

# Prevendo Despesas Hospitalares\_v2

Evanil Tiengo Junior

## Índice

Introdução .....	1
Objetivos .....	1
Etapa 1 - Coletando os Dados .....	3
Etapa 2 - EDA (Exploratory Data Analysis) .....	3
Etapa 3 - Modelagem .....	11
Etapa 4 - Interpretando o Modelo .....	12
Etapa 5 - Previsão .....	16
Etapa 6 - Avaliando a Performance .....	21
Etapa 7 - Otimização do Modelo .....	21
Conclusão .....	28

## Introdução

Este projeto faz parte da formação Big Data Analytics com R e Microsoft Azure da DSA. Para esta análise, é usado um conjunto de dados simulando despesas médicas hipotéticas para um conjunto de pacientes espalhados por 4 regiões do Brasil. Esse dataset possui 1.338 observações e 7 variáveis. Todo o projeto será descrito de acordo com suas etapas.

## Objetivos

- 1) Empregar técnicas de análise para verificar as correlações entre as variáveis;
- 2) Criar um modelo que faça as previsões, quanto aos gastos de um usuário, quando receber novos conjuntos de dados;
- 3) Analisar as métricas do modelo e tirar conclusões o mesmo.

```
# Local armazenamento
setwd("~/Desktop/Projeto/Projeto3")
getwd()

## [1] "~/Desktop/Projeto/Projeto3"
```

```
# Pacotes utilizados.
#install.packages("dplyr")
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##   filter, lag

## The following objects are masked from 'package:base':
##
##   intersect, setdiff, setequal, union

#install.packages("tidyr")
library(tidyr)
#install.packages("ggplot2")
library(ggplot2)
#install.packages("psych")
library(psych)

##
## Attaching package: 'psych'

## The following objects are masked from 'package:ggplot2':
##
##   %+%, alpha

#install.packages("corrplot")
library(corrplot)

## corrplot 0.84 loaded

#install.packages("caTools")
library(caTools)
#install.packages("car")
library(car)

## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:psych':
##
##   logit

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

## Etapa 1 - Coletando os Dados

Os dados foram fornecidos pela DSA. Sendo assim preciso realizar a carga. Antes de realizar a carga do arquivo é necessário saber o formato do mesmo, que neste caso é .csv!

```
Despesas <- read.csv("~/Desktop/Projeto/Projeto3/despesas.csv")
```

## Etapa 2 - EDA (Exploratory Data Analysis)

*# Com o comando abaixo é possível identificar as classes de cada variável*  
`str(Despesas)`

```
## 'data.frame': 1338 obs. of 7 variables:
## $ idade : int 19 18 28 33 32 31 46 37 37 60 ...
## $ sexo : Factor w/ 2 levels "homem","mulher": 2 1 1 1 1 2 2 2 1 2 .
..
## $ bmi : num 27.9 33.8 33 22.7 28.9 25.7 33.4 27.7 29.8 25.8 ...
## $ filhos : int 0 1 3 0 0 0 1 3 2 0 ...
## $ fumante: Factor w/ 2 levels "nao","sim": 2 1 1 1 1 1 1 1 1 1 ...
## $ regioao : Factor w/ 4 levels "nordeste","norte",...: 3 4 4 1 1 4 4 1
2 1 ...
## $ gastos : num 16885 1726 4449 21984 3867 ...
```

*# Para a previsão irei utilizar a Regressão Linear Múltipla. Sendo assim, irei transformar algumas variáveis em numéricas. Essa transformação é possível pois as variáveis não numéricas são fatores (Têm níveis)*

```
# $sexo: mulher = 1; homem = 2
Despesas$sexo <- as.numeric(Despesas$sexo)
```

```
# $fumante: sim = 1; não = 2
Despesas$fumante <- as.numeric(Despesas$fumante)
```

```
# $regiao: nordeste = 1; norte = 2; sudeste = 3; sul = 4
Despesas$regiao <- as.numeric(Despesas$regiao)
```

```
str(Despesas)
```

```
## 'data.frame': 1338 obs. of 7 variables:
## $ idade : int 19 18 28 33 32 31 46 37 37 60 ...
## $ sexo : num 2 1 1 1 1 2 2 2 1 2 ...
## $ bmi : num 27.9 33.8 33 22.7 28.9 25.7 33.4 27.7 29.8 25.8 ...
## $ filhos : int 0 1 3 0 0 0 1 3 2 0 ...
## $ fumante: num 2 1 1 1 1 1 1 1 1 1 ...
## $ regioao : num 3 4 4 1 1 4 4 1 2 1 ...
## $ gastos : num 16885 1726 4449 21984 3867 ...
```

*# Identificação de NA's e Vazios!*  
`any(is.na(Despesas))`

```
## [1] FALSE

any(Despesas == "")

## [1] FALSE

# False significa que não existe nenhum campo com NA ou Vazio!

# Abaixo temos um resumo dos dados
str(Despesas)

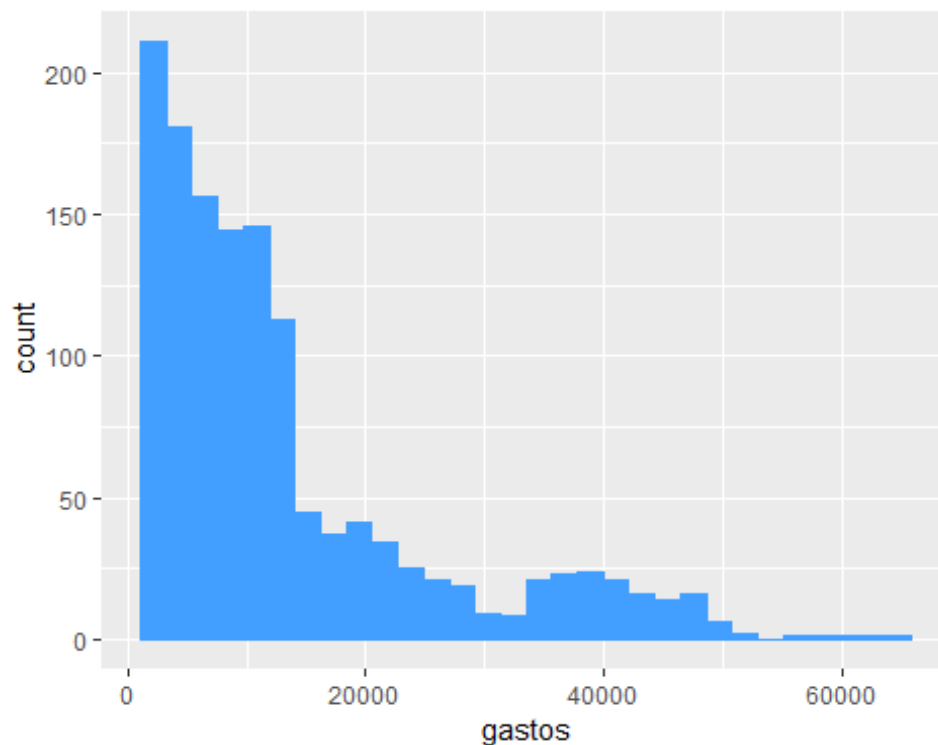
## 'data.frame': 1338 obs. of 7 variables:
## $ idade : int 19 18 28 33 32 31 46 37 37 60 ...
## $ sexo : num 2 1 1 1 1 2 2 2 1 2 ...
## $ bmi : num 27.9 33.8 33 22.7 28.9 25.7 33.4 27.7 29.8 25.8 ...
## $ filhos : int 0 1 3 0 0 0 1 3 2 0 ...
## $ fumante: num 2 1 1 1 1 1 1 1 1 1 ...
## $ regiao : num 3 4 4 1 1 4 4 1 2 1 ...
## $ gastos : num 16885 1726 4449 21984 3867 ...

