  Df Sum of Sq  RSS  AIC
## - generosity      1      0.34 40.6 -196
## - corruption      1      0.52 40.8 -195
## <none>                       40.3 -195
## - life_expect     1      2.17 42.5 -189
## - gdp             1      3.77 44.1 -183
## - freedom_life    1      5.25 45.5 -178
## - social_support  1      7.55 47.8 -170
##
## Step:  AIC=-196
```

```
## score ~ gdp + social_support + life_expect + freedom_life + corruption
##
##                  Df Sum of Sq  RSS  AIC
## <none>                        40.6 -196
## + generosity      1      0.34 40.3 -195
## - corruption      1      0.87 41.5 -194
## - life_expect     1      2.21 42.8 -189
## - gdp             1      3.56 44.2 -185
## - freedom_life    1      6.03 46.6 -176
## - social_support  1      7.55 48.2 -171
```

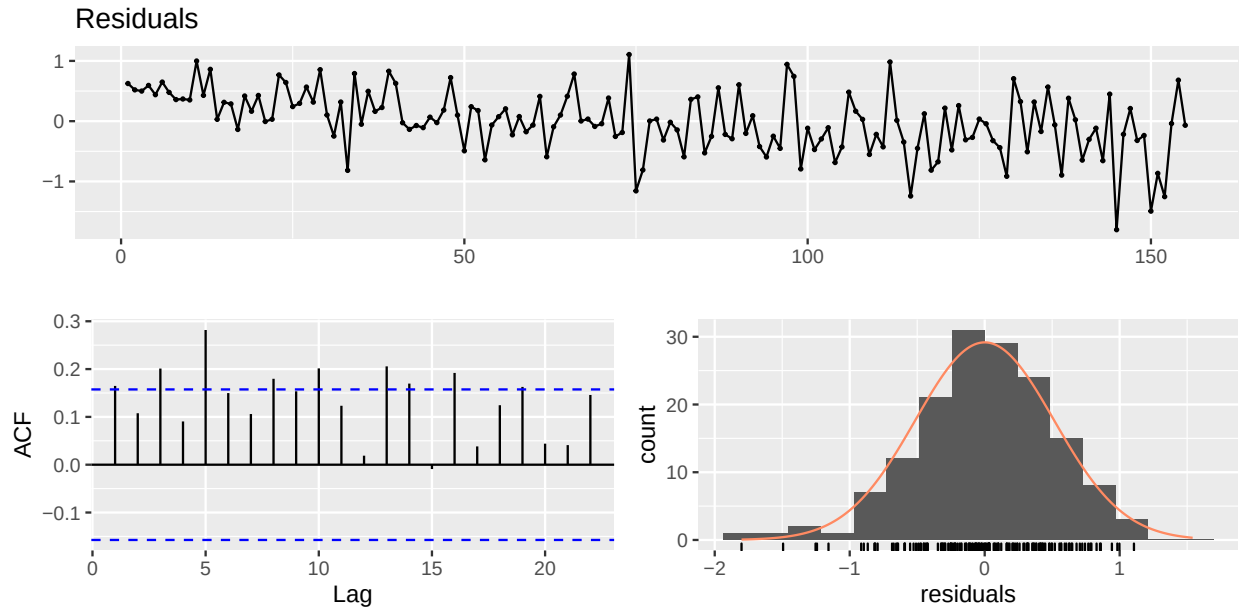
```
summary(lm_model)
```

```
##
## Call:
## lm(formula = score ~ gdp + social_support + life_expect + freedom_life +
##     corruption, data = dt2018[, 3:9])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.8016 -0.2934  0.0034  0.3650  1.1059
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.890      0.189   10.01  < 2e-16 ***
## gdp               0.870      0.241    3.61  0.00041 ***
## social_support    1.115      0.212    5.26  4.9e-07 ***
## life_expect       0.975      0.343    2.85  0.00505 **
## freedom_life      1.469      0.312    4.70  5.8e-06 ***
## corruption        0.900      0.505    1.78  0.07667 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.522 on 149 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.789,  Adjusted R-squared:  0.782
## F-statistic:  111 on 5 and 149 DF,  p-value: <2e-16
```

Como visto na tabela acima, a função *step()* com o argumento *both* selecionou a variável corruption que, sem a presença da variável *generosity*, tem um nível de confiança de 90%, conforme o teste do *p-value*. Todas as outras apresentam um nível de confiança maior que 95%. Já o R^2 aponta que estas variáveis explicam $78,9\%$ do *score*.

Dando uma olhada nos resíduos da regressão, vemos que os erros não apresentam uma distribuição normal. O teste de Shapiro nos confirma isto apresentando um *p-value* acima de 0.05.

