

# Previsão de Consumo de Energia de Carros Elétricos com Machine Learning

## Descrição e Introdução ao Problema de Negócio

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, nós devemos construir 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 o número de motores elétricos, o peso do veículo, a capacidade de carga, entre outros atributos.

Para a construção deste projeto, utilizaremos as linguagens R.

O dataset se encontra no link: <https://data.mendeley.com/datasets/tb9yrptydn/2>

Este conjunto de dados lista todos os carros totalmente elétricos com seus atributos (propriedades) disponíveis atualmente no mercado. 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 veículos elétricos) do veículo, incluindo os diferentes métodos de carregamento.

O conjunto de dados inclui carros que, a partir de 2 de dezembro de 2020, poderiam ser adquiridos na Polônia como novos e em revendedor autorizado e aqueles disponíveis em pré-venda pública e geral, mas apenas se uma lista de preços publicamente disponível. A lista não inclui carros descontinuados que não podem ser adquiridos como novos de um revendedor autorizado (também quando não estão disponíveis em estoque).

O conjunto de dados de carros elétricos inclui todos os carros totalmente elétricos no mercado primário que foram obtidos de materiais oficiais (especificações técnicas e catálogos) fornecidos por fabricantes de automóveis com licença para vender carros na Polônia.

Esses materiais foram baixados de seus sites oficiais. Caso os dados fornecidos pelo fabricante estivessem incompletos, as informações eram complementadas com dados do AutoCatálogo SAMAR.

Nosso trabalho é construir um modelo de ML capaz de prever o consumo de energia de veículos elétricos.

## Dicionário de Dados

Nome da Variável	Descrição
Car Full Name	Nome completo do Veículo
Make	Fabricante do veículo
Model	Modelo do Veículo
Minimal price	Preço Mínimo de comercialização do veículo em [U\$]
Engine power	Potência Declarada pelo fabricante do motor do veículo, medida em [kW]
Maximum torque	Torque Máximo do motor medido em [Nm]
Type of brakes	Tipo de Sistema de Freios
Drive type	Tipo de Tração
Battery capacity	Capacidade de carga da bateria em [kWh]
Range	Autonomia da bateria em [km]
Wheelbase	Distância entre eixos em [cm]
Length	Comprimento do Veículo em [cm]
Width	Largura do Veículo em [cm]
Height	Altura do Veículo em [cm]
Minimal Empty weight	Peso em vazio medido em [kg]
Permissible gross weight	Peso máximo permitido medido em [kg]
Maximum load capacity	Capacidade máxima de carga medida em [kg]
Number of seats	Número de assentos do veículo
Number of doors	Número de portas do veículo
Tire size	Tamanho do raio dos pneus medido em [in] polegadas
Maximum speed	Velocidade máxima medida em [km/h]
Boot capacity	Litragem do porta-malas medida em [l]
Acceleration 0-100	Tempo de aceleração de 0-100 [s]
Maximum DC charging power	Capacidade máxima de carga da bateria medida em [kW]
Mean Energy Consumption [kWh/100 km]	Consumo médio de bateria medida em kWh por 100 km percorridos

## Pacotes Utilizados

Utilizando os dados do dataset, temos como objetivo prever o consumo de energia elétrica dos modelos da base e identificar insights importantes e possíveis oportunidades de melhoria.

```
# Pacotes utilizados
library(readxl)
library(thinkr)
library(usefun)
library(Amelia)
```

## Carregando os Dados e Ajustando o Dataset

```
# Carregando e Ajustando o Dataset
getwd()
?read_xlsx

dadosv00 <- read_xlsx('dados/fev_dataset.xlsx', na = 'NaN')

View(dadosv00)
dim(dadosv00)
str(dadosv00)
```

```
> str(dadosv00)
tibble [53 x 25] (S3: tbl_df/tbl/data.frame)
 $ Car full name      : chr [1:53] "Audi e-tron 55 quattro" "Audi e-tron 50 quattro" "Audi e-tron S quattro" "Audi e-tron Sportback 50 quattro" ...
 $ Make              : chr [1:53] "Audi" "Audi" "Audi" "Audi" ...
 $ Model            : chr [1:53] "e-tron 55 quattro" "e-tron 50 quattro" "e-tron S quattro" "e-tron Sportback 50 quattro" ...
 $ Minimal price (gross) [PLN] : num [1:53] 345700 300400 414900 319700 357000 ...
 $ Engine power [kW] : num [1:53] 360 313 503 313 360 503 170 184 286 136 ...
 $ Maximum torque [Nm] : num [1:53] 664 540 973 540 664 973 250 270 400 260 ...
 $ Type of brakes     : chr [1:53] "disc (front + rear)" "disc (front + rear)" "disc (front + rear)" "disc (front + rear)" ...
 $ Drive type        : chr [1:53] "4WD" "4WD" "4WD" "4WD" ...
 $ Battery capacity [kWh] : num [1:53] 95 71 95 71 95 95 42.2 42.2 80 50 ...
 $ Range (WLTP) [km] : num [1:53] 438 340 364 346 447 369 359 345 460 350 ...
 $ Wheelbase [cm] : num [1:53] 293 293 293 293 293 ...
 $ Length [cm] : num [1:53] 490 490 490 490 490 ...
 $ Width [cm] : num [1:53] 194 194 198 194 194 ...
 $ Height [cm] : num [1:53] 163 163 163 162 162 ...
 $ Minimal empty weight [kg] : num [1:53] 2565 2445 2695 2445 2595 ...
 $ Permissible gross weight [kg] : num [1:53] 3130 3040 3130 3040 3130 ...
 $ Maximum load capacity [kg] : num [1:53] 640 670 565 640 670 565 440 440 540 459 ...
 $ Number of seats : num [1:53] 5 5 5 5 5 4 4 5 5 ...
 $ Number of doors : num [1:53] 5 5 5 5 5 5 5 5 ...
 $ Tire size [in] : num [1:53] 19 19 20 19 19 20 19 20 19 16 ...
 $ Maximum speed [kph] : num [1:53] 200 190 210 190 200 210 160 160 180 150 ...
 $ Boot capacity (VDA) [l] : num [1:53] 660 660 660 615 615 615 260 260 510 380 ...
 $ Acceleration 0-100 kph [s] : num [1:53] 5.7 6.8 4.5 6.8 5.7 4.5 8.1 6.9 6.8 9.5 ...
 $ Maximum DC charging power [kW] : num [1:53] 150 150 150 150 150 150 50 50 150 100 ...
 $ mean - Energy consumption [kWh/100 km]: num [1:53] 24.4 23.8 27.6 23.3 23.9 ...
```

```
# Convertendo o objeto para DataFrame
dadosv00 <- as.data.frame(dadosv00)

# Verificando Dados NaN
summary(is.na(dadosv00))
colSums(is.na(dadosv00))
missmap(dadosv00)
```

```
> summary(is.na(dadosv00))
Car full name      Make      Model      Minimal price (gross) [PLN] Engine power [KW]
Mode :logical      Mode :logical  Mode :logical  Mode :logical      Mode :logical
FALSE:53           FALSE:53      FALSE:53      FALSE:53           FALSE:53

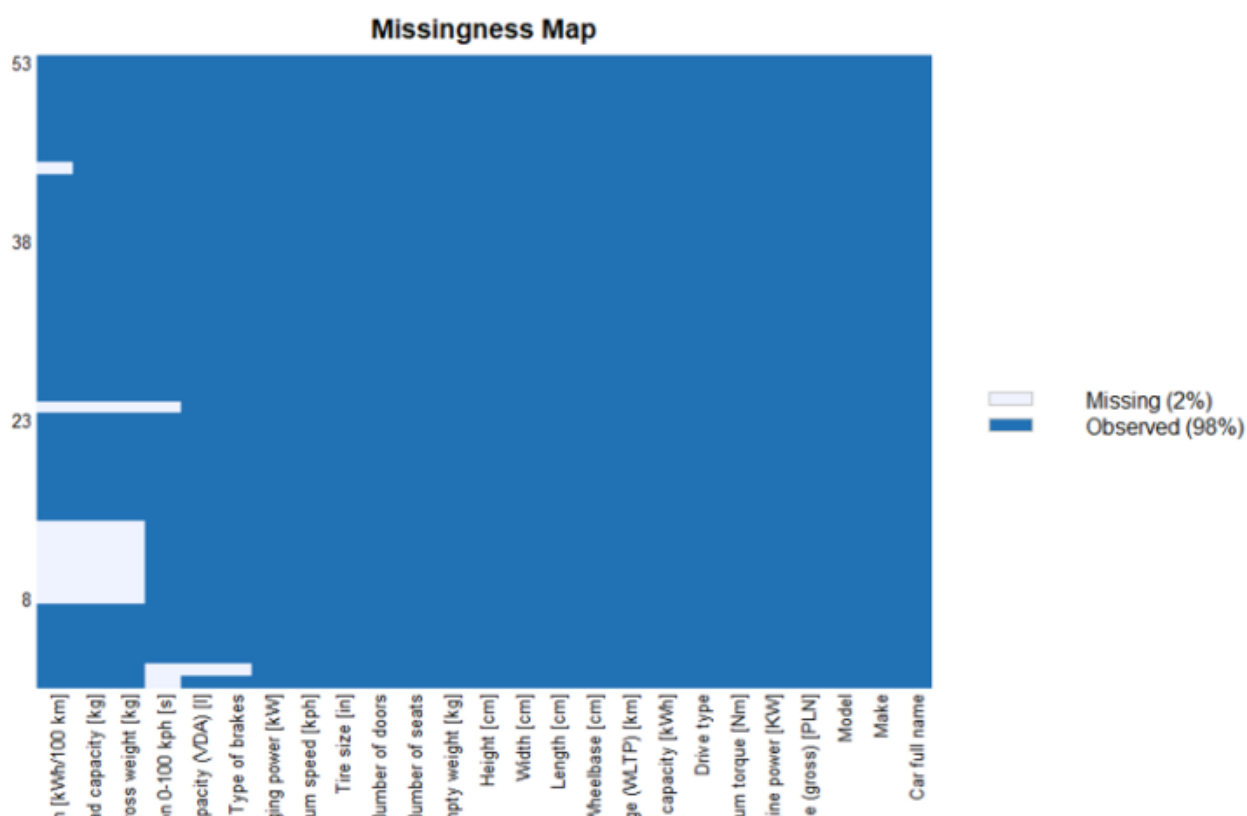
Maximum torque [Nm] Type of brakes Drive type      Battery capacity [kWh] Range (WLTP) [km]
Mode :logical      Mode :logical  Mode :logical  Mode :logical      Mode :logical
FALSE:53           FALSE:52      FALSE:53      FALSE:53           FALSE:53
                        TRUE :1

Wheelbase [cm] Length [cm] Width [cm] Height [cm] Minimal empty weight [kg]
Mode :logical      Mode :logical  Mode :logical  Mode :logical      Mode :logical
FALSE:53           FALSE:53      FALSE:53      FALSE:53           FALSE:53

Permissible gross weight [kg] Maximum load capacity [kg] Number of seats Number of doors
Mode :logical           Mode :logical      Mode :logical  Mode :logical
FALSE:45                FALSE:45           FALSE:53      FALSE:53
TRUE :8                 TRUE :8

Tire size [in] Maximum speed [kph] Boot capacity (VDA) [l] Acceleration 0-100 kph [s]
Mode :logical      Mode :logical      Mode :logical  Mode :logical
FALSE:53           FALSE:53           FALSE:52      FALSE:50
                        TRUE :1                 TRUE :3

Maximum DC charging power [kW] mean - Energy consumption [kWh/100 km]
Mode :logical           Mode :logical
FALSE:53                FALSE:44
                        TRUE :9
```