# Aqui podemos tirar algumas observações:
```

Após os testes acima, conclui-se que os dados estão ok!

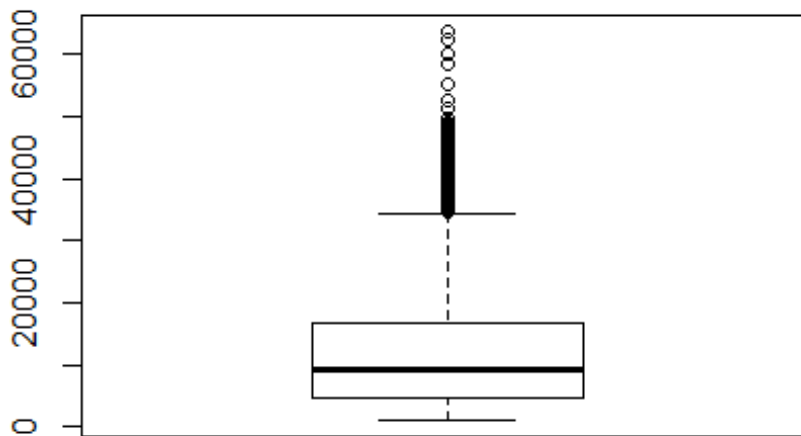
```
# Medidas da tendencia central da variavel $gastos
ggplot(Despesas, aes(gastos)) +
  geom_histogram(color = "#429FFF", fill = "#429FFF")

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



Pelo histograma podemos identificar uma distribuição Unimodal e um enviesamento a direita (Mediana < Media). Com isso podemos observar que a distribuição dos dados concentra-se em torno dos valores mínimos. Como pode ser observado no resumo, o 3rd Quartil que representa 75% dos valores, é igual a 16640 reais. Isso quer dizer que 75% dos gastos são inferiores a 16640 reais. Vamos fazer um boxplot para confirmar.

```
boxplot(Despesas$gastos)
```



Como pode ser observado temos muitos outliers superiores.

```
# O valor do primeiro outlier é dado pela eq: Q1 + (1.5*IQR)
# O IQR é o Q3 - Q1
IQR = 16640 - 4740
IQR

## [1] 11900

Outlier_Superior = 16640 + (1.5*IQR)
Outlier_Superior

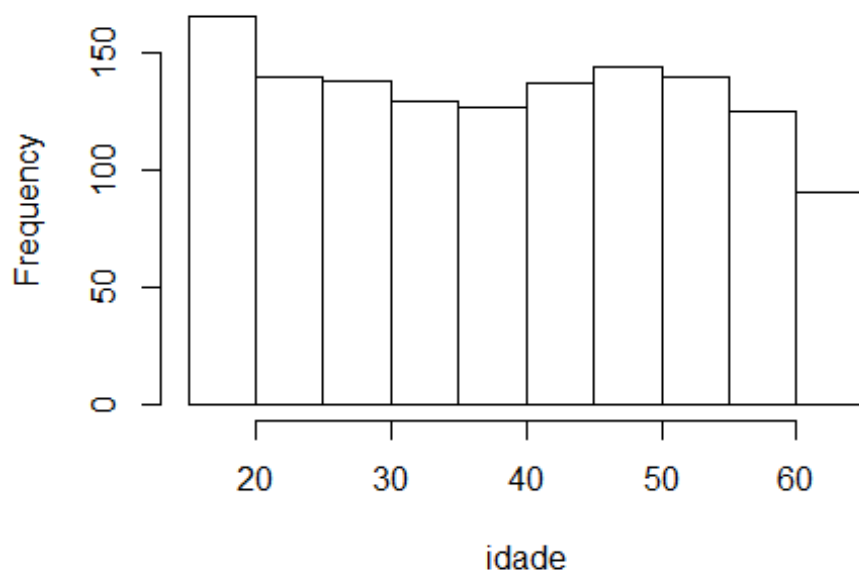
## [1] 34490

#34490
```

Como os valores dos outliers são elevados não irei retirá-los do modelo nesse momento. Mais abaixo farei um teste retirando os outliers para saber a influência deles.

Gráfico e Tabela de contingência para analisar a distribuição das variáveis

```
# $idade: distribuição uniforme
hist(Despesas$idade, xlab = "idade")
```



```
count(Despesas, idade)
```

```
## # A tibble: 47 x 2
```

```
##   idade     n
```

```
##   <int> <int>
```

```
## 1     18    69
```

```
## 2     19    68
```

```
## 3     20    29
```

```
## 4     21    28
```

```
## 5     22    28
```

```
## 6     23    28
```

```
## 7     24    28
```

```
## 8     25    28
```

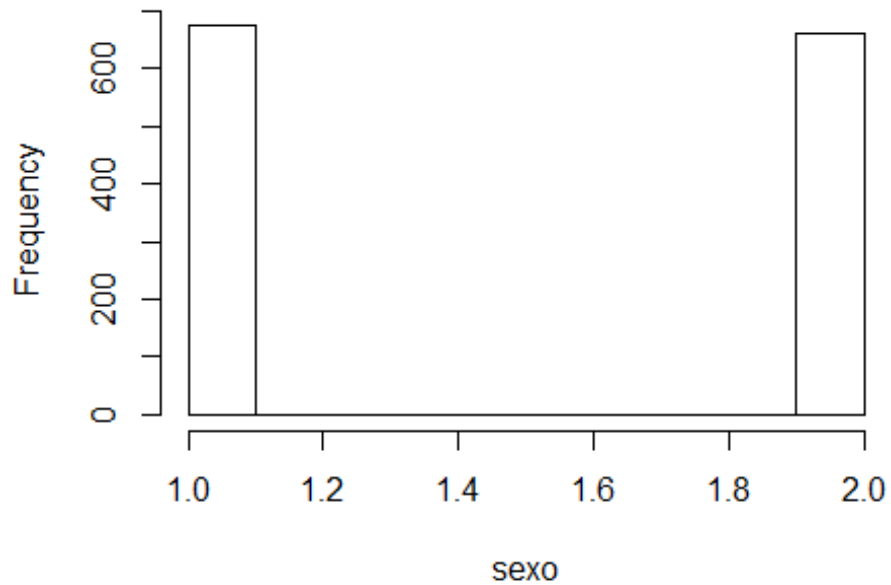
```
## 9     26    28
```

```
## 10    27    28
```

```
## # ... with 37 more rows
```

```
# $sexo: distribuição uniforme
```

```
hist(Despesas$sexo, xlab = "sexo")
```



```
round(prop.table(table(Despesas$sexo)) * 100, digits = 1)
```

```
##
##      1      2
## 50.5 49.5
```

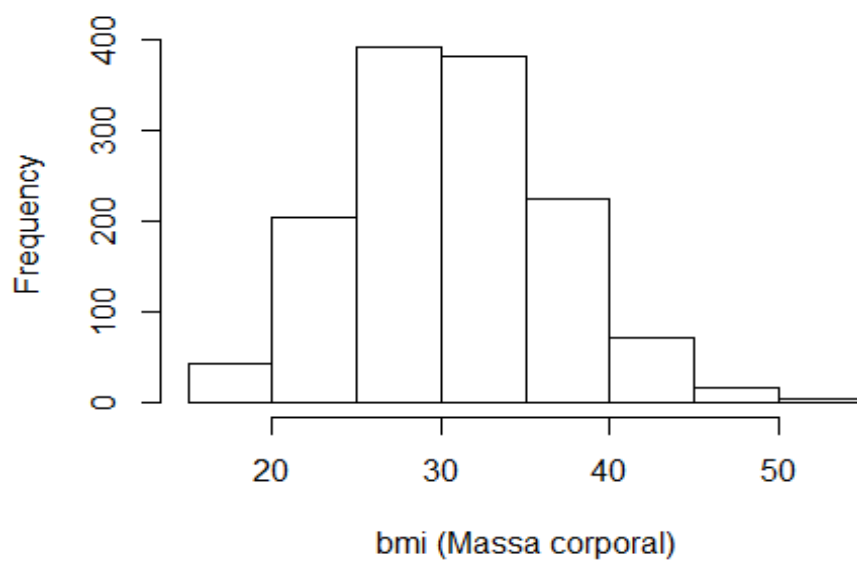
*# Em termos de porcentagem temos:*

*# mulher = 50.5%*

*# homem = 49.5%*

*# \$bmi: distribuição simétrica.*

```
hist(Despesas$bmi, xlab = "bmi (Massa corporal)")
```