```
checkresiduals(lm_model)
```



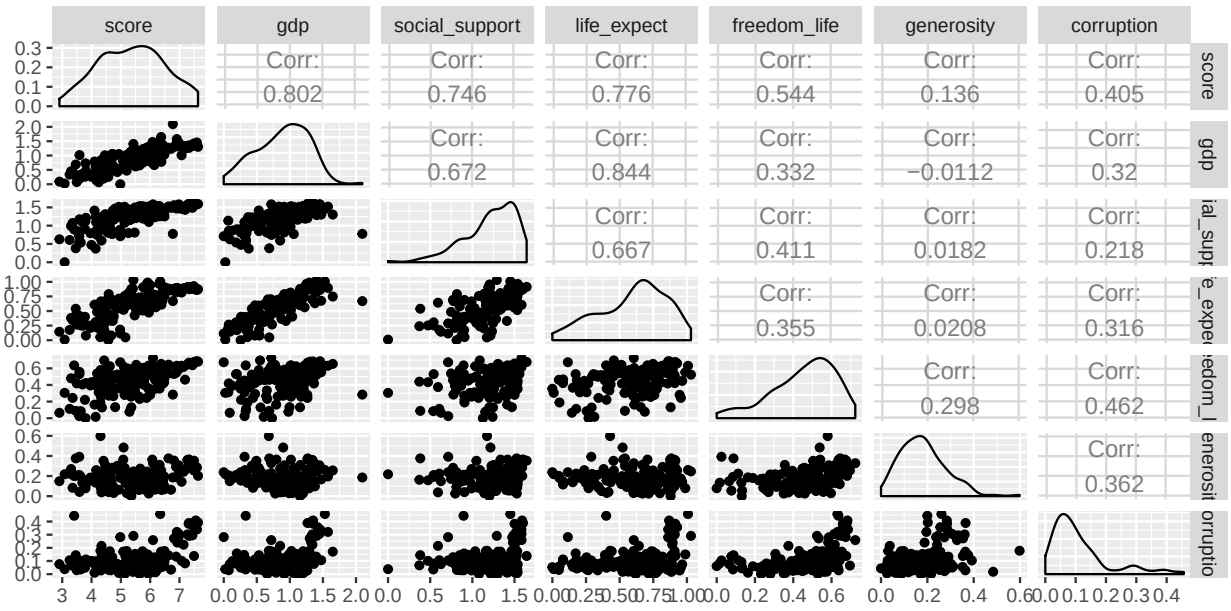

```
##
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  Residuals
## LM test = 52, df = 10, p-value = 1e-07
```

```
shapiro.test(lm_model$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  lm_model$residuals
## W = 1, p-value = 0.07
```

Um dos problemas comuns em regressão e que impacta bastante o modelo é a multicolinearidade. Vamos verificar a existência usando o código abaixo:

