

Projeto 01 - Prevendo Consumo de Energia de Carros Elétricos

Descrição e Introdução do Problema de Negócio

Utilizando os dados do dataset, documentado no arquivo `readme_projeto01.pdf`, temos como objetivo prever o consumo de energia elétrica dos modelos da base e identificar insights importantes a 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)
```

Car full name	Make	Model	Minimal price (gross) [PLN]	Engine power (kW)	Maximum torque (Nm)	Type of brakes	Drive type	Battery capacity (kWh)	Range (NEDC) (km)	Wheelbase (cm)	Length (cm)	Width (cm)	Height (cm)	Minimal empty weight (kg)	Permissible gross weight (kg)	Maximum load capacity (kg)	Number of seats	Number of doors	Tire size (in)	Maximum speed (km/h)	Boot capacity (VDA) (kg)	Acceleration 0-100 (s)	Maximum DC charging power (kW)	mean - Energy consumption (kWh/100 km)
Audi e-tron S quattro	Audi	e-tron S quattro	345700	300	680	disc (front + rear)	4WD	95.0	438	292.8	495.1	193.5	162.9	2005	3130	640	5	5	19	200	640	5.7	150	24.43
Audi e-tron S quattro	Audi	e-tron S quattro	308400	313	560	disc (front + rear)	4WD	75.0	340	292.8	495.1	193.5	162.9	2445	3040	670	5	5	19	190	640	6.9	150	23.80
Audi e-tron S quattro	Audi	e-tron S quattro	434800	303	670	disc (front + rear)	4WD	95.0	364	292.8	495.2	197.6	162.8	2005	3130	565	5	5	20	210	640	4.5	150	27.55
Audi e-tron Sportback 55 quattro	Audi	e-tron Sportback 55 quattro	319700	313	560	disc (front + rear)	4WD	75.0	340	292.8	495.1	193.5	161.8	2445	3040	640	5	5	19	190	615	6.9	150	23.50
Audi e-tron Sportback 55 quattro	Audi	e-tron Sportback 55 quattro	317000	300	664	disc (front + rear)	4WD	95.0	447	292.8	495.1	193.5	161.6	2005	3130	670	5	5	19	200	615	5.7	150	23.85
Audi e-tron Sportback S quattro	Audi	e-tron Sportback S quattro	424200	303	670	disc (front + rear)	4WD	95.0	369	292.8	495.2	197.6	161.5	2005	3130	565	5	5	20	210	615	4.5	150	27.20
BMW i3	BMW	i3	169700	170	260	disc (front + rear)	2WD (rear)	42.2	350	257.0	400.0	179.1	157.0	1440	1730	440	4	5	19	140	260	8.1	50	13.10
BMW i4	BMW	i4	184200	184	270	disc (front + rear)	2WD (rear)	42.2	345	257.0	400.0	179.1	159.0	1440	1730	440	4	5	20	160	260	6.6	50	14.50
BMW iX3	BMW	iX3	232000	236	400	disc (front + rear)	2WD (rear)	60.0	440	281.4	473.4	189.1	166.0	2200	2725	540	5	5	19	180	510	6.8	150	16.80
Citroen e-C4	Citroen	e-C4	125000	136	280	disc (front + rear)	2WD (front)	55.0	350	265.7	415.4	160.0	153.2	1941	2000	450	5	5	18	150	380	9.0	100	—
DS DS1 Crossback e-tense	DS	DS1 Crossback e-tense	159800	136	280	disc (front + rear)	2WD (front)	55.0	320	253.8	413.8	205.0	153.4	1520	1875	450	5	5	17	160	380	8.7	100	15.60
Honda e	Honda	e	152000	136	315	disc (front + rear)	2WD (rear)	35.5	222	253.8	389.4	176.2	151.2	1554	1855	342	5	5	16	145	171	8.0	100	17.20
Honda e Advance	Honda	e Advance	165000	154	315	disc (front + rear)	2WD (rear)	35.5	222	253.8	389.4	175.2	151.2	1941	1870	350	5	5	17	145	171	8.0	100	17.50
Hyundai Ioniq electric	Hyundai	Ioniq electric	154500	136	280	disc (front + rear)	2WD (front)	39.5	311	275.0	447.0	182.0	147.5	1527	1870	510	5	5	16	185	287	9.0	100	13.80
Hyundai Kona electric 39 240Wh	Hyundai	Kona electric 39 240Wh	154300	136	395	disc (front + rear)	2WD (front)	39.2	289	260.0	418.0	180.0	157.0	1535	2020	405	5	5	17	155	332	9.7	100	15.00
Hyundai Kona electric 64kWh	Hyundai	Kona electric 64kWh	178400	204	395	disc (front + rear)	2WD (front)	64.0	440	260.0	418.0	180.0	157.0	1605	2170	405	5	5	17	167	332	7.4	100	15.40
Jaguar i-Pace	Jaguar	i-Pace	165000	400	680	disc (front + rear)	4WD	90.0	470	288.0	483.2	201.1	155.6	2200	2670	537	5	5	20	200	656	4.0	100	23.20
Kia e-Niro 39 240Wh	Kia	e-Niro 39 240Wh	144990	136	395	disc (front + rear)	2WD (front)	39.2	289	270.0	473.5	180.5	156.0	1582	2080	400	5	5	17	155	451	9.0	100	15.30
Kia e-Niro 64kWh	Kia	e-Niro 64kWh	197000	204	395	disc (front + rear)	2WD (front)	64.0	450	270.0	473.5	180.5	156.0	1707	2230	490	5	5	17	167	451	7.8	100	15.50
Kia e-Soul 39 240Wh	Kia	e-Soul 39 240Wh	139900	136	395	disc (front + rear)	2WD (front)	39.2	276	260.0	418.5	180.0	160.5	1535	1682	490	5	5	17	157	315	9.9	100	16.00
Kia e-Soul 64kWh	Kia	e-Soul 64kWh	160990	204	395	disc (front + rear)	2WD (front)	64.0	452	260.0	418.5	180.0	160.5	1535	1682	490	5	5	17	167	315	7.9	100	15.70
Mazda MX-30	Mazda	MX-30	142000	145	270	disc (front + rear)	2WD (front)	35.5	200	205.5	439.5	179.5	155.5	1545	2118	474	5	5	18	140	350	9.7	37	14.50

```
> str(dadosv00)
tibble [53 × 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 308400 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 5 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 [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 50 150 100 ...
 $ 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