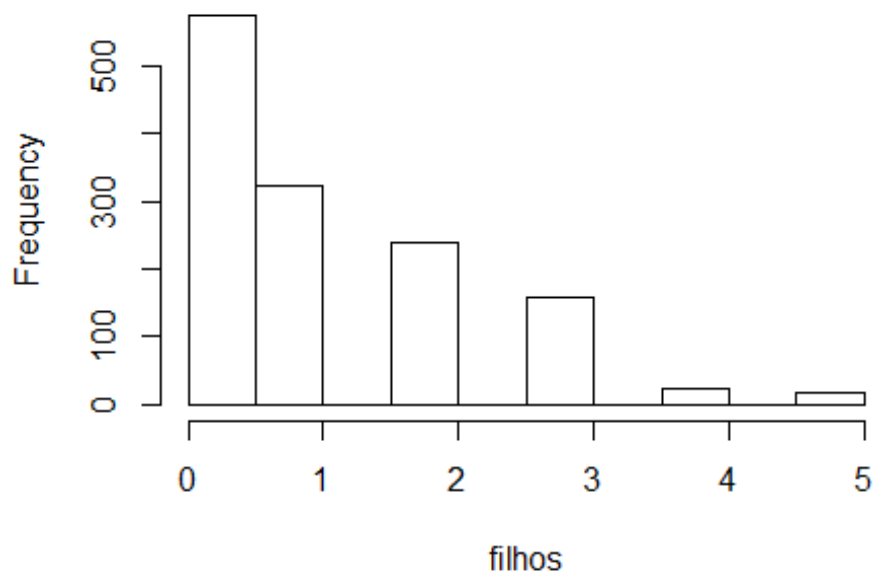


```
count(Despesas, bmi)

## # A tibble: 275 x 2
##   bmi      n
##   <dbl> <int>
## 1 16      1
## 2 16.8    2
## 3 17.2    1
## 4 17.3    3
## 5 17.4    2
## 6 17.5    1
## 7 17.7    1
## 8 17.8    2
## 9 17.9    1
## 10 18     1
## # ... with 265 more rows
```

Os dados de bmi (massa corporal) fornecidos tendem a centralidade e os valores são mais comuns entre 25 a 35 bmi.

```
# $filhos: distribuição enviesada a direita.
hist(Despesas$filhos, xlab = "filhos")
```



```
table(Despesas$filhos)

##
##  0    1    2    3    4    5
## 574 324 240 157  25  18
```

Os dados fornecidos tendem a ter um enviesamento a direita, o que indica que a distribuição dos dados concentra-se em torno dos valores mínimos. Em termos de porcentagem temos:



0 filhos ->  $574 / 1338 = 42,9\%$

1 filhos ->  $324 / 1338 = 24,2\%$

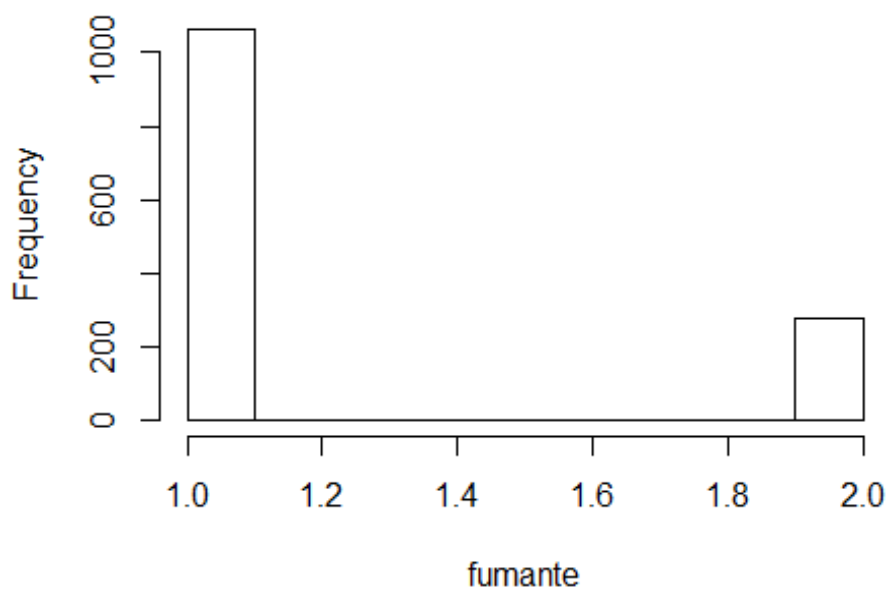
2 filhos ->  $240 / 1338 = 17,9\%$

3 filhos ->  $157 / 1338 = 11,8\%$

4 filhos ->  $25 / 1338 = 1,9\%$

5 filhos ->  $18 / 1338 = 1,3\%$

```
# $fumante: distribuição desequilibrada.  
hist(Despesas$fumante, xlab = "fumante")
```



```
table(Despesas$fumante)
```

```
##  
##      1      2  
## 1064   274
```

```
round(prop.table(table(Despesas$fumante)) * 100, digits = 1)
```

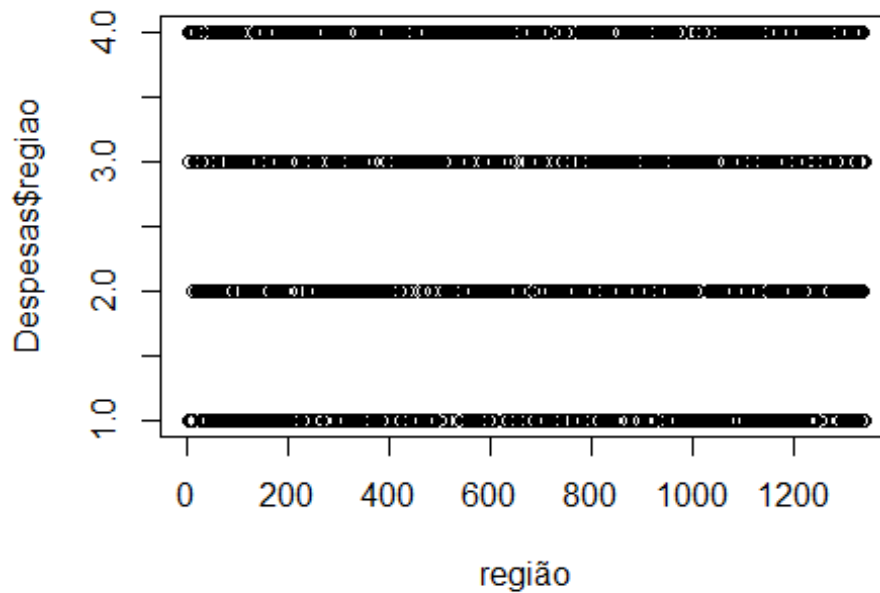
```
##  
##      1      2  
## 79.5  20.5
```

Em termos de porcentagem temos:

Não Fumante ->  $1064 / 1338 = 79,5\%$

Fumante ->  $274 / 1338 = 20,5\%$

```
# $regiao: distribuição uniforme
plot(Despesas$regiao, xlab = "região")
```



```
table(Despesas$regiao)
```

```
##
##  1  2  3  4
## 325 324 325 364
```

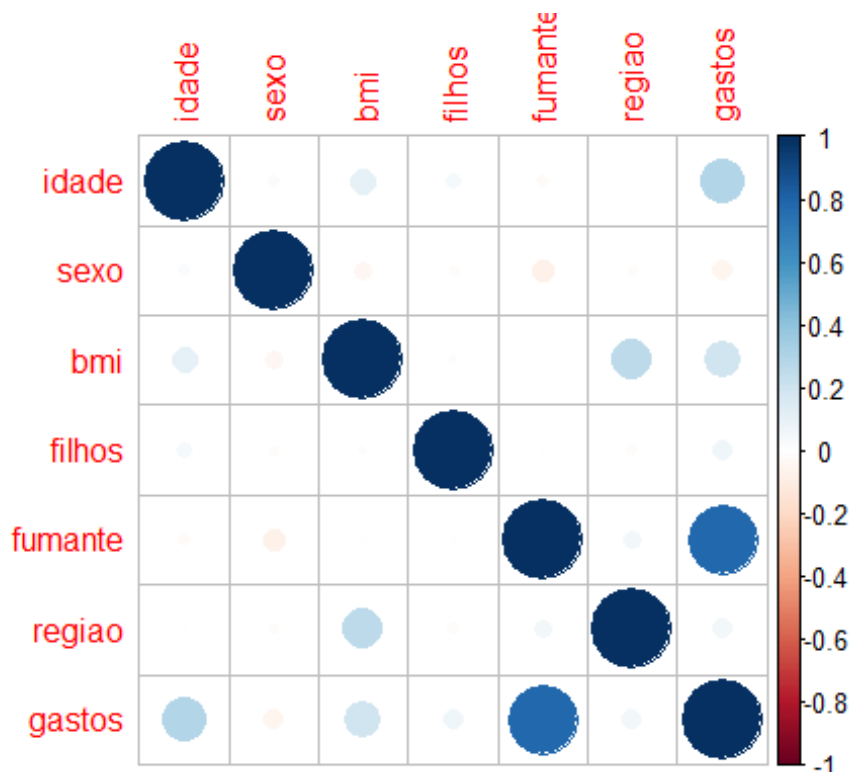
Explorando relacionamento entre as variáveis:

Matriz de Correlação

```
Data_Cor <- cor(Despesas)
Data_Cor
```

```
##          idade          sexo          bmi          filhos          fumante
## idade    1.00000000  0.02085587  0.109341015  0.04246900 -0.025018752
## sexo     0.02085587  1.00000000 -0.046380206 -0.01716298 -0.076184817
## bmi      0.10934101 -0.04638021  1.000000000  0.01264471  0.003968054
## filhos   0.04246900 -0.01716298  0.012644707  1.00000000  0.007673120
## fumante  -0.02501875 -0.07618482  0.003968054  0.00767312  1.000000000
## regiao   -0.00521169 -0.01612099  0.261848704 -0.01925722  0.053929632
## gastos    0.29900819 -0.05729207  0.198576255  0.06799823  0.787251430
##          regiao          gastos
## idade   -0.00521169  0.29900819
## sexo    -0.01612099 -0.05729207
## bmi      0.26184870  0.19857626
## filhos  -0.01925722  0.06799823
## fumante  0.05392963  0.78725143
## regiao   1.00000000  0.05699279
## gastos   0.05699279  1.00000000
```

```
# Visualizando relacionamento entre as variáveis:
corrplot(Data_Cor, method = "circle")
```



O objetivo do estudo da correlação é determinar (mensurar) o grau de relacionamento entre duas variáveis. Existem algumas associações interessantes. Pode-se destacar que a correlação entre as variáveis é positiva e moderada. Por ordem de força temos a fumante, idade e bmi. Estas associações implicam que, à medida que elas aumentam, o custo esperado do seguro saúde sobe!

### Etapa 3 - Modelagem

Treinando o Modelo. No modelo é usado a Regressão Linear Múltipla. Formula da Regressão Linear múltiplas Estimada:

$$y = a + b_0x_1 + b_1x_2 + \dots + X_nx_n$$

Criando amostras randômicas:

```
# Criando amostras randômicas
set.seed(101)
Amostra <- sample.split(Despesas, SplitRatio = 0.70)

# Treinamos o nosso modelo nos dados de treino
# Dados_Treino
Dados_Treino <- subset(Despesas, Amostra == TRUE)
summary(Dados_Treino)
```

```
##      idade      sexo      bmi      filhos
## Min.   :18.00   Min.   :1.000   Min.   :16.0   Min.   :0.000
## 1st Qu.:26.00   1st Qu.:1.000   1st Qu.:26.3   1st Qu.:0.000
## Median :40.00   Median :1.000   Median :30.5   Median :1.000
## Mean   :39.28   Mean   :1.486   Mean   :30.7   Mean   :1.111
## 3rd Qu.:52.00   3rd Qu.:2.000   3rd Qu.:34.7   3rd Qu.:2.000
## Max.   :64.00   Max.   :2.000   Max.   :53.1   Max.   :5.000
##      fumante      regioao      gastos
## Min.   :1.000   Min.   :1.000   Min.   : 1136
## 1st Qu.:1.000   1st Qu.:2.000   1st Qu.: 4828
## Median :1.000   Median :3.000   Median : 9635
## Mean   :1.205   Mean   :2.558   Mean   :13551
## 3rd Qu.:1.000   3rd Qu.:4.000   3rd Qu.:17180
## Max.   :2.000   Max.   :4.000   Max.   :63770
```

*# Fazemos as predições nos dados de teste*

*# Dados\_Testes*

```
Dados_Testes <- subset(Despesas, Amostra == FALSE)
```

*# O modelo utilizado aqui é o de Regressão Linear. No modelo1 estou considerando*

*# todas as variáveis do dataset.*

```
Modelo_v1 <- lm(gastos ~ ., Despesas)
```

## Etapa 4 -Interpretando o Modelo

Nesta etapa iremos analisar o resumo dos parâmetros do Modelo\_v1.

```
summary(Modelo_v1)
```

```
##
## Call:
## lm(formula = gastos ~ ., data = Despesas)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11573.9  -2899.6   -974.5   1418.2  29955.5
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -35842.8     1213.2  -29.545 < 2e-16 ***
## idade       256.9       11.9    21.597 < 2e-16 ***
## sexo        129.0       333.0     0.387  0.69853
## bmi         337.1       28.4    11.872 < 2e-16 ***
## filhos      468.6       137.7     3.402  0.00069 ***
## fumante     23866.0     412.7    57.832 < 2e-16 ***
## regioao     -298.0      152.3    -1.957  0.05057 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6063 on 1331 degrees of freedom
```

```
## Multiple R-squared:  0.7505, Adjusted R-squared:  0.7493
## F-statistic: 667.1 on 6 and 1331 DF,  p-value: < 2.2e-16
```

Avaliando os parâmetros:

O primeiro parâmetro a ser analisado é o nível de significância de acordo com o p-value. O nível de significância é o limite para o p-valor, abaixo do qual assume-se que a hipótese nula é falsa. O p-valor é a probabilidade de se obter uma estatística de teste igual ou mais extrema que a estatística observada a partir de uma amostra de uma população quando ela é verdadeira. Isto significa que o nível de significância é a probabilidade de se rejeitar incorretamente a hipótese nula quando ela é verdadeira. O nível de significância corresponde ao erro do tipo I, cujos valores mais comuns são 10%, 5% e 1%. Iremos considerar o de 10%.

Pode-se observar que as variáveis idade, bmi, filhos, fumante, regioao têm o nível de significância < 10%. Já as outras variáveis não influenciam tanto no modelo. Nesse sentido teremos que elaborar outro modelo, retirando as variáveis que não influenciam ou pouco influenciam na variável dependente. Retirando essas variáveis os valores das variáveis significativas e da interceptação serão alterados.

```
Modelo_v2 <- lm(gastos ~ idade + bmi + filhos + fumante + regioao, Despesas)
summary(Modelo_v2)

##
## Call:
## lm(formula = gastos ~ idade + bmi + filhos + fumante + regioao,
##     data = Despesas)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11634.9  -2890.5   -964.4   1392.1  29899.4
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -35623.28    1072.30  -33.221  < 2e-16 ***
## idade        257.07      11.89   21.620  < 2e-16 ***
## bmi          336.60      28.35   11.871  < 2e-16 ***
## filhos       467.64     137.68    3.397 0.000702 ***
## fumante     23853.93     411.38   57.986  < 2e-16 ***
## regioao     -297.97     152.23   -1.957 0.050517 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6061 on 1332 degrees of freedom
## Multiple R-squared:  0.7504, Adjusted R-squared:  0.7495
## F-statistic: 801 on 5 and 1332 DF,  p-value: < 2.2e-16
```

Comparando os parâmetros dos dois modelos conforme pressuposições abaixo: a) Multicolinearidade

```
vif(Modelo_v1)

##      idade      sexo      bmi      filhos      fumante      regiao
## 1.016252 1.008878 1.090591 1.002746 1.009406 1.078632

vif(Modelo_v2)

##      idade      bmi      filhos      fumante      regiao
## 1.015620 1.088195 1.002453 1.003691 1.078632
```

Não há multicolinearidade nos dois modelos

b) Parcimônia No Modelo\_v1 temos algumas variáveis que não contribuem para o modelo. Sendo assim, o Modelo\_v2 ganha nesse ponto