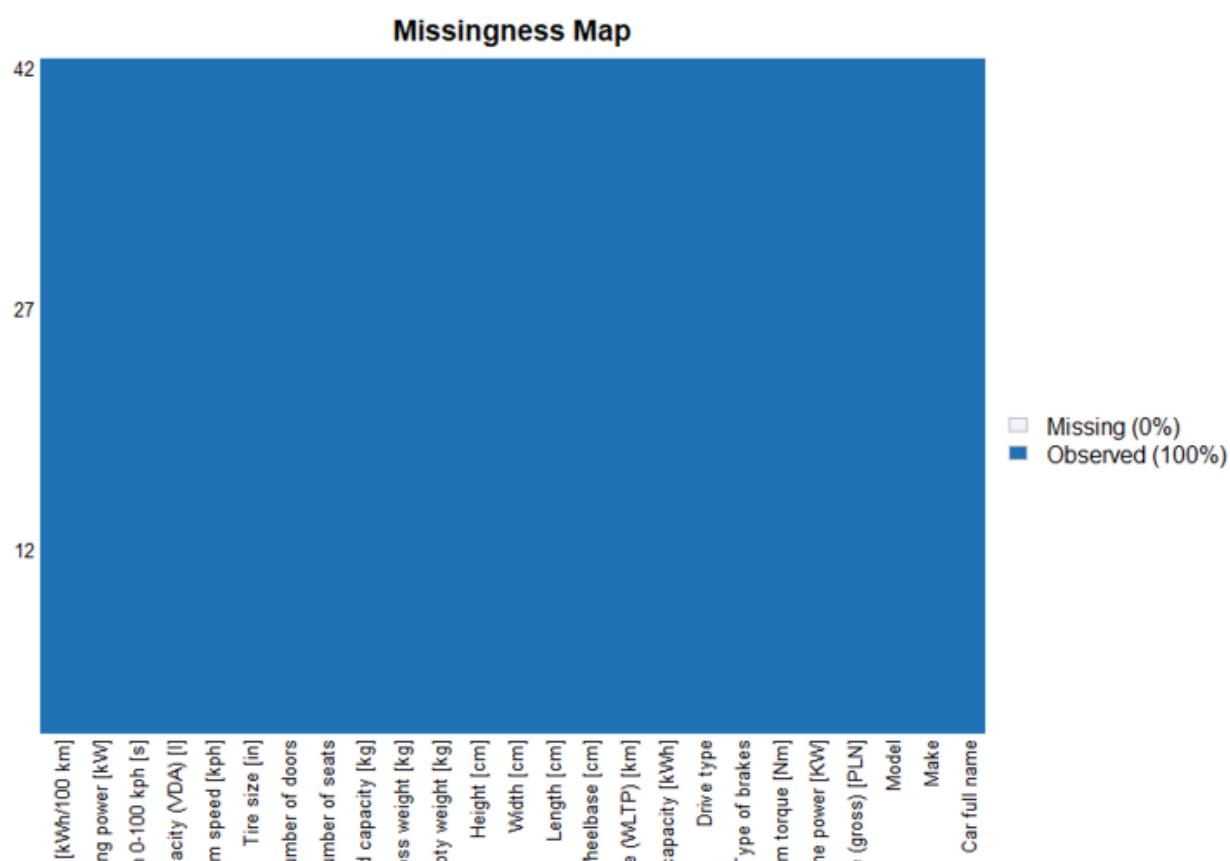


Temos algumas linhas vazias no dataset, especificamente na coluna Média de Consumo, sendo esta nossa variável resposta. Vamos apenas excluir esses dados incompletos para não tendenciar o erro de previsão durante a nossa modelagem preditiva. Poderíamos ter aplicado

alguma técnica de imputação, mas neste primeiro momento vamos prosseguir apenas sem os dados incompletos e verificar quais resultados podemos alcançar.

```
# Eliminando os dados Na
dadosv01 <- na.omit(dadosv00)

summary(is.na(dadosv01))
colSums(is.na(dadosv01))
missmap(dadosv01)
```



```
# Renomeando as Colunas
nomesCol <- c('Carro', 'Fabricante', 'Modelo', 'PrecoMin', 'Potencia',
              'TorqMax', 'Freios', 'Cambio', 'CapBat', 'Autonomia',
              'DistEixos', 'Comprimento', 'Largura', 'Altura', 'PesoVazio',
              'PesoCheio', 'CapMax', 'NumAssentos', 'NumeroPortas',
              'TampNeu', 'VelMax', 'BootCap', 'Acc', 'CapMaxBat',
              'ConsMedio')
colnames(dadosv01) <- nomesCol
View(dadosv01)
```

Carro	Fabricante	Modelo	PrecoMin	Potencia	TorqMax	Freios	Can
Audi e-tron 55 quattro	Audi	e-tron 55 quattro	345700	360	664	disc (front + rear)	4Wl
Audi e-tron 50 quattro	Audi	e-tron 50 quattro	308400	313	540	disc (front + rear)	4Wl
Audi e-tron S quattro	Audi	e-tron S quattro	414900	503	973	disc (front + rear)	4Wl
Audi e-tron Sportback 50 quattro	Audi	e-tron Sportback 50 quattro	319700	313	540	disc (front + rear)	4Wl
Audi e-tron Sportback 55 quattro	Audi	e-tron Sportback 55 quattro	357000	360	664	disc (front + rear)	4Wl
Audi e-tron Sportback S quattro	Audi	e-tron Sportback S quattro	426200	503	973	disc (front + rear)	4Wl
BMW i3	BMW	i3	169700	170	250	disc (front + rear)	2Wl
BMW i3s	BMW	i3s	184200	184	270	disc (front + rear)	2Wl

## Realizando Label Encoding das Variáveis Categóricas

Como pretendemos criar um modelo de regressão, precisamos identificar quais variáveis categóricas vamos transformar em numéricas.

Temos como Variáveis Categóricas: Carro, Fabricante, Modelo, Freios e Cambio. Vamos descartar Modelo e Carro, pois não são pertinentes a análise da variável resposta, conforme orientado pela área de negócio.

```
# Realizando Label Encoding das Variáveis Categóricas

dadosv01$Fabricante <- as.numeric(as.factor(dadosv01$Fabricante))

dadosv01$Freios <- as.numeric(as.factor(dadosv01$Freios))

dadosv01$Cambio <- as.numeric(as.factor(dadosv01$Cambio))

str(dadosv01)
```

## Dicionário da Variável Fabricante

- Audi = 1
- BMW = 2
- Citroen = 3
- DS = 4
- Honda = 5
- Hyundai = 6
- Jaguar = 7
- Kia = 8
- Mazda = 9
- Mercedes-Benz = 10
- Mini = 11
- Nissan = 12
- Opel = 13
- Peugeot = 14
- Porsche = 15
- Renault = 16
- Skoda = 17
- Smart = 18
- Volkswagen = 19

## Dicionário da Variável Fabricante

- Freio a Disco nas 4 rodas = 1
- Freio a Disco na Dianteira e Tambor na traseira = 2

## Dicionário da Variável Fabricante

- 4WD = 3
- 2WD(rear) = 2
- 2WD(front) = 1

## Salvando Dataset Modificado em Disco

```
file = 'dados/dados_ajustados.csv'
save_df_to_file(dadosv01, file = file)
```

Neste momento estamos prontos para iniciar nossas análises exploratórias e preparar nosso dataset para construção dos modelos de Machine Learning.

## Análise Exploratória

```
# Pacotes
```

```
library(kim)
library(plotly)
library(ggplot2)
library(dplyr)
library(corrplot)
library(GGally)
library(gridExtra)
```

```
# Verificando e ajustando os tipos de variáveis
```

```
dados <- as.data.frame(read_csv(name = 'dados_ajustados', head = TRUE,
                                dirname = 'dados' ))
```

```
dados <- dados[, -1]
dados$Fabricante <- as.factor(dados$Fabricante)
dados$Freios <- as.factor(dados$Freios)
dados$Cambio <- as.factor(dados$Cambio)
```

```
str(dados)
View(dados)
```

```
'data.frame': 42 obs. of 25 variables:
 $ Carro : chr "Audi e-tron 55 quattro" "Audi e-tron 50 quattro" "Audi e-tron Sportback 50 quattro" ...
 $ Fabricante : Factor w/ 19 levels "1","2","3","4",...: 1 1 1 1 1 1 2 2
 $ Modelo : chr "e-tron 55 quattro" "e-tron 50 quattro" "e-tron Sportback 50 quattro" ...
 $ PrecoMin : int 345700 308400 414900 319700 357000 426200 169700 180000 ...
 $ Potencia : int 360 313 503 313 360 503 170 184 286 136 ...
 $ TorqMax : int 664 540 973 540 664 973 250 270 400 260 ...
 $ Freios : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
 $ Cambio : Factor w/ 3 levels "1","2","3": 3 3 3 3 3 3 2 2 2 1 ...
 $ CapBat : num 95 71 95 71 95 95 42.2 42.2 80 50 ...
 $ Autonomia : int 438 340 364 346 447 369 359 345 460 320 ...
 $ DistEixos : num 293 293 293 293 293 ...
 $ Comprimento : num 490 490 490 490 490 ...
 $ Largura : num 194 194 198 194 194 ...
 $ Altura : num 163 163 163 162 162 ...
 $ PesoVazio : int 2565 2445 2695 2445 2595 2695 1440 1460 2260 1523 ...
 $ PesoCheio : int 3130 3040 3130 3040 3130 3130 1730 1730 2725 1975 ...
 $ CapMax : int 640 670 565 640 670 565 440 440 540 450 ...
 $ NumAssentos : int 5 5 5 5 5 5 4 4 5 5 ...
 $ NumeroPortas : int 5 5 5 5 5 5 5 5 5 5 ...
 $ TamPneu : int 19 19 20 19 19 20 19 20 19 17 ...
 $ VelMax : int 200 190 210 190 200 210 160 160 180 150 ...
 $ BootCap : int 660 660 660 615 615 615 260 260 510 350 ...
 $ Acc : num 5.7 6.8 4.5 6.8 5.7 4.5 8.1 6.9 6.8 8.7 ...
 $ CapMaxBat : int 150 150 150 150 150 150 50 50 150 100 ...
 $ ConsMedio : num 24.4 23.8 27.6 23.3 23.9 ...
```



Iniciamos nossa Análise Exploratória analisando um sumário geral dos dados, porém como temos muitas variáveis, é mais prático analisarmos de forma separada Variáveis Numéricas, Categóricas e a Variável Resposta.

```
# Análise Exploratória dos Dados

summary(dados)

# Podemos reparar que a Média e a Mediana possuem valores próximos em todas
# as variáveis numéricas, o que nos indica uma possível distribuição normal.

# Vamos separar as Variáveis em Numéricas, Categóricas e Resposta para
# analisarmos de forma separada.

VarNum <- dados %>% select(c(4:6), c(9:24))
VarCat <- dados %>% select(c(1:3), c(7:8))
VarResp <- dados$ConsMedio
```

## Análise da Variável Resposta

```
# Variável Resposta

summary(VarResp) # Min. 13.10 / 1st Qu.15.60 / Median 16.88 / Mean 18.61
# 3rd Qu. 22.94 / Max. 27.55

diff(range(VarResp)) # 14.45

sd(VarResp) # 4.134293

var(VarResp) # 17.09238

hist(VarResp, main = 'Distribuição do Consumo Médio em Carros Elétricos',
     xlab = 'Consumo de Energia [kW/h]')
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
13.10	15.60	16.88	18.61	22.94	27.55

Logo de início percebemos que a mediana é inferior a Média dos Dados, o que mostra que nossa curva está deslocada à direita (região caudal), provavelmente com um Skewness positivo.