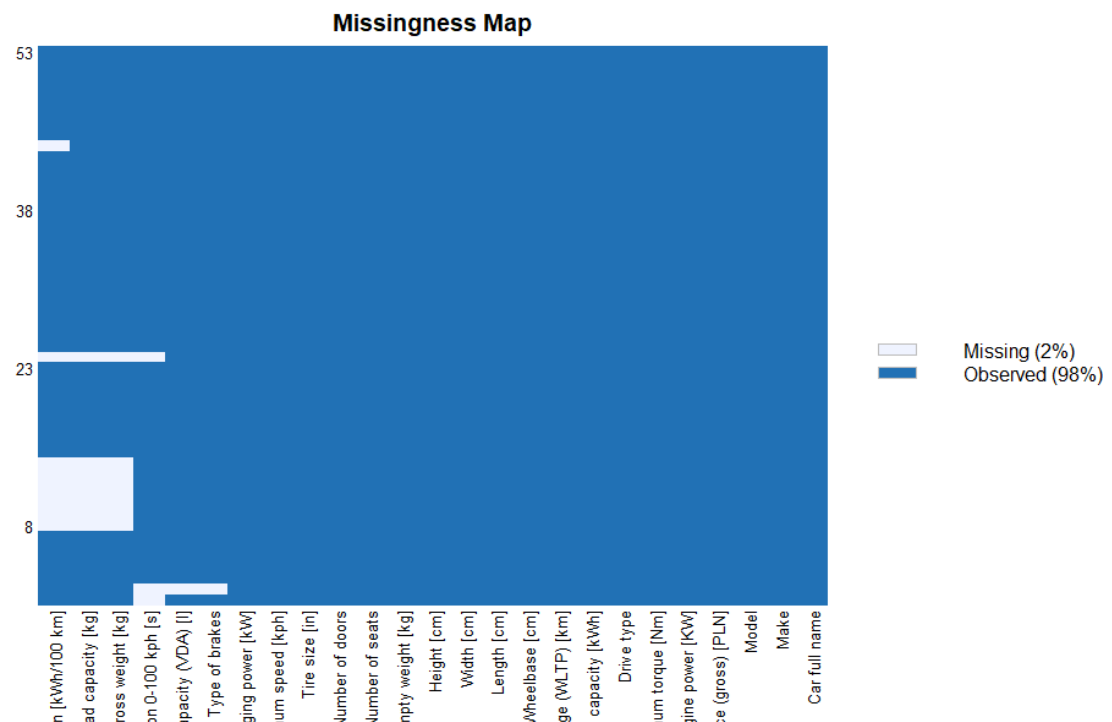
Maximum torque [Nm] Type of brakes Drive type      Battery capacity [kwh] Range (WLTP) [km]
Mode :logical      Mode :logical  Mode :logical  Mode :logical      Mode :logical
FALSE:53            FALSE:52      FALSE:53      FALSE:53            FALSE:53
TRUE :1

