Linguagem R

IN

Aula 7

- Objetivo da aula:
- Gráficos básicos;
- Pacote Lattice;
- Pacote Tables.

IN

Gráficos: BarPlot

BarPlot:
A sintaxe básica para criar um gráfico de barras em R é:

barplot(v,xlab,ylab,main, names.arg,col)

Onde, v é um vetor que contém dados, xlab é o rótulo para o eixo x, ylab é o rótulo para eixo y, main é o título do gráfico de barras, names.arg é um vetor de nomes que aparecem sob cada barra e col é usado para dar cores às barras no gráfico.

IN

Exemplo de gráfico BarPlot:

```
v1 <- c(7,12,28,3,41)
v2 <- c("Mar","Abr","Mai","Jun","Jul")
barplot(v1,names.arg=v2,xlab="Mês",ylab="Vendas",col="blue"
        ,main="Vendas por Mês")
```

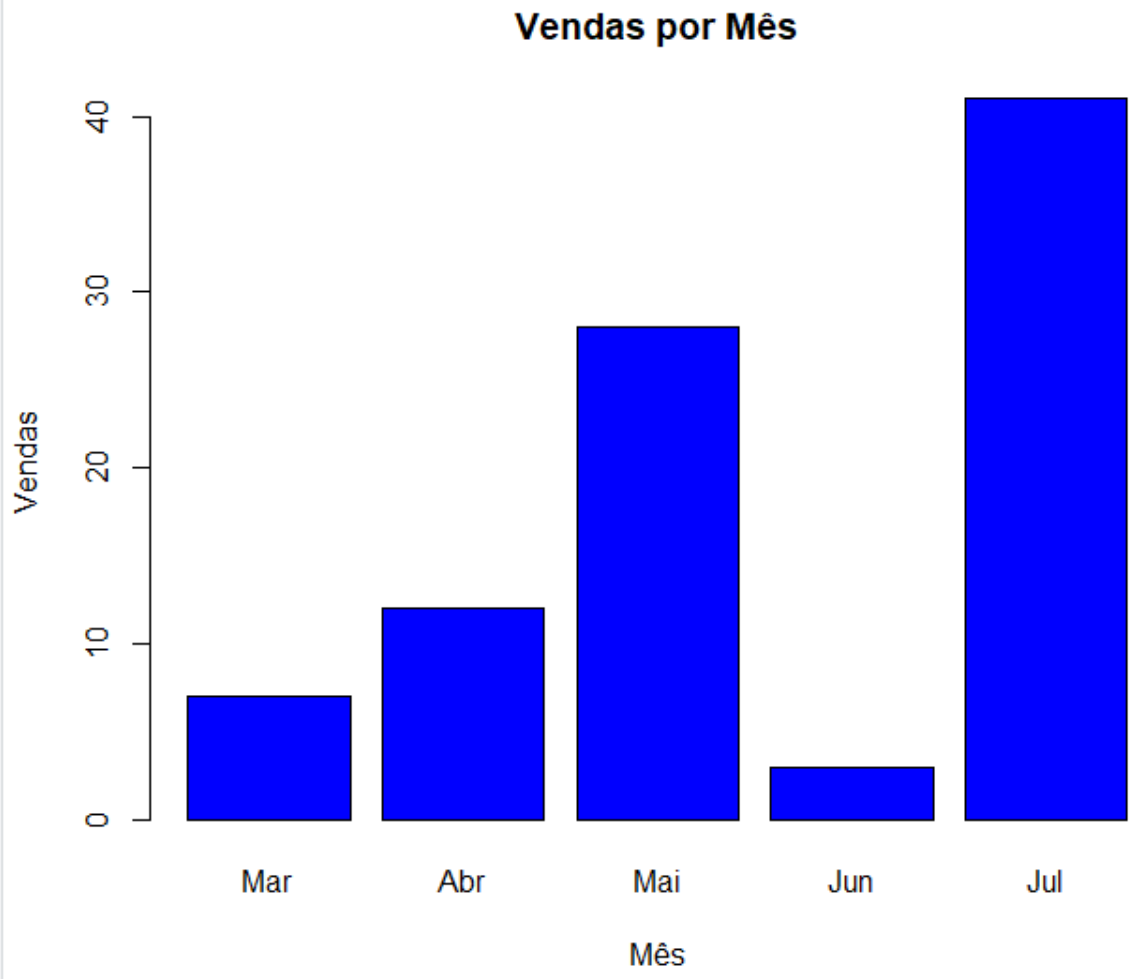
```
}
valor_moda <- moda(conj)
print(valor_moda)
sd(conj)
var(conj)
v1 <- c(7,12,28,3,41)
v2 <- c("Mar","Abr","Mai","Jun","Jul")
barplot(v1,xlab="Mês",ylab="Vendas",col="blue"
        ,main="Vendas por Mês",names.arg=v2)
```

