SME0820 - Modelos de Regressão e Aprendizado Supervisionado I - Trabalho I

Brenda da Silva Muniz 11811603 Francisco Rosa Dias de Miranda 4402962
Heitor Carvalho Pinheiro 11833351- Mônica Amaral Novelli 11810453

Setembro 2021

Neste trabalho, nosso objetivo é ajustar um modelo de regressão linear simples ao conjunto de dados fornecido, utilizando linguagem R. Para esta tarefa, descreveremos cada etapa de nosso *pipeline*.

Primeiramente, vamos carregar os módulos utilizados nesta análise. Caso não possua algum dos pacotes, utilize o comando `install_packages("Nome_do_pacote")`.

```
library(tidyverse)
library(ggpubr)
library(corrplot)
library(DataExplorer)
library(GGally)
library(knitr)
library(data.table)
```

Com os pacotes carregados em nosso ambiente, lemos o arquivo `.csv` disponibilizado colocando-o na mesma pasta de nosso projeto. Vamos inspecionar o que foi carregado com auxílio do comando `head()`, que exibe as 5 primeiras observações.

```
dados <- fread("data-table-B3.csv")

head(dados) %>% kable()
```

y	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11
18,9	350	165	260	8	2,56	4	3	200,3	69,9	3910	1
17	350	170	275	8,5	2,56	4	3	199,6	72,9	3860	1
20	250	105	185	8,25	2,73	1	3	196,7	72,2	3510	1
18,25	351	143	255	8	3	2	3	199,9	74	3890	1
20,07	225	95	170	8,4	2,76	1	3	194,1	71,8	3365	0
11,2	440	215	330	8,2	2,88	4	3	184,5	69	4215	1

Parte 0: Limpeza dos dados

Por padrão, o R utiliza o ponto (.) como separador decimal. No caso do arquivo `.csv` fornecido, algumas colunas utilizavam a vírgula, que quando lidas eram identificadas como *strings*. Utilizamos a função `parse_number()` para corrigir isso, e o comando `str()` para mostrar o tipo de cada uma das colunas de nosso dataset.

```r
dados$y <- dados$y %>% parse_number(locale = locale(decimal_mark = ","))
dados$x1 <-dados$x1 %>% parse_number(locale = locale(decimal_mark = ","))
dados$x4 <- dados$x4 %>% parse_number(locale = locale(decimal_mark = ","))
dados$x5 <- dados$x5 %>% parse_number(locale = locale(decimal_mark = ","))
dados$x8 <- dados$x8 %>% parse_number(locale = locale(decimal_mark = ","))
dados$x9 <- dados$x9 %>% parse_number(locale = locale(decimal_mark = ","))

str(dados)
```

```
## Classes 'data.table' and 'data.frame':   32 obs. of  12 variables:
##  $ y  : num  18.9 17 20 18.2 20.1 ...
##  $ x1 : num  350 350 250 351 225 440 231 262 89.7 96.9 ...
##  $ x2 : int  165 170 105 143 95 215 110 110 70 75 ...
##  $ x3 : int  260 275 185 255 170 330 175 200 81 83 ...
##  $ x4 : num  8 8.5 8.25 8 8.4 8.2 8 8.5 8.2 9 ...
##  $ x5 : num  2.56 2.56 2.73 3 2.76 2.88 2.56 2.56 3.9 4.3 ...
##  $ x6 : int  4 4 1 2 1 4 2 2 2 2 ...
##  $ x7 : int  3 3 3 3 3 3 3 3 4 5 ...
##  $ x8 : num  200 200 197 200 194 ...
##  $ x9 : num  69.9 72.9 72.2 74 71.8 69 65.4 65.4 64 65 ...
##  $ x10: int  3910 3860 3510 3890 3365 4215 3020 3180 1905 2320 ...
##  $ x11: int  1 1 1 1 0 1 1 1 0 0 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

Parte a):

- Descrição do banco de dados

Poderíamos também descrever as colunas do banco de dados com auxílio da função `introduce()` do pacote `DataExplorer`. A tabela obtida é exibida abaixo:

```r
a <- introduce(dados)

a %>% select(rows,
             discrete_columns,
             continuous_columns,
             total_observations,
             complete_rows,
             total_missing_values) %>%
  kable(col.names = c("Linhas", "Colunas Discretas", "Colunas Contínuas", "Total de observações","Atribu
```

Linhas	Colunas Discretas	Colunas Contínuas	Total de observações	Atributos sem NA	Atributos com NA
32	0	12	384	30	2

- Definição das variáveis
- Análise exploratória inicial

