Evergy Consuption of Electric Cars

Entender o Problema

Uma empresa da área de transporte e logística deseja migrar sua frota para carros elétricos com o objetivo de reduzir os custos.

Antes de tomar a decisão, a empresa gostaria de prever o consumo de energia de carros elétricos com base em diversos fatores de utilização e características dos veículos.

Usando um dataset com dados reais disponíveis publicamente, será construido um modelo de Machine Learning capaz de prever o consumo de energia de carros elétricos com base em diversos fatores, tais como o tipo e número de motores elétricos do veículo, o peso do veículo, a capacidade de carga, entre outros atributos.

Este conjunto de dados lista carros totalmente elétricos que, a partir de 2 de dezembro de 2020, poderiam ser adquiridos na Polônia como novos em uma concessionária autorizada e aqueles disponíveis em pré-venda pública e geral, mas somente se uma lista de preços disponível publicamente com versões de equipamentos e parâmetros técnicos completos estivesse disponível. com seus atributos

A coleção não contém dados sobre carros híbridos e carros elétricos dos chamados "extensores de alcance". Os carros a hidrogênio também não foram incluídos no conjunto de dados devido ao número insuficiente de modelos produzidos em massa e à especificidade diferente (em comparação com veículo elétrico) do veículo, incluindo os diferentes métodos de carregamento.

O banco de dados composto por 53 carros elétricos (cada variante de um modelo – que difere em termos de capacidade da bateria, potência do motor etc. – é tratada separadamente) e 22 variáveis (25 variáveis, incluindo marca, modelo e "nome do carro" mesclando estes dois anteriores).

```
library(readxl)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- 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
```

```
library(moments)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(vcd)
```

```
## Loading required package: grid
```

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
##
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
library(gridExtra)
```

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

```
library(car)
```

```
## Loading required package: carData
##
## Attaching package: 'car'
##
## The following object is masked from 'package:dplyr':
##
##     recode
##
## The following object is masked from 'package:purrr':
##
##     some
```

```
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:gridExtra':
##
##     combine
##
## The following object is masked from 'package:dplyr':
##
##     combine
##
## The following object is masked from 'package:ggplot2':
##
```

```
##     margin
```

```
library(Boruta)
library(glmnet)
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
##
## Loaded glmnet 4.1-7
```

```
library(rpart)
library(randomForest)
library(e1071)
```

```
##
## Attaching package: 'e1071'
##
## The following objects are masked from 'package:moments':
##
##     kurtosis, moment, skewness
```

Carregando os dados

```
dados <- read_excel("FEV-data-Excel.xlsx", col_types = "guess")
```

PASSO 01 - Descrição dos Dados

1.1 Renomear colunas

```
colnames(dados) <- c('nomeCarro', 'marca', 'modelo', 'precoMinimoBrutoPLN', 'potenciaMotorKM', 'torqueMa

colnames(dados)
```

```
##  [1] "nomeCarro"                    "marca"
##  [3] "modelo"                       "precoMinimoBrutoPLN"
##  [5] "potenciaMotorKM"              "torqueMaximoNm"
##  [7] "tipoFreios"                   "tipoTracao"
##  [9] "capacidadeBateriaKWh"         "autonomiaWLTPkm"
## [11] "distanciaEntreEixosCm"        "comprimentoCm"
## [13] "larguraCm"                    "alturaCm"
## [15] "pesoMinimoVazioKg"            "pesoBrutoPermissivelKg"
## [17] "capacidadeMaximaCargaKg"      "numeroAssentos"
## [19] "numeroPortas"                 "tamanhoPneuPol"
## [21] "velocidadeMaximaKmH"          "capacidadePortaMalasLitros"
## [23] "aceleracao0a100KmHS"          "potenciaMaximaCarregamentoDCKW"
## [25] "mediaConsumoEnergiaKWh100Km"
```

```
# Vizualiza dos dados
head(dados)
```

```
## # A tibble: 6 x 25
##   nomeCarro      marca modelo precoMinimoBrutoPLN potenciaMotorKM torqueMaximoNm
##   <chr>          <chr> <chr>                <dbl>           <dbl>          <dbl>
## 1 Audi e-tron 5~ Audi  e-tro~              345700             360            664
## 2 Audi e-tron 5~ Audi  e-tro~              308400             313            540
## 3 Audi e-tron S~ Audi  e-tro~              414900             503            973
## 4 Audi e-tron S~ Audi  e-tro~              319700             313            540
## 5 Audi e-tron S~ Audi  e-tro~              357000             360            664
## 6 Audi e-tron S~ Audi  e-tro~              426200             503            973
## # i 19 more variables: tipoFreios <chr>, tipoTracao <chr>,
## #   capacidadeBateriaKWh <dbl>, autonomiaWLTPkm <dbl>,
## #   distanciaEntreEixosCm <dbl>, comprimentoCm <dbl>, larguraCm <dbl>,
## #   alturaCm <dbl>, pesoMinimoVazioKg <dbl>, pesoBrutoPermissivelKg <dbl>,
## #   capacidadeMaximaCargaKg <dbl>, numeroAssentos <dbl>, numeroPortas <dbl>,
## #   tamanhoPneuPol <dbl>, velocidadeMaximaKmH <dbl>,
## #   capacidadePortaMalasLitros <dbl>, aceleracao0a100KmHS <dbl>, ...
```

1.2 Data Dimension

```
print(paste("Number of rows ", nrow(dados)))
```

```
## [1] "Number of rows  53"
```

```
print(paste("Number of columns ", ncol(dados)))
```

```
## [1] "Number of columns  25"
```

1.3 Data Types

```
str(dados)
```

```
## tibble [53 x 25] (S3: tbl_df/tbl/data.frame)
##  $ nomeCarro                    : chr [1:53] "Audi e-tron 55 quattro" "Audi e-tron 50 quattro" "Audi
##  $ marca                        : chr [1:53] "Audi" "Audi" "Audi" "Audi" ...
##  $ modelo                       : chr [1:53] "e-tron 55 quattro" "e-tron 50 quattro" "e-tron S quat
##  $ precoMinimoBrutoPLN          : num [1:53] 345700 308400 414900 319700 357000 ...
##  $ potenciaMotorKM              : num [1:53] 360 313 503 313 360 503 170 184 286 136 ...
##  $ torqueMaximoNm               : num [1:53] 664 540 973 540 664 973 250 270 400 260 ...
##  $ tipoFreios                   : chr [1:53] "disc (front + rear)" "disc (front + rear)" "disc (fro
##  $ tipoTracao                   : chr [1:53] "4WD" "4WD" "4WD" "4WD" ...
##  $ capacidadeBateriaKWh         : num [1:53] 95 71 95 71 95 95 42.2 42.2 80 50 ...
##  $ autonomiaWLTPkm              : num [1:53] 438 340 364 346 447 369 359 345 460 350 ...
##  $ distanciaEntreEixosCm        : num [1:53] 293 293 293 293 293 ...
##  $ comprimentoCm                : num [1:53] 490 490 490 490 490 ...
##  $ larguraCm                    : num [1:53] 194 194 198 194 194 ...
##  $ alturaCm                     : num [1:53] 163 163 163 162 162 ...
##  $ pesoMinimoVazioKg            : num [1:53] 2565 2445 2695 2445 2595 ...
##  $ pesoBrutoPermissivelKg       : num [1:53] 3130 3040 3130 3040 3130 ...
##  $ capacidadeMaximaCargaKg      : num [1:53] 640 670 565 640 670 565 440 440 540 459 ...
##  $ numeroAssentos               : num [1:53] 5 5 5 5 5 5 4 4 5 5 ...
##  $ numeroPortas                 : num [1:53] 5 5 5 5 5 5 5 5 5 5 ...
##  $ tamanhoPneuPol               : num [1:53] 19 19 20 19 19 20 19 20 19 16 ...
##  $ velocidadeMaximaKmH          : num [1:53] 200 190 210 190 200 210 160 160 180 150 ...
##  $ capacidadePortaMalasLitros   : num [1:53] 660 660 660 615 615 615 260 260 510 380 ...
##  $ aceleracao0a100KmHS          : num [1:53] 5.7 6.8 4.5 6.8 5.7 4.5 8.1 6.9 6.8 9.5 ...
```

```
##  $ potenciaMaximaCarregamentoDCKW: num [1:53] 150 150 150 150 150 150 50 50 150 100 ...
##  $ mediaConsumoEnergiaKWh100Km   : num [1:53] 24.4 23.8 27.6 23.3 23.9 ...
```

1.4 Check NAs

```
# Contagem de linhas com dados completos
sum(complete.cases(dados))
```

```
## [1] 42
```

```
# Contagem de linhas com dados incompletos
sum(!complete.cases(dados))
```

```
## [1] 11
```

```
colSums(is.na(dados))
```

```
##                      nomeCarro                          marca
##                              0                              0
##                         modelo             precoMinimoBrutoPLN
##                              0                              0
##                potenciaMotorKM                 torqueMaximoNm
##                              0                              0
##                     tipoFreios                     tipoTracao
##                              1                              0
##           capacidadeBateriaKWh                 autonomiaWLTPkm
##                              0                              0
##          distanciaEntreEixosCm                  comprimentoCm
##                              0                              0
##                      larguraCm                       alturaCm
##                              0                              0
##              pesoMinimoVazioKg          pesoBrutoPermissivelKg
##                              0                              8
##        capacidadeMaximaCargaKg                 numeroAssentos
##                              8                              0
##                   numeroPortas                 tamanhoPneuPol
##                              0                              0
##            velocidadeMaximaKmH     capacidadePortaMalasLitros
##                              0                              1
##            aceleracao0a100KmHS potenciaMaximaCarregamentoDCKW
##                              3                              0
##    mediaConsumoEnergiaKWh100Km
##                              9
```

1.4 Preencher NAs

Como o conjunto de dados é pequeno e 11 linhas não possuem dados completos, será feito imputation dos dados.

Mercedes-Benz EQV (long)

- tipoFreios = disc (front + rear)
- aceleracao0a100KmHS = 12.1s
- capacidadePortaMalasLitros = 978.5

```
dados[dados$nomeCarro == 'Mercedes-Benz EQV (long)', "tipoFreios"] <- "disc (front + rear)"
dados[dados$nomeCarro == 'Mercedes-Benz EQV (long)', "aceleracao0a100KmHS"] <- 12.1
dados[dados$nomeCarro == 'Mercedes-Benz EQV (long)', "capacidadePortaMalasLitros"] <- 978.5
```

Tesla Model 3 Standard Range Plus

- capacidadeMaximaCargaKg = 389 kg
- pesoBrutoPermissivelKg = 2014 kg
- mediaConsumoEnergiaKWh100Km = 14.6 kWh/100 km

```
dados[dados$nomeCarro == 'Tesla Model 3 Standard Range Plus', "capacidadeMaximaCargaKg"] <- 389
dados[dados$nomeCarro == 'Tesla Model 3 Standard Range Plus', "pesoBrutoPermissivelKg"] <- 2014
dados[dados$nomeCarro == 'Tesla Model 3 Standard Range Plus', "mediaConsumoEnergiaKWh100Km"] <- 14.6
```

Peugeot-e-2008

- capacidadeMaximaCargaKg = 482 kg
- pesoBrutoPermissivelKg = 2030 kg
- aceleracao0a100KmHS = 8.5 s
- mediaConsumoEnergiaKWh100Km = 18.2 kWh/100 km

```
dados[dados$nomeCarro == 'Peugeot e-2008', "capacidadeMaximaCargaKg"] <- 389
dados[dados$nomeCarro == 'Peugeot e-2008', "pesoBrutoPermissivelKg"] <- 2030
dados[dados$nomeCarro == 'Peugeot e-2008', "aceleracao0a100KmHS"] <- 8.5
dados[dados$nomeCarro == 'Peugeot e-2008', "mediaConsumoEnergiaKWh100Km"] <- 18.2
```

Citroën ë-C4

- mediaConsumoEnergiaKWh100Km = 16.8 kWh/100 km

```
dados[dados$nomeCarro == 'Citroën ë-C4', "mediaConsumoEnergiaKWh100Km"]  <- 16.8
```

Tesla Model 3 Long Range

- capacidadeMaximaCargaKg = 388 kg
- pesoBrutoPermissivelKg = 2232 -mediaConsumoEnergiaKWh100Km = 15,5 kWh/100 km

```
dados[dados$nomeCarro == "Tesla Model 3 Long Range", "capacidadeMaximaCargaKg"] <- 388
dados[dados$nomeCarro == "Tesla Model 3 Long Range", "pesoBrutoPermissivelKg"] <- 2232
dados[dados$nomeCarro == "Tesla Model 3 Long Range", "mediaConsumoEnergiaKWh100Km"] <- 15.5
```

Tesla Model 3 Performance

- capacidadeMaximaCargaKg = 388 kg
- pesoBrutoPermissivelKg = 2232
- mediaConsumoEnergiaKWh100Km = 16,3 kWh/100 km