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

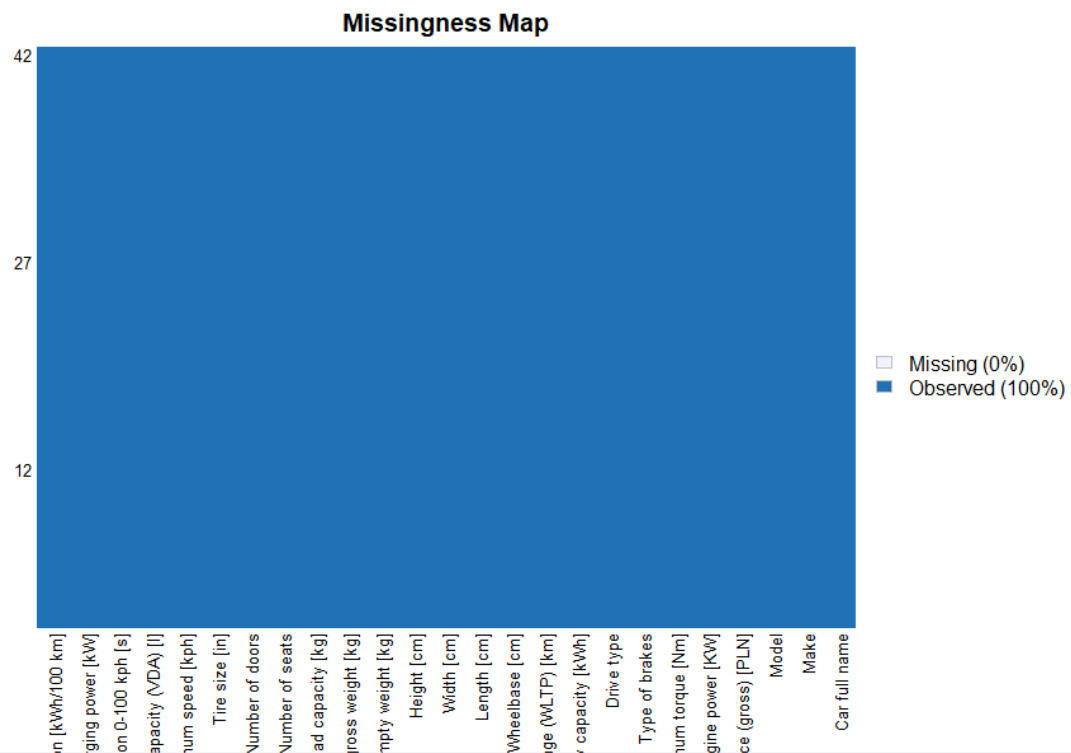
Maximum DC charging power [kw] mean - Energy consumption [kwh/100 km]
Mode :logical              Mode :logical
FALSE:53                    FALSE:44
TRUE :9
```



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 processar 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)	4Wt
Audi e-tron 50 quattro	Audi	e-tron 50 quattro	308400	313	540	disc (front + rear)	4Wt
Audi e-tron S quattro	Audi	e-tron S quattro	414900	503	973	disc (front + rear)	4Wt
Audi e-tron Sportback 50 quattro	Audi	e-tron Sportback 50 quattro	319700	313	540	disc (front + rear)	4Wt
Audi e-tron Sportback 55 quattro	Audi	e-tron Sportback 55 quattro	357000	360	664	disc (front + rear)	4Wt
Audi e-tron Sportback S quattro	Audi	e-tron Sportback S quattro	426200	503	973	disc (front + rear)	4Wt
BMW i3	BMW	i3	169700	170	250	disc (front + rear)	2Wt
BMW i3s	BMW	i3s	184200	184	270	disc (front + rear)	2Wt

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 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 180 ...
 $ 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 ...
 $ Cambio : Factor w/ 3 levels "1","2","3": 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 ...
 $ 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 pratico 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 a 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:

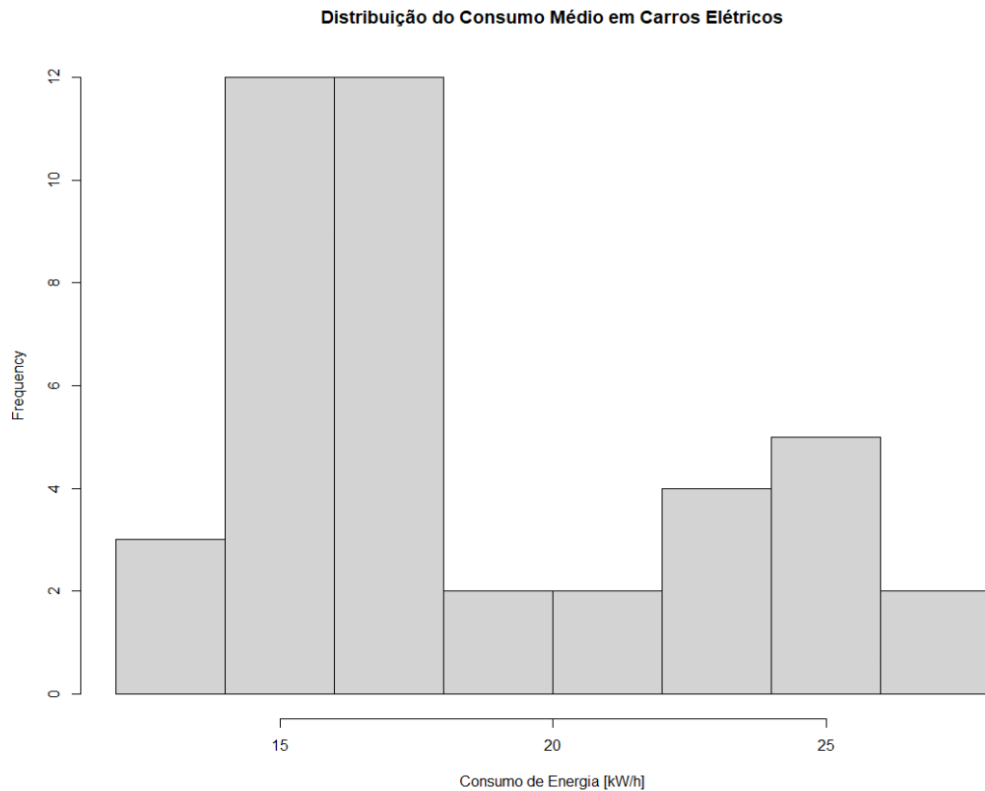
```
> diff(range(VarResp)) # 14.45
[1] 14.45
```

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

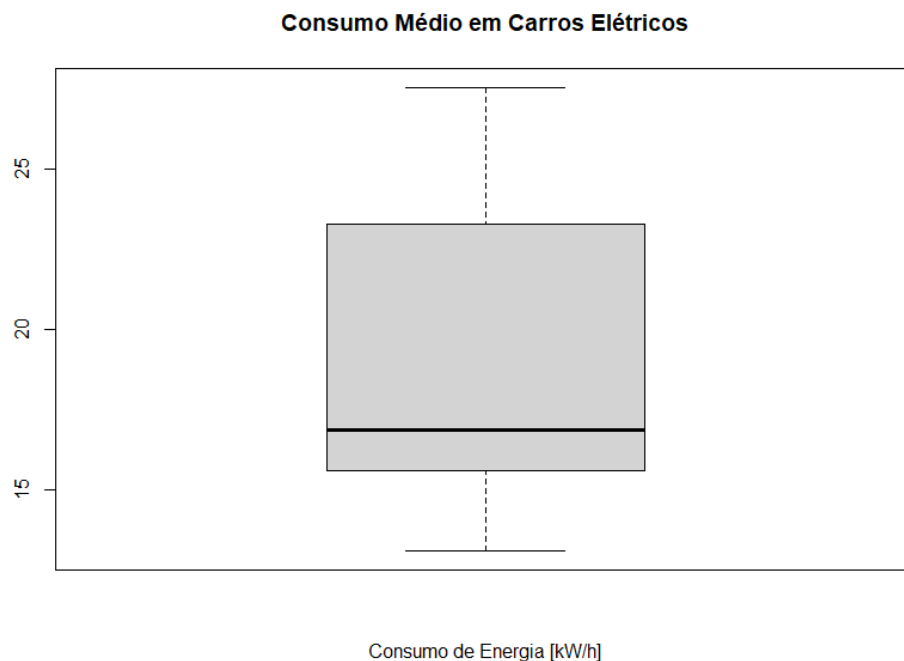
Nosso desvio padrão é de 4,13 kW/h e a variância de 17,09.

```
> 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 $p\text{-value} >$ que 0.05 não podemos rejeitar H_0
 - Se $p\text{-value} <$ que 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 $p\text{-value}$ é menor que 0.05 , sendo assim, não podemos considerar que os dados da Variável resposta seguem uma distribuição Normal, ou seja, rejeitamos H_0 .

Análise das Variáveis Numéricas

Inicialmente, vamos verificar um resumo de cada variável:

```
summary(VarNum)
```

```
> 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-Hitogramas de cada Variável

VarNum01 <- ggplot(VarNum) + geom_histogram(aes(x = PrecoMin))
```

