Introdução à Datascience com R

Cleuton Sampaio

Como fazer uma regressão linear em R

Como já devem saber, estou trabalhando em um curso de Datascience com a linguagem R em vídeo, que será totalmente gratuito. Será um curso de introdução ao assunto, tanto para datascience como para R.

Para você ter uma ideia comparativa, eu queria apresentar aqui um pequeno exemplo do curso (e também do meu livro: "Datascience para programadores", editora Ciência Moderna).

O código todo está no Github (https://github.com/cleuton/datascience/tree/master/R-course/lession1).

Regressão linear simples

Vamos supor que temos um grupo de crianças, nascidas no mesmo bairro e na mesma cidade, com os seguintes pesos e alturas:

Pesos	Alturas
58	1,58
78	1,8
70	1,7
80	1,8
77	1,76
74	1,73
61	1,63
65	1,65
55	1,56
76	1,79
54	1,56
53	1,51
69	1,69
67	1,67
72	1,74
58	1,6

53	1,52
55	1,57
57	1,57
66	1,67
65	1,64
50	1,5
63	1,64
58	1,56
55	1,56
63	1,62
73	1,71
80	1,83
76	1,76
40	1,75
130	1,6

Podemos estabelecer uma relação entre peso e altura? E podemos usar essa relação para prever os pesos das outras crianças da mesma região, conhecidas as suas alturas?

Explorando os dados

A amostra é pequena, logo, bastaria um exame visual para detectar anomalias. Porém, vamos agir como faríamos com amostras grandes. A primeira coisa a fazer é tentar "ver" esses dados.

Se tentarmos plotar um gráfico dessa amostra, veremos que há uma relação entre a altura e o peso das crianças:

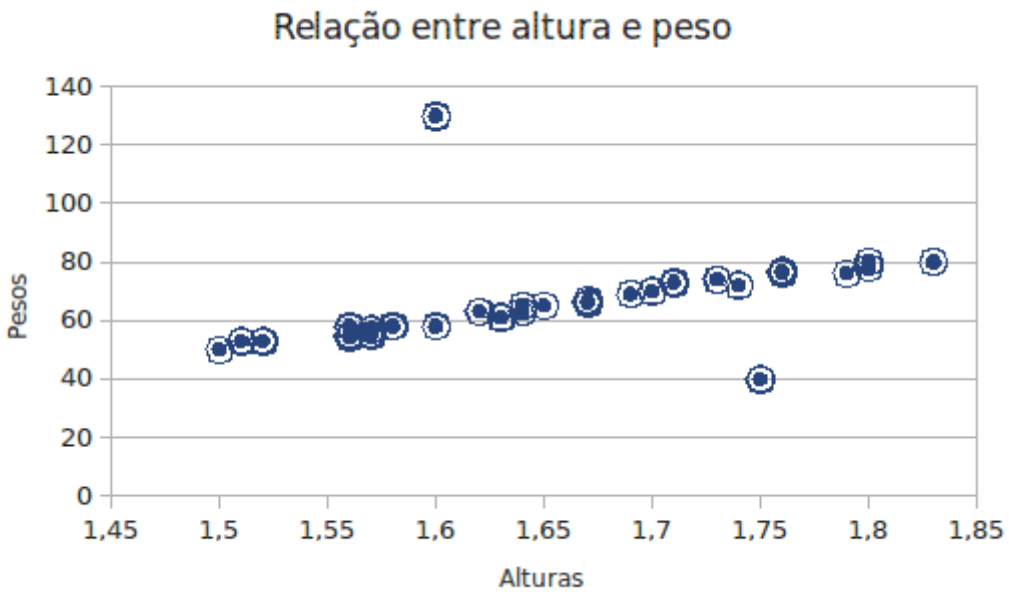


Figura 1: Gráfico

A princípio, os pesos e alturas parecem estar relacionados em uma reta, logo, é um problema que caberia perfeitamente em uma solução de **Regressão Linear**. Podemos notar que há alguns pontos muito fora da tendência normal da reta: Um acima dela e outro abaixo. Vamos ver se descobrimos o motivo.