```
dados[dados$nomeCarro == "Tesla Model 3 Performance", "capacidadeMaximaCargaKg"] <- 388
dados[dados$nomeCarro == "Tesla Model 3 Performance", "pesoBrutoPermissivelKg"] <- 2232
dados[dados$nomeCarro == "Tesla Model 3 Performance", "mediaConsumoEnergiaKWh100Km"] <- 16.3
```

Tesla Model S Long Range Plus

- capacidadeMaximaCargaKg = 394 kg
- pesoBrutoPermissivelKg = 2574 kg
- mediaConsumoEnergiaKWh100Km = 17,7 kWh/100 km

```r
dados[dados$nomeCarro == "Tesla Model S Long Range Plus", "capacidadeMaximaCargaKg"] <- 394
dados[dados$nomeCarro == "Tesla Model S Long Range Plus", "pesoBrutoPermissivelKg"] <- 2574
dados[dados$nomeCarro == "Tesla Model S Long Range Plus", "mediaConsumoEnergiaKWh100Km"] <- 17.7
```

Tesla Model S Performance

- capacidadeMaximaCargaKg = 394 kg
- pesoBrutoPermissivelKg = 2626 kg
- mediaConsumoEnergiaKWh100Km = 18,3 kWh/100 km

```r
dados[dados$nomeCarro == "Tesla Model S Performance", "capacidadeMaximaCargaKg"] <- 394
dados[dados$nomeCarro == "Tesla Model S Performance", "pesoBrutoPermissivelKg"] <- 2626
dados[dados$nomeCarro == "Tesla Model S Performance", "mediaConsumoEnergiaKWh100Km"] <- 18.3
```

Tesla Model X Long Range Plus

- capacidadeMaximaCargaKg = 610 kg
- pesoBrutoPermissivelKg = 3040 kg
- mediaConsumoEnergiaKWh100Km = 20,4 kWh/100 km

```r
dados[dados$nomeCarro == "Tesla Model X Long Range Plus", "capacidadeMaximaCargaKg"] <- 610
dados[dados$nomeCarro == "Tesla Model X Long Range Plus", "pesoBrutoPermissivelKg"] <- 3040
dados[dados$nomeCarro == "Tesla Model X Long Range Plus", "mediaConsumoEnergiaKWh100Km"] <- 20.4
```

Tesla Model X Performance

- capacidadeMaximaCargaKg = 610 kg
- pesoBrutoPermissivelKg = 3087 kg
- mediaConsumoEnergiaKWh100Km = 20,9 kWh/100 km

```r
dados[dados$nomeCarro == "Tesla Model X Performance", "capacidadeMaximaCargaKg"] <- 610
dados[dados$nomeCarro == "Tesla Model X Performance", "pesoBrutoPermissivelKg"] <- 3087
dados[dados$nomeCarro == "Tesla Model X Performance", "mediaConsumoEnergiaKWh100Km"] <- 20.9
```

Nissan e-NV200 evalia

- aceleracao0a100KmHS = 14.0s

```r
dados[dados$nomeCarro == "Nissan e-NV200 evalia", "aceleracao0a100KmHS"] <- 14.0
```

```r
# Contagem de linhas com dados incompletos
sum(!complete.cases(dados))
```

```
## [1] 0
```

1.7 Estatísticas descritivas

1.7.1 Atributos Numericos

```r
get_descriptive_analysis <- function(dados){

mean <- apply(dados, 2, mean, na.rm = TRUE)
std <- apply(dados, 2, sd, na.rm = TRUE)
min <- apply(dados, 2, min, na.rm = TRUE)
max <- apply(dados, 2, max, na.rm = TRUE)
median <- apply(dados, 2, median, na.rm = TRUE)
```

```r
quartis <- apply(dados, 2, quantile, probs = c(0.25, 0.75), na.rm = TRUE)
IQR <- apply(dados, 2, IQR, na.rm = TRUE)
skew <- apply(dados, 2, skewness, na.rm = TRUE)
kurtosis <- apply(dados, 2, kurtosis, na.rm = TRUE)

result <- data.frame(min, max, max-min, mean, median, std, skew, kurtosis)


names(result) <- c("min", "max", "range", "mean", "median", "std", "skew", "kurtosis")

return(result)
}

get_descriptive_analysis(dados %>% select_if(is.numeric))
```

```
##                                    min      max    range         mean   median
## precoMinimoBrutoPLN            82050.0 794000.0 711950.0 2.461585e+05 178400.0
## potenciaMotorKM                   82.0    772.0    690.0 2.697736e+02    204.0
## torqueMaximoNm                   160.0   1140.0    980.0 4.600377e+02    362.0
## capacidadeBateriaKWh              17.6    100.0     82.4 6.236604e+01     58.0
## autonomiaWLTPkm                  148.0    652.0    504.0 3.769057e+02    364.0
## distanciaEntreEixosCm            187.3    327.5    140.2 2.735811e+02    270.0
## comprimentoCm                    269.5    514.0    244.5 4.425094e+02    447.0
## larguraCm                        164.5    255.8     91.3 1.862415e+02    180.9
## alturaCm                         137.8    191.0     53.2 1.554226e+02    155.6
## pesoMinimoVazioKg               1035.0   2710.0   1675.0 1.868453e+03   1685.0
## pesoBrutoPermissivelKg          1310.0   3500.0   2190.0 2.317604e+03   2170.0
## capacidadeMaximaCargaKg          290.0   1056.0    766.0 5.091132e+02    485.0
## numeroAssentos                     2.0      8.0      6.0 4.905660e+00      5.0
## numeroPortas                       3.0      5.0      2.0 4.849057e+00      5.0
## tamanhoPneuPol                    14.0     21.0      7.0 1.767925e+01     17.0
## velocidadeMaximaKmH              123.0    261.0    138.0 1.781698e+02    160.0
## capacidadePortaMalasLitros       171.0    978.5    807.5 4.551604e+02    425.0
## aceleracao0a100KmHS                2.5     14.0     11.5 7.596226e+00      7.9
## potenciaMaximaCarregamentoDCKW    22.0    270.0    248.0 1.135094e+02    100.0
## mediaConsumoEnergiaKWh100Km       13.1     28.2     15.1 1.876321e+01     17.1
##                                        std        skew    kurtosis
## precoMinimoBrutoPLN           1.491875e+05  1.47946765  2.14890348
## potenciaMotorKM               1.812986e+02  1.13421026  0.33326006
## torqueMaximoNm                2.616470e+02  1.11745460  0.20238904
## capacidadeBateriaKWh          2.417091e+01  0.11533262 -1.27237589
## autonomiaWLTPkm               1.188179e+02  0.23970997 -0.45082918
## distanciaEntreEixosCm         2.274052e+01 -0.65994872  2.30382343
## comprimentoCm                 4.886328e+01 -0.88312112  1.13198608
## larguraCm                     1.428064e+01  2.06780092  8.27541700
## alturaCm                      1.127536e+01  1.10942569  2.20366304
## pesoMinimoVazioKg             4.708809e+02  0.32751414 -1.24728446
## pesoBrutoPermissivelKg        5.405027e+02  0.29490683 -1.10930407
## capacidadeMaximaCargaKg       1.373991e+02  1.48888806  3.27580373
## numeroAssentos                8.381332e-01  0.55706018  5.01443448
## numeroPortas                  4.555735e-01 -2.97937791  8.16172633
## tamanhoPneuPol                1.868500e+00 -0.05945042 -1.14248535
## velocidadeMaximaKmH           4.305620e+01  0.79795118 -0.76112845
## capacidadePortaMalasLitros    1.928944e+02  0.77330578 -0.07962097
```

```
## aceleracao0a100KmHS            2.926665e+00  0.14166333 -0.77644701
## potenciaMaximaCarregamentoDCKW 5.716697e+01  0.90400317  1.10910551
## mediaConsumoEnergiaKWh100Km    4.133728e+00  0.78381216 -0.72971416
```

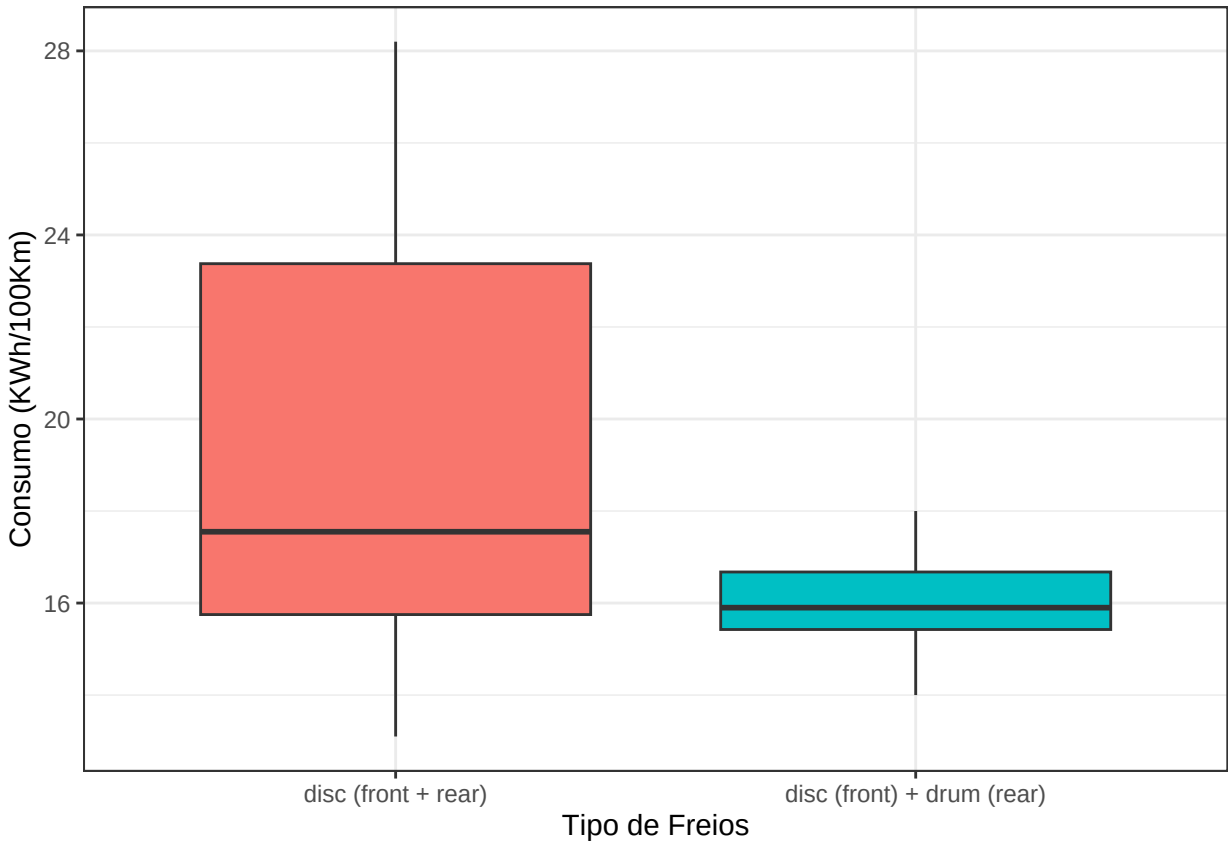
1.7.1 Atributos Categoricos