30:1 (Top Level) R Script

Console Terminal Background Jobs

R 4.2.2 · C:/Users/qualquer/Downloads/arq1/

```
> v1 <- c(7,12,28,3,41)
> v2 <- c("Mar","Abr","Mai","Jun","Jul")
> barplot(v1,xlab="Mês",ylab="Vendas",col="blue"
+         ,main="Vendas por Mês",names.arg=v2)
>
```



Gráficos: PieChart

PieChart:
A sintaxe básica para criar um gráfico de pizza em R é:

pie(x, labels, radius, main, col, clockwise)

Onde x é um vetor que contém os dados, labels é usado para dar descrição às fatias, radius indica o raio do círculo do gráfico da torta, main indica o título do gráfico, col indica a paleta de cores.

IN

Exemplo de gráfico PieChart:

```
x <- c(21, 62, 10, 53) labels <- c(“ford", “fiat", “cherry", “bmw")
pie(x, labels, main = “gráfico pizza – marcas carros", col = rainbow(length(x)))
```

IN

Gráfico de Pizza:

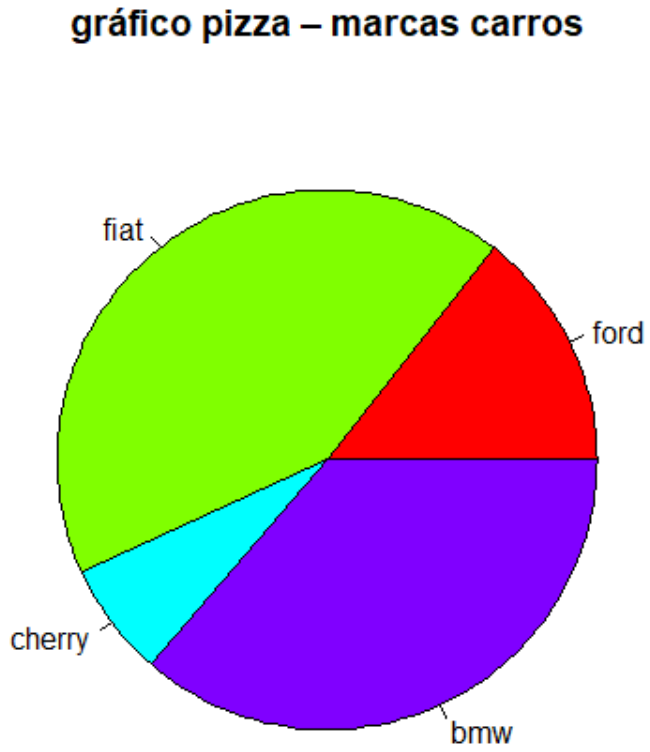
```
44
45
46
47  x <- c(21, 62, 10, 53)
48  labels <- c("ford", "fiat", "cherry", "bmw")
49  pie(x, labels, main = "gráfico pizza - marcas carros", col = rainbow(length(
50
51
52
```

46:1 (Top Level) R Script

onsole Terminal Background Jobs

R 4.2.2 · C:/Users/qualquer/Downloads/arq1/

```
x <- c(21, 62, 10, 53)
labels <- c("ford", "fiat", "cherry", "bmw")
pie(x, labels, main = "gráfico pizza - marcas carros", col = rainbow(length(x)))
```



Um gráfico de Pizza mais elaborado:

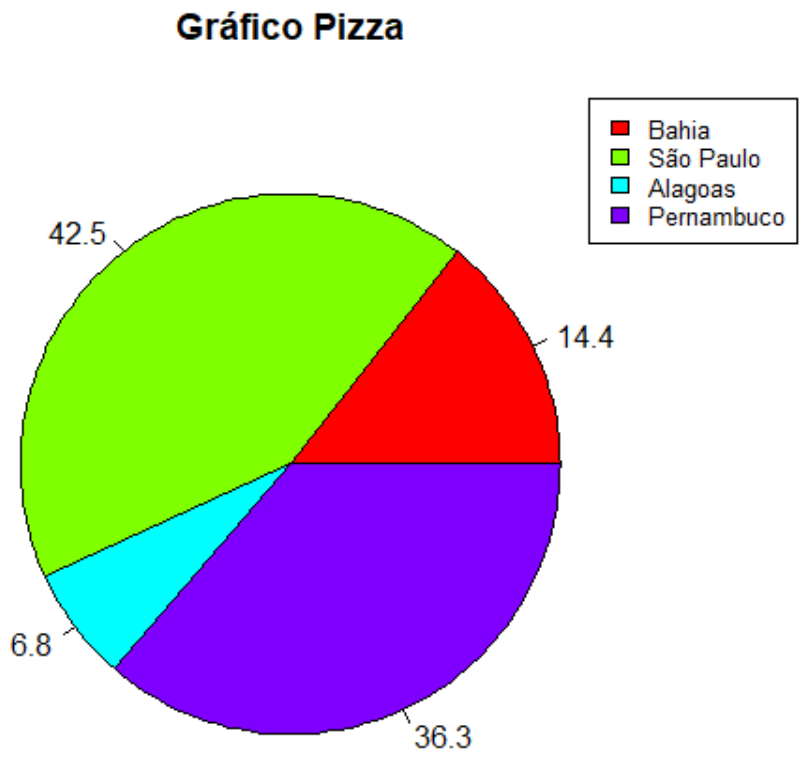
```
31  v2 <- c("Mar","Abr","Mai","Jun","Jul")
32  barplot(v1,xlab="Mês",ylab="Vendas",col="blue"
33          ,main="Vendas por Mês",names.arg=v2)
34  x <- c(21, 62, 10, 53)
35  labels <- c("Bahia", "São Paulo", "Alagoas", "Pernambuco")
36  piepercent<- round(100*x/sum(x), 1)
37  pie(x, labels = piepercent, main = "Gráfico Pizza",col = rainbow(length(x)))
38  legend("topright", c("Bahia", "São Paulo", "Alagoas", "Pernambuco"), cex = 0
39         fill = rainbow(length(x)))
40
```

34:1 (Top Level) R Script

Console Terminal Background Jobs

R 4.2.2 · C:/Users/qualquer/Downloads/arq1/

```
x <- c(21, 62, 10, 53)
labels <- c("Bahia", "São Paulo", "Alagoas", "Pernambuco")
piepercent<- round(100*x/sum(x), 1)
pie(x, labels = piepercent, main = "Gráfico Pizza",col = rainbow(length(x)))
legend("topright", c("Bahia", "São Paulo", "Alagoas", "Pernambuco"), cex = 0.8,
       fill = rainbow(length(x)))
```



IN

Gráfico BoxPlot:

BoxPlot:
A sintaxe básica para criar um boxplot em R é:

boxplot(x, xlab, ylab, main)

Onde, x é um vetor, data é o dataframe.

IN

Exemplo de um BoxPlot:

```
dados <- c(31, 21, 23, 24, 24, 25, 35)
boxplot(x = dados, xlab = "vendas",
        ylab = "quantidade de vendas", main = "análise de
vendas")
```