```
> skewness(VarResp)
[1] 0.7857014
```

A distância máxima do range é de 14,45, conforme podemos verificar:

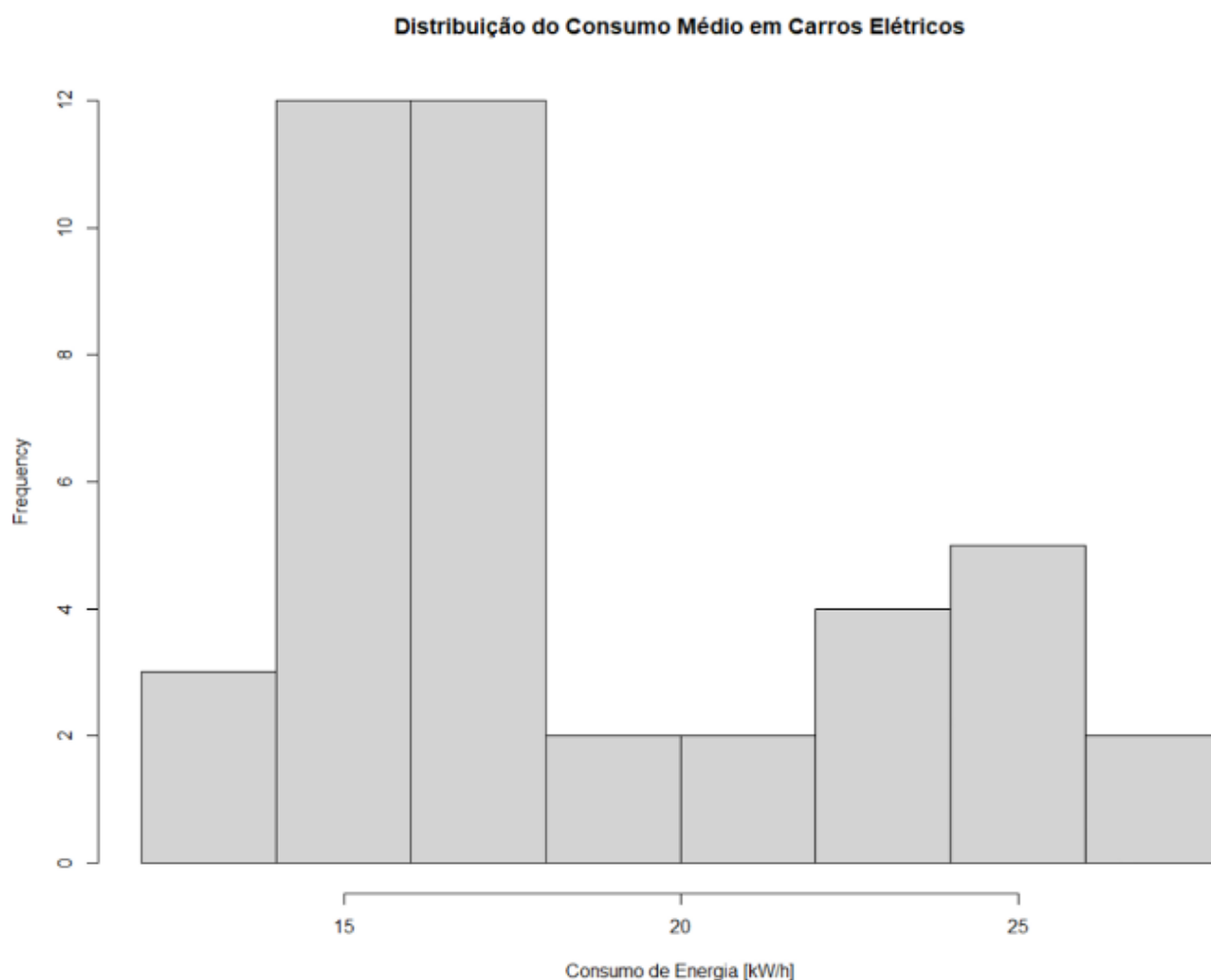
```
> diff(range(VarResp)) # 14.45
[1] 14.45
```

O que nos mostra que a maior diferença entre os modelos não ultrapassa 15 kW/h em consumo.

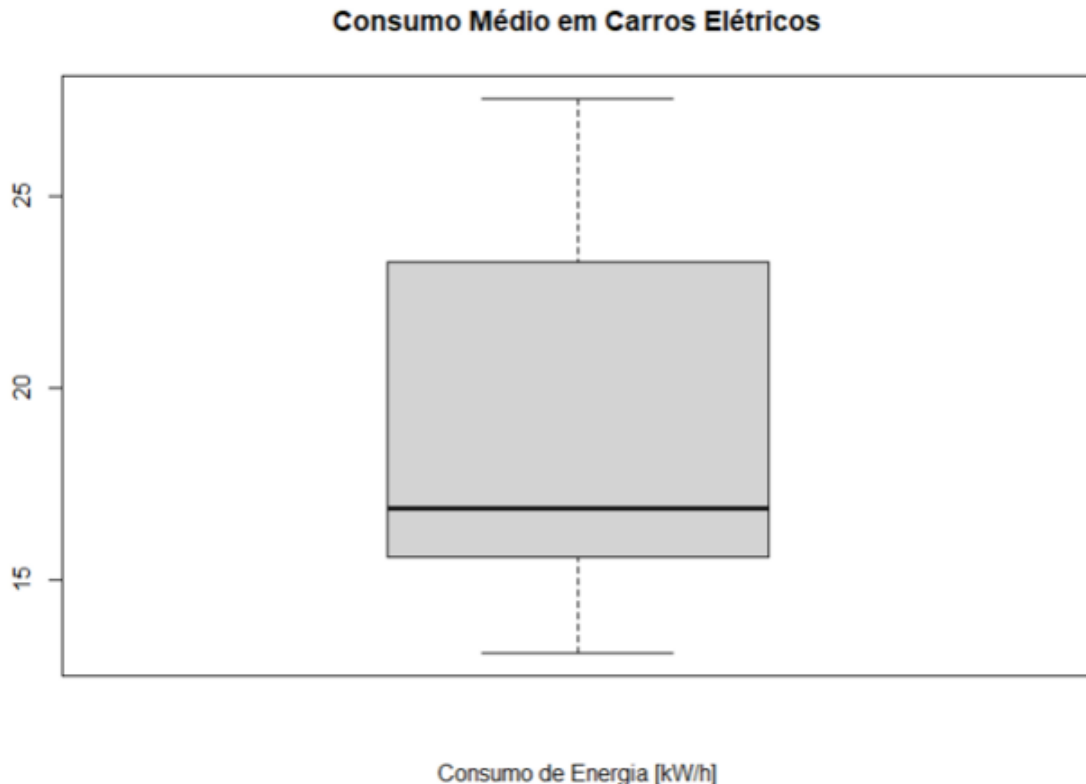
Nosso desvio padrão é de 4,13 kW/h e a variância de 17,09.

```
> sd(VarResp) # 4.134293
[1] 4.134293
>
> var(VarResp) # 17.09238
[1] 17.09238
```

Podemos ver a questão do Skewness no histograma abaixo:



Temos alguns dados discrepantes, com valores muito acima da mediana dos dados, talvez um outlier, o que podemos observar com a construção de um boxplot.



Observamos que não temos dados outliers, mantendo assim os dados com um padrão razoável para serem analisados.

Não podemos afirmar que temos uma distribuição normal nos dados, mas podemos realizar um teste de Shapiro e verificar se existe normalidade na distribuição dos dados.

- $H_0 \rightarrow$  Consideramos que os dados são normalmente distribuídos
- $H_1 \rightarrow$  Não podemos considerar que os dados são normalmente distribuídos
  - Se  $p\text{-value} > 0.05$  não podemos rejeitar  $H_0$
  - Se  $p\text{-value} < 0.05$  então rejeitamos  $H_0$ .