```
# Selecionar as variáveis categóricas
names(select(dados, where(is.character) | where(is.factor)))
```

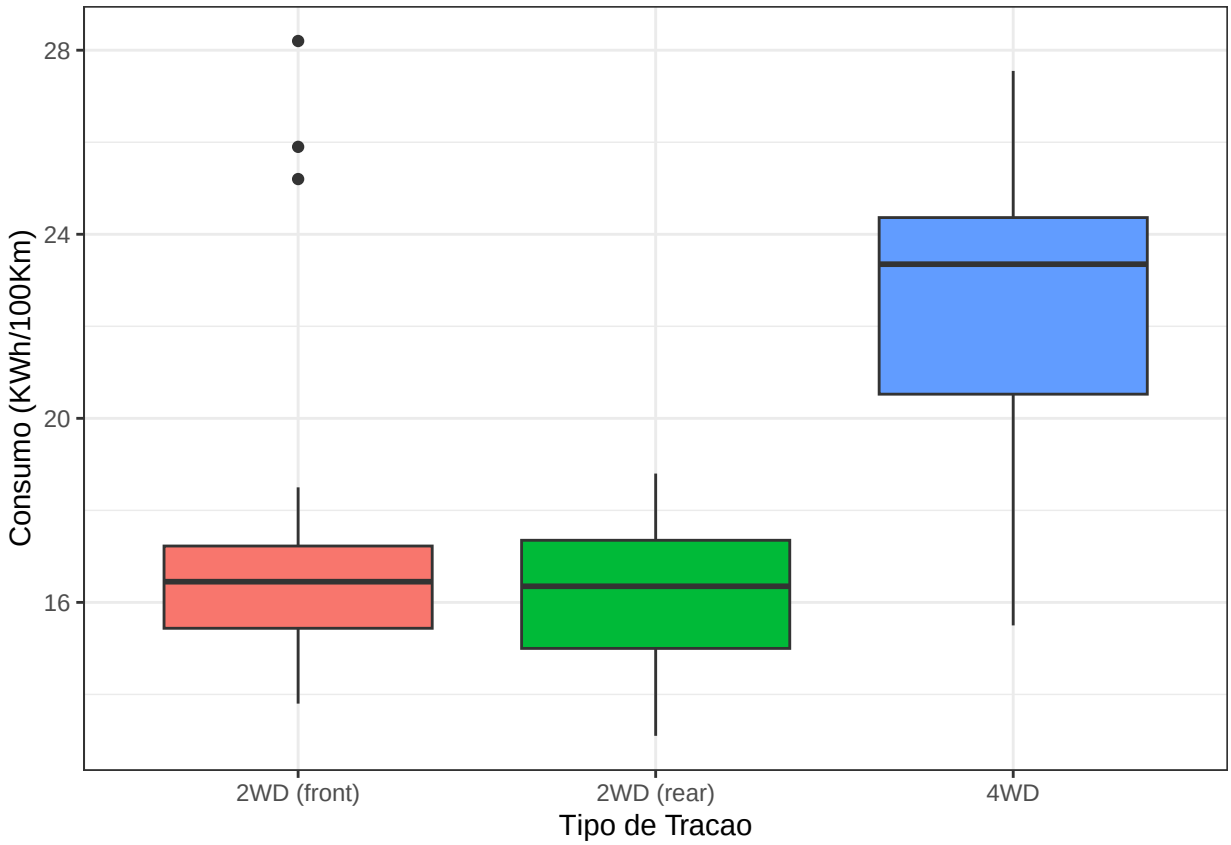
```
## [1] "nomeCarro"  "marca"      "modelo"     "tipoFreios" "tipoTracao"
```

As variáveis nomeCarro, modelo e marca serão desconsideradas durante a análise. Muitas marcas possuem apenas uma amostra na base de dados.

```
ggplot(dados, aes(x = tipoFreios, y = mediaConsumoEnergiaKWh100Km, fill=tipoFreios )) +
     geom_boxplot() +
     xlab("Tipo de Freios") +
     ylab("Consumo (KWh/100Km)") +
     theme_bw() +
     theme(legend.position = "none")
```




```
ggplot(dados, aes(x = tipoTracao, y = mediaConsumoEnergiaKWh100Km, fill=tipoTracao)) +
     geom_boxplot() +
     xlab("Tipo de Tracao") +
     ylab("Consumo (KWh/100Km)") +
     theme_bw() +
     theme(legend.position = "none")
```



PASSO 02 - Seleção da Colunas

```
# Remover colunas pelo nome
dados <- subset(dados, select = -c(nomeCarro, marca, modelo))

# Exibir as colunas do data frame resultante
colnames(dados)
```

```
##  [1] "precoMinimoBrutoPLN"            "potenciaMotorKM"
##  [3] "torqueMaximoNm"                 "tipoFreios"
##  [5] "tipoTracao"                     "capacidadeBateriaKWh"
##  [7] "autonomiaWLTPkm"                "distanciaEntreEixosCm"
##  [9] "comprimentoCm"                  "larguraCm"
## [11] "alturaCm"                       "pesoMinimoVazioKg"
## [13] "pesoBrutoPermissivelKg"         "capacidadeMaximaCargaKg"
## [15] "numeroAssentos"                 "numeroPortas"
## [17] "tamanhoPneuPol"                 "velocidadeMaximaKmH"
## [19] "capacidadePortaMalasLitros"     "aceleracao0a100KmHS"
## [21] "potenciaMaximaCarregamentoDCKW" "mediaConsumoEnergiaKWh100Km"
```

```
# Transformar variáveis categóricas em fatores
dados$tipoFreios <- as.factor(dados$tipoFreios)
dados$tipoTracao <- as.factor(dados$tipoTracao)
str(dados)
```

```
## tibble [53 x 22] (S3: tbl_df/tbl/data.frame)
##  $ precoMinimoBrutoPLN           : num [1:53] 345700 308400 414900 319700 357000 ...
```

```
##  $ potenciaMotorKM               : num [1:53] 360 313 503 313 360 503 170 184 286 136 ...
##  $ torqueMaximoNm                : num [1:53] 664 540 973 540 664 973 250 270 400 260 ...
##  $ tipoFreios                    : Factor w/ 2 levels "disc (front + rear)",..: 1 1 1 1 1 1 1 1 1 1
##  $ tipoTracao                    : Factor w/ 3 levels "2WD (front)",..: 3 3 3 3 3 3 2 2 2 1 ...
##  $ capacidadeBateriaKWh          : num [1:53] 95 71 95 71 95 95 42.2 42.2 80 50 ...
##  $ autonomiaWLTPkm               : num [1:53] 438 340 364 346 447 369 359 345 460 350 ...
##  $ distanciaEntreEixosCm         : num [1:53] 293 293 293 293 293 ...
##  $ comprimentoCm                 : num [1:53] 490 490 490 490 490 ...
##  $ larguraCm                     : num [1:53] 194 194 198 194 194 ...
##  $ alturaCm                      : num [1:53] 163 163 163 162 162 ...
##  $ pesoMinimoVazioKg             : num [1:53] 2565 2445 2695 2445 2595 ...
##  $ pesoBrutoPermissivelKg        : num [1:53] 3130 3040 3130 3040 3130 ...
##  $ capacidadeMaximaCargaKg       : num [1:53] 640 670 565 640 670 565 440 440 540 459 ...
##  $ numeroAssentos                : num [1:53] 5 5 5 5 5 5 4 4 5 5 ...
##  $ numeroPortas                  : num [1:53] 5 5 5 5 5 5 5 5 5 5 ...
##  $ tamanhoPneuPol                : num [1:53] 19 19 20 19 19 20 19 20 19 16 ...
##  $ velocidadeMaximaKmH           : num [1:53] 200 190 210 190 200 210 160 160 180 150 ...
##  $ capacidadePortaMalasLitros    : num [1:53] 660 660 660 615 615 615 260 260 510 380 ...
##  $ aceleracao0a100KmHS           : num [1:53] 5.7 6.8 4.5 6.8 5.7 4.5 8.1 6.9 6.8 9.5 ...
##  $ potenciaMaximaCarregamentoDCKW: num [1:53] 150 150 150 150 150 150 50 50 150 100 ...
##  $ mediaConsumoEnergiaKWh100Km   : num [1:53] 24.4 23.8 27.6 23.3 23.9 ...
```

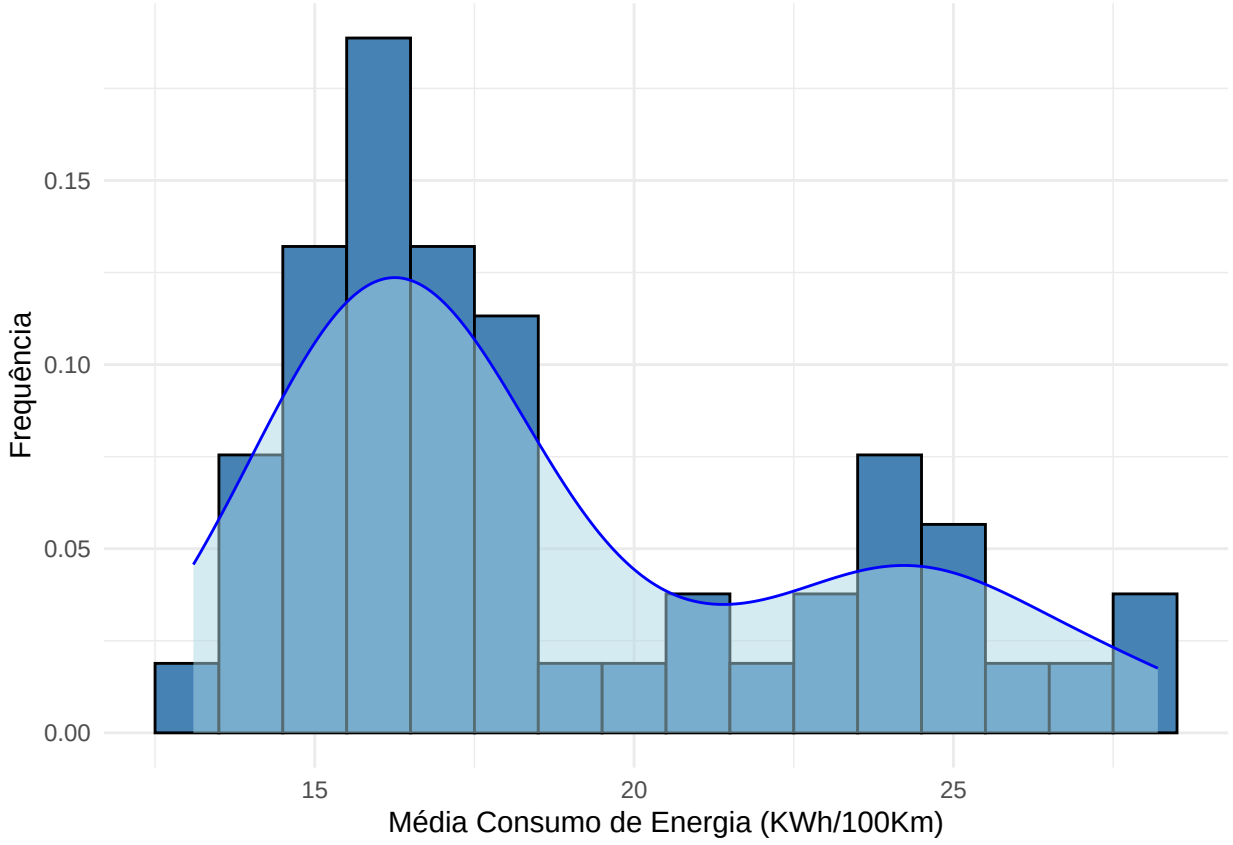
Passo 3 - Análise Exploratória dos dados

Como há poucas amostras no conjunto de dados a separação dos dados de treino e teste será realizada após a a analse exploratória dos dados.

3.1 Análise Univariada

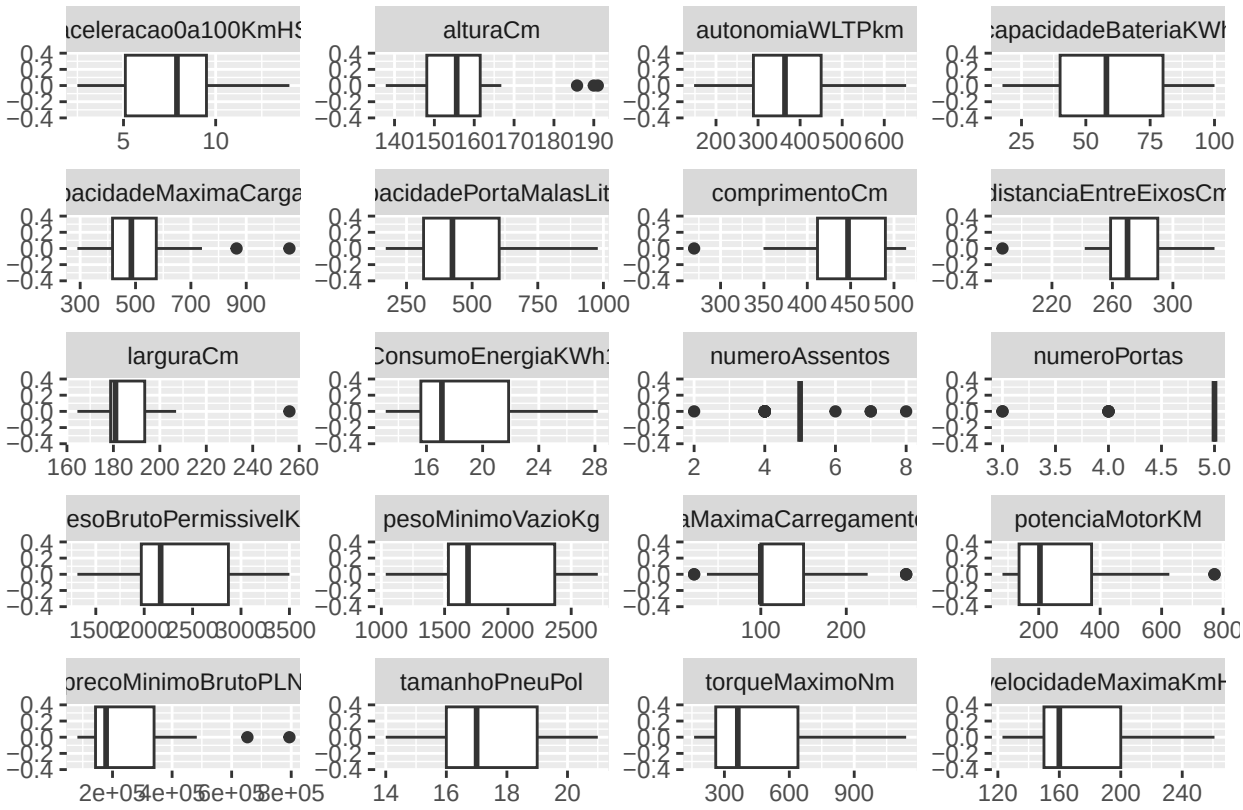
3.1.1 Variável Resposta