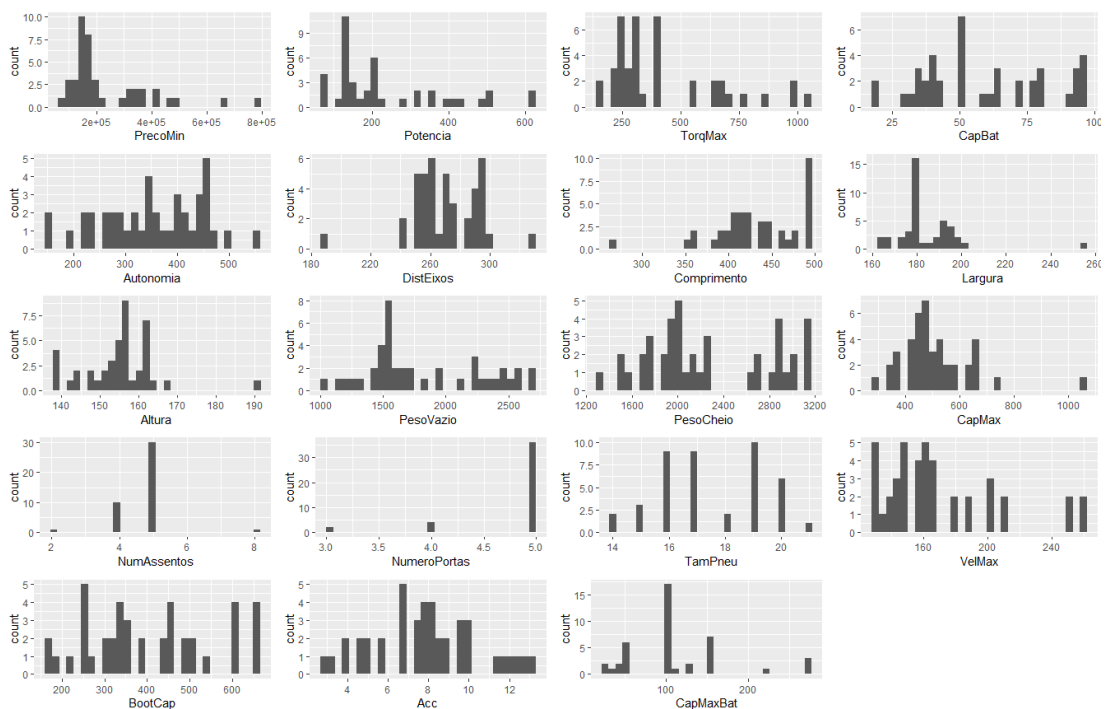


```

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)

```

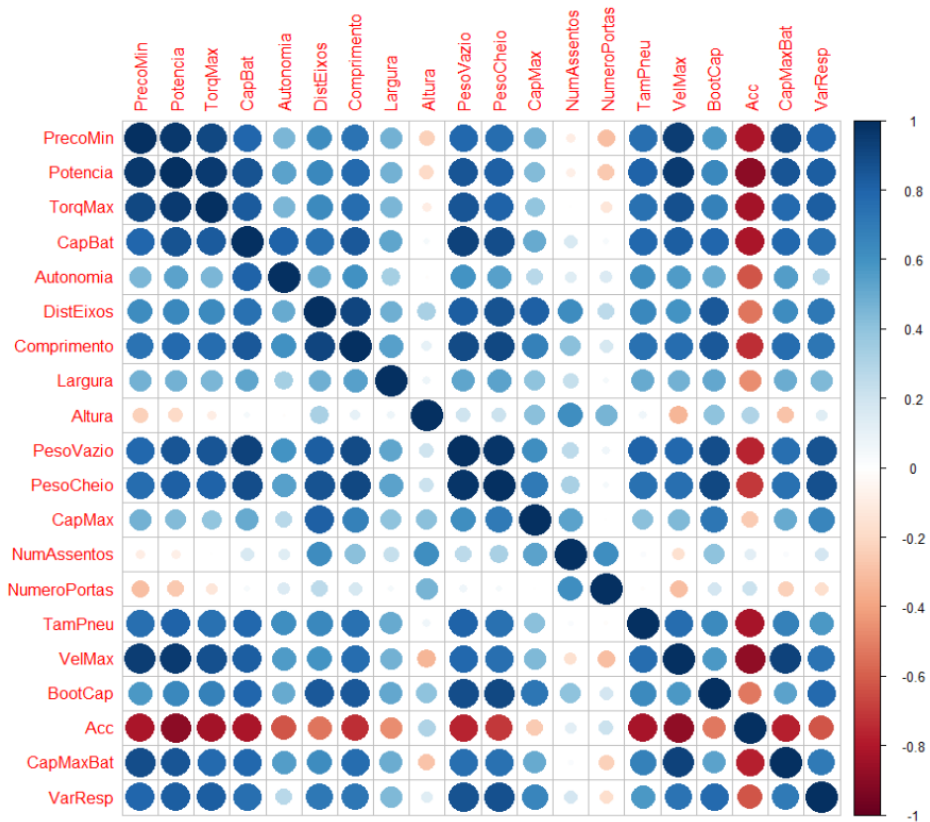


Percebemos que em nosso dataset temos carros de categorias diferentes, apesar de mesmo combustível propulsor, sendo assim um dataset misto, não condicionando a categoria dos veículos. Talvez seria interessante obtermos mais dados de veículos e categorizar os mesmos (etiquetar) para que possamos analisar 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)))
```

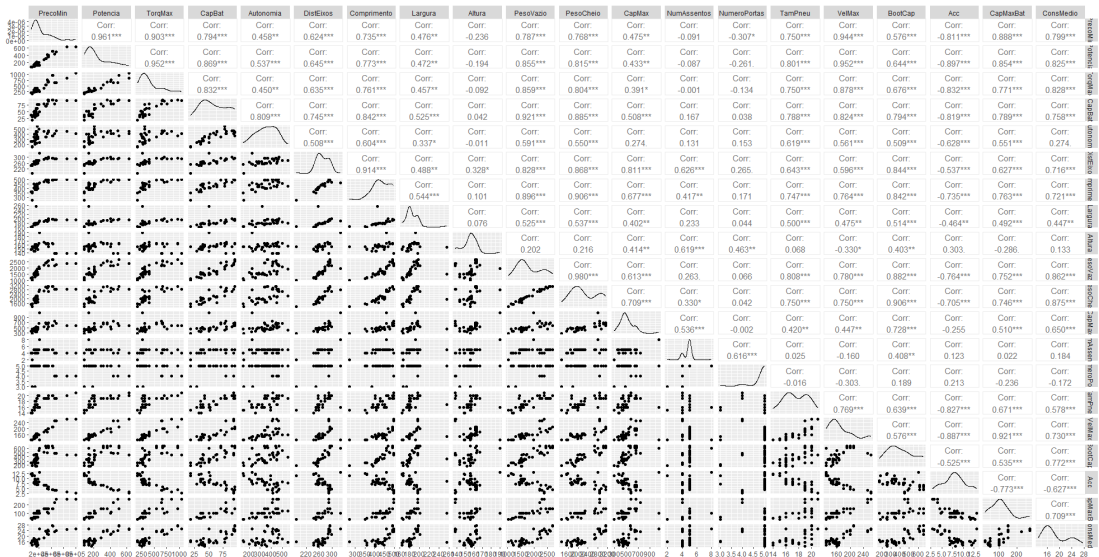


Podemos identificar aqui uma excessiva correlação entre diversas variáveis, o que possivelmente causa confusão durante nossa modelagem preditiva. Correlações como Peso/potencia, DistEixos/CapMax, são relativamente óbvias ao modelo de negócio. Porém observando a nossa Variável Resposta, podemos ver altas correlações com as variáveis, Preço, Potencia, TorqMax, CapBat, Peso Vazio, Pezo 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 multi variável.