IN

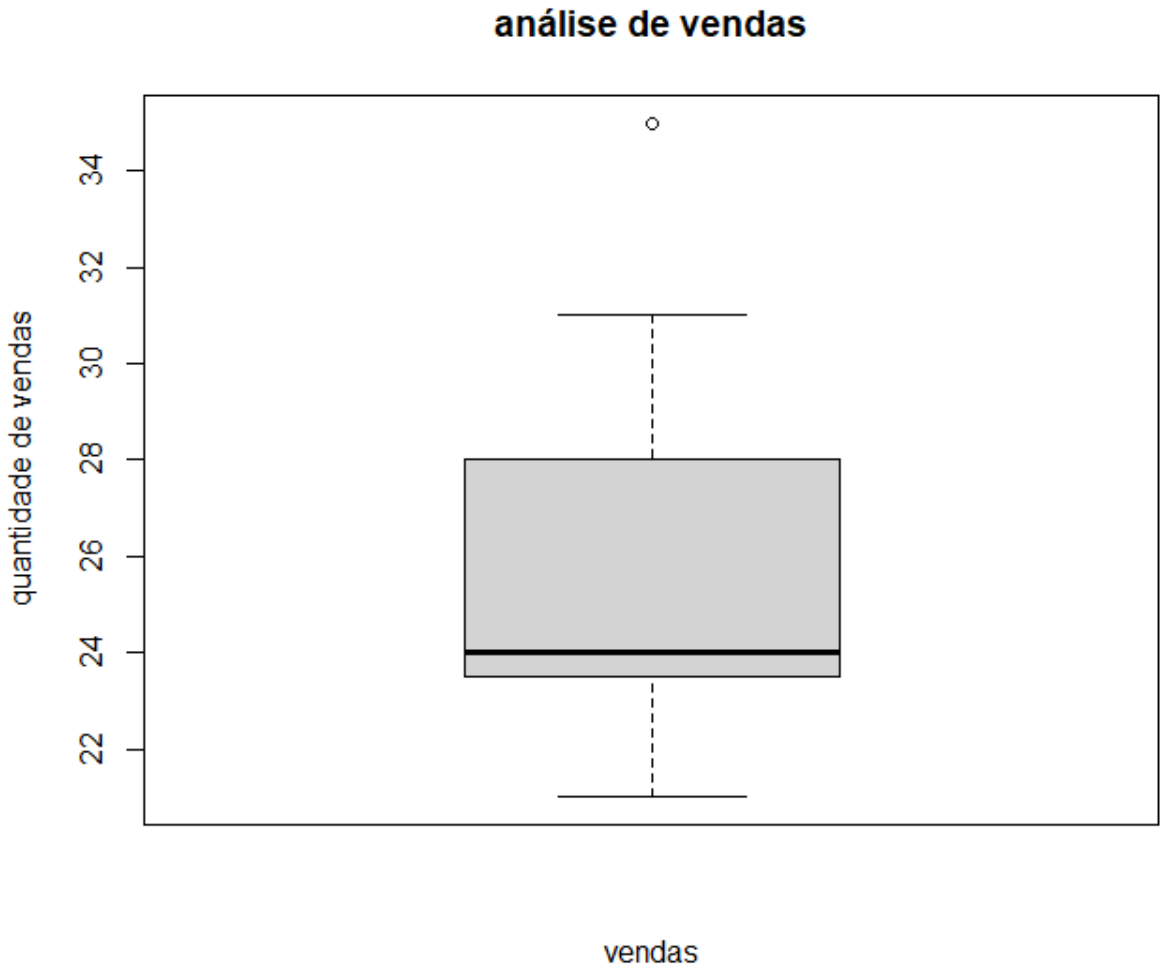
```
48  labels <- c("ford", "fiat", "cherry", "bmw")
49  pie(x, labels, main = "gráfico pizza - marcas carros", col = rainbow(length(
50
51
52  dados <- c(31, 21, 23, 24, 24, 25, 35)
53  boxplot(x = dados, xlab = "vendas",
54          ylab = "quantidade de vendas", main = "análise de vendas")
55
56
57
```

52:1 (Top Level) R Script

Console Terminal Background Jobs

R 4.2.2 · C:/Users/qualquer/Downloads/arq1/

```
> dados <- c(31, 21, 23, 24, 24, 25, 35)
> boxplot(x = dados, xlab = "vendas",
+         ylab = "quantidade de vendas", main = "análise de vendas")
>
```



IN

Gráfico Histograma:

Histograma:
A sintaxe básica para criar um histograma em R é:

hist(v,main,xlab,xlim,ylim,breaks,col,border)

Onde, xlab é usado para dar descrição do eixo x, xlim é usado para especificar a faixa de valores no eixo x, ylim é usado para especificar a faixa de valores no eixo y, break é usado para indicar a largura de cada barra.