Usando o LibreOffice, podemos plotar a estatística descritiva dos dados. O arquivo que usamos é o "mod-preditivo.ods" e está na mesma pasta do Github.

	Pesos	**Alturas**
Média	66,16	1,65
Erro padrão	2,79	0,02
Modo	55	1,56
Mediana	65	1,64
Primeiro quartil	56	1,57
Terceiro quartil	73,5	1,74
Variância	240,74	0,01
Desvio padrão	15,52	0,09
Curtose	8,92	-1,09
Inclinação	2,25	0,19
Intervalo	90	0,33

Mínimo	40	1,5
Máximo	130	1,83
Soma	2051	51,27
Contagem	31	31

Temos um desvio padrão bastante alto nos pesos, cerca de 15,6 kg e também temos uma curtose muito alta, o que pode indicar presença de *outliers*, ou pontos com grandes desvios.

Vamos estabelecer um limite de 3 desvios padrões, que seria entre 50,64 kg e 81,68 kg, e vamos listar os valores que estão fora deste intervalo:

- Peso: 50 kg, altura: 1,50 m;
- Peso: 40 kg, altura: 1,75 m;
- Peso: 130 kg, altura: 1,6 m.

É possível uma pessoa de 1,50 m pesar 50 kg, mas os outros dois pesos parecem ser valores "espúrios", ou "outliers", pois uma pessoa pesar 40 kg com 1,75 m e outra com 130 kg e 1,60 m, parecem ser anomalias. O que podemos fazer?

- Substituir os pesos dessas duas pessoas pela média dos pesos;
- Retirar essas duas pessoas da amostra.

Vamos optar por retirar essas duas pessoas da amostra.

Modelando os dados

Agora vemos que as alturas e pesos seguem uma linha, sem valores muito fora dela. Podemos, então, calcular a fórmula usando a Regressão Linear.

A fórmula de uma reta é: y = ax + b, onde:

- "y": Peso estimado;
- "a": Inclinação da reta (slope);
- "x": Altura informada;
- "b": Coeficiente linear (intercept).

Precisamos chegar aos valores de "a" e "b", o que pode ser feito por uma heurística de aproximação (Aprendizado de máquina), ou por uma solução de forma fechada, calculando estes parâmetros com duas fórmulas:

$$a = \frac{\sum_{i=0}^{n} (x_i - \overline{x}) \times \sum_{i=0}^{n} (y_i - \overline{y})}{\sum_{i=0}^{n} (x_i - \overline{x})^2}$$

$$b = \overline{y} - a \times \overline{x}$$

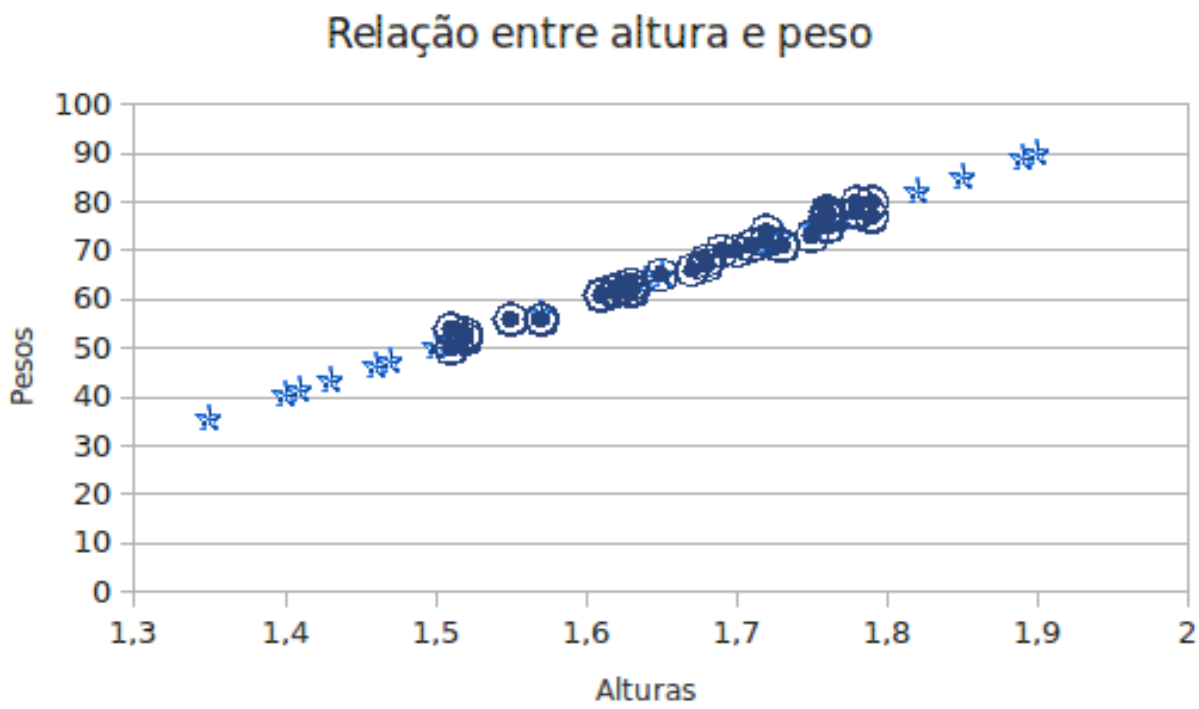
Aplicando nossas fórmulas, obtemos a equação da reta preditiva:

$$Pesoestimado \approx ❑ 95,8 \times altura - 93,5$$

(Você pode conferir isto na planilha de exemplo: mod_preditivo.ods)

Testando o modelo

Podemos jogar mais alguns valores de teste e plotar o novo gráfico, usando a fórmula que calculamos:



As estrelas são os novos valores, previstos usando o modelo, e os círculos são os valores que já tínhamos na amostra.

Avaliando o modelo

Podemos ver que eles estão na mesma reta imaginária dos outros pesos. Para saber exatamente o quanto nosso modelo é capaz de prever os pesos, podemos calcular a métrica R2 (r quadrado), que é o coeficiente de determinação de nosso modelo:

Soma dos quadrados explicada (SQE): O quanto nosso modelo está explicando a relação

$$SQE = \sum_{i=1}^{n} (\hat{y}_i - \overline{y})^2$$

Soma dos Quadrados dos resíduos (SQR): O que nosso modelo não explica da relação

$$SQR = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Finalmente, o R quadrado é calculado assim:

$$R^2 = 1 - \frac{SQR}{SQT}$$

No nosso caso, o valor do R quadrado é 0,98, o que significa que 98% da variável dependente (Pesos) consegue ser explicada pelo modelo.

Agora, usando R

R é uma linguagem de programação fantástica para trabalhos de análise de dados, mas é muito ruim para qualquer outra coisa, como programação de aplicações comerciais, por exemplo. Eu acho que esse negócio de querer usar uma ferramenta só para tudo é muito ruim. É o que fazem com Java, por exemplo.

O arquivo com o código-fonte em R está na mesma pasta do Github e se chama: "lerOds.R". Ele lê a mesma planilha ("mod-preditivo.ods") e usa os dados para calcular um modelo de regressão linear, plotando um gráfico também.

Em vez de utilizar uma solução de forma fechada, como fizemos na planilha, vamos criar um modelo que "aprende" com os dados.

Para usar R você tem que instalar duas coisas:

- A linguagem R;
- O editor Rstudio.

A forma de instalar é simples e vamos ver por sistema operacional:

- Windows:

    - Baixe o R de http://cran.us.r-project.org/, clicando em **Download R for Windows.** Instale o R com as opções padrão;
    - Baixe o Rstudio de http://rstudio.org/download/desktop e instale;
- MacOS:
    - Baixe o R de http://cran.us.r-project.org/, clicando em **Download R for (Mac) OS X.** Instale o R com as opções padrão;
    - Baixe o Rstudio de https://download1.rstudio.org/RStudio-1.1.383.dmg e instale;
- Ubuntu:
    - Abra um terminal e execute: `sudo apt-get install r-base`
    - Baixe o pacote do Rstudio e instale: https://download1.rstudio.org/rstudio-xenial-1.1.383-amd64.deb

Se você usa Ubuntu ou MacOS, pode ser necessário instalar a libxml2-dev:

- Ubuntu: sudo apt-get install libxml2-dev;
- MacOS: brew install libxml2.