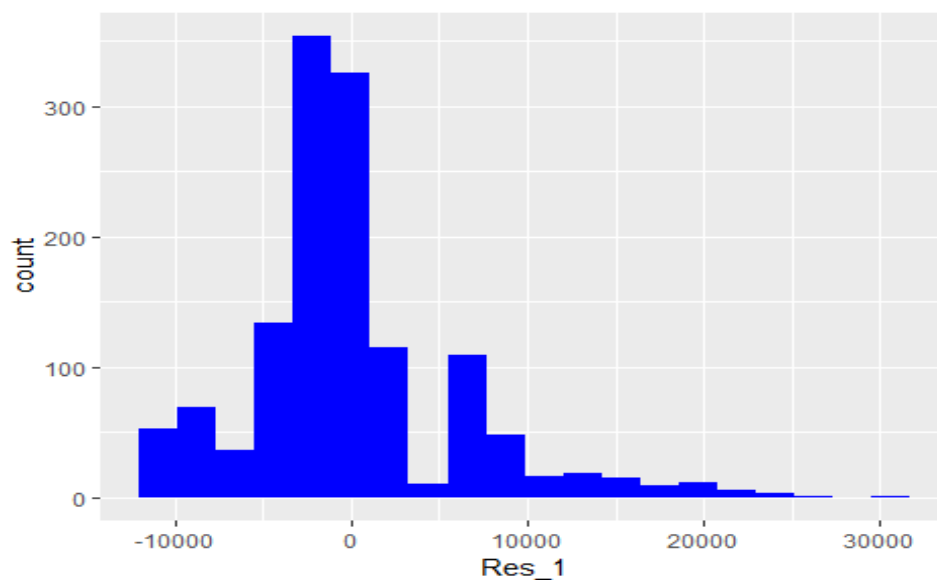
c) Residuais

```
Res_1 <- residuals(Modelo_v1)
Res_2 <- residuals(Modelo_v2)

# Convertendo o objeto para um dataframe
Res_1 <- as.data.frame(Res_1)
head(Res_1)

##      Res_1
## 1 -8655.88369
## 2 -1722.91135
## 3 -2235.91837
## 4 17998.32688
## 5 -1952.47571
## 6   38.01796

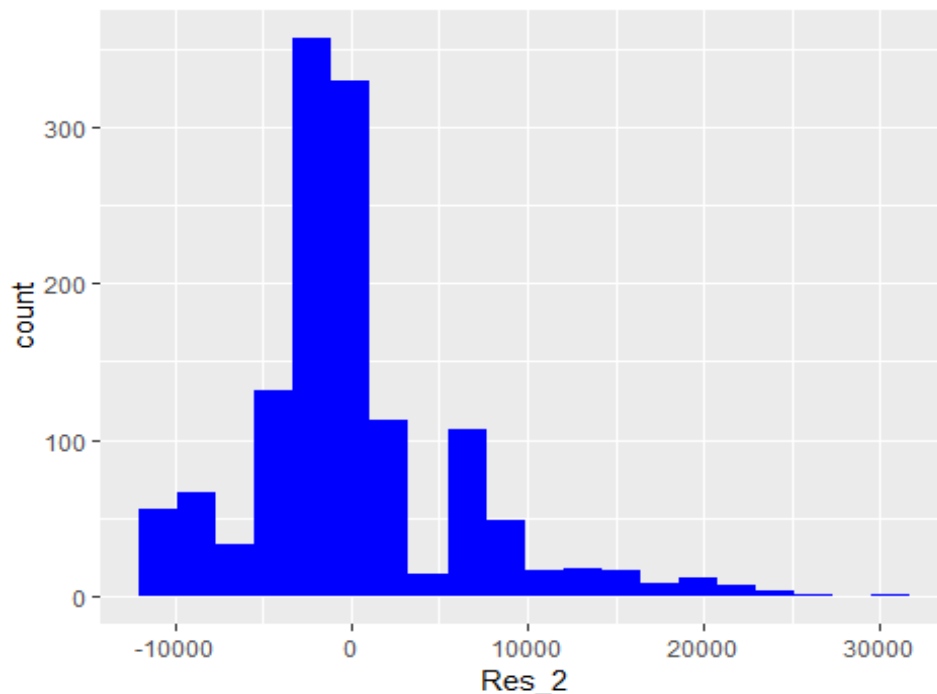
ggplot(Res_1, aes(Res_1)) +
  geom_histogram(bins = 20, fill = 'blue')
```



```
Res_2 <- as.data.frame(Res_2)
head(Res_2)

##           Res_2
## 1 -8581.2590
## 2 -1785.2731
## 3 -2298.0197
## 4 17927.6929
## 5 -2019.7981
## 6    98.0721

ggplot(Res_2, aes(Res_2)) +
  geom_histogram(bins = 20, fill = 'blue')
```



O histograma apresenta uma distribuição parecida com a normal, o que indica que a média entre os valores previstos e os valores observados é próximo de 0 (o que é bom).

d) Os valores de correlação são muito próximos.

e) p-value são iguais

Resultado: Baseado nos itens acima seguirei com o Modelo\_v2.

Interpretação dos parâmetros:

Coefficientes: Como pode ser observado os valores estimados mudaram no Modelo\_v2. Com isso temos a equação da regressão linear múltipla:

$$\text{gastos} = -35623.28 + 257.07\text{idade} + 336.60\text{bmi} + 467.64\text{filhos} + 23853.93\text{fumante} - 297.97\text{regiao}$$

Exemplo: Linha 20 do dataset Despesas

```
Despesas[20,]
```

```
##      idade sexo  bmi filhos fumante regioao  gastos
## 20      30    1 35.3      0        2        3 36837.47
```

```
gastos_20 = -35623.28 + 257.07(30) + 336.60(35.3) + 467.64(0) + 23853.93(2) -
297.97(3)
```

```
gastos_20 = -35623.28 + 257.07*30 + 336.60*35.3 + 467.64*0 + 23853.93*2 -
297.97*3
```

```
gastos_20
```

```
## [1] 30784.75
```

gastos\_20 = 30784.75 gastos real = 36837.47 -> Taxa de acerto de 83,6%

Residual standard error:

Temos um dp = 6061 dos resíduos, e um grau de liberdade = 1332

Multiple R-squared:

Quanto mais próximo de 1 melhor. No nosso caso temos um  $R^2 = 0.7504$ . o que indica que 75,04% da variável dependente consegue ser explicada pelas variáveis explanatórias presentes no modelo.

Adjusted R-squared:

Ele deve ser menor que  $R^2$ . No nosso caso temos 0.7495.

F-statistic: Esse teste obtém os parâmetros do nosso modelo e compara com um modelo que tenha menos parâmetros

p-value: Temos uma probabilidade  $< 2.2e-16$  que a variável não seja relevante.

## Etapa 5 - Previsão

Prevendo despesas médicas Usando a função predict conseguimos realizar a previsão do modelo baseado nas no arquivo de teste

```
Previsao_v2 <- predict(Modelo_v2, Dados_Testes)
```

```
# Abaixo temos o cabeçalho das 10 primeiras previsões
```

```
head(Previsao_v2, 10)
```

```
##          3          4          7          10          11          14          17
## 6747.480 4056.777 10574.033 12041.036 2880.393 14831.323 11837.198
##          18          21          24
## 1558.411 15176.431 31434.198
```

```
# Visualizando os valores previstos e observados
```

```
Resultados <- cbind(Dados_Testes$gastos, Previsao_v2)
```



```

colnames(Resultados) <- c('Real', 'Previsto')
Resultados <- as.data.frame(Resultados)
# Estamos prevendo os gastos e eles não podem ser negativos. Vamos verificar
min(Resultados)

## [1] -1518.167

# Tratando os valores negativos com os valores mínimos
Trata_min <- function(x){
  if (x < 0){
    return(1122)
  }else{
    return(x)
  }
}

# Aplicando a função para tratar valores negativos em nossa previsão
Resultados$Previsto <- sapply(Resultados$Previsto, Trata_min)
Resultados$Previsto

## [1] 6747.4797 4056.7771 10574.0329 12041.0361 2880.3933 14831.322
8
## [7] 11837.1984 1558.4106 15176.4307 31434.1979 7804.3236 14047.214
9
## [13] 28829.2311 1114.5909 31108.5508 1021.8258 33306.9650 8262.814
4
## [19] 10358.8801 14030.7157 10709.5404 5576.2500 33422.3966 40086.076
7
## [25] 32693.0750 10163.5459 13166.6753 1948.8584 17114.2792 27571.965
4
## [31] 11822.7415 13187.5804 4924.8121 9580.3312 5126.0526 16010.507
6
## [37] 36941.4264 10888.0777 5733.0270 9579.1011 39113.8886 14069.350
1
## [43] 8513.1450 3930.1009 5801.2436 5952.7489 3884.9453 12407.801
4
## [49] 13243.7233 12685.5584 8790.6448 273.0799 2835.5249 4012.630
8
## [55] 26939.5534 9720.6990 13980.6162 2152.8032 3699.2011 5109.947
8
## [61] 29084.3084 6320.0220 35280.9942 2248.1956 5509.7949 30161.950
3
## [67] 31445.0114 24598.2511 31537.0543 6528.9408 7407.6118 8736.975
1
## [73] 17203.0131 10132.0884 5311.1883 11514.7936 9824.9227 4557.903
3
## [79] 9618.7083 36249.6527 9182.7471 5410.1507 2116.1511 3117.025
8
## [85] 7135.5873 17315.4897 11435.1482 5028.0630 11092.0226 12253.090
3
## [91] 4630.7661 5233.9901 10510.8567 1017.5741 7888.1907 27788.749