```r
# Crie o histograma com curva de densidade usando ggplot2
ggplot(data = dados, aes(x = mediaConsumoEnergiaKWh100Km)) +
  geom_histogram(aes(y = after_stat(density)), binwidth = 1, fill = "steelblue", color = "black") +
  geom_density(fill = "lightblue", alpha = 0.5, color = "blue") +
  labs(x = "Média Consumo de Energia (KWh/100Km)", y = "Frequência") +
  theme_minimal()
```



3.1.2 Variaveis Numéricas

```
# Plotando boxplots para cada variável numérica em um único gráfico
dados %>% select_if(is.numeric) %>%
  gather(variable, value ) %>%
  ggplot( aes(x=value)) +
  geom_boxplot() +
  facet_wrap(~ variable, ncol = 4, scales = "free")
```




```
# Verificar quais colunas têm valores discrepantes (outliers)
outliers <- apply(dados %>% select_if(is.numeric), 2, function(x) any(boxplot.stats(x)$out))
```

```
## Warning in any(boxplot.stats(x)$out): coercing argument of type 'double' to
## logical

## Warning in any(boxplot.stats(x)$out): coercing argument of type 'double' to
## logical

## Warning in any(boxplot.stats(x)$out): coercing argument of type 'double' to
## logical

## Warning in any(boxplot.stats(x)$out): coercing argument of type 'double' to
## logical

## Warning in any(boxplot.stats(x)$out): coercing argument of type 'double' to
## logical

## Warning in any(boxplot.stats(x)$out): coercing argument of type 'double' to
## logical

## Warning in any(boxplot.stats(x)$out): coercing argument of type 'double' to
## logical

## Warning in any(boxplot.stats(x)$out): coercing argument of type 'double' to
## logical
```

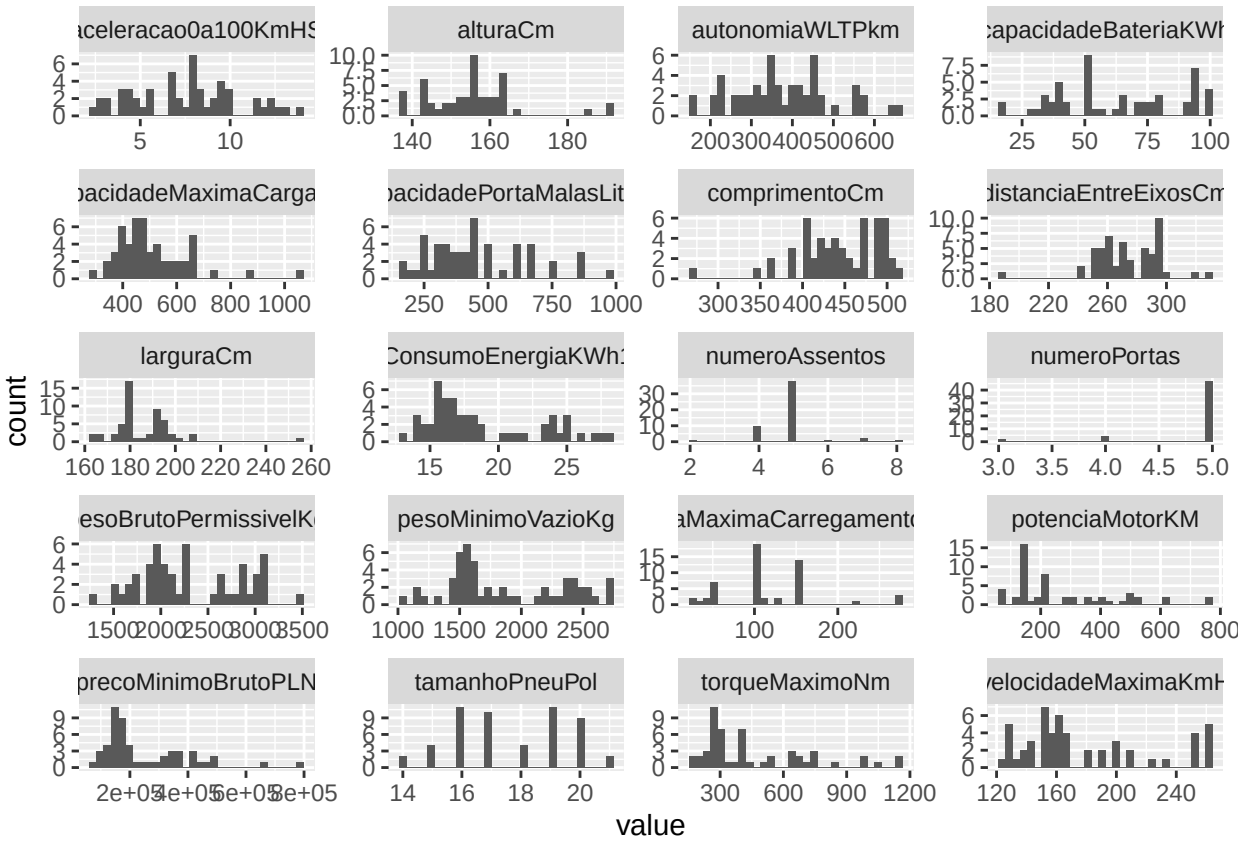
```
## Warning in any(boxplot.stats(x)$out): coercing argument of type 'double' to
## logical

## Warning in any(boxplot.stats(x)$out): coercing argument of type 'double' to
## logical
```

```
outliers
```

```
##            precoMinimoBrutoPLN                potenciaMotorKM
##                           TRUE                           TRUE
##                torqueMaximoNm           capacidadeBateriaKWh
##                          FALSE                          FALSE
##               autonomiaWLTPkm          distanciaEntreEixosCm
##                          FALSE                           TRUE
##                 comprimentoCm                      larguraCm
##                           TRUE                           TRUE
##                       alturaCm              pesoMinimoVazioKg
##                           TRUE                          FALSE
##         pesoBrutoPermissivelKg        capacidadeMaximaCargaKg
##                          FALSE                           TRUE
##                numeroAssentos                    numeroPortas
##                           TRUE                           TRUE
##                tamanhoPneuPol             velocidadeMaximaKmH
##                          FALSE                          FALSE
##     capacidadePortaMalasLitros             aceleracao0a100KmHS
##                          FALSE                          FALSE
## potenciaMaximaCarregamentoDCKW    mediaConsumoEnergiaKWh100Km
##                           TRUE                          FALSE
```

```
# Plotando histogramas para cada variável numérica em um único gráfico
dados %>% select_if(is.numeric) %>%
  gather(variable, value) %>%
  ggplot( aes(x=value)) +
  geom_histogram() +
  facet_wrap(~ variable, ncol = 4, scales = "free")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# Define a estrutura da figura
par(mar = c(2, 2, 2, 2))  # Definir margens menores
layout(matrix(1:20, 5, 4, byrow = TRUE))

# Inicializar a lista para salvar os valores-p
p_values <- list()


for (coluna in colnames(dados)) {
  if (is.numeric(dados[[coluna]])) {
    # Gerar o qqplot para a coluna numérica

    # qqPlot(dados[[coluna]], ylab = coluna)

    # Calcular o valor-p para o teste de normalidade
    shapiro <- shapiro.test(dados[[coluna]])
    p_value <- shapiro$p.value

    # Adicionar o valor-p à lista
    p_values[[coluna]] <- p_value
  }
}



as.data.frame(p_values) %>%
  gather(variable, 'p_value' ) %>%
```

```
  mutate(rejeitou_H0 = p_value < 0.05)
```

```
##                           variable      p_value rejeitou_H0
## 1             precoMinimoBrutoPLN 2.171747e-06        TRUE
## 2                 potenciaMotorKM 4.431479e-06        TRUE
## 3                  torqueMaximoNm 7.000158e-06        TRUE
## 4            capacidadeBateriaKWh 5.186537e-03        TRUE
## 5                 autonomiaWLTPkm 6.546809e-01       FALSE
## 6           distanciaEntreEixosCm 2.016518e-03        TRUE
## 7                   comprimentoCm 3.632577e-03        TRUE
## 8                       larguraCm 1.139571e-06        TRUE
## 9                        alturaCm 8.876149e-05        TRUE
## 10              pesoMinimoVazioKg 8.702493e-04        TRUE
## 11         pesoBrutoPermissivelKg 1.042015e-02        TRUE
## 12        capacidadeMaximaCargaKg 1.225622e-04        TRUE
## 13                numeroAssentos 1.892320e-09        TRUE
## 14                  numeroPortas 1.056700e-13        TRUE
## 15                tamanhoPneuPol 7.853066e-03        TRUE
## 16            velocidadeMaximaKmH 1.711430e-05        TRUE
## 17     capacidadePortaMalasLitros 7.638741e-03        TRUE
## 18            aceleracao0a100KmHS 2.670510e-01       FALSE
## 19 potenciaMaximaCarregamentoDCKW 4.782810e-05        TRUE
## 20    mediaConsumoEnergiaKWh100Km 7.582362e-05        TRUE
```

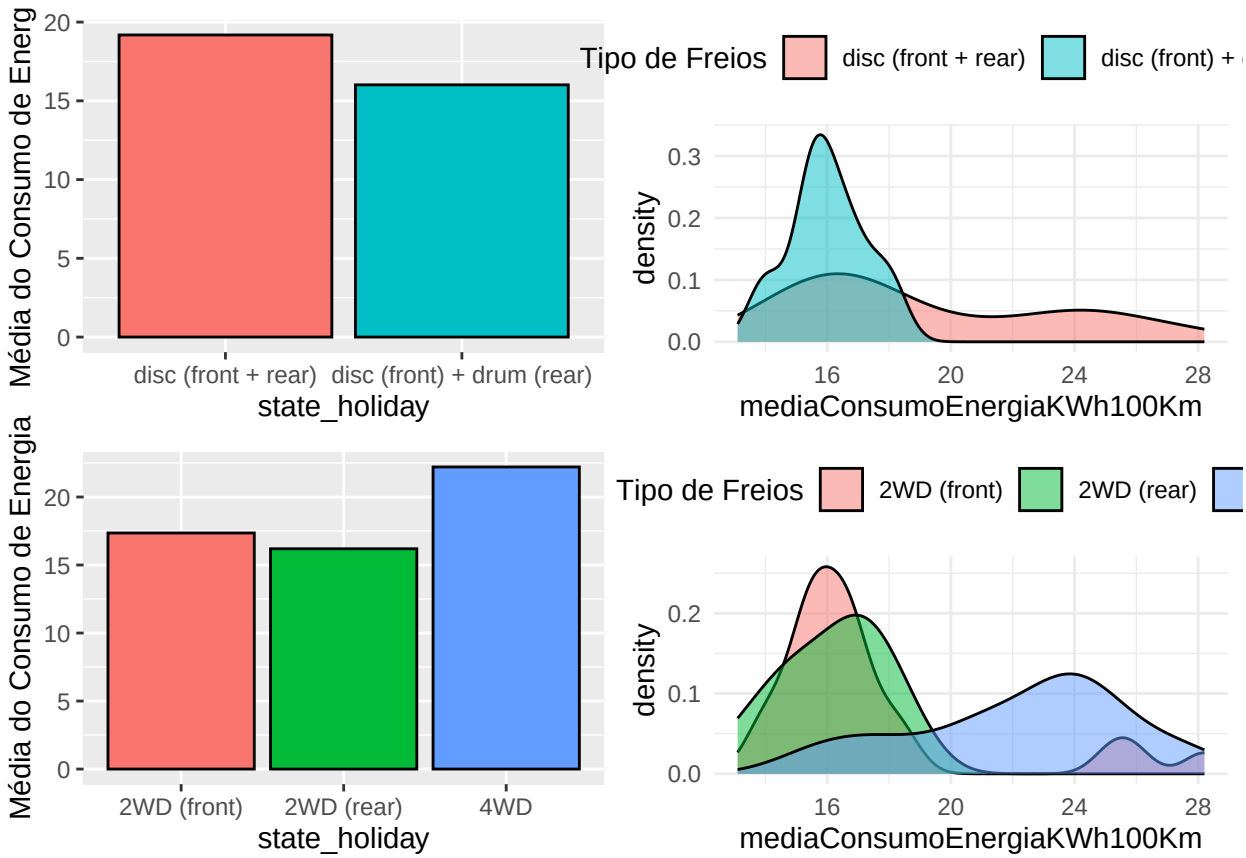
```
rm(p_values, p_value, shapiro, coluna)
```

Apenas as variáveis autonomiaWLTPkm e aceleracao0a100KmHS possuem evidências suficientes para não rejeitarem a hipótese nula de que as suas distribuições são normais.

3.1.3 Variáveis Categóricas

