Linguagem R

IN

Aula 10

- Objetivo da aula:
- Regressão linear;
- Regressão linear múltipla;
- Séries temporais.

IN

Regressão linear:

Regressão Linear é uma técnica estatística que visa determinar a equação da 'melhor' reta que ajusta a um conjunto de dados (x,y).

No modelo de regressão linear a variável dependente (x) é chamada de variável preditora (predict) cujo valor é obtido por meio de experimentos. A variável independente (y) é chamada de variável alvo (target).

A equação matemática geral para uma regressão linear é dada por: $y = ax + b$, onde a é o coeficiente angular e b o linear (chamado intercept).

IN

Regressão em R:

Como estabelecer um modelo:
Colete os dados de x e y.
Crie um modelo de relacionamento usando as funções lm() em R.
Encontre os coeficientes do modelo criado e crie a equação matemática usando estes coeficientes.
Obtenha um resumo do modelo de relacionamento para saber o erro médio na previsão.
Para prever o peso de novas pessoas, use a função predict() em R.

IN

Exemplo:

Digite no RStudio os dados:
Dados de peso: 151, 174, 138, 186, 128, 136, 179, 163, 152, 131
Dados de altura: 63, 81, 56, 91, 47, 57, 76, 72, 62, 48

Crie a função lm(y~x) no RStudio. Para isto usa-se:

```
rela <- lm(y~x)
print(rela)
print(summary(rela))
```

IN

Continuação do Exemplo:

Para se fazer uma previsão, ou seja inserir um valor de x no modelo e obter um valor de y usa-se a função predict().
Para isto 1º define-se o novo valor de x da seguinte maneira:

```
x_new<- data.frame(x=valor)
res <- predict(rela,x_new)
print(res)
```

IN

```
149
150 x<- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
151 y<- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
152 relacao <- lm(y~x)
153 print(relacao)
154 print(summary(relacao))
155 a <- data.frame(x = 170)
156 resultado <-  predict(relacao,a)
157 print(resultado)
158
159
```

149:1 (Top Level)

Console | Terminal | Background Jobs

R 4.2.2 · ~/

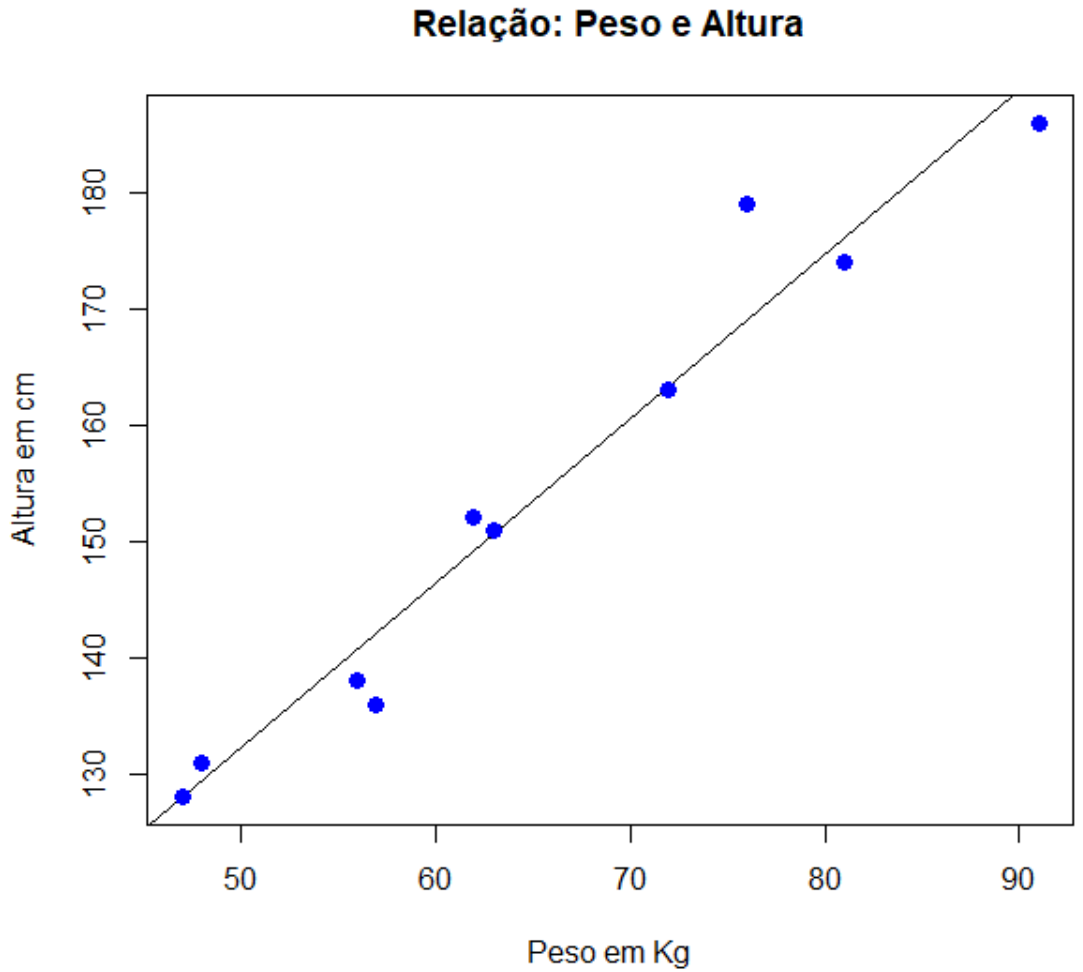
```
Residuals:
    Min      1Q  Median      3Q     Max
-6.3002 -1.6629  0.0412  1.8944  3.9775

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -38.45509    8.04901  -4.778  0.00139 **
x             0.67461    0.05191  12.997 1.16e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.253 on 8 degrees of freedom
Multiple R-squared:  0.9548,    Adjusted R-squared:  0.9491
F-statistic: 168.9 on 1 and 8 DF,  p-value: 1.164e-06

> a <- data.frame(x = 170)
> resultado <-  predict(relacao,a)
> print(resultado)
       1
76.22869
>
```