```
# 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 cambio, o que é melhor para nosso modelo preditivo.

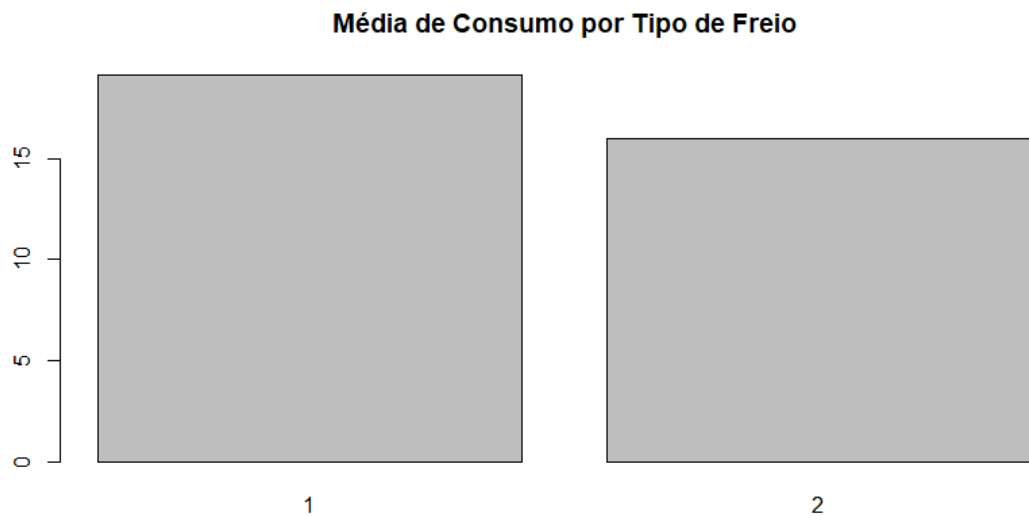
Vamos analisar como é Consumo de Energia para cada tipo de variável, calculando a média para cada categoria.

```
media_por_tipo_de_freio = dados %>% group_by(Freios) %>%
  summarise(avg_consumo = mean(ConsMedio))

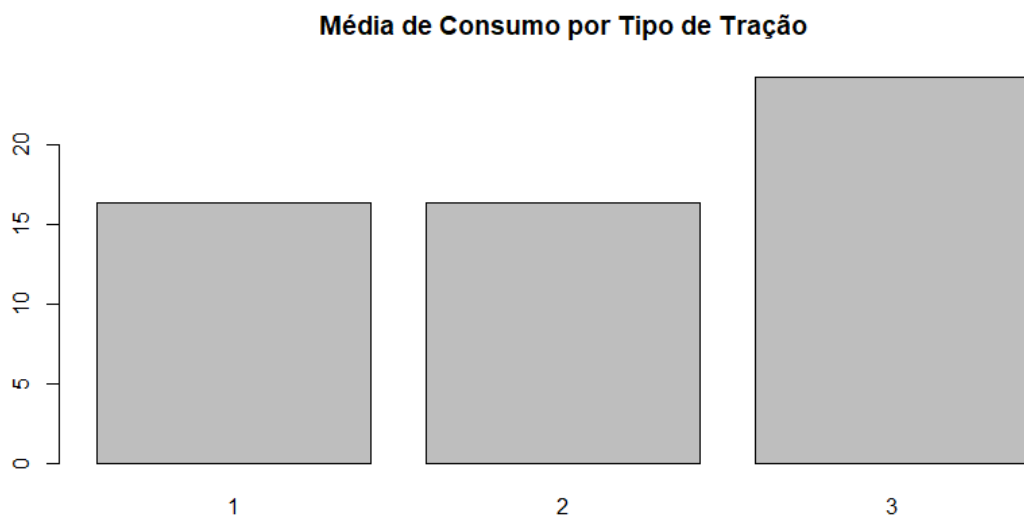
media_por_tipo_de_tracao = dados %>% group_by(Cambio) %>%
  summarise(avg_consumo = mean(ConsMedio))

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 cambio 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 uma fator descoberto é interessante, o consumo médio é muito próximo em carros com tração dianteira ou trazeira.

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)

# Carregando o Dataset

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

dim(dados)
dados <- dados[, 5:26]
dim(dados)
View(dados)
```

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
-1.25615 -0.54454 -0.04457  0.50914  1.49087

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.958e+01  1.373e+01   2.884  0.00918 **
PrecoMin     -2.417e-07  6.370e-06  -0.038  0.97011
Potencia     1.683e-02  1.130e-02   1.489  0.15209
TorqMax       9.646e-04  4.213e-03   0.229  0.82124
Freios       -8.488e-01  1.047e+00  -0.811  0.42700
Cambio       2.126e+00  8.627e-01   2.464  0.02292 *
CapBat       1.625e-01  6.469e-02   2.512  0.02068 *
Autonomia    -2.250e-02  6.994e-03  -3.216  0.00433 **
DistEixos    -3.495e-02  5.713e-02  -0.612  0.54761
Comprimento  9.998e-03  3.030e-02   0.330  0.74487
Largura     -1.359e-02  1.403e-02  -0.969  0.34410
Altura       1.474e-02  6.264e-02   0.235  0.81631
PesoVazio   -3.630e-03  5.665e-03  -0.641  0.52896
PesoCheio    2.290e-04  2.990e-03   0.077  0.93971
CapMax       8.274e-03  5.136e-03   1.611  0.12289
NumAssentos  5.048e-01  1.376e+00   0.367  0.71751
NumeroPortas -5.646e-01  7.909e-01  -0.714  0.48351
TamPneu     -6.330e-01  2.655e-01  -2.384  0.02715 *
VelMax      -8.463e-02  5.522e-02  -1.533  0.14103
BootCap      5.269e-03  4.814e-03   1.095  0.28672
Acc         -1.533e-01  3.061e-01  -0.501  0.62193
CapMaxBat    2.012e-02  1.427e-02   1.410  0.17390
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.946 on 20 degrees of freedom
Multiple R-squared:  0.9745,    Adjusted R-squared:  0.9476
F-statistic: 36.34 on 21 and 20 DF,  p-value: 1.242e-11

```

Podemos verificar que Tamanho do Pneu, Autonomia, Capacidade de Bateria e Cambio são estatisticamente relevantes ao nosso modelo.

As métricas mostram que com R^2 Ajustado de 94,76%, conseguimos afirmar que nosso modelo consegue explicar 94% 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 Regularização ao Dataset

# Criando um função de normalização

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

# Aplicando a Função as Variáveis do Modelo

dados_MinMax <- as.data.frame(lapply(dados[-22], normalizar))
dados_MinMax <- dados_MinMax %>% cbind(dados$ConsMedio)
colnames(dados_MinMax)[22] <- 'ConsMedio'
View(dados_MinMax)
str(dados_MinMax)