IN

Exemplo de gráfico de histograma:

```
v <-  c(13,21,8,36,22,12,31,33,19,28,28,27)
hist(v,xlab = "idades",col = "red")
```

IN

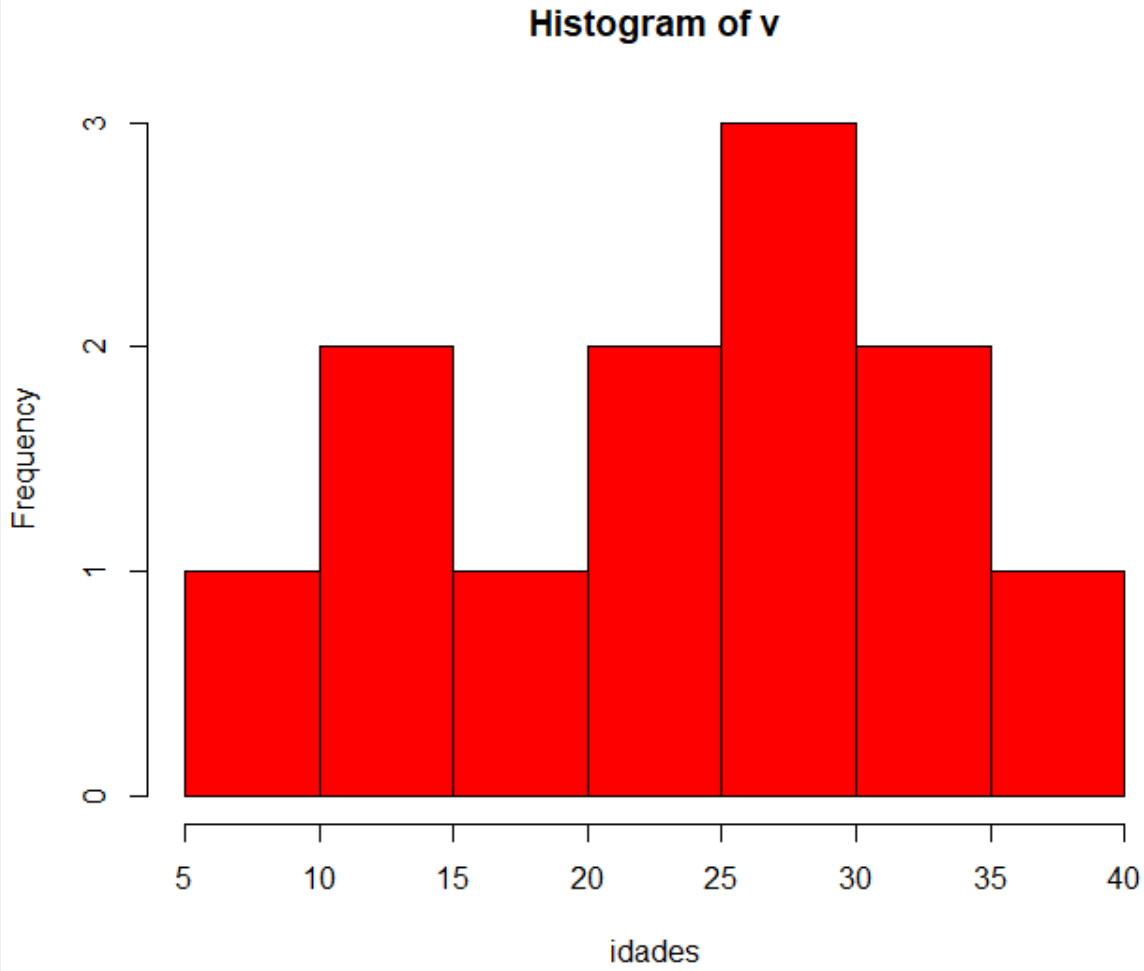
```
50
51
52  dados <- c(31, 21, 23, 24, 24, 25, 35)
53  boxplot(x = dados, xlab = "vendas",
54          ylab = "quantidade de vendas", main = "análise de vendas")
55
56
57  v <-  c(13,21,8,36,22,12,31,33,19,28,28,27)
58  hist(v,xlab = "idades",col = "red")
59
```

57:1 (Top Level) R Script

Console Terminal Background Jobs

R 4.2.2 · C:/Users/qualquer/Downloads/arq1/

```
> v <-  c(13,21,8,36,22,12,31,33,19,28,28,27)
> hist(v,xlab = "idades",col = "red")
>
```



IN

Gráfico de Linha:

Gráfico de Linha:
A sintaxe básica para criar um gráfico de linha em R é:

```
plot(v,type,col,xlab,ylab)
```

Onde, type tem o valor "p" para desenhar apenas os pontos, "l" para desenhar apenas as linhas e "o" para desenhar tanto pontos quanto linhas.