Abra o Rstudio e crie um novo arquivo (r-script), colando o texto abaixo, ou então baixe o arquivo "lerOds.R" do Github e o abra no editor:

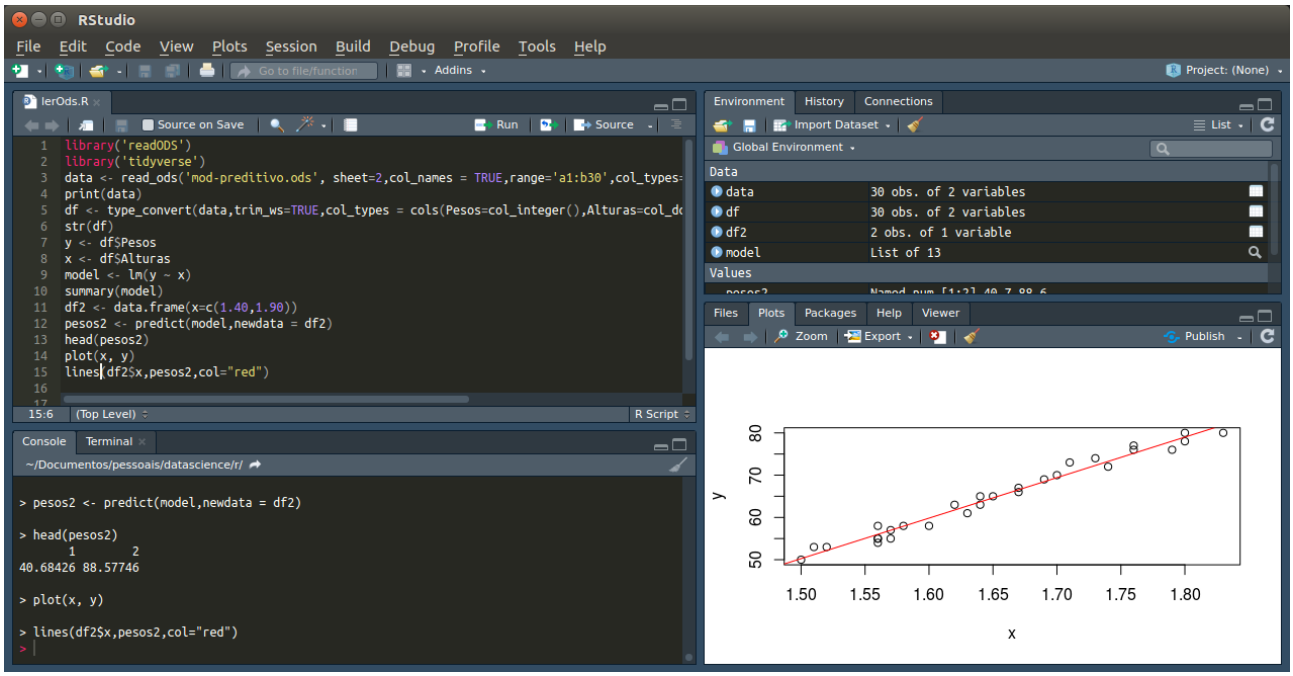
```
library('readODS')
library('tidyverse')
data <- read_ods('mod-preditivo.ods', sheet=2,col_names =
TRUE,range='a1:b30',col_types=NA)
print(data)
df <- type_convert(data,trim_ws=TRUE,col_types =
cols(Pesos=col_integer(),Alturas=col_double()),locale = locale(decimal_mark =
","))
str(df)
y <- df$Pesos
x <- df$Alturas
model <- lm(y ~ x)
summary(model)
df2 <- data.frame(x=c(1.40,1.90))
pesos2 <- predict(model,newdata = df2)
head(pesos2)
plot(x, y)
lines(df2$x,pesos2,col="red")
```

Calma! Respire fundo!