```
# Criar o gráfico de barras com ggplot2
p1 <- ggplot(data = dados, aes(x = tipoFreios, y = mediaConsumoEnergiaKWh100Km, fill = tipoFreios)) +
  geom_bar(stat = "summary", fun = "mean", color = "black") +
  labs(x = "state_holiday", y = "Média do Consumo de Energia") +
  theme(legend.position = "none")

# Criação do gráfico de densidade
p2 <- ggplot(dados, aes(x = mediaConsumoEnergiaKWh100Km, fill = tipoFreios)) +
  geom_density(alpha = 0.5) +
  scale_fill_discrete(name = "Tipo de Freios") +
  theme_minimal() +
  theme(legend.position = "top")

# Criar o gráfico de barras com ggplot2
p3 <- ggplot(data = dados, aes(x = tipoTracao, y = mediaConsumoEnergiaKWh100Km, fill = tipoTracao)) +
  geom_bar(stat = "summary", fun = "mean", color = "black") +
  labs(x = "state_holiday", y = "Média do Consumo de Energia") +
  theme(legend.position = "none")

# Criação do gráfico de densidade
p4 <- ggplot(dados, aes(x = mediaConsumoEnergiaKWh100Km, fill = tipoTracao)) +
  geom_density(alpha = 0.5) +
  scale_fill_discrete(name = "Tipo de Freios") +
```

```
  theme_minimal() +
  theme(legend.position = "top")


# Organização dos subplots lado a lado
grid.arrange(p1, p2, p3, p4, ncol = 2)
```

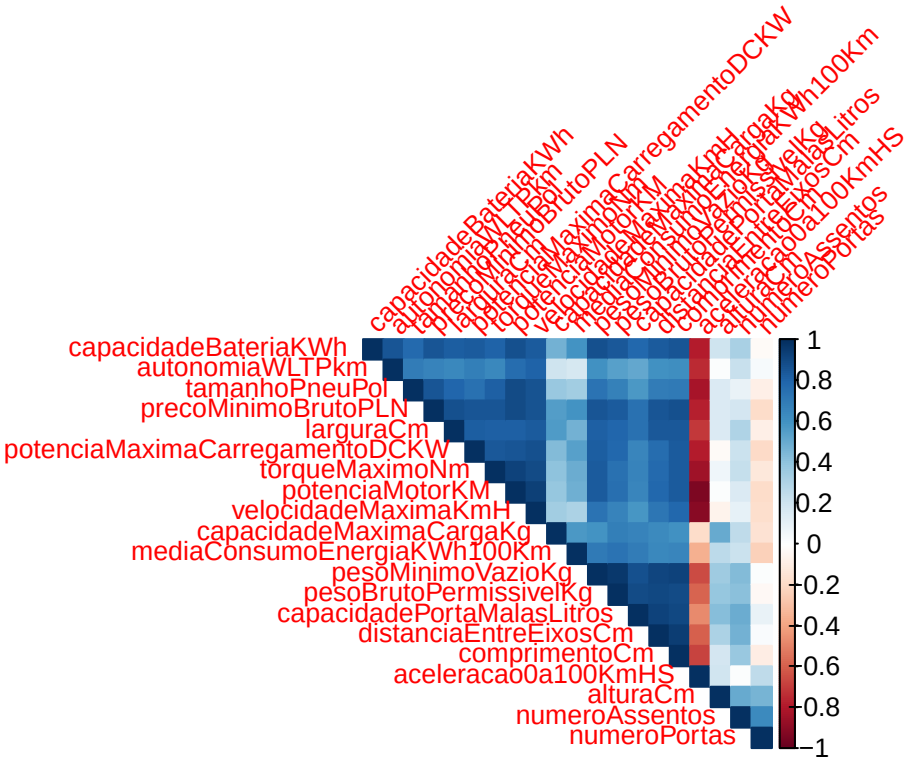



```
rm(p1 ,p2 ,p3 , p4)
```

3.2 Análise Multivariada

3.2.1 Variáveis Numericas

```
# Explorando relacionamento entre as variáveis: Matriz de Correlação - corrplot
corrplot(cor(dados %>% select_if(is.numeric), method = "spearman"), method = "color", type = "upper", o
```



3.2.2 Variáveis Categoricas

```
assocstats(table(dados$tipoTracao, dados$tipoFreios))
```

```
##                     X^2 df  P(> X^2)
## Likelihood Ratio 12.447  2 0.0019824
## Pearson          13.215  2 0.0013501
##
## Phi-Coefficient   : NA
## Contingency Coeff.: 0.447
## Cramer's V        : 0.499
```

Valor-p: O valor-p associado ao teste de qui-quadrado é menor que 0,05 ($p < 0,05$). Isso indica que há evidências estatísticas para rejeitar a hipótese nula de que não há associação entre as variáveis "tipoTracao" e "tipoFreios". Em outras palavras, existe uma associação significativa entre essas variáveis.

Coeficiente de Cramér-V: O coeficiente de Cramér-V é uma medida de associação entre variáveis categóricas. No caso, o valor de Cramér's V é 0,499. Esse valor varia de 0 a 1, onde 0 indica nenhuma associação e 1 indica associação completa. Um valor de 0,499 sugere uma associação moderada entre as variáveis "tipoTracao" e "tipoFreios".

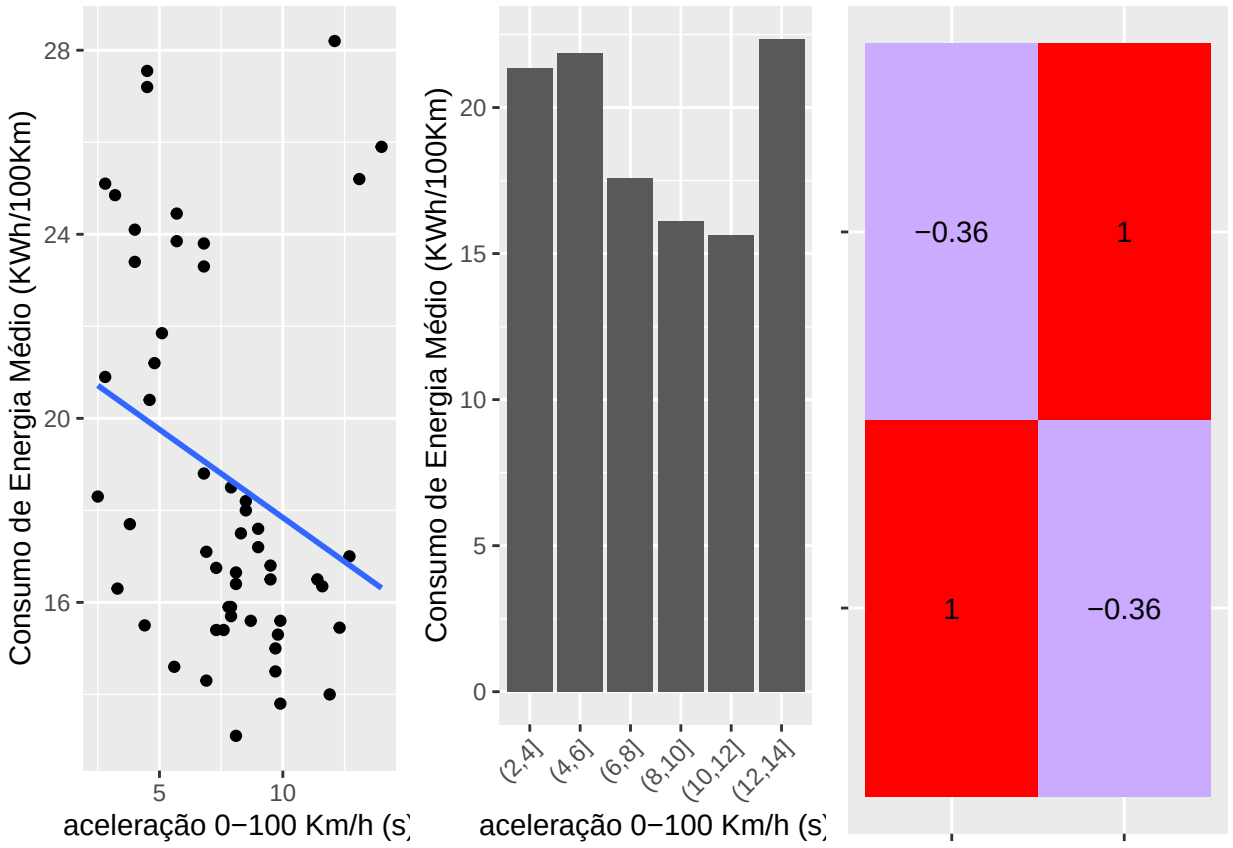
3.3 Análise Bivariada

```
# Primeiro subplot: scatterplot
p1 <- ggplot(dados, aes(x = aceleracao0a100KmHS, y = mediaConsumoEnergiaKWh100Km)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(x = "aceleração 0-100 Km/h (s)", y = "Consumo de Energia Médio (KWh/100Km)")
```

```r
# Segundo subplot: barplot
cuts <- seq(floor(min(dados$aceleracao0a100KmHS)), ceiling(max(dados$aceleracao0a100KmHS)), length.out =

dados$aceleracao0a100KmHS_binned  <- cut(dados$aceleracao0a100KmHS, breaks = cuts, order_result = ordere

p2 <- dados %>%
  group_by(aceleracao0a100KmHS_binned) %>%
  summarise(media_ConsumoEnergia = mean(mediaConsumoEnergiaKWh100Km)) %>%
    ggplot( aes(x = aceleracao0a100KmHS_binned, y = media_ConsumoEnergia)) +
    geom_bar(stat = "identity") +
    labs(x = "aceleração 0-100 Km/h (s)", y = "Consumo de Energia Médio (KWh/100Km)") +
    theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Terceiro subplot: heatmap de correlação
cor_mat <- cor(dados %>% select(aceleracao0a100KmHS, mediaConsumoEnergiaKWh100Km), method = "spearman")

# Transforme a matriz de correlação em um data.frame
cor_df <- melt(cor_mat)

# Plot a matriz de correlação como um gráfico de azulejos
p3 <- ggplot(cor_df, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile() +
  geom_text(aes(label = round(value, digits = 2)), color = "black", size = 4) + # Adiciona rótulos
  scale_fill_gradient2(low = "blue", mid = "white", high = "red", midpoint = 0) +
  theme(axis.text.x = element_blank(),  # Oculta os rótulos do eixo x
        axis.text.y = element_blank(),  # Oculta os rótulos do eixo y
        axis.title.x = element_blank(), # Oculta o título do eixo x
        axis.title.y = element_blank(), # Oculta o título do eixo y
        legend.position = "none") +
  theme(legend.position = "none")



# Organização dos subplots lado a lado
grid.arrange(p1, p2, p3, ncol = 3)
```

Hipótese 1: Carros que possuem menor tempo de aceleração de 0 a 100Km/h possuem menor media de consumo de Energia. Qual a correlação entre essas duas variáveis?

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