```

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

```
FeaturesLMS <- lm(ConsMedio ~ ., data = dados_MinMax)  
summary(FeaturesLMS)
```

```
Call:  
lm(formula = ConsMedio ~ ., data = dados_MinMax)  
  
Residuals:  
    Min       1Q   Median       3Q      Max   
-1.25615 -0.54454 -0.04457  0.50914  1.49087  
  
Coefficients:  
              Estimate Std. Error t value Pr(>|t|)      
(Intercept)   17.1539      3.1959   5.368 2.97e-05 ***  
PrecoMin       -0.1720      4.5348  -0.038  0.97011      
Potencia        9.1368      6.1362   1.489  0.15209      
TorqMax         0.8585      3.7499   0.229  0.82124      
Freios        -0.8488      1.0468  -0.811  0.42700      
Cambio         4.2517      1.7254   2.464  0.02292 *     
CapBat        12.5794      5.0070   2.512  0.02068 *     
Autonomia     -9.0209      2.8047  -3.216  0.00433 **    
DistEixos     -4.8996      8.0094  -0.612  0.54761      
Comprimento    2.2676      6.8725   0.330  0.74487      
Largura       -1.2409      1.2805  -0.969  0.34410      
Altura         0.7697      3.2700   0.235  0.81631      
PesoVazio     -6.0253      9.4034  -0.641  0.52896      
PesoCheio      0.4167      5.4414   0.077  0.93971      
CapMax         6.3376      3.9344   1.611  0.12289      
NumAssentos    3.0290      8.2546   0.367  0.71751      
NumeroPortas  -1.1293      1.5818  -0.714  0.48351      
TamPneu       -4.4307      1.8585  -2.384  0.02715 *     
VelMax       -11.0022      7.1786  -1.533  0.14103      
BootCap        2.5767      2.3541   1.095  0.28672      
Acc           -1.5793      3.1531  -0.501  0.62193      
CapMaxBat      4.9887      3.5380   1.410  0.17390      
---  
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.946 on 20 degrees of freedom  
Multiple R-squared:  0.9745,    Adjusted R-squared:  0.9476  
F-statistic: 36.34 on 21 and 20 DF,  p-value: 1.242e-11
```

Temos então exatamente a mesma resposta do algoritmo de regressão linear, sendo assim, vamos manter os dados na mesma escala para continuar com nosso processo de criação do modelo base.

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)
```

```
# Carregando o Dataset
```

```
dados <- read.csv2('dados/dados_preparados.csv', header = TRUE)  
dim(dados)
```

```
summary(is.na(dados))
dados <- dados[-1]
str(dados)
View(dados)
```

Construindo o Modelo Base e Analisando o Resultado

Divisão dos Dados em Treino e Teste

```
# 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, -5]

View(dados_teste)
View(dados_treino)
```

Modelo Base

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

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

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

Residuals:
    Min       1Q   Median       3Q      Max
-2.8029 -0.8202 -0.0275  0.9220  4.1377

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  16.1859     0.6447   25.108 < 2e-16 ***
Cambio         1.3460     1.0881    1.237  0.2264
CapBat        21.5268     2.3071    9.331 4.34e-10 ***
Autonomia    -15.7830     1.9945   -7.913 1.28e-08 ***
TamPneu       -3.1759     1.8531   -1.714  0.0976 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.401 on 28 degrees of freedom
Multiple R-squared:  0.8967,    Adjusted R-squared:  0.882
F-statistic: 60.78 on 4 and 28 DF,  p-value: 2.127e-13
```

Nosso modelo base possui em treinamento, um R^2 de 88% ajustado, o que significa que conseguimos explicar o consumo médio com 88% de variabilidade das variáveis independentes. Um valor bom para a métrica.

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

```
# Testando e Avaliando o Modelo

Previsao01 <- data.frame(Previsao = predict(ModeloBase, dados_teste))
Previsao01

# Analisando o resíduo do modelo

ConsumoTeste <- data.frame(Target = dados[-separador, 5])
```



```

Res_ModeloBase <- data.frame(Residuo = ConsumoTeste$Target - Previsao01)

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

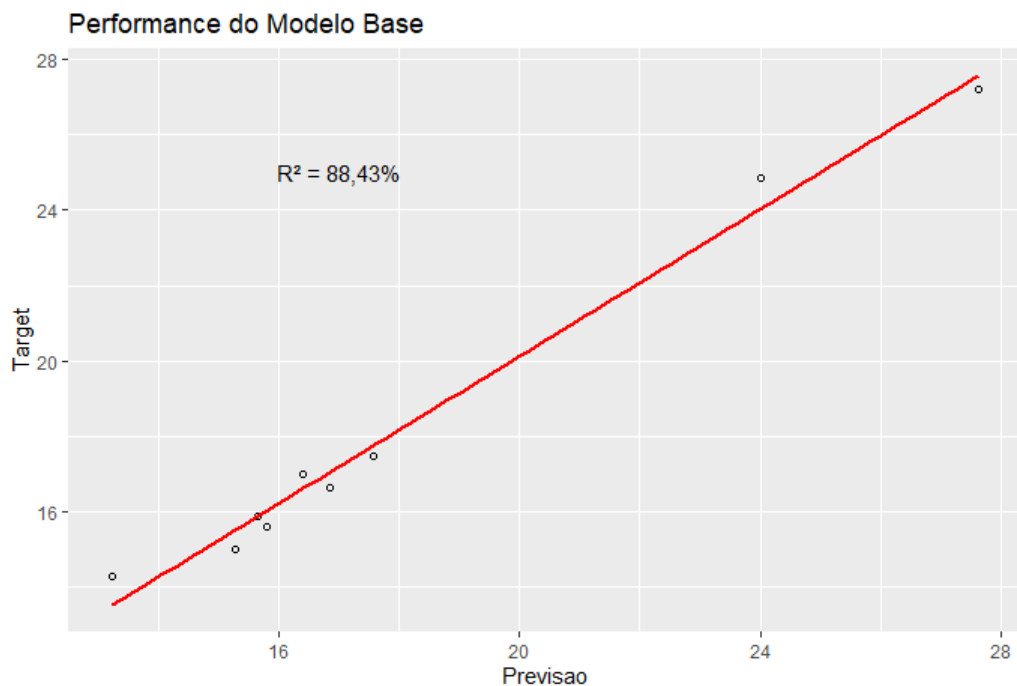
head(FitModeloBase)
summary(FitModeloBase)

```

Target	Previsao	Residuo	Target	Previsao	Residuo
Min.:14.30	Min.:13.22	Min.: -0.43820	1	27.2	27.63820 -0.43820137
1st Qu.:15.60	1st Qu.:15.65	1st Qu.: -0.20091	2	14.3	13.22488 1.07512269
Median:16.65	Median:16.41	Median: -0.06369	3	17.5	17.56369 -0.06369144
Mean:18.22	Mean:18.05	Mean: 0.17648	4	15.0	15.28271 -0.28270557
3rd Qu.:17.50	3rd Qu.:17.56	3rd Qu.: 0.59477	5	15.9	15.64659 0.25341266
Max.:27.20	Max.:27.64	Max.: 1.07512	6	15.6	15.79437 -0.19437256

Podemos ver que temos uma média nos resíduos próximo 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.