```
shapiro.test(VarResp)
```

Shapiro-Wilk normality test

```
data: VarResp  
W = 0.86663, p-value = 0.0001665
```

Podemos perceber que p-value é menor que 0.05, sendo assim, não podemos considerar que os dados da Variável resposta seguem uma distribuição Normal, ou seja, rejeitamos H0.

## Análise das Variáveis Numéricas

Inicialmente, vamos verificar um resumo de cada variável:

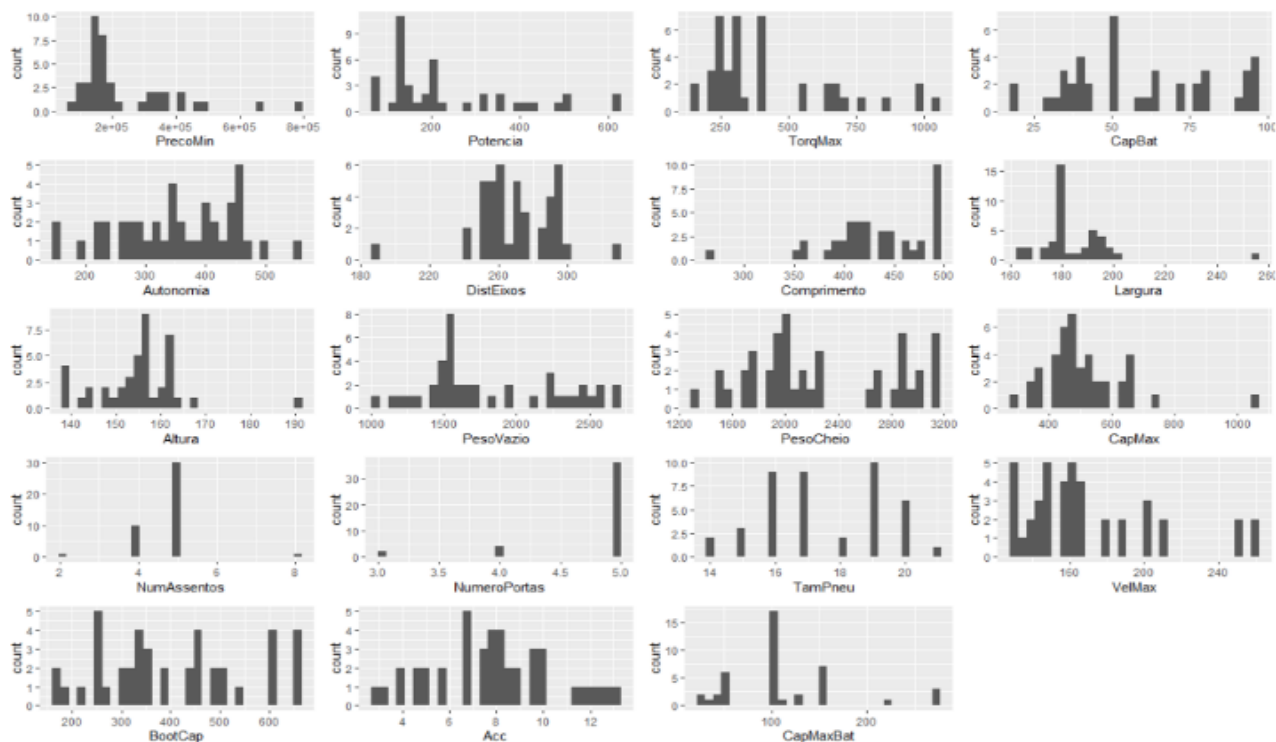
```
> summary(VarNum)
      PrecoMin      Potencia      TorqMax      CapBat      Autonomia
Min.   : 82050   Min.   : 82.0   Min.   : 160.0   Min.   :17.60   Min.   :148.0
1st Qu.:140650   1st Qu.:136.0   1st Qu.: 260.0   1st Qu.:39.20   1st Qu.:279.2
Median :166945   Median :184.0   Median : 317.5   Median :52.00   Median :352.5
Mean   :235066   Mean   :237.7   Mean   : 425.2   Mean   :58.84   Mean   :351.7
3rd Qu.:316875   3rd Qu.:313.0   3rd Qu.: 540.0   3rd Qu.:78.65   3rd Qu.:434.8
Max.   :794000   Max.   :625.0   Max.   :1050.0   Max.   :95.00   Max.   :549.0
      DistEixos      Comprimento      Largura      Altura      PesoVazio      PesoCheio
Min.   :187.3   Min.   :269.5   Min.   :164.5   Min.   :137.8   Min.   :1035   Min.   :1310
1st Qu.:256.3   1st Qu.:406.6   1st Qu.:178.7   1st Qu.:151.2   1st Qu.:1516   1st Qu.:1882
Median :270.0   Median :431.8   Median :180.2   Median :156.0   Median :1622   Median :2100
Mean   :269.8   Mean   :433.5   Mean   :184.8   Mean   :155.0   Mean   :1821   Mean   :2268
3rd Qu.:290.0   3rd Qu.:475.5   3rd Qu.:193.5   3rd Qu.:160.5   3rd Qu.:2249   3rd Qu.:2855
Max.   :327.5   Max.   :496.3   Max.   :255.8   Max.   :190.0   Max.   :2695   Max.   :3130
      CapMax      NumAssentos      NumeroPortas      TamPneu      VelMax
Min.   : 290.0   Min.   :2.000   Min.   :3.00   Min.   :14.00   Min.   :130.0
1st Qu.: 440.0   1st Qu.:4.250   1st Qu.:5.00   1st Qu.:16.00   1st Qu.:146.2
Median : 485.5   Median :5.000   Median :5.00   Median :17.00   Median :160.0
Mean   : 510.5   Mean   :4.762   Mean   :4.81   Mean   :17.55   Mean   :169.5
3rd Qu.: 565.0   3rd Qu.:5.000   3rd Qu.:5.00   3rd Qu.:19.00   3rd Qu.:187.5
Max.   :1056.0   Max.   :8.000   Max.   :5.00   Max.   :21.00   Max.   :260.0
      BootCap      Acc      CapMaxBat
Min.   :171.0   Min.   : 2.800   Min.   : 22.0
1st Qu.:310.2   1st Qu.: 6.800   1st Qu.: 62.5
Median :371.0   Median : 7.900   Median :100.0
Mean   :404.3   Mean   : 7.893   Mean   :109.7
3rd Qu.:497.0   3rd Qu.: 9.650   3rd Qu.:143.8
Max.   :660.0   Max.   :13.100   Max.   :270.0
```

Vamos analisar de forma gráfica um multi-histograma:

```
# Multi-Histogramas de cada Variável
```

```
VarNum01 <- ggplot(VarNum) + geom_histogram(aes(x = PrecoMin))
VarNum02 <- ggplot(VarNum) + geom_histogram(aes(x = Potencia))
VarNum03 <- ggplot(VarNum) + geom_histogram(aes(x = TorqMax))
VarNum04 <- ggplot(VarNum) + geom_histogram(aes(x = CapBat))
VarNum05 <- ggplot(VarNum) + geom_histogram(aes(x = Autonomia))
VarNum06 <- ggplot(VarNum) + geom_histogram(aes(x = DistEixos))
VarNum07 <- ggplot(VarNum) + geom_histogram(aes(x = Comprimento))
VarNum08 <- ggplot(VarNum) + geom_histogram(aes(x = Largura))
VarNum09 <- ggplot(VarNum) + geom_histogram(aes(x = Altura))
VarNum10 <- ggplot(VarNum) + geom_histogram(aes(x = PesoVazio))
VarNum11 <- ggplot(VarNum) + geom_histogram(aes(x = PesoCheio))
VarNum12 <- ggplot(VarNum) + geom_histogram(aes(x = CapMax))
VarNum13 <- ggplot(VarNum) + geom_histogram(aes(x = NumAssentos))
VarNum14 <- ggplot(VarNum) + geom_histogram(aes(x = NumeroPortas))
VarNum15 <- ggplot(VarNum) + geom_histogram(aes(x = TamPneu))
VarNum16 <- ggplot(VarNum) + geom_histogram(aes(x = VelMax))
VarNum17 <- ggplot(VarNum) + geom_histogram(aes(x = BootCap))
VarNum18 <- ggplot(VarNum) + geom_histogram(aes(x = Acc))
VarNum19 <- ggplot(VarNum) + geom_histogram(aes(x = CapMaxBat))
```

```
grid.arrange(VarNum01, VarNum02, VarNum03, VarNum04, VarNum05,
              VarNum06, VarNum07, VarNum08, VarNum09, VarNum10,
              VarNum11, VarNum12, VarNum13, VarNum14, VarNum15,
              VarNum16, VarNum17, VarNum18, VarNum19)
```

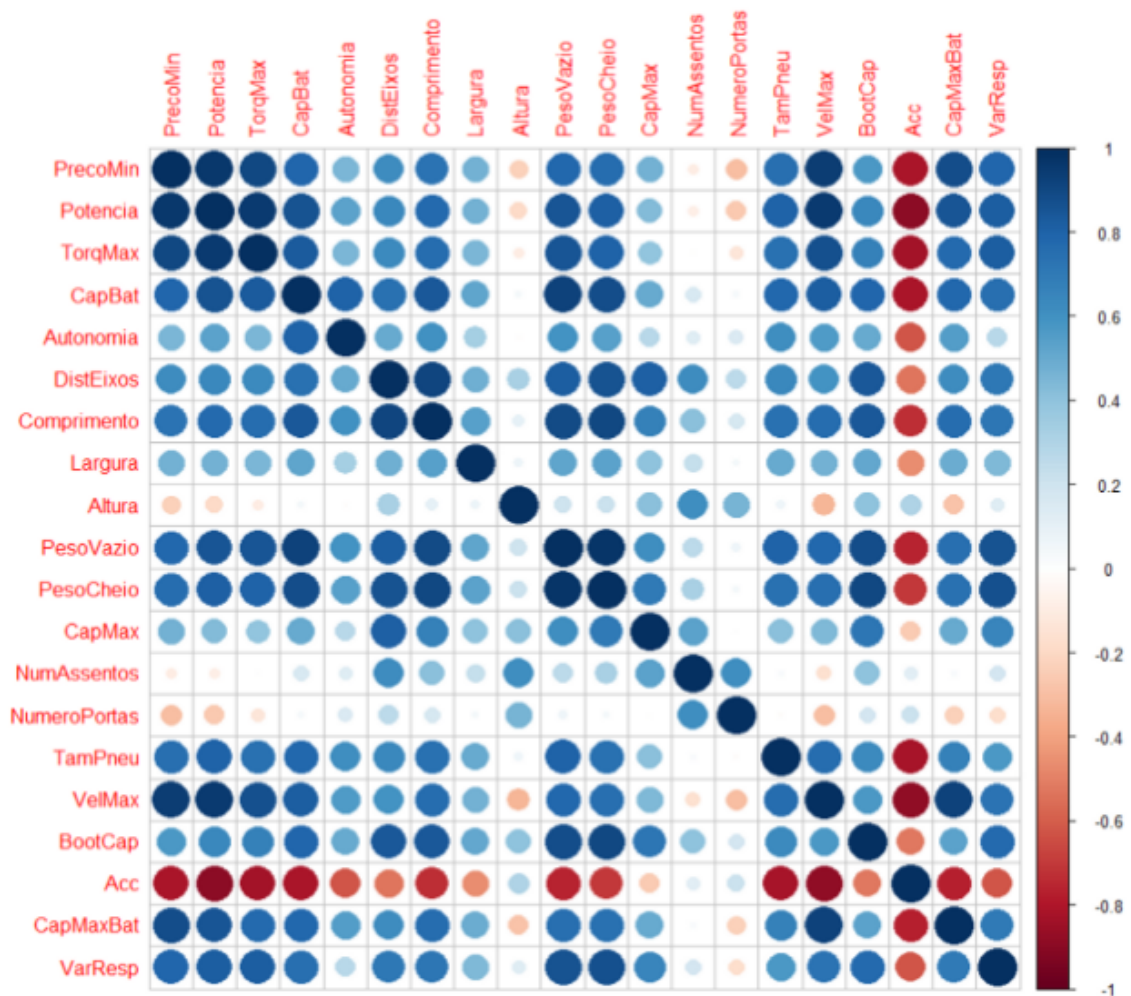


Percebemos que em nosso dataset temos carros de categorias diferentes, apesar de mesmo combustível propulsor, sendo assim um dataset misto, não condicionando a categoria dos veículos. Talvez seria interessante obtermos mais dados de veículos e categorizar os mesmos (etiquetar) para que possamos analisar de forma separada. Podemos corroborar com essa afirmação quando olhamos dados extremos em TorqMax, Potencia, PrecoMin, CapBat e VelMax.

Um outro ponto importante para nossa análise é identificar como as variáveis se relacionam entre si e entre elas e a variável resposta. Vamos realizar uma análise de Correlação a fim de identificar problemas de Multicolinearidade por exemplo.