IN

Gráfico:

Para construir o gráfico use:

```
plot(y,x,col = "blue",main = "Relação: Peso e Altura",
     abline(lm(x~y)),cex = 1.3,pch = 16,xlab = "Peso em
Kg",ylab = "Altura em cm")
```


Relação: Peso e Altura
Altura em cm
130
140
150
160
170
180
50
60
70
80
90
Peso em Kg

IN

Regressão linear múltipla:

Regressão linear múltipla é uma regressão linear entre mais de duas variáveis preditoras e uma alvo.
A equação matemática geral para regressão múltipla é:

$$y = a + b1.x1 + b2.x2 + ...bn.xn$$

A sintaxe básica para a função **lm()** na regressão múltipla é:

```
lm(y ~ x1+x2+x3+...,dados)
```

IN

Exemplo:

Considere um modelo linear múltiplo com as variáveis y(km/l), x1(potência), x2(peso) de um veículo.
O objetivo é estabelecer uma relação entre o consumo (km/l) do veículo a partir da relação entre as outras variáveis; potência e peso.

IN

Exemplo:

Como exemplo utilize os seguintes dados:
y : Relação km/l: (21,21,23,22,19,18)
x1: Potência (HP):(110,110,93,110,175,105)
x2: Peso (kg): (2620,2875,2320,3215,3440,3460)

A partir dai cria-se um dataframe com as variáveis:

IN

Criação do DataFrame:

```
y<- c(21,21,23,22,19,18)
x1<- c(110,110,93,110,175,105)
x2<- c(2620,2875,2320,3215,3440,3460)
dataframe <- data.frame(y,x1,x2)
print(dataframe)
```

IN

O modelo:

A seguir cria-se o modelo:

```
relacao <- lm(y~(x1+x2),data=dataframe)
print(relacao)
print(summary(relacao))
```

IN

```
163 y<- c(21,21,23,22,19,18)
164 x1<- c(110,110,93,110,175,105)
165 x2<- c(2620,2875,2320,3215,3440,3460)
166 dataframe <- data.frame(y,x1,x2)
167 print(dataframe)
168 relacao <- lm(y~(x1+x2),data=dataframe)
169 print(relacao)
170 print(summary(relacao))
171 a <- data.frame(x = 170)
172 resultado <-  predict(relacao,a)
173 print(resultado)
174
```

168:1 (Top Level)