IN

Exemplo de gráfico de linha:

```
v <- c(8,12,27,5,41)
plot(v,type = "o")
```

IN

```
56
57  v <-  c(13,21,8,36,22,12,31,33,19,28,28,27)
58  hist(v,xlab = "idades",col = "red")
59
60
61  v <- c(8,12,27,5,41)
62  plot(v,type = "o")
63
```

60:1 (Top Level) R Script

Console Terminal Background Jobs

R 4.2.2 · C:/Users/qualquer/Downloads/arq1/

```
> v <-  c(13,21,8,36,22,12,31,33,19,28,28,27)
> hist(v,xlab = "idades",col = "red")
> v <- c(8,12,27,5,41)
> plot(v,type = "o")
>
```

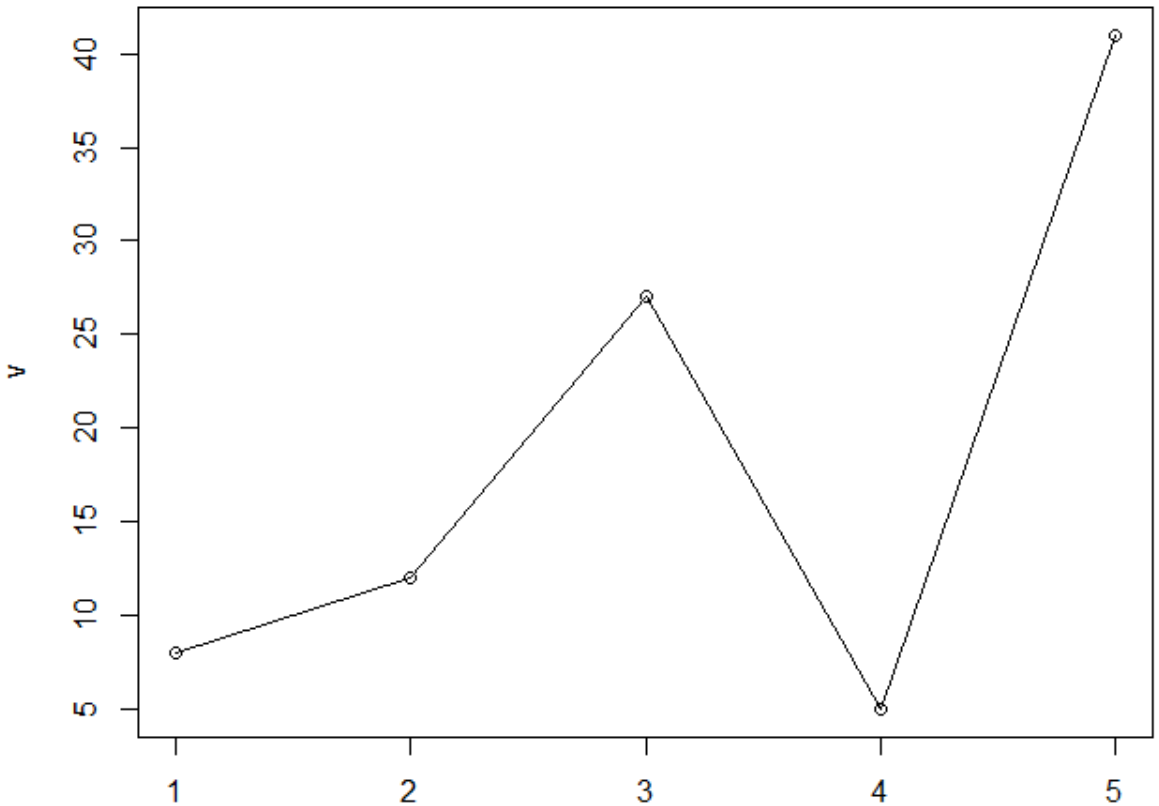


Gráfico de Dispersão de Pontos:

Scatter Plot:
A sintaxe básica para criar um gráfico de dispersão de pontos em R é:
plot(x, y, main, xlab, ylab, xlim, ylim, axes)

Onde, x e y são os dados e axes indica se ambos os eixos devem ser desenhados.