```
# Análise de Multicolinearidade
```

```
corrplot(cor(cbind(VarNum, VarResp)))
```



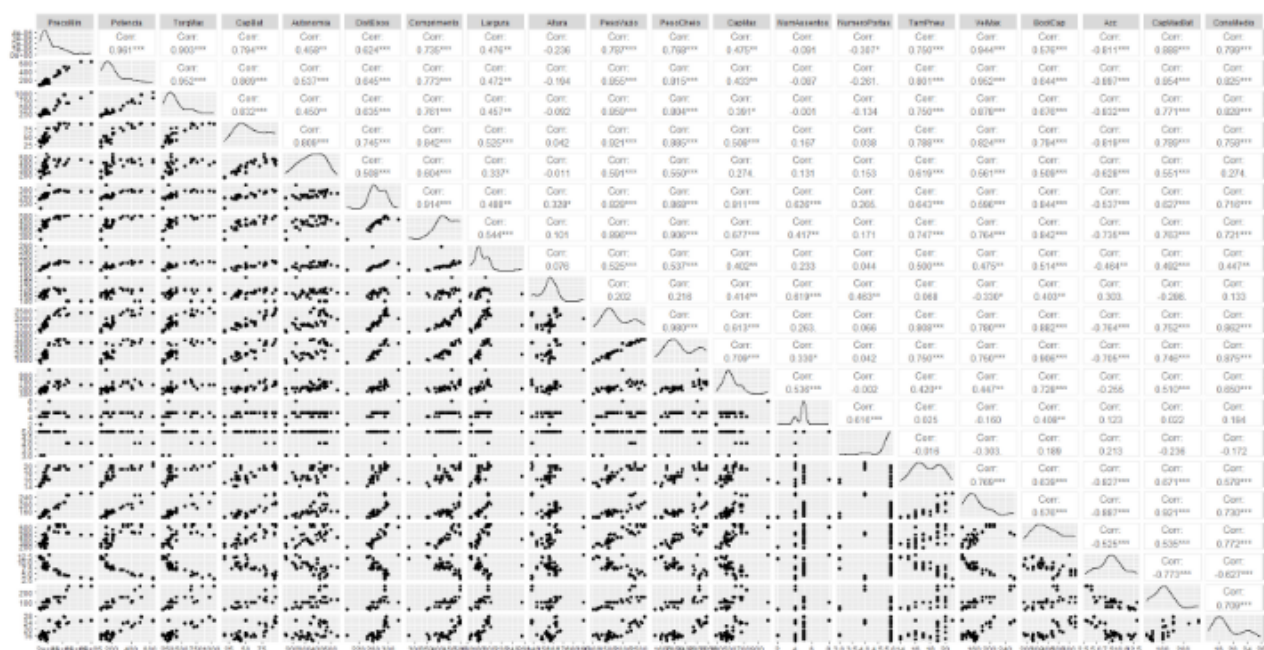


Podemos identificar aqui uma excessiva correlação entre diversas variáveis, o que possivelmente causa confusão durante nossa modelagem preditiva. Correlações como Peso/potencia, DistEixos/CapMax, são relativamente óbvias ao modelo de negócio. Porém observando a nossa Variável Resposta, podemos ver altas correlações com as variáveis, Preço, Potencia, TorqMax, CapBat, Peso Vazio, Peso Cheio, BootCap e Acc. De qualquer forma, vamos utilizar um algoritmo de regressão linear para nos ajudar a melhor selecionar nossas variáveis.

Para melhor analisarmos a correlação entre as variáveis, vamos diagramar um gráfico multivariável.

```
# Scatterplots de Cada Variável Numérica Dependente e a Variável Resposta
```

```
DadosIntResp <- dados %>% select(c(4:6), c(9:25))
Multiplot <- function(DadosIntResp, mapping, method = "loess", ...){
  p <- ggplot(data = data, mapping = mapping) +
    geom_point()+
    geom_smooth(method = method, ...)
  p
}
ggpairs(DadosIntResp, lower = list(continuous = Multiplot))
```



## Análise de Variáveis Categóricas

Vamos ignorar o fabricante neste momento, pois aparentemente não parece relevante, visto que temos 19 fabricantes para 43 observações, o que não nos dá uma amostra razoável para analisar por esse ângulo de visão.

Iniciamos então analisando as proporções de dados para cada fator das variáveis Freio e Cambio.

```
prop.table(table(VarCat$Freios))*100  
prop.table(table(VarCat$Cambio))*100
```

```
> prop.table(table(VarCat$Freios))*100  
  
      1      2  
83.33333 16.66667  
> prop.table(table(VarCat$Cambio))*100  
  
      1      2      3  
47.61905 23.80952 28.57143
```

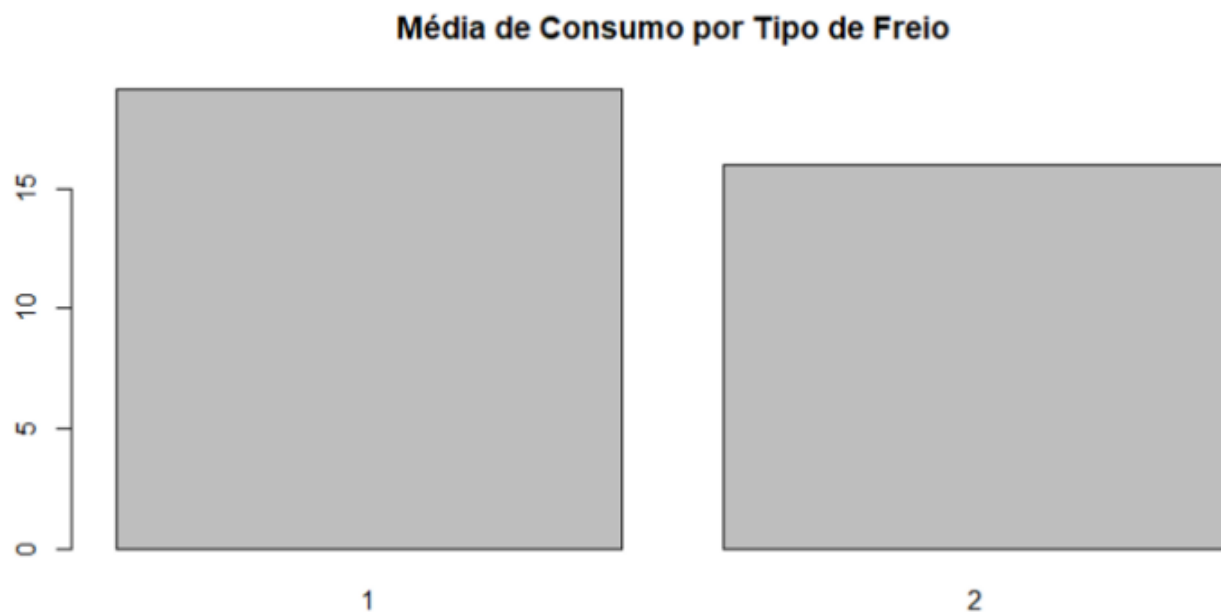
Percebemos q temos muito mais dados de carro com freios a disco nas quatro rodas que apenas disco frontal.

Também é possível perceber que temos um certo equilíbrio na variável de câmbio, o que é melhor para nosso modelo preditivo.

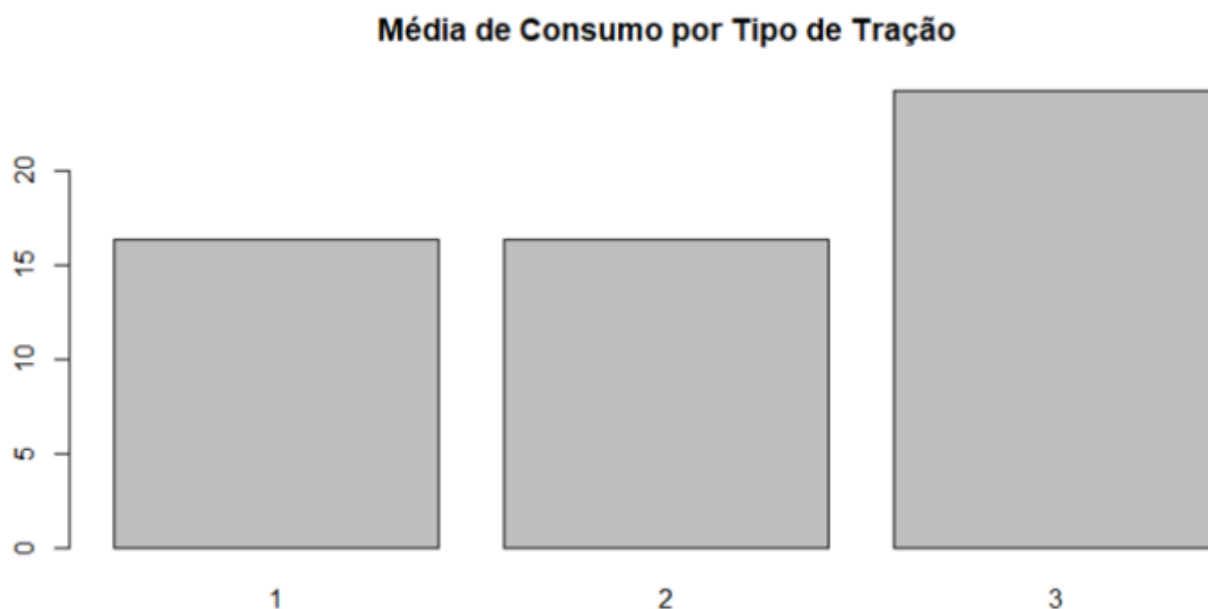
Vamos analisar como é Consumo de Energia para cada tipo de variável, calculando a média para cada categoria.

```
media_por_tipo_de_freio = dados %>% group_by(Freios) %>%  
  summarise(avg_consumo = mean(ConsMedio))  
  
media_por_tipo_de_tracao = dados %>% group_by(Cambio) %>%  
  summarise(avg_consumo = mean(ConsMedio))  
  
barplot(media_por_tipo_de_freio$avg_consumo,  
        names.arg = media_por_tipo_de_freio$Freios,  
        main = 'Média de Consumo por Tipo de Freio')  
  
barplot(media_por_tipo_de_tracao$avg_consumo,  
        names.arg = media_por_tipo_de_tracao$Cambio,  
        main = 'Média de Consumo por Tipo de Tração')
```





Podemos verificar claramente que carro que possuem frenagem frontal e traseira a disco, possuem maior consumo de energia por hora. Aqui não estamos vendo causalidade, apenas constatando o fato de que carros com essa característica em média consome mais energia elétrica.



Nesta variável, percebemos que carros que possuem tração nas 4 rodas possuem um consumo de energia maior. Aqui é um fato diretamente relacionado a natureza do câmbio 4WD,

visto que se necessita admitir energia em 4 rodas para aplicar tração, conseqüentemente usa-se maior quantidade de energia que para apenas 2 rodas.

Porém um fator descoberto é interessante, o consumo médio é muito próximo em carros com tração dianteira ou traseira.

Finalizamos nossa Exploração dos Dados e estamos prontos para iniciar a etapa de Preparação para a construção do modelo preditivo e solução do problema de negócio definido.

## Pre-Processamento dos Dados

Nesta etapa vamos aplicar Feature Selection e definir quais variáveis vamos levar em consideração para a construção do nosso Modelo Base. A regra aqui é encontrar uma boa eficiência na métrica do modelo, sendo ele o mais simples possível.