```
dados <- subset(dados, select = -aceleracao0a100KmHS_binned)
rm(cuts, p1 ,p2 ,p3 , cor_mat, cor_df)
```

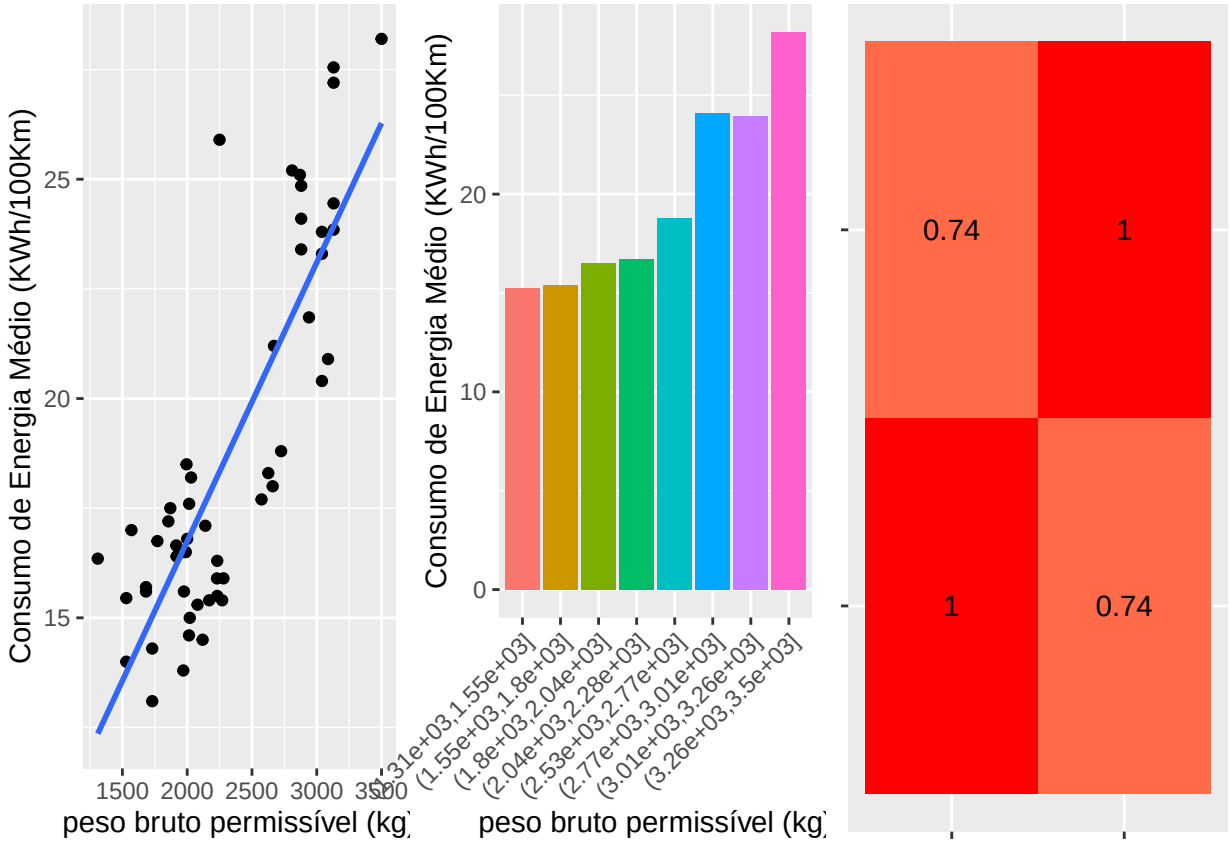
De acordo com os gráficos o coeficiente de correlação é ligeiramente negativo, indicando que há uma correlação negativa fraca entre as variáveis. Isso significa que, à medida que o tempo de aceleração de 0 a 100Km/h aumenta, o consumo medio de Energia tende a diminuir, mas a relação não é muito forte.

```
# Primeiro subplot: scatterplot
p1 <- ggplot(dados, aes(x = pesoBrutoPermissivelKg, y = mediaConsumoEnergiaKWh100Km)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(x = "peso bruto permissível (kg)", y = "Consumo de Energia Médio (KWh/100Km)")

# Segundo subplot: barplot
cuts <- seq(floor(min(dados$pesoBrutoPermissivelKg)-1), ceiling(max(dados$pesoBrutoPermissivelKg)), leng

dados$pesoBrutoPermissivelKg_binned  <- cut(dados$pesoBrutoPermissivelKg, breaks = cuts, order_result =

p2 <- dados %>%
  group_by(pesoBrutoPermissivelKg_binned) %>%
  summarise(media_ConsumoEnergia = mean(mediaConsumoEnergiaKWh100Km)) %>%
    ggplot( aes(x = pesoBrutoPermissivelKg_binned, y = media_ConsumoEnergia, fill=pesoBrutoPermissivelKg
    geom_bar(stat = "identity") +
    labs(x = "peso bruto permissível (kg)", y = "Consumo de Energia Médio (KWh/100Km)") +
    theme(axis.text.x = element_text(angle = 45, hjust = 1), legend.position = "none")
```

```r
# Terceiro subplot: heatmap de correlação
cor_mat <- cor(dados %>% select(pesoBrutoPermissivelKg, mediaConsumoEnergiaKWh100Km), method = "spearma

# Transforme a matriz de correlação em um data.frame
cor_df <- melt(cor_mat)

# Plot a matriz de correlação como um gráfico de azulejos
p3 <- ggplot(cor_df, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile() +
  geom_text(aes(label = round(value, digits = 2)), color = "black", size = 4) + # Adiciona rótulos
  scale_fill_gradient2(low = "blue", mid = "white", high = "red", midpoint = 0) +
  theme(axis.text.x = element_blank(),  # Oculta os rótulos do eixo x
        axis.text.y = element_blank(),  # Oculta os rótulos do eixo y
        axis.title.x = element_blank(), # Oculta o título do eixo x
        axis.title.y = element_blank(), # Oculta o título do eixo y
        legend.position = "none")



# Combine os subplots em uma única plotagem usando a função cowplot::plot_grid
# Organização dos subplots lado a lado
grid.arrange(p1, p2, p3, ncol = 3)
```

Hipótese 2: Carros que possuem maior peso bruto permissivel possuem maior media de consumo de energia. Qual a correlação entre essas duas variáveis?

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



```r
dados <- subset(dados, select = -pesoBrutoPermissivelKg_binned)
rm(cuts, p1 ,p2 ,p3 , cor_mat, cor_df)
```

De acordo com os gráficos o coeficiente de correlação é positivo. Isso significa que, à medida que o peso bruto permissivel aumenta, o consumo medio de energia tende a aumentar.

```r
# Cálculo do coeficiente de correlação e valor-p
print(cor.test(dados$autonomiaWLTPkm, dados$mediaConsumoEnergiaKWh100Km, method = "kendall"))
```

Hipótese 3: Carros que possuem maior autonomia (km) possuem menor media de consumo de Energia (KWh/100km). Qual a correlação entre essas duas variáveis?

```
##
##  Kendall's rank correlation tau
##
## data:  dados$autonomiaWLTPkm and dados$mediaConsumoEnergiaKWh100Km
## z = 0.98981, p-value = 0.3223
## alternative hypothesis: true tau is not equal to 0
## sample estimates:
##        tau
## 0.09395487
```

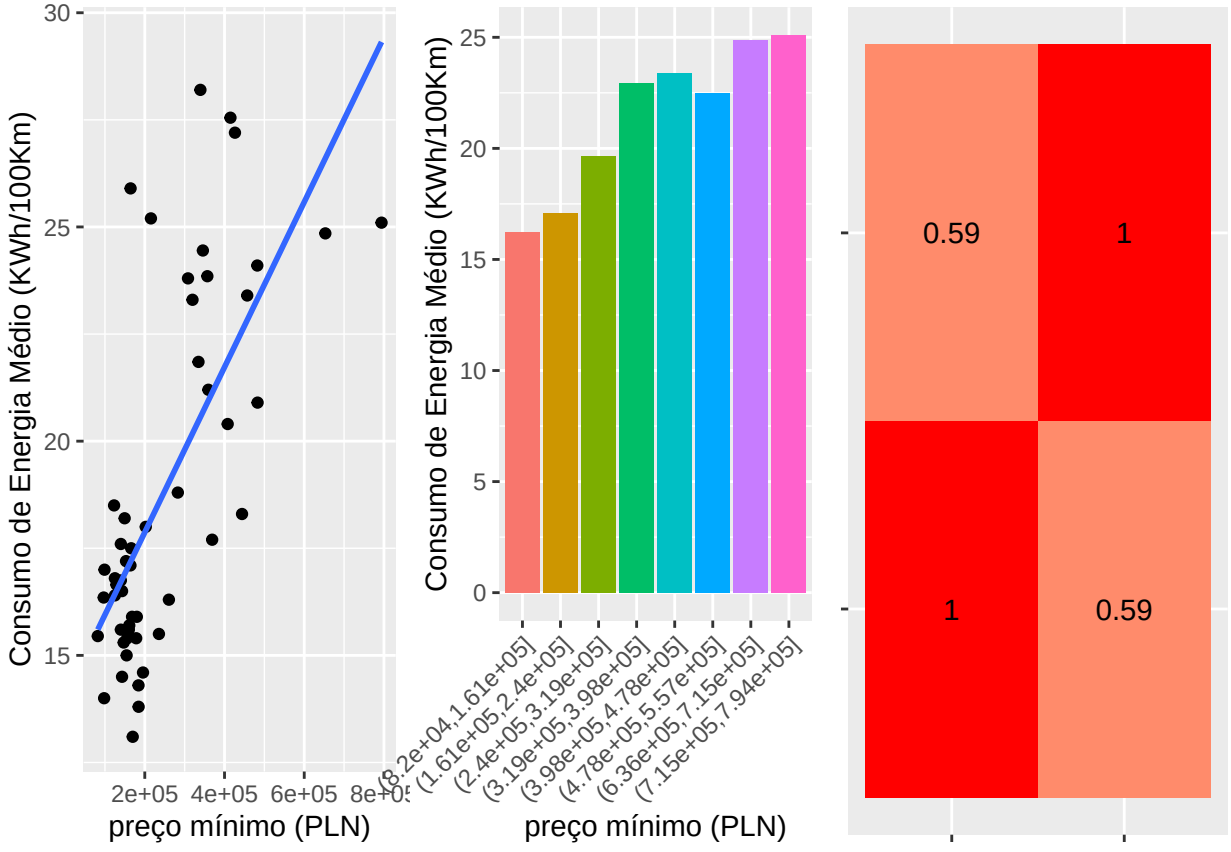
Inconclusivo. Como o valor p retornado é maior que 0.05, então a hipótese nula não pode ser rejeitada. Em outras palavras, não há evidências estatísticas suficientes para suportar a hipótese alternativa de que existe uma correlação significativa entre a autonomia e o consumo de energia.

```r
# Primeiro subplot: scatterplot
p1 <- ggplot(dados, aes(x = precoMinimoBrutoPLN, y = mediaConsumoEnergiaKWh100Km)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(x = "preço mínimo (PLN)", y = "Consumo de Energia Médio (KWh/100Km)")

# Segundo subplot: barplot
cuts <- seq(floor(min(dados$precoMinimoBrutoPLN)-1), ceiling(max(dados$precoMinimoBrutoPLN)), length.ou

dados$precoMinimoBrutoPLN_binned  <- cut(dados$precoMinimoBrutoPLN, breaks = cuts, order_result = ordere

p2 <- dados %>%
  group_by(precoMinimoBrutoPLN_binned) %>%
  summarise(media_ConsumoEnergia = mean(mediaConsumoEnergiaKWh100Km)) %>%
    ggplot( aes(x = precoMinimoBrutoPLN_binned, y = media_ConsumoEnergia, fill=precoMinimoBrutoPLN_binn
    geom_bar(stat = "identity") +
    labs(x = "preço mínimo (PLN)", y = "Consumo de Energia Médio (KWh/100Km)") +
    theme(axis.text.x = element_text(angle = 45, hjust = 1), legend.position = "none")


# Terceiro subplot: heatmap de correlação
cor_mat <- cor(dados %>% select(precoMinimoBrutoPLN, mediaConsumoEnergiaKWh100Km), method = "spearman")

# Transforme a matriz de correlação em um data.frame
cor_df <- melt(cor_mat)

# Plot a matriz de correlação como um gráfico de azulejos
```

```r
p3 <- ggplot(cor_df, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile() +
  geom_text(aes(label = round(value, digits = 2)), color = "black", size = 4) + # Adiciona rótulos
  scale_fill_gradient2(low = "blue", mid = "white", high = "red", midpoint = 0) +
  theme(axis.text.x = element_blank(),  # Oculta os rótulos do eixo x
        axis.text.y = element_blank(),  # Oculta os rótulos do eixo y
        axis.title.x = element_blank(), # Oculta o título do eixo x
        axis.title.y = element_blank(), # Oculta o título do eixo y
        legend.position = "none")



# Combine os subplots em uma única plotagem usando a função cowplot::plot_grid
# Organização dos subplots lado a lado
grid.arrange(p1, p2, p3, ncol = 3)
```

Hipótese 4: Carros elétricos mais caros possuam uma maior média de consumo de energia. Qual a correlação entre essas duas variáveis?

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




```r
dados <- subset(dados, select = -precoMinimoBrutoPLN_binned)
rm(cuts, p1 ,p2 ,p3 , cor_mat, cor_df)
```

De acordo com os gráficos o coeficiente de correlação é positivo. Isso significa que, à medida que o valor de uma variável aumenta, o valor da outra variável tende a aumentar.

Hipótese 5: Carros elétricos com tração nas quatro rodas (4WD) possuam uma maior média de consumo de energia. Qual a correlação entre essas duas variáveis? O teste ANOVA é usado para comparar as médias de grupos diferentes e determinar se há diferenças significativas entre eles.

As premissas do teste ANOVA são as seguintes:

1. Independência: As observações devem ser independentes umas das outras. Isso significa que os valores em uma condição não devem ser influenciados pelos valores em outras condições.
2. Normalidade: As distribuições dos dados dentro de cada grupo devem ser aproximadamente normais. Isso significa que os resíduos do modelo devem seguir uma distribuição normal.
3. Homogeneidade das variâncias: As variâncias dos grupos devem ser aproximadamente iguais. Isso significa que a variabilidade dos dados deve ser semelhante em cada grupo.

Sera considerada verdadeira a premissa 1.