Modelo V02

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

```

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

```

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

Residuals:
    Min       1Q   Median       3Q      Max
-2.8029 -0.8202 -0.0275  0.9220  4.1377

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   16.1859     0.6447   25.108 < 2e-16 ***
Cambio         1.3460     1.0881    1.237  0.2264
CapBat        21.5268     2.3071    9.331 4.34e-10 ***
Autonomia    -15.7830     1.9945   -7.913 1.28e-08 ***
TamPneu       -3.1759     1.8531   -1.714  0.0976 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.401 on 28 degrees of freedom
Multiple R-squared:  0.8967,    Adjusted R-squared:  0.882
F-statistic: 60.78 on 4 and 28 DF,  p-value: 2.127e-13
```

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

```
Previsao02 <- data.frame(Previsao = predict(ModeloV02, dados_teste))
Previsao02

# Analisando o resíduo do modelo

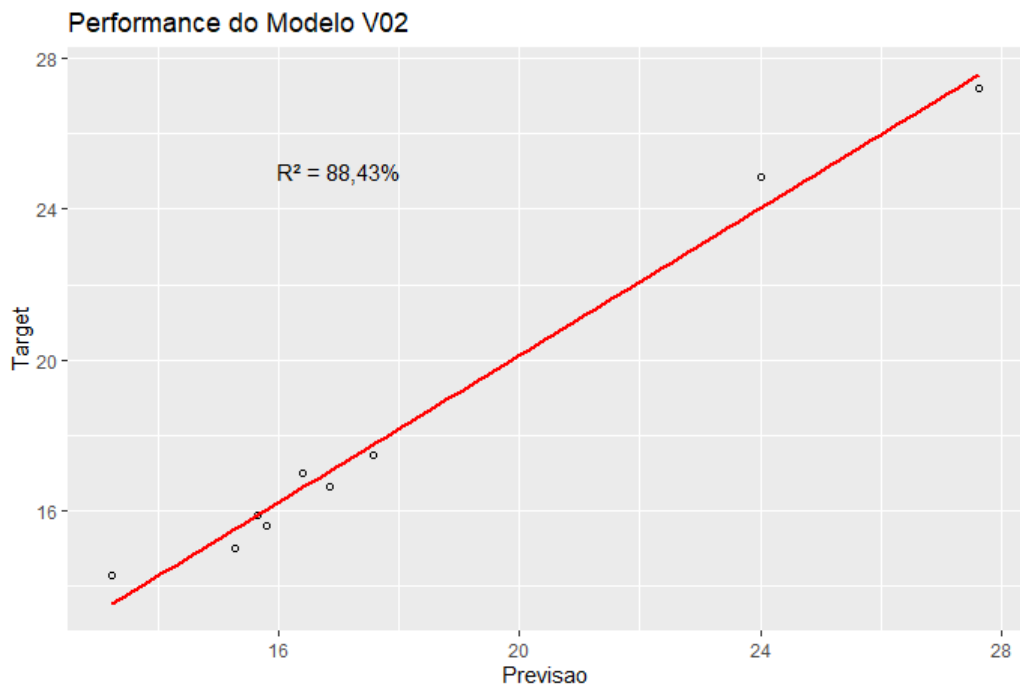
Res_ModeloV02 <- data.frame(Residuo = ConsumoTeste$Target - Previsao02$Previsao)

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

head(FitModeloV02)
summary(FitModeloV02)
```

Target	Previsao	Residuo
Min. :14.30	Min. :13.22	Min. :-0.43820
1st Qu.:15.60	1st Qu.:15.65	1st Qu.: -0.20091
Median :16.65	Median :16.41	Median : -0.06369
Mean :18.22	Mean :18.05	Mean : 0.17648
3rd Qu.:17.50	3rd Qu.:17.56	3rd Qu.: 0.59477
Max. :27.20	Max. :27.64	Max. : 1.07512

	Target	Previsao	Residuo
1	27.2	27.63820	-0.43820137
2	14.3	13.22488	1.07512269
3	17.5	17.56369	-0.06369144
4	15.0	15.28271	-0.28270557



Modelo V03

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

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

```
Boosted Linear Model

33 samples
 4 predictor

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

  mstop  RMSE      Rsquared  MAE
    50   3.137419  0.6272396  2.667200
    100  2.951202  0.6280721  2.393349
    150  2.845333  0.6398828  2.265874

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.
```

Neste modelo tivemos um R^2 de 63,98% 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.

```
Previsao03 <- data.frame(Previsao = predict(ModeloV03, dados_teste))
Previsao03
```