```
# Carregando Pacotes
#
library(dplyr)
library(kim)
library(caret)

# Carregando o Dataset

dados <- as.data.frame(read_csv(name = 'dados_ajustados', head = TRUE,
                                dirname = 'dados' ))

dim(dados)
dados <- dados[, 5:26]
dim(dados)
str(dados)
dados$Freios <- as.factor(dados$Freios)
dados$Cambio <- as.factor(dados$Cambio)
dados$NumAssentos <- as.factor(dados$NumAssentos)
dados$NumeroPortas <- as.factor(dados$NumeroPortas)
str(dados)
View(dados)FeaturesLM <- lm(ConsMedio ~ .,
                             data = dados)
summary(FeaturesLM)
```

## Feature Selection

Vamos utilizar um modelo linear simples com todas as variáveis para determinar quais são as variáveis mais significantes para explicar nossa variável alvo.

```
FeaturesLM <- lm(ConsMedio ~ .,
                 data = dados)
summary(FeaturesLM)
```

```
Call:
lm(formula = ConsMedio ~ ., data = dados)

Residuals:
    Min       1Q   Median       3Q      Max
-1.2542 -0.3364  0.0000  0.3086  1.1067

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  6.217e+01  1.628e+01   3.820  0.00151 **
PrecoMin      4.431e-06  6.499e-06   0.682  0.50506
Potencia      6.931e-03  1.138e-02   0.609  0.55091
TorqMax     -1.455e-03  4.577e-03  -0.318  0.75469
Freios2     -2.390e-02  1.318e+00  -0.018  0.98576
Cambio2      3.808e+00  1.278e+00   2.981  0.00883 **
Cambio3      5.277e+00  2.220e+00   2.378  0.03024 *
CapBat       8.619e-02  6.756e-02   1.276  0.22028
Autonomia   -1.652e-02  7.108e-03  -2.324  0.03364 *
DistEixos   -2.622e-01  1.198e-01  -2.188  0.04384 *
Comprimento  2.159e-02  4.208e-02   0.513  0.61499
Largura     -1.965e-02  1.428e-02  -1.377  0.18762
Altura       4.024e-02  7.665e-02   0.525  0.60674
PesoVazio    2.635e-03  7.014e-03   0.376  0.71209
PesoCheio   -2.122e-03  3.314e-03  -0.640  0.53114
CapMax      -3.181e-03  7.710e-03  -0.413  0.68541
NumAssentos4 1.823e+01  7.864e+00   2.318  0.03402 *
NumAssentos5 2.157e+01  9.505e+00   2.269  0.03748 *
NumAssentos8 4.114e+01  1.861e+01   2.211  0.04196 *
NumeroPortas4 1.676e+00  2.917e+00   0.575  0.57353
NumeroPortas5 -4.450e+00  2.216e+00  -2.008  0.06180 .
TamPneu     -7.817e-01  2.807e-01  -2.785  0.01325 *
VelMax       1.474e-02  7.138e-02   0.207  0.83896
BootCap      2.489e-02  8.726e-03   2.852  0.01153 *
Acc         -2.294e-01  3.102e-01  -0.739  0.47040
CapMaxBat   -8.928e-03  1.731e-02  -0.516  0.61304
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.8818 on 16 degrees of freedom
Multiple R-squared:  0.9822,    Adjusted R-squared:  0.9545
F-statistic: 35.41 on 25 and 16 DF,  p-value: 8.055e-10
```

Podemos verificar que Autonomia, Cambio, Distância entre Eixos, Numero de Assentos, Tamanho do Pneu e Capacidade do porta-malas são estatisticamente relevantes ao nosso modelo.

As métricas mostram que com  $R^2$  Ajustado de 95,45%, conseguimos afirmar que nosso modelo consegue explicar 95% da variabilidade do consumo médio, pelas variáveis que listamos.

Essa informação pode indicar que nosso modelo está com overfitting, ou seja, treinado demais e pouco generalizável. De qualquer forma, o intuito deste modelo era apenas identificar as variáveis mais importantes, o que nos serviu perfeitamente.

Uma observação importante que fizemos é que os dados estão em escalas diferentes e isso poderia alterar o resultado do modelo de identificação das variáveis mais importantes. Então refizemos o modelo com dados normalizados. Utilizamos uma função MinMax para deixar todas as variáveis na mesma escala.

```
# Aplicando Padronização ao Dataset

# Criando um função de Padronizar

padronizar <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}

# Primeiramente vamos dividir nosso modelo em dados de treino e teste.

separador <- createDataPartition(y = dados$ConsMedio, p = 0.75,
                                  list = FALSE)

dados_treino <- dados[separador,]
dados_teste <- dados[-separador,]

str(dados_teste)
str(dados_treino)
```

```
# Aplicando a Função aos Dados Treino Numéricos

dados_testeNumP <- dados_teste[,!unlist(lapply(dados_teste,
                                                is.factor)))]
str(dados_testeNumP)

dados_testeFac <- dados_teste[, unlist(lapply(dados_teste
                                                , is.factor)))]
str(dados_testeFac)

dados_treinoNumP <- dados_treino[,!unlist(lapply(dados_treino,
                                                is.factor)))]
str(dados_treinoNumP)

dados_treinoFac <- dados_treino[, unlist(lapply(dados_treino
                                                , is.factor)))]
str(dados_treinoFac)
```

```
dados_testeNumP <- as.data.frame(lapply(dados_testeNumP[, -18],
                                         padronizar))

dados_testeNumP <- cbind(dados_testeNumP, dados_teste$ConsMedio)
colnames(dados_testeNumP)[18] <- 'ConsMedio'
str(dados_testeNumP)

dados_treinoNumP <- as.data.frame(lapply(dados_treinoNumP[, -18],
                                         padronizar))

dados_treinoNumP <- cbind(dados_treinoNumP, dados_treino$ConsMedio)
colnames(dados_treinoNumP)[18] <- 'ConsMedio'
str(dados_treinoNumP)

# Unificando os Data Frames

dados_treino_final <- cbind(dados_treinoNumP, dados_treinoFac)
dados_teste_final <- cbind(dados_testeNumP, dados_testeFac)

str(dados_treino_final)
str(dados_teste_final)
```

```
# Reavaliando as Variáveis de Maior Significância
```

```
dados_final <- rbind(dados_treino_final, dados_teste_final)
FeaturesLMS <- lm(ConsMedio ~ ., data = dados_final)
summary(FeaturesLMS)
```

```
Call:
lm(formula = ConsMedio ~ ., data = dados_final)

Residuals:
    Min       1Q   Median       3Q      Max
-0.95879 -0.33334 -0.01356  0.34427  1.05642

Coefficients:
(Intercept)    16.6138    2.2395    7.419 1.46e-06 ***
PrecoMin         1.2369    4.2953    0.288 0.777061
Potencia         8.0887    5.1687    1.565 0.137158
TorqMax        -0.3851    2.7555   -0.140 0.890606
CapBat         10.4525    3.6385    2.873 0.011048 *
Autonomia       -7.5258    2.1371   -3.522 0.002832 **
DistEixos      -12.6267    4.5659   -2.765 0.013789 *
Comprimento     4.8813    4.8253    1.012 0.326780
Largura         -0.5382    1.0920   -0.493 0.628791
Altura           2.6975    2.7209    0.991 0.336245
PesoVazio        3.7471    8.2349    0.455 0.655199
PesoCheio       -4.0409    5.0726   -0.797 0.437338
CapMax           3.9142    1.7945    2.181 0.044432 *
```

```
TamPneu      -7.2467      1.7481     -4.145 0.000761 ***
VelMax       -10.6552      5.2359     -2.035 0.058761 .
BootCap       5.1904      2.7796      1.867 0.080280 .
Acc          -2.4776      2.4640     -1.006 0.329618
CapMaxBat     2.7448      3.0502      0.900 0.381527
Freios2       -1.2319      1.0308     -1.195 0.249484
Cambio2       3.1542      0.8691      3.629 0.002255 **
Cambio3       3.8343      1.6382      2.341 0.032540 *
NumAssentos4  5.0579      2.2070      2.292 0.035825 *
NumAssentos5  5.1798      2.6637      1.945 0.069624 .
NumAssentos8  9.4808      4.2587      2.226 0.040716 *
NumeroPortas4 1.2353      2.2536      0.548 0.591154
NumeroPortas5 -2.3233      1.3482     -1.723 0.104113
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.8516 on 16 degrees of freedom
Multiple R-squared:  0.9834,    Adjusted R-squared:  0.9576
F-statistic: 38.02 on 25 and 16 DF,  p-value: 4.666e-10
```

Temos então uma resposta semelhante de  $R^2$  ajustado, mas com variáveis diferentes, sendo assim, vamos admitir as variáveis: CapBat, Autonomia, DistEixos, CapMax, Cambio e NumAssentos.

```
dados_prep_teste <- dados_teste_final %>% select(CapBat, Autonomia, DistEixos,
                                                CapMax, Cambio, NumAssentos,
                                                ConsMedio)

str(dados_prep_teste)

dados_prep_treino <- dados_treino_final %>% select(CapBat, Autonomia, DistEixos,
                                                CapMax, Cambio, NumAssentos,
                                                ConsMedio)

str(dados_prep_treino)

write.csv2(dados_prep_treino, file = 'dados/dados_prep_treino.csv')

write.csv2(dados_prep_teste, file = 'dados/dados_prep_teste.csv')
```

## Modelagem Preditiva

```
# Parte 04 - Modelagem Preditiva

# Neste etapa, vamos criar modelos de machine learning e analisar as métricas
# de cada modelo e definir qual realiza previsões com mais eficiência,
# utilizando o dataset preparado e levando em consideração nosso modelo
# preditivo base.
```

```
# Carregando Pacotes

library(dplyr)
library(caret)
library(ggplot2)
library(forecast)

# Carregando o Dataset

dados_treino <- read.csv2('dados/dados_prep_treino.csv', header = TRUE)[-1]
str(dados_treino)
dados_teste <- read.csv2('dados/dados_prep_teste.csv', header = TRUE)[-1]
str(dados_teste)

summary(is.na(dados_treino))
summary(is.na(dados_teste))
```

## Construindo o Modelo Base e Analisando o Resultado

### Modelo Base

Vamos construir o modelo base com a função `lm()` do pacote `utils` base da linguagem R.

```
# Construindo nosso Modelo Base

# Vamos utilizar a função lm() para construir nosso modelo base, sendo o
# algoritmo mais simples que conhecemos.

ModeloBase <- lm(ConsMedio ~ ., dados_treino)
summary(ModeloBase)

# Nosso modelo Base possui em treinamento, um R² de 92,67% ajustado, o
# que significa que conseguimos explicar o consumo com 92,67% de
# variabilidade dessas variáveis.