```
# Teste de Normalidade Shapiro-Wilk em cada grupo de tração
# H0: Os dados são normalmente distribuídos
# H1: Os dados não são normalmente distribuídos

print(shapiro.test(dados$mediaConsumoEnergiaKWh100Km[dados$tipoTracao == '2WD (front)']) )
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dados$mediaConsumoEnergiaKWh100Km[dados$tipoTracao == "2WD (front)"]
## W = 0.70614, p-value = 1.216e-05
```

```
print(shapiro.test(dados$mediaConsumoEnergiaKWh100Km[dados$tipoTracao == '2WD (rear)']) )
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dados$mediaConsumoEnergiaKWh100Km[dados$tipoTracao == "2WD (rear)"]
## W = 0.97966, p-value = 0.9643
```

```
print(shapiro.test(dados$mediaConsumoEnergiaKWh100Km[dados$tipoTracao == '4WD']) )
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dados$mediaConsumoEnergiaKWh100Km[dados$tipoTracao == "4WD"]
## W = 0.94692, p-value = 0.3788
```

O p-value em todos os grupos é maior que 0,05, logo não háevidências suficientes para rejeitar a hipótese nula em cada um dos grupos.

```
# Validamos primeiro a suposição 2 usando o Teste de Levene

# Hipótese nula (H0): A variância é igual entre os grupos.
# Hipótese alternativa (HA): A variância é diferente entre os grupos.

leveneTest(mediaConsumoEnergiaKWh100Km ~ tipoTracao, data = dados)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value Pr(>F)
## group  2  0.9085 0.4097
##       50
```

O p-valor obtido é maior que 0.05, logo não há evidências suficientes para rejeitar a hipótese nula de igualdade de variâncias entre os grupos.

```
# Teste ANOVA
# H0: Não há diferença significativa na média de crescimento dos dentes entre os diferentes níveis/grup
# e suplemento.
# Ha: Há pelo menos uma diferença significativa na média de crescimento dos dentes entre os diferentes
# níveis de dose e suplemento.

aov(mediaConsumoEnergiaKWh100Km ~ tipoTracao, dados)
```

```
## Call:
##    aov(formula = mediaConsumoEnergiaKWh100Km ~ tipoTracao, data = dados)
##
## Terms:
##                 tipoTracao Residuals
## Sum of Squares    333.6757  554.8851
## Deg. of Freedom          2        50
##
## Residual standard error: 3.331321
## Estimated effects may be unbalanced
```

O valor p é menor que 0.05 indicando que há evidências estatísticas para rejeitar a hipótese nula de que não há diferenças significativas entre os grupos de tração. Assim, podemos concluir que há uma relação significativa entre o tipo de tração dos carros e a media de consumo de energia.

Hipótese 5: Carros elétricos com greios a disco nas 4 rodas possuam uma maior média de consumo de energia. Qual a correlação entre o tipo de freio e o consumo de nergia? Será aplicado o teste ANOVA

Sera considerada verdadeira a premissa de independência entre as amostra.

```
# Teste de Normalidade Shapiro-Wilk em cada grupo para validar a premissa de normalidade
# H0: Os dados são normalmente distribuídos
# H1: Os dados não são normalmente distribuídos

niveis <- levels(dados$tipoFreios)

print(shapiro.test(dados$mediaConsumoEnergiaKWh100Km[dados$tipoFreios == niveis[1]]) )
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dados$mediaConsumoEnergiaKWh100Km[dados$tipoFreios == niveis[1]]
## W = 0.90084, p-value = 0.0008774
```

```
print(shapiro.test(dados$mediaConsumoEnergiaKWh100Km[dados$tipoFreios == niveis[2]]) )
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dados$mediaConsumoEnergiaKWh100Km[dados$tipoFreios == niveis[2]]
## W = 0.98396, p-value = 0.9765
```

```
rm(niveis)
```

O p-value em um dos grupos é menor que 0,05, logo não há evidências suficientes para não rejeitar a hipótese nula em um dos grupos.

Como a premissa de normalidade não foi atendida a aplicação do tesste ANOVA não é apropriada. Assim será utilizado o teste não parámetrico de Kruskal-Wallis.

```
# Teste de Kruskal-Wallis
# HO (Hipótese Nula): As distribuições das variáveis nos diferentes grupos são iguais.
# HA (Hipótese Alternativa): Pelo menos uma das distribuições das variáveis nos grupos é diferente.

kruskal.test(mediaConsumoEnergiaKWh100Km ~ tipoFreios, data = dados)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  mediaConsumoEnergiaKWh100Km by tipoFreios
## Kruskal-Wallis chi-squared = 3.5782, df = 1, p-value = 0.05854
```

Como o valor de p (0.05854) é maior que o nível de significância comum de 0.05, não temos evidências estatisticamente significativas para rejeitar a hipótese nula. Isso significa que não há diferenças estatisticamente significativas nas medianas da variável mediaConsumoEnergiaKWh100Km entre os grupos definidos por tipoFreios.

Passo 5 - Preparação dos Dados

5.1 Divisão dos Dados em Conjunto de Treinamento e Teste

```
# Divisão em treino e teste
set.seed(123)  # Define uma semente para a reprodutibilidade
indexes <- createDataPartition(dados$mediaConsumoEnergiaKWh100Km, p = 0.8, list = FALSE)
train_data <- dados[indexes, ]
test_data <- dados[-indexes, ]

rm(indexes)
```

5.2 Normalização (Zscore)

De acordo com a análise univariada das variáveis numéricas (seção 3.1.2), há apenas duas variáveis com distribuição normal ou próxima de normal (autonomiaWLTPkm e aceleracao0a100KmHS).

```
# Cria o objeto de pré-processamento
preprocess_obj <- preProcess(train_data[, c('autonomiaWLTPkm', 'aceleracao0a100KmHS')], method = c("cen

# Aplica a transformação Zscore nos dados de treino
train_data_normalized_1 <- predict(preprocess_obj, train_data[, c('autonomiaWLTPkm', 'aceleracao0a100Km

# Aplica a transformação Zscore nos dados de teste
test_data_normalized_1 <- predict(preprocess_obj, test_data[, c('autonomiaWLTPkm', 'aceleracao0a100KmHS

rm(preprocess_obj)
```

5.3 Rescaling

A normalização Min-Max é mais adequada quando os dados não seguem uma distribuição normal.

```
# Obtém todos os nomes de colunas numéricas, exceto as especificadas
colunas_selecionadas <- setdiff(names(train_data %>% select_if(is.numeric)),
                                c('autonomiaWLTPkm', 'aceleracao0a100KmHS', 'mediaConsumoEnergiaKWh100Km
```

```r
# A normalização MinMax é mais recomendada em dados que apresentam outliers
preprocess_obj <- preProcess(train_data[, colunas_selecionadas], method = "range")

# Aplica a transformação MinMax nos dados de treino
train_data_normalized_2 <- predict(preprocess_obj, train_data[, colunas_selecionadas])

# Aplica a transformação MinMax nnos dados de teste
test_data_normalized_2 <- predict(preprocess_obj, test_data[, colunas_selecionadas])


rm(colunas_selecionadas, preprocess_obj)
```

```r
# Concatenando com as variáveis categoricas
train_data_normalized <- cbind(train_data_normalized_1,
                               train_data_normalized_2,
                               train_data %>% select(where(is.factor), mediaConsumoEnergiaKWh100Km))

test_data_normalized <- cbind(test_data_normalized_1,
                              test_data_normalized_2,
                              test_data %>% select(where(is.factor), mediaConsumoEnergiaKWh100Km))

rm(train_data_normalized_1, train_data_normalized_2)
rm(test_data_normalized_1, test_data_normalized_2)
rm(train_data, test_data)
```

Passo 6 - Seleção de Variáveis

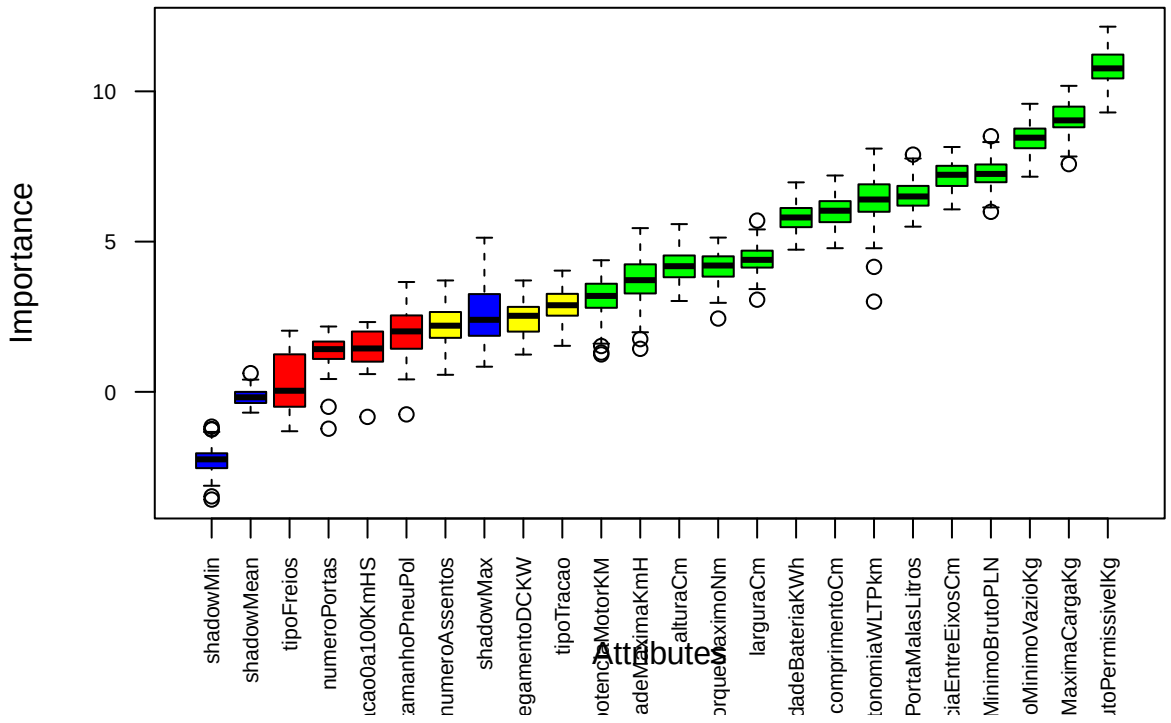
```r
# Separe as variáveis preditoras e a variável alvo
pred_vars <- select(train_data_normalized, -mediaConsumoEnergiaKWh100Km)
target_var <- train_data_normalized$mediaConsumoEnergiaKWh100Km

# Realize a seleção de variáveis usando o Boruta
boruta_obj <- Boruta(pred_vars, target_var)

# Acesse as variáveis confirmadas como importantes
important_variables <- names(boruta_obj$finalDecision[boruta_obj$finalDecision == "Confirmed"])

# Exiba as variáveis
print(important_variables)
```

```
##  [1] "autonomiaWLTPkm"            "precoMinimoBrutoPLN"
##  [3] "potenciaMotorKM"            "torqueMaximoNm"
##  [5] "capacidadeBateriaKWh"       "distanciaEntreEixosCm"
##  [7] "comprimentoCm"              "larguraCm"
##  [9] "alturaCm"                   "pesoMinimoVazioKg"
## [11] "pesoBrutoPermissivelKg"     "capacidadeMaximaCargaKg"
## [13] "velocidadeMaximaKmH"        "capacidadePortaMalasLitros"
```

```r
rm(pred_vars, target_var)
```

Serão utilizados as 13 variáveis mais importantes identificadas pelo algoritmo Boruta inicialmente.