```
# Analisando o resíduo do modelo

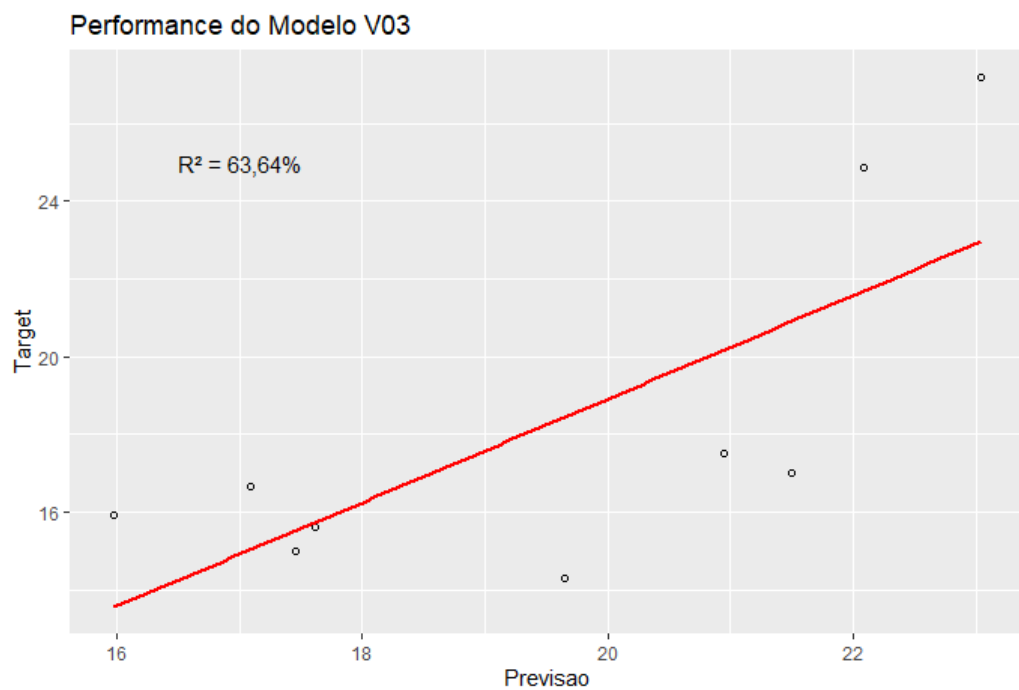
Res_ModeloV03 <- data.frame(Residuo = ConsumoTeste$Target - Previsao03$Previsao)

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

head(FitModeloV03)
summary(FitModeloV03)
```

Target	Previsao	Residuo
Min.:14.30	Min.:15.97	Min.: -5.35011
1st Qu.:15.60	1st Qu.:17.46	1st Qu.: -3.44976
Median :16.65	Median :19.65	Median : -2.01082
Mean :18.22	Mean :19.49	Mean : -1.26334
3rd Qu.:17.50	3rd Qu.:21.50	3rd Qu.: -0.07321
Max.:27.20	Max.:23.05	Max.: 4.15415

	Target	Previsao	Residuo
1	27.2	23.04585	4.15414786
2	14.3	19.65011	-5.35011209
3	17.5	20.94976	-3.44975553
4	15.0	17.46152	-2.46151719
5	15.9	15.97321	-0.07320515
6	15.6	17.61082	-2.01081709



Modelo V04

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

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

```

glmnet

33 samples
4 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  0.006041044  1.735157  0.8300400  1.309803
0.10  0.060410437  1.796147  0.8125245  1.339386
0.10  0.604104372  2.407918  0.6758986  1.800427
0.55  0.006041044  1.732466  0.8305748  1.307999
0.55  0.060410437  1.780290  0.8159601  1.324401
0.55  0.604104372  2.647387  0.6119737  1.958751
1.00  0.006041044  1.729771  0.8311295  1.306058
1.00  0.060410437  1.766342  0.8190449  1.311602
1.00  0.604104372  2.877818  0.5569342  2.145376

RMSE was used to select the optimal model using the
smallest value.
The final values used for the model were alpha = 1 and lambda
= 0.006041044.

```

Atingimos aqui 83% de R^2 , uma melhora drástica em relação ao modelo V03, porém ainda menor que nosso modelo base.

```

Previsao04 <- data.frame(Previsao = predict(ModeloV04, dados_teste))
Previsao04

# Analisando o resíduo do modelo

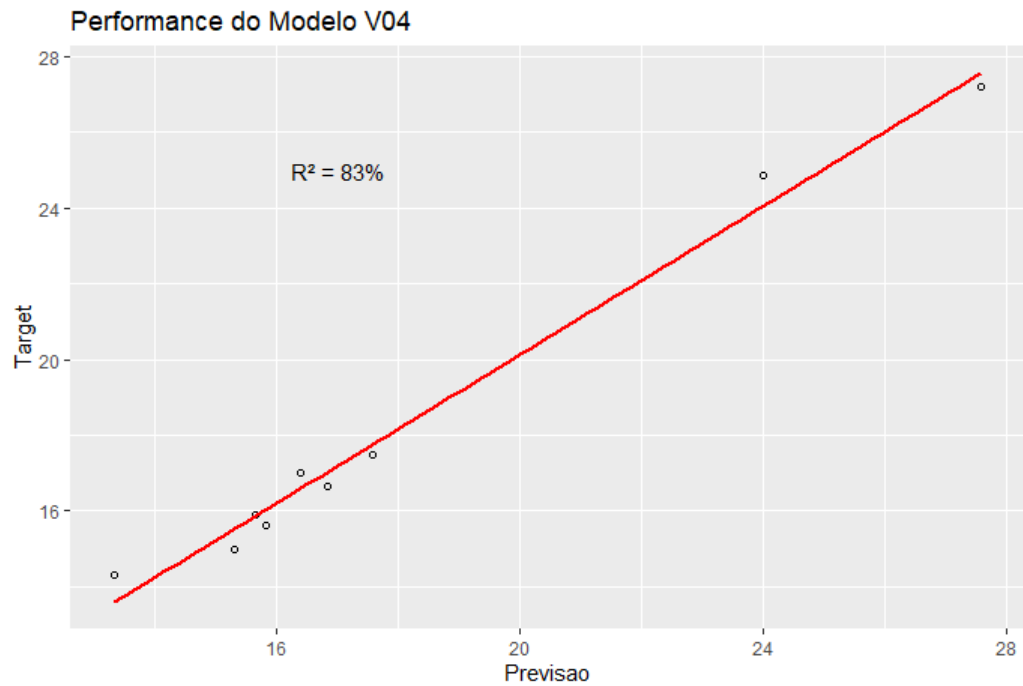
Res_ModeloV04 <- data.frame(Residuo = ConsumoTeste$Target - Previsao04$Previsao)

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

head(FitModeloV03)
summary(FitModeloV04)

```

Target	Previsao	Residuo	Target	Previsao	Residuo
Min. :14.30	Min. :13.32	Min. :-0.39316	1	27.2 23.04585	4.15414786
1st Qu.:15.60	1st Qu.:15.66	1st Qu.: -0.21751	2	14.3 19.65011	-5.35011209
Median :16.65	Median :16.40	Median :-0.07526	3	17.5 20.94976	-3.44975553
Mean :18.22	Mean :18.06	Mean : 0.16495	4	15.0 17.46152	-2.46151719
3rd Qu.:17.50	3rd Qu.:17.58	3rd Qu.: 0.60119	5	15.9 15.97321	-0.07320515
Max. :27.20	Max. :27.59	Max. : 0.97545	6	15.6 17.61082	-2.01081709



Conclusão

Portanto finalizamos nosso trabalho, disponibilizando o Modelo Base 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 pre-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.