# Acurácia do Modelo Base de Treino
prev_treino_ModeloB <- predict(ModeloBase, dados_treino[-8])
accuracy(prev_treino_ModeloB, dados_treino$ConsMedio)

# Acurácia de RMSE 1.407
```

```
Call:
lm(formula = ConsMedio ~ ., data = dados_treino)

Residuals:
    Min       1Q   Median       3Q      Max
-3.3493 -0.7016  0.0883  0.5107  2.4363

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  14.0363     1.9469   7.209 1.17e-07 ***
CapBat       17.6938     2.1127   8.375 7.41e-09 ***
Autonomia    -14.1841     1.6552  -8.570 4.76e-09 ***
DistEixos     -3.8796     4.1306  -0.939  0.35626
CapMax        6.8222     2.0469   3.333  0.00259 **
Cambio        0.7917     0.4980   1.590  0.12399
NumAssentos   0.2588     0.5429   0.477  0.63760
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.157 on 26 degrees of freedom
Multiple R-squared:  0.9404,    Adjusted R-squared:  0.9267
F-statistic: 68.41 on 6 and 26 DF,  p-value: 1.114e-14
```

Nosso modelo base possui em treinamento, um  $R^2$  de 92,67% ajustado, o que significa que conseguimos explicar o consumo médio com 92% de variabilidade das variáveis independentes. Um valor bom para a métrica, assim como a acurácia do modelo de treino base com um RMSE de 1.027.

```
> accuracy(prev_treino_ModeloB, dados_treino$ConsMedio)
              ME      RMSE      MAE      MPE      MAPE
Test set 2.691476e-15 1.027163 0.7549713 -0.3647261 4.476753
```

Vamos realizar as previsões dos dados de teste e analisar os resíduos.

```
# Testando e Avaliando o Modelo

Previsao01 <- predict(ModeloBase, dados_teste[-8])
Previsao01

# Analisando a Acurácia do Teste para o Modelo Base

accuracy(Previsao01, dados_teste$ConsMedio)

# Obtivemos uma acurácia relativamente mais alta que em treino
# no valor de RMSE 2.907. !! Ponto de Atenção
```



```
# Analisando o resíduo do modelo

ConsumoTeste <- dados_teste$ConsMedio

Res_ModeloBase <- ConsumoTeste - Previsao01

FitModeloBase <- data.frame(Target = ConsumoTeste,
                             Previsao = Previsao01,
                             Residuo = Res_ModeloBase)

head(FitModeloBase)
summary(FitModeloBase)

# Scatter Plot Comparativo

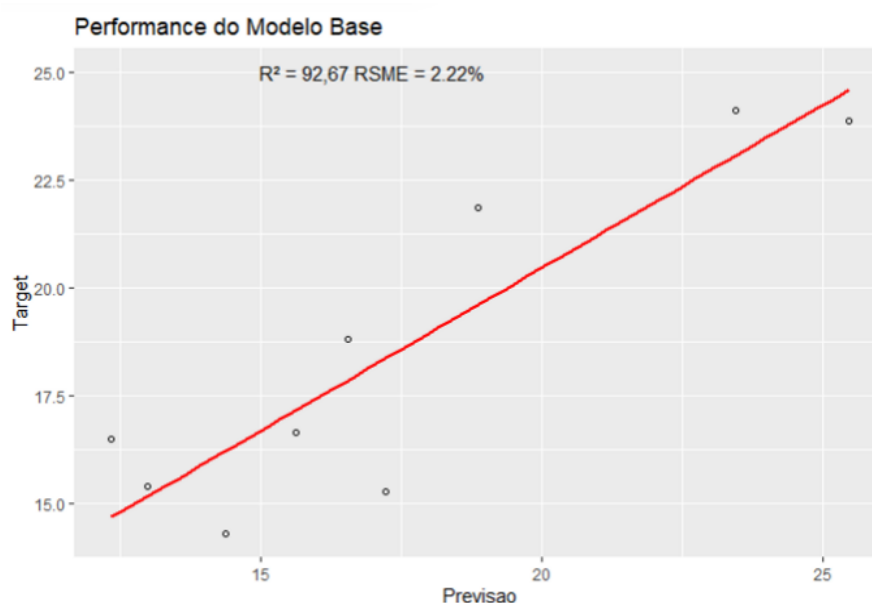
camada1 <- geom_point(shape = 1)

camada2 <- geom_smooth(method = lm, color = 'red', se = FALSE)

ggplot(FitModeloBase, aes(x = Previsao, y = Target)) + camada1 + camada2 +
  ggtitle('Performance do Modelo Base') + annotate(geom = 'text', x = 17,
                                                    y = 25,
                                                    label = 'R² = 92,67 RSME = 2,22%')
```

Podemos ver que temos uma média nos resíduos próximos a zero e um range pequeno entre os valores mínimos e máximos. Vamos ver a regressão graficamente.

Consequentemente temos uma ótima acurácia e poucos erros em relação aos resultados medidos no teste.



```
> head(FitModeloBase)
  Target Previsao   Residuo
1  23.85 25.46181 -1.61180984
2  14.30 14.36639 -0.06639241
3  18.80 16.56364  2.23636045
4  15.40 12.98300  2.41699755
5  15.30 17.23469 -1.93468759
6  21.85 18.87666  2.97333844
```

```
> summary(FitModeloBase)
      Target      Previsao      Residuo
Min.   :14.30  Min.   :12.35  Min.   :-1.93469
1st Qu.:15.40  1st Qu.:14.37  1st Qu.: -0.06639
Median :16.65  Median :16.56  Median : 1.01638
Mean   :18.53  Mean   :17.43  Mean   : 1.09301
3rd Qu.:21.85  3rd Qu.:18.88  3rd Qu.: 2.41700
Max.   :24.10  Max.   :25.46  Max.   : 4.15171
```

## Modelo V02

Para o modelo V02 vamos trocar o algoritmo mas manter o método de regressão. Vamos utilizar o Caret com o método 'lm'.

```
# Construindo Modelo Versão 02

# Para este modelo, utilizaremos o pacote Caret com o método de Regressão
# Linear, sem Trainig Control e Tuning.

ModeloV02 <- train(ConsMedio ~ ., data = dados_treino, method = 'lm')
summary(ModeloV02)

# Podemos reparar que obtivemos um R² de 92,67%, mesmo valor do modelo base.
# Vamos realizar a Previsão de teste e avaliar
# graficamente.

Previsao02 <- predict(ModeloV02, dados_teste[-8])
Previsao02

accuracy(Previsao02, dados_teste$ConsMedio)

# Mesma acurácia do modelo base RMSE 2.907

# Analisando o resíduo do modelo

Res_ModeloV02 <- ConsumoTeste - Previsao02

FitModeloV02 <- data.frame(Target = ConsumoTeste,
                           Previsao = Previsao02,
                           Residuo = Res_ModeloV02)

head(FitModeloV02)
summary(FitModeloV02)
```

```
# Scatter Plot Comparativo

camada1 <- geom_point(shape = 1)

camada2 <- geom_smooth(method = lm, color = 'red', se = FALSE)

ggplot(FitModeloV02, aes(x = Previsao, y = Target)) + camada1 + camada2 +
  ggtitle('Performance do Modelo V02') + annotate(geom = 'text', x = 17,
                                                  y = 25,
                                                  label = 'R² = 92,67 RSME = 2.22%')
```

Call:

```
lm(formula = .outcome ~ ., data = dat)
```

Residuals:

```
      Min       1Q   Median       3Q      Max
-3.3493 -0.7016  0.0883  0.5107  2.4363
```

Coefficients:

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   14.0363     1.9469   7.209 1.17e-07 ***
CapBat         17.6938     2.1127   8.375 7.41e-09 ***
Autonomia     -14.1841     1.6552  -8.570 4.76e-09 ***
DistEixos      -3.8796     4.1306  -0.939  0.35626
CapMax          6.8222     2.0469   3.333  0.00259 **
Cambio         0.7917     0.4980   1.590  0.12399
NumAssentos    0.2588     0.5429   0.477  0.63760
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Residual standard error: 1.157 on 26 degrees of freedom

Multiple R-squared: 0.9404, Adjusted R-squared: 0.9267

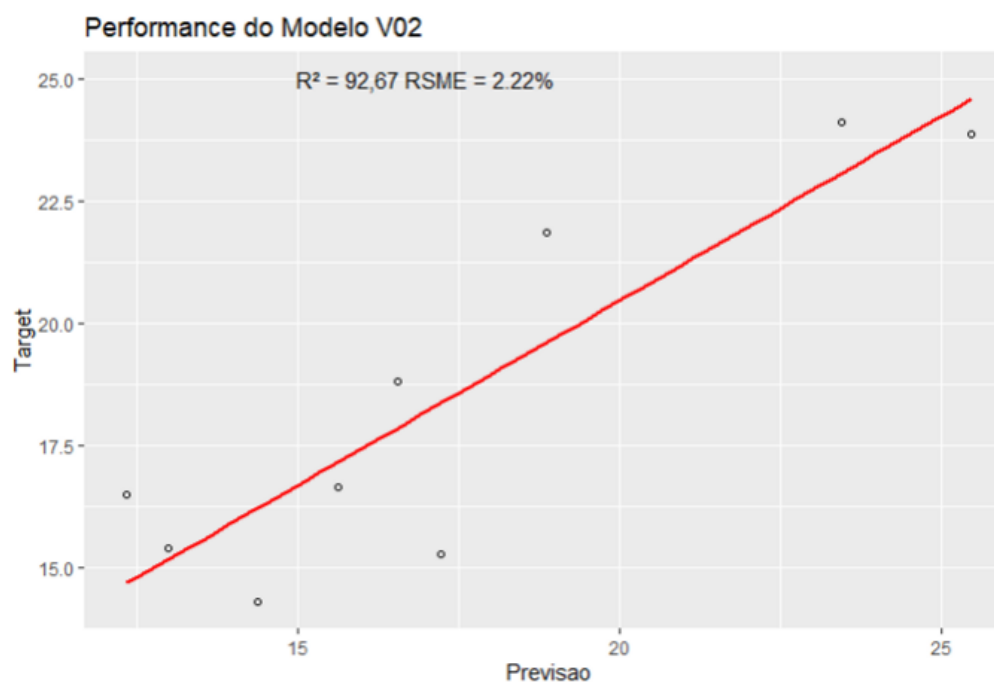
F-statistic: 68.41 on 6 and 26 DF, p-value: 1.114e-14

Perceba que apenas alterando o pacote, não tivemos nenhuma diferença entre os resultados.

```
> accuracy(Previsao02, dados_teste$ConsMedio)
              ME      RMSE      MAE      MPE      MAPE
Test set 1.093011 2.229308 1.895876 6.146204 10.56118
```

```
> head(FitModeloV02)
  Target Previsao  Residuo
1  23.85 25.46181 -1.61180984
2  14.30 14.36639 -0.06639241
3  18.80 16.56364  2.23636045
4  15.40 12.98300  2.41699755
5  15.30 17.23469 -1.93468759
6  21.85 18.87666  2.97333844
```

```
summary(y ~ x, data = dados_teste)
              Target      Previsao      Residuo
Min.   :14.30   Min.   :12.35   Min.   : -1.93469
1st Qu.:15.40   1st Qu.:14.37   1st Qu.: -0.06639
Median :16.65   Median :16.56   Median :  1.01638
Mean   :18.53   Mean   :17.43   Mean   :  1.09301
3rd Qu.:21.85   3rd Qu.:18.88   3rd Qu.:  2.41700
Max.   :24.10   Max.   :25.46   Max.   :  4.15171
```



## Modelo V03

Para este modelo vamos mudar o método de regressão linear para o Método Boosted Linear Regression.