```
ggcorr(dados, geom = "circle")
```

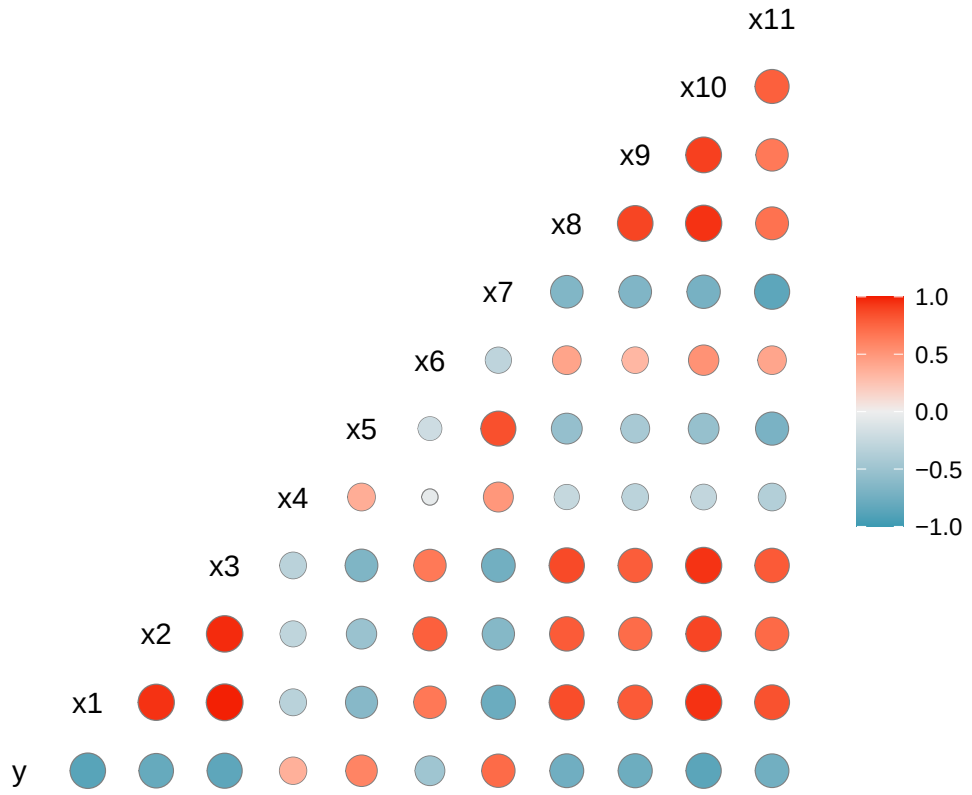
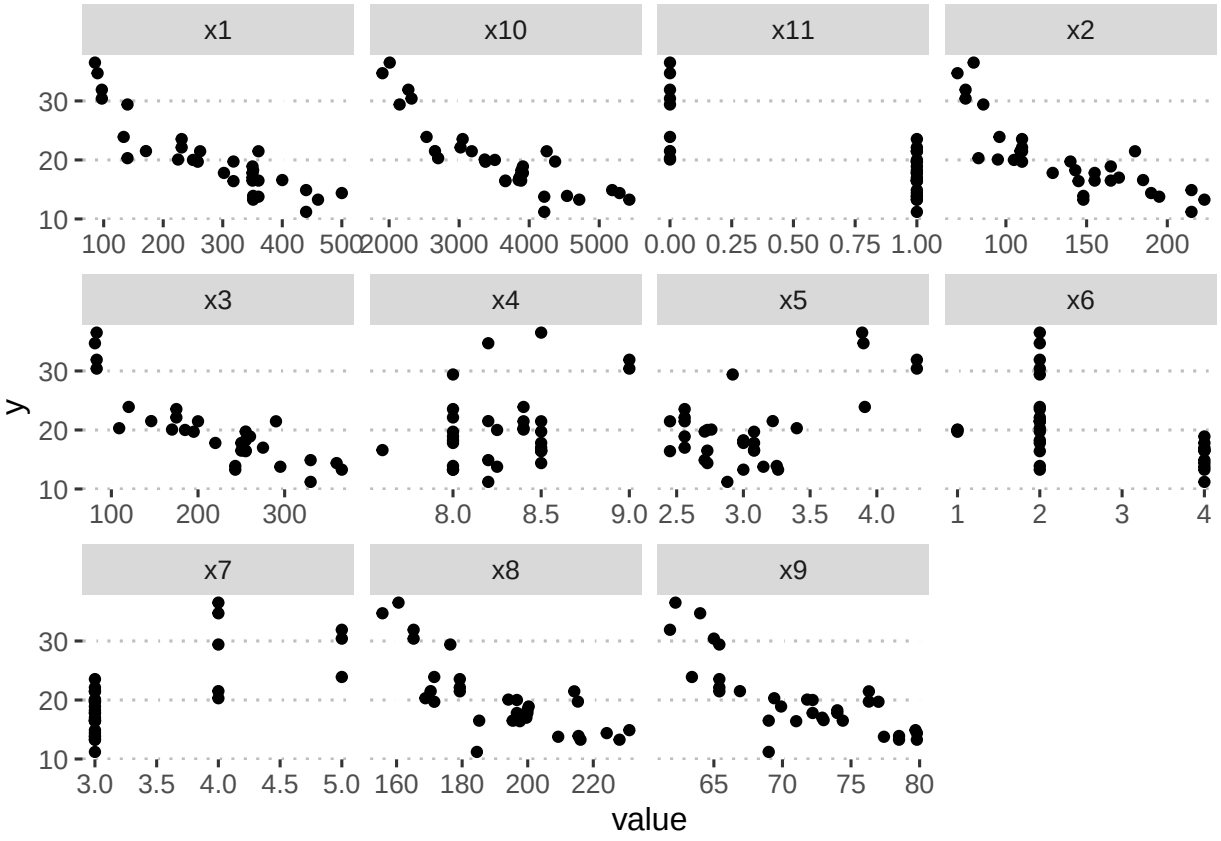


Figure 1: Correlograma entre as variáveis

- Graficos de dispersão Y versus X_i, $i = 1, ..., 11$.

```
dados %>%
  pivot_longer(cols = !"y") %>% #todas as variaveis como funcao de y
  ggplot(aes(y = y)) +
    geom_point(aes(x = value)) +
    facet_wrap(~name, scales = "free_x") + theme_pubclean()
```



-

Interpretação de cada gráfico

Parte b):