```
ggpairs(dt2018[,3:9])
```

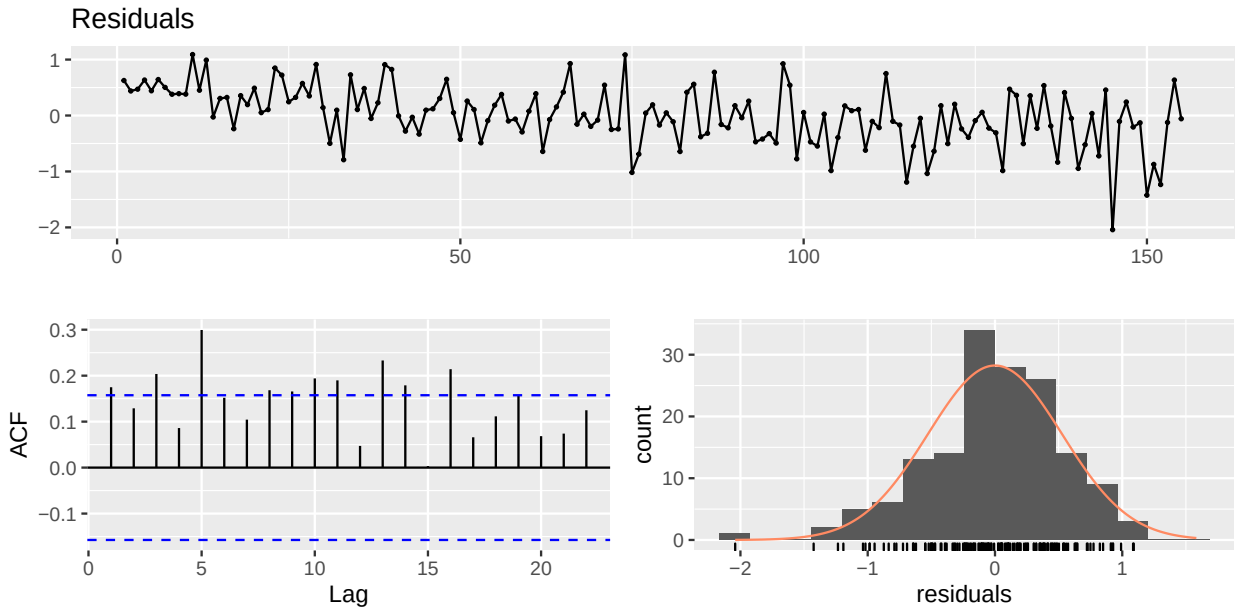


Podemos perceber uma forte correlação entre as variáveis *GDP* e *life expect*, com um índice de 0.844. Vamos tirar a variável *life expect* do modelo para realizar o ajuste, pois a variável *GDP* apresenta uma correlação maior com o *score*.

```
lm_model <- lm(score~gdp+social_support+freedom_life+corruption, dt2018[,3:9])
summary(lm_model)
```

```
##
## Call:
## lm(formula = score ~ gdp + social_support + freedom_life + corruption,
##     data = dt2018[, 3:9])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0412 -0.2866  0.0367  0.3785  1.0902
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.921      0.193    9.97  < 2e-16 ***
## gdp               1.367      0.170    8.04  2.4e-13 ***
## social_support    1.187      0.215    5.51  1.5e-07 ***
## freedom_life      1.498      0.320    4.69  6.2e-06 ***
## corruption        0.994      0.516    1.93    0.056 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.534 on 150 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.777,  Adjusted R-squared:  0.771
## F-statistic:  131 on 4 and 150 DF,  p-value: <2e-16
```