```
# Construindo Modelo V03

# Para este modelo vamos alterar o método de Regressão Linear para Boosted
# Linear Regression e analisar os resultados.
# Utilizaremos o mesmo pacote Caret.

ModeloV03 <- train(ConsMedio ~ ., data = dados_treino, method = 'BstLm')
ModeloV03

# Analisando o resíduo do modelo

Res_ModeloV03 <- ConsumoTeste - Previsao03

FitModeloV03 <- data.frame(Target = ConsumoTeste,
                             Previsao = Previsao03,
                             Residuo = Res_ModeloV03)

head(FitModeloV03)
summary(FitModeloV03)

# Scatter Plot Comparativo

ggplot(FitModeloV03, aes(x = Previsao, y = Target)) +
  geom_point(shape = 1) +
  geom_smooth(method = lm, color = 'red', se = FALSE) +
  ggtitle('Performance do Modelo V03')+
  annotate(geom = 'text', x = 17, y = 25, label = 'R² = 65,12% RSME = 3.34')
```

```
> ModeloV03
Boosted Linear Model

33 samples
6 predictor

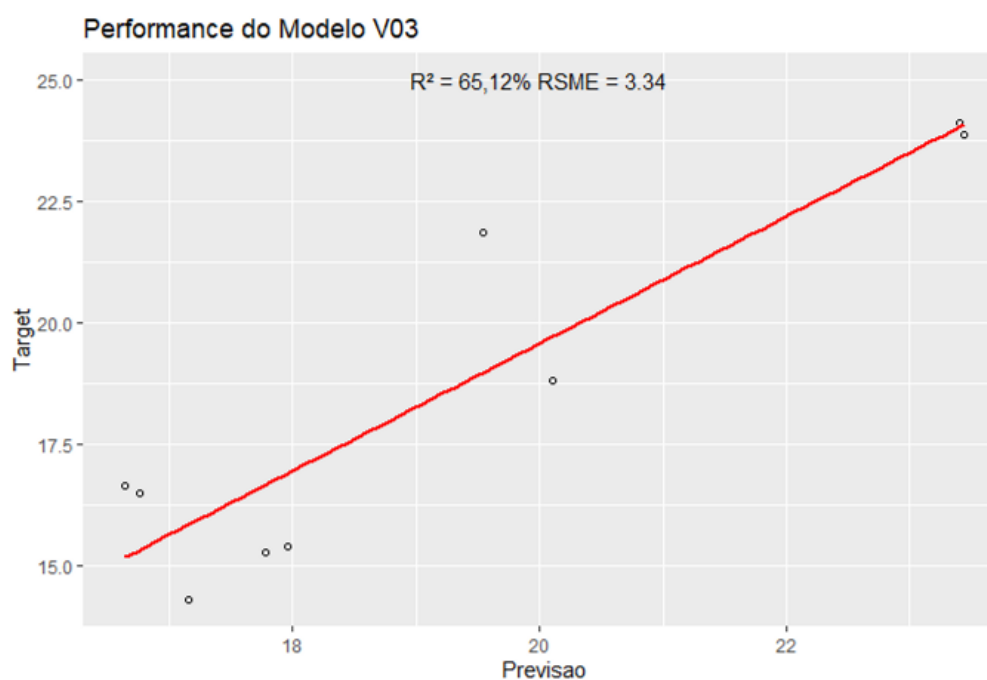
No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 33, 33, 33, 33, 33, ...
Resampling results across tuning parameters:

  mstop  RMSE      Rsquared  MAE
    50   3.833294  0.5773610  3.431336
   100   3.566225  0.6157972  3.111424
   150   3.341078  0.6512431  2.854399

Tuning parameter 'nu' was held constant at a value of 0.1
RMSE was used to select the optimal model using the
smallest value.
The final values used for the model were mstop = 150 and nu
= 0.1.
```

	Target	Previsao	Residuo
1	23.85	23.43588	0.4141222
2	14.30	17.16161	-2.8616109
3	18.80	20.10335	-1.3033470
4	15.40	17.96319	-2.5631871
5	15.30	17.78603	-2.4860301
6	21.85	19.54537	2.3046302

	Target	Previsao	Residuo
Min.	:14.30	Min. :16.65	Min. : -2.8616
1st Qu.	:15.40	1st Qu.:17.16	1st Qu.: -2.4860
Median	:16.65	Median :17.96	Median : -0.2657
Mean	:18.53	Mean :19.20	Mean : -0.6733
3rd Qu.	:21.85	3rd Qu.:20.10	3rd Qu.: 0.4141
Max.	:24.10	Max. :23.44	Max. : 2.3046



Neste modelo tivemos um  $R^2$  de 65,12% aproximadamente, com 25 repetições de re-amostragem. Neste método, realizamos diversas iterações com amostragens menores do dataset e identificamos a que possui menor RMSE e consequentemente maior  $R^2$ . De qualquer forma tivemos um resultado pior que em nosso Modelo Base.

## Modelo V04

Para este modelo usaremos outro método do pacote Caret, chamado de glmnet.

```
# Construindo Modelo V04

# Para este modelo vamos alterar o método de Regressão Linear para glmnet
# e analisar os resultados.
# Utilizaremos o mesmo pacote Caret.

ModeloV04 <- train(ConsMedio ~ ., data = dados_treino, method = 'glmnet')
ModeloV04

# Tivemos uma piora drástica do  $R^2$  em relação ao Modelo V03, porém ainda
# assim nosso modelo base é melhor em relação a métrica de variabilidade
# das variáveis em relação a variável resposta.
```

```
Previsao04 <- predict(ModeloV04, dados_teste[-8])
Previsao04

# Analisando o resíduo do modelo

Res_ModeloV04 <- ConsumoTeste - Previsao04

FitModeloV04 <- data.frame(Target = ConsumoTeste,
                           Previsao = Previsao04,
                           Residuo = Res_ModeloV04)

head(FitModeloV04)
summary(FitModeloV04)

# Scatter Plot Comparativo

ggplot(FitModeloV04, aes(x = Previsao, y = Target)) +
  geom_point(shape = 1) +
  geom_smooth(method = lm, color = 'red', se = FALSE) +
  ggtitle('Performance do Modelo V04')+
  annotate(geom = 'text', x = 17, y = 25, label = ' $R^2 = 77,77\%$ ')
```

```
> ModeloV04
glmnet
```

```
33 samples
6 predictor
```

No pre-processing

Resampling: Bootstrapped (25 reps)

Summary of sample sizes: 33, 33, 33, 33, 33, 33, ...

Resampling results across tuning parameters:

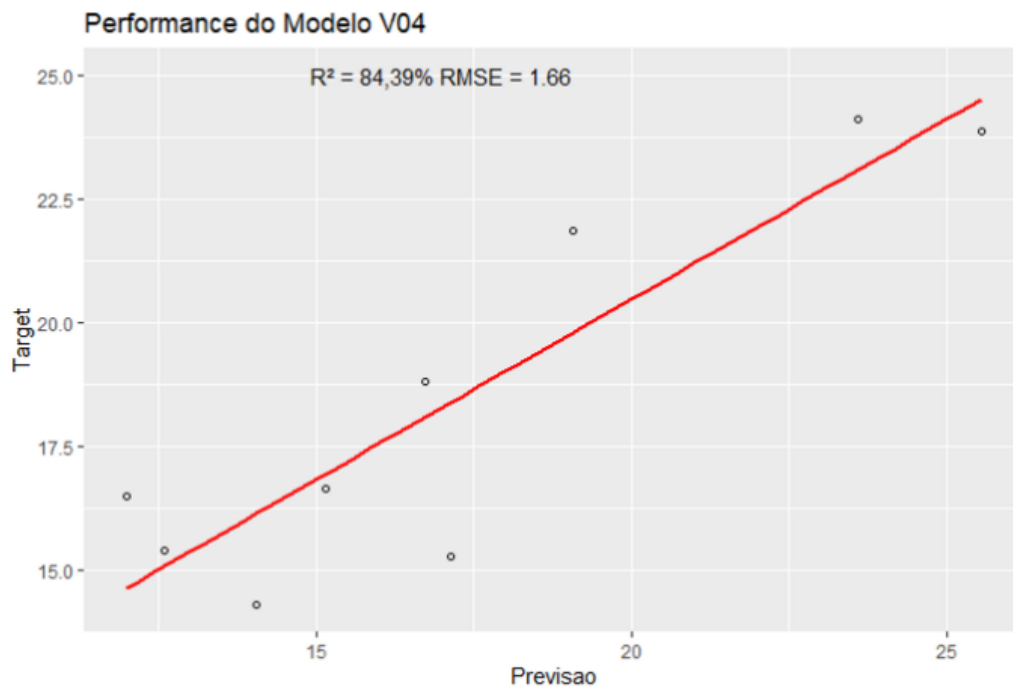
alpha	lambda	RMSE	Rsquared	MAE
0.10	0.006295296	1.664000	0.8439565	1.236413
0.10	0.062952965	1.704398	0.8422168	1.290303
0.10	0.629529645	2.169265	0.7640937	1.710190
0.55	0.006295296	1.660722	0.8450119	1.232211
0.55	0.062952965	1.687921	0.8464261	1.276365
0.55	0.629529645	2.483541	0.6968285	1.942585
1.00	0.006295296	1.656760	0.8460774	1.228075
1.00	0.062952965	1.686541	0.8458511	1.275791
1.00	0.629529645	2.813177	0.6168897	2.166624

RMSE was used to select the optimal model using the smallest value.

The final values used for the model were alpha = 1 and lambda = 0.006295296.

	Target	Previsao	Residuo
1	23.85	23.43588	0.4141222
2	14.30	17.16161	-2.8616109
3	18.80	20.10335	-1.3033470
4	15.40	17.96319	-2.5631871
5	15.30	17.78603	-2.4860301
6	21.85	19.54537	2.3046302

	Target	Previsao	Residuo
Min.	:14.30	Min. :12.00	Min. :-1.8369
1st Qu.	:15.40	1st Qu.:14.05	1st Qu.: 0.2538
Median	:16.65	Median :16.73	Median : 1.5001
Mean	:18.53	Mean :17.32	Mean : 1.2085
3rd Qu.	:21.85	3rd Qu.:19.07	3rd Qu.: 2.7768
Max.	:24.10	Max. :25.55	Max. : 4.5012



Atingimos aqui 84,39% de  $R^2$ , uma melhora drástica em relação ao modelo V03, porém ainda menor que nosso modelo base, porém uma acurácia melhor de 1.66.

## Conclusão

Portanto finalizamos nosso trabalho, disponibilizando o Modelo V04 como apto a realizar novas previsões para a área de negócio, pois apresenta melhor performance nas métricas e nos resíduos gerados.

Importante ressaltar que os novos dados a serem aplicados as variáveis do modelo, necessariamente precisam ser tratados da mesma forma que fizemos durante o processo de pré-processamento. Ou seja, sem outliers, sem dados NaN e normalizados para que possam ter a mesma escala de valor.

Concluimos o projeto com este relatório, incluindo a descrição do script realizado em linguagem R.