Primeiro, temos que instalar os pacotes necessários no Rstudio. Na parte de baixo da janela dele, à esquerda, fica a console. Digite os seguintes comandos:

- install.packages('xml2')
- install.packages('readODS')
- install.packages('tinyverse')

Agora sim! Seu código está pronto para ser executado. Selecione o menu "Code / Source" ou clique CTRL+SHIFT+S (ou COMMAND+SHIFT+S, no MacOS). Isto executará o código. Se tudo deu certo, você verá uma figura como a segunte:



Seu código no painel topo-esquerda, a console, no painel baixo-esquerda, as variáveis e objetos, no painel topo-direita, e o plot dos gráficos no painel baixo-direita.

Como diria Jack, o Estripador, vamos por partes…

1 - Ler a planilha

```
data <- read_ods('mod-preditivo.ods', sheet=2,col_names =
TRUE,range='a1:b30',col_types=NA)
```

Este comando usa a função "read_ods" para ler a segunda "spreadsheet" do arquivo "mod-preditivo.ods", selecionando o intervalo (range) de células a1:b30. Ele lerá os valores como caracteres.

Essa função retorna um objeto Data Frame, contendo as linhas e colunas lidas.

2 - Converter os valores

Como lemos os dados de uma planilha em Português, os pesos virão com vírgula como separador de decimais. E isso dá problema! Por isso importamos como caracteres. Agora, é hora de converter tudo em números:

```
df <- type_convert(data,trim_ws=TRUE,col_types =
cols(Pesos=col_integer(),Alturas=col_double()),
locale = locale(decimal_mark = ","))
```

Usamos a função "type_convert" para converter o dataframe original, transformando os pesos em inteiros e as alturas em números reais, mas entendendo a vírgula como separador de decimais (no R, o separador default é o ponto). Agora, vejamos se ficou tudo ok:

```
str(df)
```

Este comando lista a estrutura de um dataframe. Veja a saída na Console:

```
> str(df)
'data.frame':   30 obs. of  2 variables:
 $ Pesos  : int  58 78 70 80 77 74 61 65 55 76 ...
 $ Alturas: num  1.58 1.8 1.7 1.8 1.76 1.73 1.63 1.65 1.56 1.79 ...
```

Ele entendeu perfeitamente os números.

3 – Criar um modelo linear

Criamos um modelo linear de maneira bem simples:

```
y <- df$Pesos
x <- df$Alturas
model <- lm(y ~ x)
```

Pegamos a coluna "Pesos", do dataframe, e atribuímos a uma variável "y" (para facilitar), e a coluna "Alturas", à variável "x", que é a variável independente. Depois, usamos a sintaxe de fórmulas do R para criar a fórmula de regressão: "y ~ x" ou "y" depende de "x". A função "lm" cria um modelo linear com base em uma fórmula.

E podemos ver como esse modelo se saiu com a função "summary":

```
> summary(model)

Call:
lm(formula = y ~ x)

Residuals:
     Min       1Q   Median       3Q      Max
-2.04096 -1.01008  0.07419  1.00118  2.62196

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -93.417      4.627  -20.19   <2e-16 ***
x             95.786      2.796   34.26   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.418 on 27 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared:  0.9775,    Adjusted R-squared:  0.9767
F-statistic:  1174 on 1 and 27 DF,  p-value: < 2.2e-16
```

Podemos ver os coeficientes gerados e o valor do R quadrado ajustado (98%).

4 – Executar predições

Vamos criar duas novas alturas e prever seus pesos, usando o modelo criado:

```
df2 <- data.frame(x=c(1.40,1.90))
pesos2 <- predict(model,newdata = df2)
```

Agora, vamos plotar tudo junto: Dados originais e a reta de regressão, usando as duas novas alturas e os dois novos pesos estimados:

```
plot(x, y)
lines(df2$x,pesos2,col="red")
```

Ecco le previsioni!