Console | Terminal | Background Jobs

R 4.2.2 · ~/

```
> y<- c(21,21,23,22,19,18)
> x1<- c(110,110,93,110,175,105)
> x2<- c(2620,2875,2320,3215,3440,3460)
> dataframe <- data.frame(y,x1,x2)
> print(dataframe)
   y  x1   x2
1 21 110 2620
2 21 110 2875
3 23  93 2320
4 22 110 3215
5 19 175 3440
6 18 105 3460
> relacao <- lm(y~(x1+x2),data=dataframe)
> print(relacao)

Call:
lm(formula = y ~ (x1 + x2), data = dataframe)

Coefficients:
(Intercept)           x1           x2
  30.258095    -0.004232    -0.003044
```

A equação da reta:

A partir dos resultados, a equação da reta é dada por:
y=30.258095-0.004232.x1-0.003044.x2

```
Coefficients:
(Intercept)           x1           x2
  30.258095    -0.004232    -0.003044
```

IN

Realizando a previsão:

1ª maneira:
Escreve-se a função com os coeficientes, os novos valores e a partir dai faz-se o cálculo:

```
new_x1<-105
new_x2<-3460
y<-30.258095-0.004232*new_x1-0.003044*new_x2
print(y)
```

IN

Realizando a previsão:

2ª maneira:
Faz-se um novo modelo.

```
new_dataframe <- data.frame(new_x1,new_x2)
new_relacao <- lm(y~(x1+x2),data=new_dataframe)
resultado <-  predict(new_relacao,data=new_dataframe)
print(resultado)
```

IN

```
170 new_x1<-105
171 new_x2<-3460
172 y<-30.258095-0.004232*new_x1-0.003044*new_x2
173 print(y)
174 new_dataframe <- data.frame(new_x1,new_x2)
175 new_relacao <- lm(y~(x1+x2),data=new_dataframe)
176 resultado <-  predict(new_relacao,data=new_dataframe)
177 print(resultado)
178
179
180
```

174:1 (Top Level)

Console Terminal Background Jobs

R 4.2.2 · ~/

```
> new_x1<-105
> new_x2<-3460
> y<-30.258095-0.004232*new_x1-0.003044*new_x2
> print(y)
[1] 19.28149
> new_dataframe <- data.frame(new_x1,new_x2)
> new_relacao <- lm(y~(x1+x2),data=new_dataframe)
> resultado <-  predict(new_relacao,data=new_dataframe)
> print(resultado)
       1
19.28149
>
```

IN

Séries Temporais:

Uma série temporal é um conjunto de dados que varia no tempo.
Exemplos simples de séries temporais pode-se dizer que é o valor de uma ação no mercado de ações em diferentes pontos do tempo em um determinado dia. A previsão diária de clima numa cidade também é uma série temporal.

IN

Séries Temporais:

O objeto de série temporal em R é criado usando a função ts().

Em R uma série temporal é definida como:
timeseries.object.name <- ts(data, start, end, frequency)

Onde, data é um vetor que contém os dados usados na série temporal, start especifica a hora de início para a primeira observação na série temporal, end especifica a hora de término da última observação na série temporal e a frequência especifica o número de observações por unidade de tempo.

IN

Séries Temporais:

Exemplo de série temporal.
Admita que os valores a seguir são dados coletados de temperatura (Kelvin) medida todo dia 05 de cada mês ao longo do ano de 2012.

```
temp <- c(799,1174,865,1334,635,918,685,998,784,985,882,1071)
```

Criando o modelo de série temporal. Para isto use:

```
temp.timeseries <- ts(temp,start = c(2012,1),frequency = 12)
```

A partir daí faz-se um print da serie temporal para obter a tabela de dados.

IN

Séries Temporais:

Imprimindo a tabela e gerando o gráfico.

```
print(temp.timeseries)
plot(temp.timeseries)
```

IN

```
179
180
181 temp <- c(799,1174,865,1334,635,918,685,998,784,985,882,1071)
182 temp.timeseries <- ts(temp,start = c(2012,1),frequency = 12)
183 print(temp.timeseries)
184 plot(temp.timeseries)
185
186
```

181:1 (Top Level) R Script

Console Terminal Background Jobs

R 4.2.2 · ~/

```
> temp <- c(799,1174,865,1334,635,918,685,998,784,985,882,1071)
> temp.timeseries <- ts(temp,start = c(2012,1),frequency = 12)
> print(temp.timeseries)
     Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
2012 799 1174  865 1334  635  918  685  998  784  985  882 1071
> plot(temp.timeseries)
>
```