```r
# Visualize as variáveis selecionadas pelo Boruta
plot(boruta_obj, cex.axis = 0.7, las = 2)
```




```
rm(boruta_obj)
```

```
X_train <- train_data_normalized[, important_variables]
y_train <- train_data_normalized$mediaConsumoEnergiaKWh100Km

X_test <- test_data_normalized[,important_variables]
y_test <- test_data_normalized$mediaConsumoEnergiaKWh100Km
```

Passo 07 - Modelagem de Machine Learning

```
# Função auxiliar para o cálculos das métricas de performance para cada modelo
ml_error <- function( model_name, y, y_hat, R2){
# k = número de variáveis independentes
  e <- y- y_hat
  MAE <- mean(abs(e))
  MAPE <- mean(abs((e)/y)*100)
  MSE <- mean(e^2)
  RMSE <- sqrt(MSE)

  return(data.frame('Model Name' = model_name, MAE = MAE, MAPE = MAPE, MSE = MSE, RMSE = RMSE))
}
```

7.1 Modelo de Regressão Linear

```
# model
modelo_lr <- lm(y_train ~ ., data = X_train)

# prediction
yhat_lr = predict(modelo_lr, newdata = X_test)
```

```r
# performance
lr_result = ml_error( 'Linear Regression', y_test, yhat_lr)
lr_result
```

```
##          Model.Name      MAE     MAPE      MSE    RMSE
## 1 Linear Regression 2.076462 10.73705 9.226348 3.03749
```

7.2 Modelo de Regressão Linear - Ridge

```r
# model
modelo_lrr <- glmnet(as.matrix(X_train), y_train, alpha = 1)

# prediction
yhat_lrr = predict(modelo_lrr, newx = as.matrix(X_test), s= 0.01)

# performance
lrr_result = ml_error( 'Linear Regression - Ridge', y_test, yhat_lrr)
lrr_result
```

```
##                  Model.Name      MAE     MAPE      MSE     RMSE
## 1 Linear Regression - Ridge 1.975674 10.28498 8.206758 2.864744
```

7.3 Modelo de Árvore de Decisão

```r
# Construir o modelo de regressão com rpart
modelo_ad <- rpart(y_train ~ ., data = X_train)

# prediction
yhat_ad= predict(modelo_ad, newdata = X_test)

# performance
ad_result = ml_error( 'Decision Tree', y_test, yhat_ad)
ad_result
```

```
##      Model.Name      MAE     MAPE      MSE     RMSE
## 1 Decision Tree 1.617262 7.501639 8.896838 2.982757
```

7.4 Modelo Random Forest

```r
modelo_rf <- randomForest(y_train ~ ., data = X_train)

# prediction
yhat_rf= predict(modelo_rf, newdata = X_test)

# performance
rf_result = ml_error( 'Random Forest', y_test, yhat_rf)
rf_result
```

```
##      Model.Name     MAE     MAPE      MSE     RMSE
## 1 Random Forest 1.87033 9.079062 8.283471 2.878102
```

7.5 Modelo SVM

```
# Treinamento do modelo SVM com valores padrão
modelo_svm <- svm(y_train ~ ., data = X_train,
                  type = 'eps-regression',
                  kernel = 'linear',
                  C = 1,
                  epsilon = 0.1)

# Fazer previsões nos dados de treinamento
yhat_svm <- predict(modelo_svm, X_test)

# performance
svm_result = ml_error( 'SVM', y_test, yhat_svm)
svm_result
```

```
##   Model.Name      MAE     MAPE     MSE     RMSE
## 1        SVM 2.073125 10.77034 10.0009 3.162419
```

7.6 Comparação de Desempenho

```
resultados <- rbind(lr_result, lrr_result, ad_result, rf_result, svm_result)
resultados[order(resultados$RMSE), ]
```

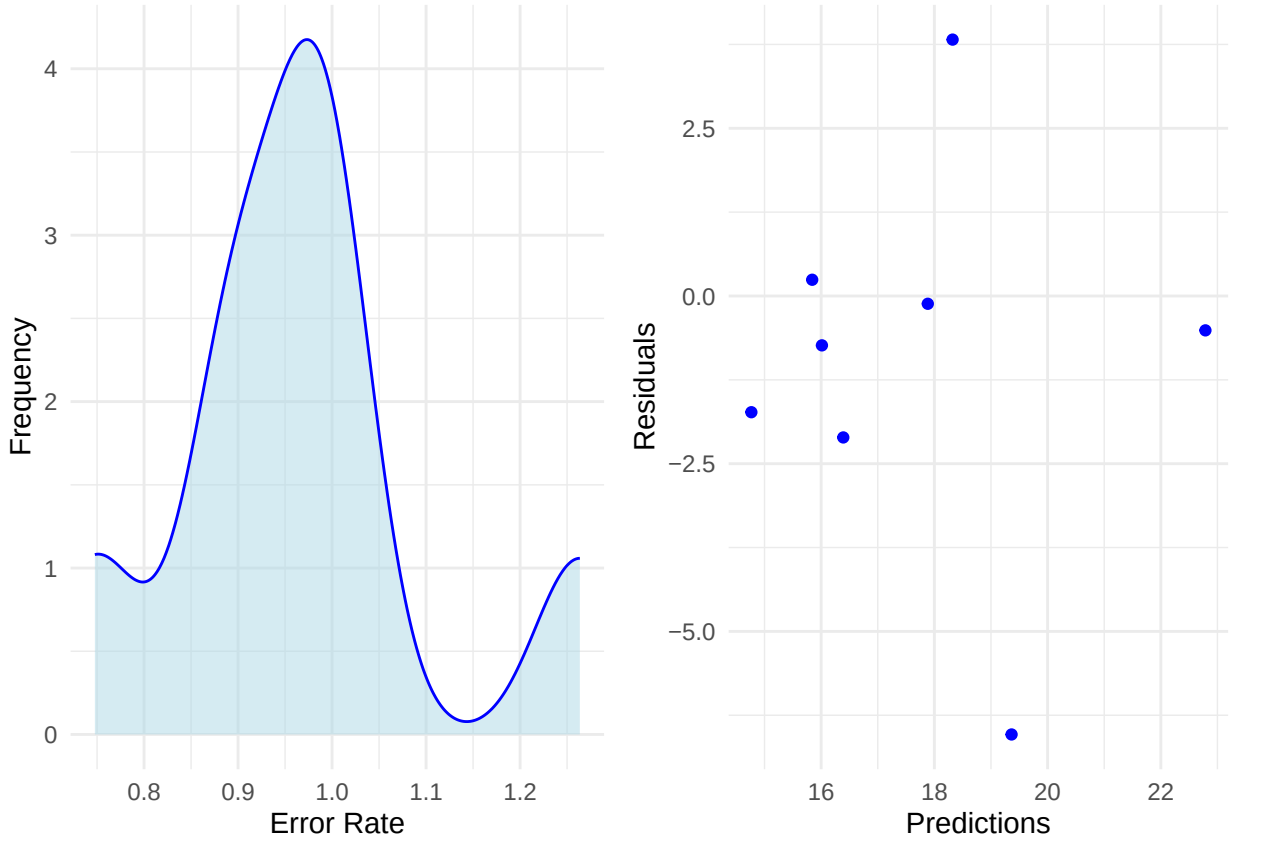
```
##                  Model.Name      MAE      MAPE       MSE     RMSE
## 2 Linear Regression - Ridge 1.975674 10.284977  8.206758 2.864744
## 4             Random Forest 1.870330  9.079062  8.283471 2.878102
## 3             Decision Tree 1.617262  7.501639  8.896838 2.982757
## 1         Linear Regression 2.076462 10.737045  9.226348 3.037490
## 5                       SVM 2.073125 10.770340 10.000897 3.162419
```

O modelo Linear Regression com regularização Ridge apresenta o menor RMSE. A métrica RMSE atribui maior peso aos erros maiores, sendo aconselhavel para selecionar modelos> selecionar o melho modelos.

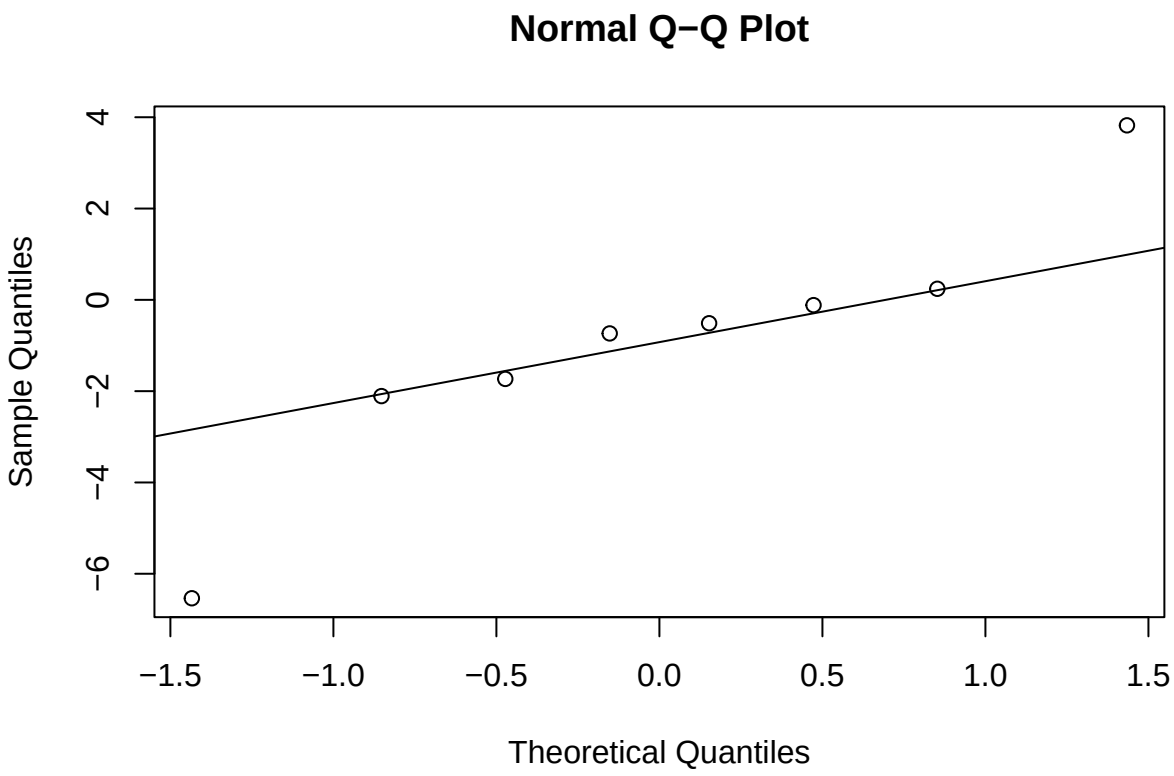
Passo 08 - Tradução e Interpretação do Erro

```
resultadoFinal <- as.data.frame(cbind(y_test, yhat_lrr))
names(resultadoFinal) <- c('mediaConsumoEnergiaKWh100Km', 'predictions')


resultadoFinal <-  mutate(resultadoFinal, residuals = predictions - mediaConsumoEnergiaKWh100Km,
                                          error_rate = predictions/mediaConsumoEnergiaKWh100Km)

# Plotando os resíduos
p1 <- ggplot(resultadoFinal, aes(x = error_rate)) +
  #geom_histogram(aes(y = after_stat(density)), binwidth = 1, fill = "steelblue", color = "black") +
  geom_density(fill = "lightblue", alpha = 0.5, color = "blue") +
  labs(x = "Error Rate", y = "Frequency") +
  theme_minimal()

p2 <- ggplot(resultadoFinal, aes(x = predictions, y = residuals)) +
  geom_point(color = "blue") +
  labs(x = "Predictions", y = "Residuals") +
  theme_minimal()
```

```
# Combine os subplots em uma única plotagem usando a função cowplot::plot_grid
# Organização dos subplots lado a lado
grid.arrange(p1, p2, ncol = 2)
```




```
qqnorm(resultadoFinal$residuals)
qqline(resultadoFinal$residuals)
```


Normal Q−Q Plot
Sample Quantiles
Theoretical Quantiles
4
2
0
−2
−4
−6
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5