```

2							
##	[97]	7061.2437	16052.3242	11382.1707	12227.8968	7140.3406	8868.702
3							
##	[103]	30689.8109	4024.4839	37007.7378	5292.4496	1122.0000	39159.766
4							
##	[109]	35045.9561	11726.0120	11126.7525	25522.6497	34438.4738	38392.676
2							
##	[115]	9537.8028	9382.9781	11895.6879	9605.7688	686.2609	7887.432
0							
##	[121]	5104.6095	13146.9638	16585.6967	10695.4779	5777.1732	2388.426
3							
##	[127]	25398.8581	31992.1891	7582.2852	6279.8403	15715.7499	6427.314
4							
##	[133]	9143.7183	535.1200	28700.8234	12556.8999	8533.8659	6152.472
0							
##	[139]	3947.2922	9863.1992	39487.7601	34376.9991	14115.7659	15401.818
7							
##	[145]	35681.8977	8655.3880	12348.9540	7044.4403	8507.6632	2174.533
6							
##	[151]	4372.6605	7966.3617	15043.6577	1122.0000	11781.4798	2089.119
1							
##	[157]	14305.6483	8079.8411	10716.0617	29883.9426	3389.0137	30593.773
4							
##	[163]	25423.5736	35961.0219	7319.8507	10271.0119	2223.6537	5405.935
6							
##	[169]	11267.3476	6456.1445	13341.3186	15577.2207	15492.8521	14863.031
0							
##	[175]	5312.5490	5830.6523	24897.2240	10543.3642	11884.4133	37070.084
8							
##	[181]	33279.5386	4361.4526	7060.1572	6509.1522	3362.5967	13027.136
7							
##	[187]	3960.5190	7464.8347	7376.3280	14959.3662	35637.2800	10979.111
2							
##	[193]	9196.3617	4224.4642	11852.2638	12851.8783	15061.9719	32085.569
2							
##	[199]	1299.3622	28631.8780	3479.6828	2393.6875	2251.8023	37461.177
6							
##	[205]	4857.8556	3924.5890	2235.3331	6072.1515	10454.1588	12526.876
3							
##	[211]	710.6656	17626.3992	4184.9276	16469.7270	30224.7687	25976.059
5							
##	[217]	4625.1471	2277.5272	13303.0486	2453.7649	31209.7828	10702.363
6							
##	[223]	9710.9718	10142.7949	30942.4815	9599.1404	13189.5992	14243.164
2							
##	[229]	17123.4749	7295.7803	8790.0298	3819.3860	15452.6704	34002.512
6							
##	[235]	3688.2438	37497.6561	10478.1691	3481.0501	11169.9730	4766.387
2							
##	[241]	4543.9544	15439.5873	11834.9654	38727.1912	5100.9728	16461.146

3  
## [247] 1715.2175 32749.9466 11852.6581 3710.0447 1122.0000 31159.940  
5  
## [253] 12008.0978 7079.8687 11143.6330 8051.5190 10299.1605 11679.746  
3  
## [259] 26196.7608 11627.3472 4856.1541 9754.3595 3437.1610 35762.158  
1  
## [265] 26918.5517 13963.3949 7570.2884 4139.3070 8019.8708 36225.080  
8  
## [271] 6443.1385 6493.5488 1531.3786 14648.5938 12430.1403 7029.267  
9  
## [277] 7244.9286 7196.8113 14943.2914 11760.7589 14054.1411 6984.005  
1  
## [283] 13605.2339 10237.6792 35086.8234 34045.1480 38198.0705 5591.679  
7  
## [289] 37915.8111 6703.9849 15286.7749 1122.0000 34033.6528 11080.844  
7  
## [295] 7998.6484 28769.5115 4447.6192 8147.8907 13066.8105 7115.117  
2  
## [301] 14632.3518 6026.9188 36827.6897 6213.0273 3982.0692 6408.505  
2  
## [307] 8603.8141 12003.7690 14444.8761 4136.7868 10077.6299 26749.163  
1  
## [313] 29506.0700 8632.2134 14711.5558 2933.2574 29474.4284 27662.323  
7  
## [319] 9672.3375 11385.4131 12004.2404 3092.5911 17142.2136 5829.315  
1  
## [325] 9338.6112 28378.1679 29263.4608 13289.4640 10758.8043 5200.252  
6  
## [331] 11231.6748 26398.7234 10022.9208 11285.0468 28518.7094 33801.887  
1  
## [337] 15513.0351 5751.7656 11134.4073 31552.2268 9631.2129 3522.569  
0  
## [343] 7122.2169 15608.7552 1871.3024 5136.6521 1797.7475 12259.103  
7  
## [349] 8807.6154 2783.1258 32823.7522 574.2923 2134.3518 15726.235  
7  
## [355] 33798.6748 5797.5298 4076.9901 10888.5792 7194.1539 13668.954  
5  
## [361] 28781.1138 35576.2702 12426.6171 13111.9361 31367.4920 10205.673  
3  
## [367] 34373.4695 26325.4192 37659.7543 3587.5132 8996.8592 17166.755  
4  
## [373] 12375.1203 4119.9533 7720.1693 9102.7310 8038.2922 3535.795  
8  
## [379] 5217.2232 29314.3125 6288.0631 7925.2307 9594.7815 17828.361  
8  
## [385] 5231.1121 390.5304 4910.6425 5149.8723 6951.7119 27697.244  
3  
## [391] 13489.8322 8371.9049 31895.1018 8015.5890 11689.9449 8723.360

5							
##	[397]	1032.8131	9222.8922	9842.2276	10786.8457	8721.8432	8442.277
5							
##	[403]	1122.0000	15877.1730	16407.3800	32979.5863	9265.1698	39378.669
6							
##	[409]	30352.0900	8771.3277	34791.2666	3509.7065	15550.0450	5923.668
0							
##	[415]	2619.1055	10948.0717	387.5387	26651.7885	12909.4355	6357.616
9							
##	[421]	29280.6821	5343.2243	10521.2290	24558.6073	1152.8609	6800.821
6							
##	[427]	10243.4420	15654.6330	28175.7639	9639.3992	6647.2570	34763.870
2							
##	[433]	7350.4894	29647.5608	6395.5291	6467.6033	15267.6014	28385.799
2							
##	[439]	14836.6911	4381.6355	11855.0047	36879.8016	4737.9514	10363.125
4							
##	[445]	3018.0268	2941.5102	26963.7074	32089.8510	7137.3188	34187.976
0							
##	[451]	10729.3956	34706.4972	9403.7761	8003.6589	3979.6925	8216.859
5							
##	[457]	12868.5617	12033.2913	33359.7990	12179.4217	9218.5333	1784.807
9							
##	[463]	1122.0000	5984.0327	15072.4878	13188.4827	16929.8723	11804.153
0							
##	[469]	13404.1671	4701.8309	11403.2194	11991.0437	8055.2328	11422.392
9							
##	[475]	4586.2255	6876.3888	33352.8363	12652.0050	10476.1034	3779.665
2							
##	[481]	39234.7251	5949.1121	11539.8434	1122.0000	7447.3862	7459.974
3							
##	[487]	7530.3940	2089.0120	12847.7336	8871.9446	11997.6421	3811.677
6							
##	[493]	2461.0084	15128.5707	11533.6695	2589.6968	3076.1219	10885.336
9							
##	[499]	8465.9300	5417.3942	8346.6044	32167.1197	15538.1920	2114.168
9							
##	[505]	5294.7191	28892.5509	3962.5013	8081.8234	30313.9571	2674.178
9							
##	[511]	13918.1621	12591.5698	26768.9882	4925.4572	6558.8874	12085.038
8							
##	[517]	15152.7482	32622.9061	8055.7707	12848.3787	5876.8173	7984.305
2							
##	[523]	3185.9646	6671.7988	6686.9008	11292.5045	13013.6592	23669.958
5							
##	[529]	8899.6217	5084.7909	6189.9899	40734.5635	2160.7218	5052.270
5							
##	[535]	5062.8099	31492.2931	25521.2825	10890.8187	14698.3656	8811.365
7							
##	[541]	7504.6991	8922.6893	11741.8060	9564.2499	6494.9226	5794.000

```

1
## [547] 3522.9333 6155.9416 10586.8653 2231.7264 39558.0726 5776.808
9
## [553] 5070.5847 27822.4096 11026.9712 4091.9119 6607.7569 39528.694
0
## [559] 31501.1544 2532.2937 27782.7658 5403.1946 10984.9509 2255.439
0
## [565] 11865.7714 7746.2284 16037.1152 14591.9859 9505.7668 14632.381
9
## [571] 17153.3080 12623.6358 1419.5171