```
checkresiduals(lm_model)
```



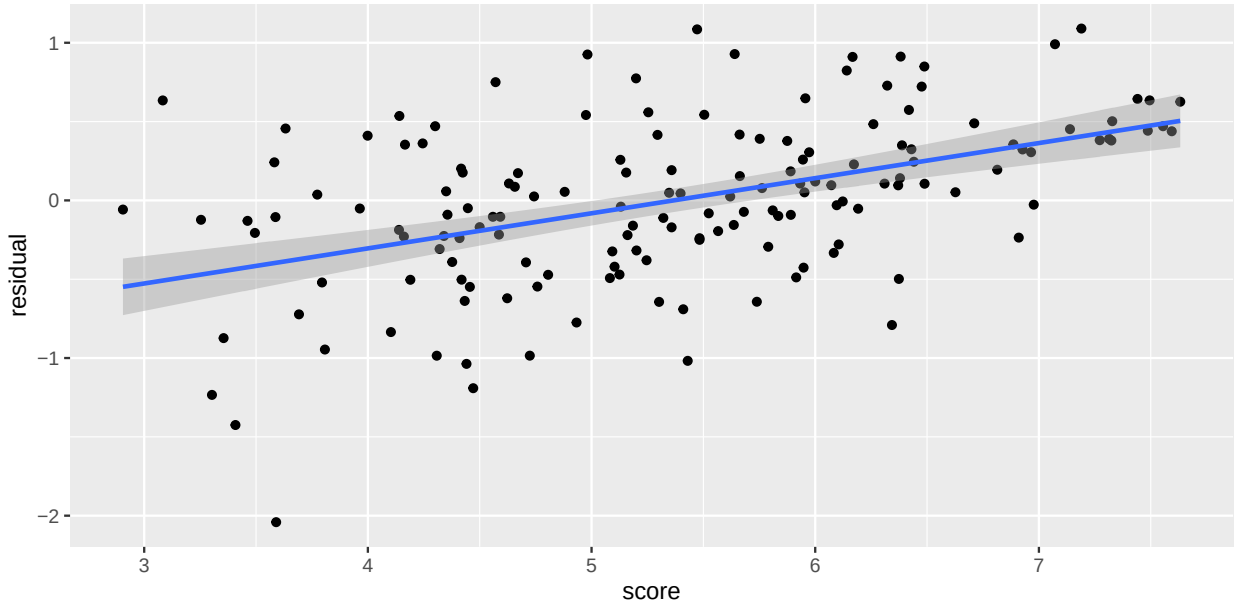

```
##
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  Residuals
## LM test = 57, df = 10, p-value = 1e-08
```

```
shapiro.test(lm_model$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  lm_model$residuals
## W = 1, p-value = 0.03
```

Outro problema comum nas regressões de cortes trasversais é a *heterocedasticidade*. Podemos gerar um gráfico de correlação entre os resóduos e a variável dependente do modelo e rodar o teste de Breusch-Pagan para verificar a existência deste problema:

```
resid <- as.data.frame(lm_model$residuals)
colnames(resid) <- c("residual")
dtcheck <- cbind(dt2018[-20,], resid)

ggplot(dtcheck) +
  geom_point(mapping = aes(x = score, y= residual))+
  geom_smooth(mapping = aes(x = score, y= residual), method = lm)
```




```
bptest(lm_model)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  lm_model
## BP = 5, df = 4, p-value = 0.2
```

Como o *p-value* é maior que $0,05$, temos que rejeitar a hipótese nula e assumir que este modelo apresenta heterocedasticidade, ou seja, os estimadores pelo MQO não são os melhores estimadores lineares não viesados.

Embora exista heterocedasticidade, a distribuição dos erros ocorre de forma normal (conforme demonstrado no teste de Shapiro após exclusão da variável life_expect) o que indica que o MQO ainda é o melhor estimador e que este problema provavelmente ocorre por ausência de outras variáveis que explicam o *score*.

Com o modelo ajustado em mãos, vamos utilizar os indicadores de 2019 para prever o *score* e verificar a margem de erro.

```
fcast <- forecast(lm_model, newdata = dt2019, level = 95)
accuracy(fcast$mean, dt2019$score)
```

```
##            ME  RMSE   MAE  MPE MAPE
## Test set 0.11 0.559 0.455 1.15    9
```

É visível que o erro médio absoluto percentual (MAPE) e o erro médio absoluto (MAE) está relativamente alto. Pode-se dizer que o modelo necessita de mais variáveis para tentar explicar o *score* de felicidade.

```
dtfcast <- as.data.frame(cbind(dt2019$country,
                               round(fcast$mean,3),dt2019$score))
colnames(dtfcast) <- c("country","forecast_score","real_score")

print(xtable(head(dtfcast,10)))
```

% latex table generated in R 3.6.1 by xtable 1.8-4 package % Mon Jan 20 02:00:14 2020

	country	forecast_score	real_score
1	Finland	6.921	7.769
2	Denmark	6.904	7.6
3	Norway	7.142	7.554
4	Iceland	6.758	7.494
5	Netherlands	6.826	7.488
6	Switzerland	6.868	7.48
7	Sweden	6.732	7.343
8	New Zealand	6.814	7.307
9	Canada	6.83	7.278
10	Austria	6.651	7.246

5 Conclusão

Os dados obtidos do rank de felicidade por países nos fornece variáveis que explicam de fato o *score*. Entretanto, foi identificado alguns problemas que necessitaram ajustes, como multicolinearidade entre variáveis explicativas e a ausência de correlação entre a generosidade e o score. O modelo ajustado de regressão foi capaz de explicar em $77,7\%$ o *score*, porém, não foi suficiente para fazer predições com alta acurácia. O modelo necessita de mais informações para predizer com um nível maior de assertividade.