```

## Etapa 6 - Avaliando a Performance

Calculando o erro médio: Quão distantes seus valores previstos estão dos valores observados. Serve para avaliar as versões do modelo

```

# MSE
mse <- mean((Resultados$Real - Resultados$Previsto)^2)
print(mse)

## [1] 33844649

# RMSE
rmse <- mse^0.5
rmse

## [1] 5817.615

# Calculando R Squared
SSE = sum((Resultados$Previsto - Resultados$Real)^2)
SST = sum((mean(Despesas$gastos) - Resultados$Real)^2)

# R-Squared
# Ajuda a avaliar o nível de precisão do nosso modelo. Quanto maior, melh
or, sendo 1 o valor ideal.
R2 = 1 - (SSE/SST)
R2

## [1] 0.7607692

```

## Etapa 7 - Otimização do Modelo

Nesta etapa será realizado alguns teste para ver se é possível melhorar o modelo

Modelo\_v3: Retirando as observações de outlier.

```

Despesas_v3 <- Despesas
Despesas_v3 <- subset(Despesas_v3, Despesas_v3$gastos < 34490)
summary(Despesas_v3)

##      idade      sexo      bmi      filhos
## Min.   :18.00  Min.   :1.00  Min.   :16.0  Min.   :0.000

```

```
## 1st Qu.:26.00 1st Qu.:1.00 1st Qu.:25.8 1st Qu.:0.000
## Median :39.00 Median :2.00 Median :29.7 Median :1.000
## Mean :38.99 Mean :1.51 Mean :30.1 Mean :1.084
## 3rd Qu.:51.00 3rd Qu.:2.00 3rd Qu.:33.8 3rd Qu.:2.000
## Max. :64.00 Max. :2.00 Max. :53.1 Max. :5.000
## fumante regioao gastos
## Min. :1.000 Min. :1.0 Min. : 1122
## 1st Qu.:1.000 1st Qu.:1.0 1st Qu.: 4409
## Median :1.000 Median :2.0 Median : 8410
## Mean :1.115 Mean :2.5 Mean : 9928
## 3rd Qu.:1.000 3rd Qu.:4.0 3rd Qu.:12954
## Max. :2.000 Max. :4.0 Max. :34473

Modelo_v3 <- lm(gastos ~ ., Despesas_v3)
summary(Modelo_v3)

##
## Call:
## lm(formula = gastos ~ ., data = Despesas_v3)
##
## Residuals:
## Min 1Q Median 3Q Max
## -5327.3 -1862.6 -1290.4 -579.4 24491.8
##
## Coefficients:
## Estimate Std. Error t value Pr(>|t|)
## (Intercept) -18043.572 1090.779 -16.542 < 2e-16 ***
## idade 242.377 9.495 25.527 < 2e-16 ***
## sexo 363.552 264.796 1.373 0.170026
## bmi 67.960 23.609 2.879 0.004067 **
## filhos 406.064 109.047 3.724 0.000205 ***
## fumante 14688.923 429.272 34.218 < 2e-16 ***
## regioao -357.245 120.928 -2.954 0.003196 **
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4579 on 1192 degrees of freedom
## Multiple R-squared: 0.6022, Adjusted R-squared: 0.6002
## F-statistic: 300.7 on 6 and 1192 DF, p-value: < 2.2e-16
```

Comparação:

Modelo\_v1 -> Adjust R-squared: 0.7493 e Multiple R-squared: 0.7505

Modelo\_v2 -> Adjust R-squared: 0.7495 e Multiple R-squared: 0.7504

Modelo\_v3 -> Adjust R-squared: 0.6002 e Multiple R-squared: 0.6022

Fazendo a previsão com o Modelo\_v1

```

Previsao_v1 <- predict(Modelo_v1, Dados_Testes)
# Abaixo temos o cabeçalho das 10 primeiras previsões
head(Previsao_v1, 10)

##           3           4           7          10          11          14          17
## 6685.378 3986.143 10637.244 12097.895 2812.455 14895.774 11898.437
##          18          21          24
## 1489.464 15238.520 31510.101

# Visualizando os valores previstos e observados
Resultados <- cbind(Dados_Testes$gastos, Previsao_v1)
colnames(Resultados) <- c('Real', 'Previsto')
Resultados <- as.data.frame(Resultados)
# Estamos prevendo os gastos e eles não podem ser negativos. Vamos verificar
min(Resultados)

## [1] -1460.366

# Tratando os valores negativos com os valores mínimos
Trata_min <- function(x){
  if (x < 0){
    return(1122)
  }else{
    return(x)
  }
}

# Aplicando a função para tratar valores negativos em nossa previsão
Resultados$Previsto <- sapply(Resultados$Previsto, Trata_min)
Resultados$Previsto

## [1] 6685.3784 3986.1431 10637.2443 12097.8948 2812.4548 14895.773
6
## [7] 11898.4368 1489.4644 15238.5198 31510.1007 7737.7847 14110.079
5
## [13] 28778.4838 1176.5056 31058.4374 950.9595 33256.2304 8330.310
7
## [19] 10295.9785 13965.0238 10765.6429 5643.4346 33365.6537 40033.793
2
## [25] 32762.0966 10231.1441 13094.4208 2011.9701 17179.6452 27517.252
2
## [31] 11884.2665 13119.6935 4988.0180 9643.0857 5058.6964 16080.329
8
## [37] 37013.4425 10946.0607 5801.3456 9516.2981 39186.9166 14004.145
2
## [43] 8575.1205 3861.2268 5862.8207 5888.9694 3817.3663 12469.010
3
## [49] 13178.5085 12614.8347 8849.4036 204.7084 2898.6306 4074.915
2
## [55] 27009.3559 9657.4425 14045.2517 2215.1292 3635.9967 5176.375
5

```

##	[61]	29028.6649	6255.2791	35227.6495	2182.9634	5440.5640	30231.640
1							
##	[67]	31385.4727	24542.6299	31608.4226	6592.8447	7340.0757	8803.600
3							
##	[73]	17138.5387	10063.4769	5376.0512	11445.9514	9886.7536	4625.948
0							
##	[79]	9552.6712	36203.3064	9245.9430	5469.9167	2047.8975	3052.074
0							
##	[85]	7189.5102	17378.8205	11491.0788	5091.0801	11020.2224	12192.072
8							
##	[91]	4565.9791	5295.1261	10568.9323	1077.1437	7952.9645	27737.934
2							
##	[97]	6999.0387	16117.6280	11447.3713	12157.6034	7071.7672	8808.129
3							
##	[103]	30770.3638	4083.4203	37077.2312	5229.5700	1122.0000	39233.373
4							
##	[109]	34990.0601	11785.9436	11058.5706	25597.4748	34381.2143	38345.381
2							
##	[115]	9470.5105	9312.9861	11833.3812	9666.3275	617.9753	7943.984
7							
##	[121]	5037.7292	13208.6824	16657.4555	10627.2154	5841.5708	2451.113
1							
##	[127]	25344.3824	31934.1920	7516.5374	6345.8252	15781.0085	6492.313
1							
##	[133]	9073.0842	596.0663	28776.3332	12489.0314	8593.4915	6218.813
1							
##	[139]	3879.3809	9798.5210	39561.1661	34318.6333	14175.2938	15333.648
7							
##	[145]	35627.1706	8715.6605	12406.7121	7108.7390	8445.5198	2108.138
1							
##	[151]	4433.8894	7902.1441	14985.1076	1122.0000	11844.8625	2151.623
1							
##	[157]	14369.0226	8140.1781	10770.3821	29833.1656	3325.2533	30545.139
4							
##	[163]	25494.0252	35903.9842	7249.4367	10201.6384	2282.7700	5473.487
1							
##	[169]	11330.9601	6382.5728	13270.3820	15518.5897	15552.9633	14929.099
1							
##	[175]	5243.1628	5764.7223	24967.3780	10608.0546	11817.3877	37139.245
8							
##	[181]	33226.2186	4295.7826	7121.2001	6574.9084	3298.8767	13086.361
4							
##	[187]	3895.7610	7399.8435	7440.9882	15022.6818	35579.0046	11036.377
9							
##	[193]	9132.1828	4155.8080	11795.7522	12911.5050	14997.3510	32029.636
0							
##	[199]	1231.6262	28706.3788	3540.3282	2457.5615	2315.3781	37403.152
6							
##	[205]	4795.8364	3860.1229	2299.8202	6135.0471	10512.2231	12459.237
4							



## [211]	771.9617	17563.1628	4245.7497	16533.2075	30172.5982	25922.8957
## [217]	4561.1741	2210.6616	13235.5003	2388.6528	31154.2289	10636.1943
## [223]	9769.7761	10212.2726	30885.4200	9657.8872	13126.9641	14302.8063
## [229]	17058.3699	7228.1868	8850.5085	3885.6716	15514.5121	33945.4887
## [235]	3750.5400	37578.9293	10403.4748	3413.3228	11104.2443	4703.9619
## [241]	4477.9744	15379.2194	11902.8943	38677.9130	5162.8086	16524.3807
## [247]	1647.8759	32829.3909	11781.3621	3769.8638	1122.0000	31234.9954
## [253]	12066.7790	7006.9385	11081.0683	8116.1151	10234.2465	11611.6325
## [259]	26271.6489	11689.4338	4790.1053	9820.1518	3502.6334	35838.9329
## [265]	26995.8289	14026.5971	7626.9449	4199.8315	7954.7934	36302.6126
## [271]	6508.4753	6423.9039	1593.1900	14584.5465	12374.1617	7097.8554
## [277]	7176.3207	7257.4356	15011.8641	11826.4915	14115.8754	6921.2965
## [283]	13549.0587	10294.7798	35153.7213	34121.7650	38140.8986	5654.8567
## [289]	37997.6056	6631.3016	15223.0825	1122.0000	33982.6192	11011.6134
## [295]	8056.9784	28716.7518	4384.3917	8216.0814	13129.6185	7184.9250
## [301]	14569.0267	6087.9860	36903.8672	6273.5952	3920.4293	6348.5143
## [307]	8661.1981	11931.7680	14510.4228	4074.2526	10137.1906	26820.4330
## [313]	29450.7586	8559.4644	14774.4536	2865.7084	29424.3172	27607.6734
## [319]	9601.6574	11446.5491	12068.7060	3026.5845	17075.8539	5763.2309
## [325]	9278.6059	28329.1922	29207.9777	13349.7609	10692.3649	5258.2135
## [331]	11295.8606	26473.9209	9958.2542	11215.3109	28464.0012	33873.7054
## [337]	15446.6425	5689.8322	11201.8431	31490.3089	9695.4523	3582.2627
## [343]	7182.1757	15670.3274	1803.6911	5203.7911	1732.5670	12330.6152
## [349]	8744.4054	2847.4346	32895.1705	501.8849	2198.6842	15787.5217
## [355]	33746.0308	5726.7048	4138.3173	10825.5046	7127.6552	13728.8154

## [361]	28859.5987	35647.4787	12489.6949	13043.9812	31441.5718	10264.0988
## [367]	34447.4137	26268.4550	37744.2896	3644.0595	8925.4914	17105.0445
## [373]	12309.4359	4183.4524	7650.5632	9042.0833	7970.7378	3469.6454
## [379]	5153.2153	29259.8439	6227.2981	7990.0559	9522.3757	17894.4321
## [385]	5300.1879	450.3996	4845.8894	5214.2881	6889.5764	27768.6734
## [391]	13550.0030	8432.3327	31966.5098	8080.4771	11746.1039	8659.3659
## [397]	1094.9113	9278.0106	9785.4945	10850.2692	8660.5366	8505.2800
## [403]	1122.0000	15946.4727	16342.1955	32927.3329	9197.9357	39330.1467
## [409]	30296.9102	8710.4108	34738.7361	3570.1223	15612.2307	5855.9266
## [415]	2553.1910	11016.1958	445.8773	26593.7172	12846.0147	6419.2619
## [421]	29226.6323	5407.2328	10455.9943	24628.8706	1211.3873	6733.3256
## [427]	10309.6695	15716.7841	28119.4800	9570.5417	6574.0919	34709.2247
## [433]	7407.1232	29588.2018	6455.7870	6534.4654	15333.0387	28464.3820
## [439]	14905.9231	4447.4564	11786.4888	36954.0241	4799.3120	10421.9058
## [445]	2951.8251	2876.6791	26910.6921	32161.9469	7195.7417	34131.7024
## [451]	10661.4660	34652.6194	9463.5550	7939.7740	3914.8021	8151.6560
## [457]	12805.4678	11972.2511	33308.9836	12240.3618	9271.4964	1717.2260
## [463]	1122.0000	5917.0542	15133.3620	13119.6277	16870.7698	11863.6200
## [469]	13469.8352	4770.0194	11463.5983	11923.8292	7994.2364	11482.6395
## [475]	4521.0137	6805.8372	33425.8014	12584.8238	10412.6343	3839.7144
## [481]	39312.8730	6014.0487	11470.3361	1122.0000	7510.1481	7513.8854
## [487]	7589.5328	2147.9220	12783.3957	8807.3071	11931.7690	3876.2785
## [493]	2524.9855	15191.1760	11596.6342	2651.2893	3140.0239	10826.3268
## [499]	8527.1723	5477.2521	8284.2952	32240.1895	15603.7256	2176.0078
## [505]	5360.4934	28837.7501	4025.6453	8012.0679	30266.2301	2736.3670

```
## [511] 13979.5360 12655.3760 26844.7529 4987.4136 6490.4408 12018.168
3
## [517] 15216.1270 32571.2376 7989.9309 12911.7886 5941.2154 7923.394
9
## [523] 3245.1427 6603.2826 6616.8486 11230.0486 12946.3285 23741.323
1
## [529] 8960.9718 5019.2924 6121.6008 40681.5045 2222.3607 4987.534
2
## [535] 5131.6280 31440.3049 25466.4856 10955.9275 14764.4570 8867.774
6
## [541] 7565.7525 8984.4694 11801.6053 9497.3876 6431.7788 5726.488
0
## [547] 3586.5023 6218.0290 10648.8842 2167.4055 39504.1151 5837.331
2
## [553] 5128.7749 27771.6462 10954.7245 4154.5454 6541.4199 39473.716
7
## [559] 31444.2877 2594.1836 27857.8868 5463.6203 10925.4709 2190.298
8
## [565] 11807.2883 7807.5805 15971.3636 14523.6738 9568.3263 14569.527
2
## [571] 17223.5067 12558.0607 1480.8006

# MSE
mse <- mean((Resultados$Real - Resultados$Previsto)^2)
print(mse)

## [1] 33787068

# RMSE
rmse <- mse^0.5
rmse

## [1] 5812.664

# Calculando R Squared
SSE = sum((Resultados$Previsto - Resultados$Real)^2)
SST = sum((mean(Despesas$gastos) - Resultados$Real)^2)
# R-Squared
# Ajuda a avaliar o nível de precisão do nosso modelo. Quanto maior, melh
or, sendo 1 o valor ideal.
R2 = 1 - (SSE/SST)
R2

## [1] 0.7611762
```

Previsão\_v1 -> Multiple R-squared: 0.7611

Previsão\_v2 -> Multiple R-squared: 0.7607

## Conclusão

Neste projeto foi possível utilizar algumas técnicas de manipulação de dados, técnicas estatísticas, machine learning com a linguagem R. Foi um desafio prazeroso utilizar o R em todas as etapas do projeto. Tive dificuldades em algumas técnicas de manipulação de dados, mas com a ajuda do material fornecido pela DSA e das documentações do R, consegui vencer as mesmas.

Nessa perspectiva, afim de otimizar o modelo, foi realizado alguns testes manipulando o dataset, mas os resultados foram inferiores ou bem próximos do conquistado pelo Modelo\_v2. Para melhorarmos o o resultado teríamos que possuir mais observações ou mais variáveis para treinar o modelo.

Baseado na comparação realizada no item anterior o melhor os Modelos v1 e v2 são muito parecidos. Porém o Modelo\_v2 é um pouco melhor pois possui menos variaveis e tem o R ajustado um pouco maior. Neste modelo de machine learning podemos dizer que 76,07% da variável dependente consegue ser explicada pelo modelo.

***Evanil Tiengo Junior***

***Belo Horizonte, 22 de outubro de 2018***