IN

Exemplo de gráfico de dispersão de pontos:

```
x <- c(8,12,27,5,41)
y <- c(19,3,8,16,13)
plot(x = x,y = y)
```

IN

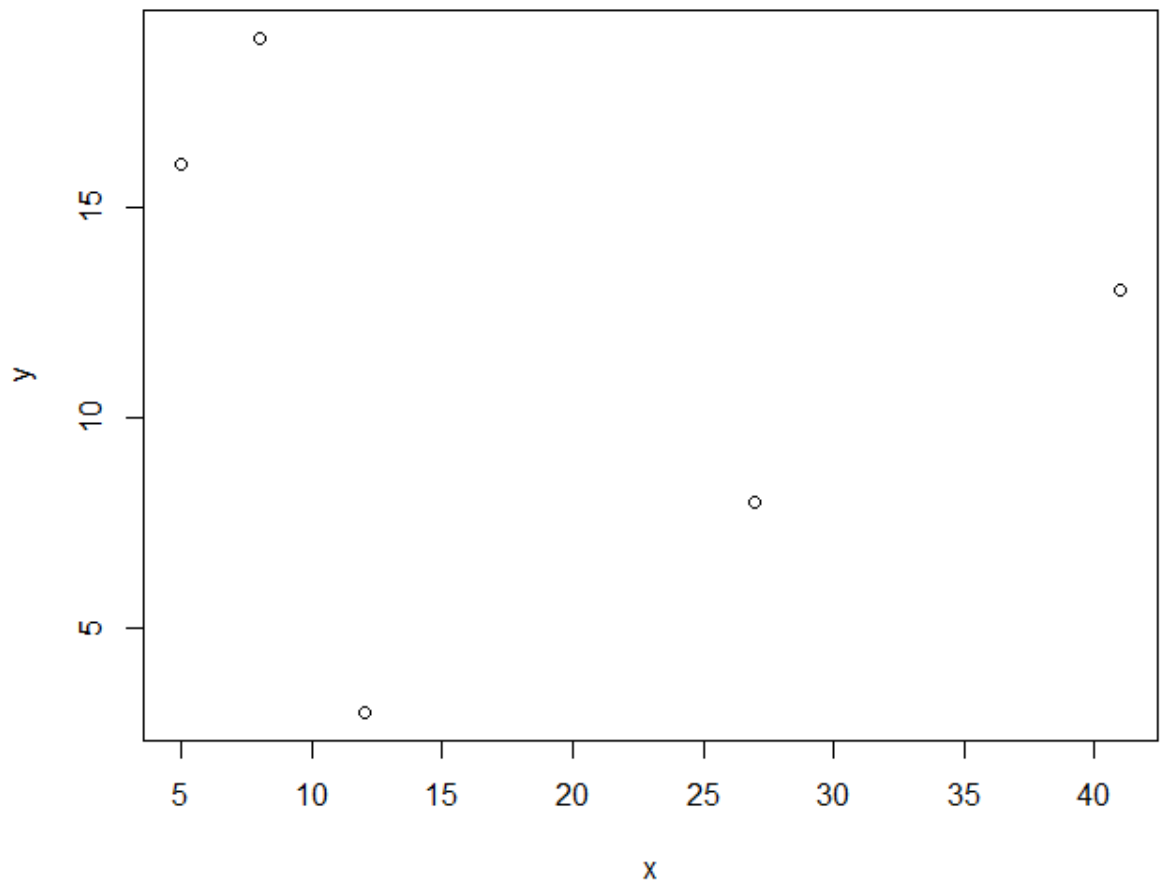
```
65  x <- c(8,12,27,5,41)
66  y <- c(19,3,8,16,13)
67  plot(x = x,y = y)
68
69
70
71
```

65:1 (Top Level) R Script

Console Terminal Background Jobs

R 4.2.2 · C:/Users/qualquer/Downloads/arq1/

```
> x <- c(8,12,27,5,41)
> y <- c(19,3,8,16,13)
> plot(x = x,y = y)
>
```



IN

Pacote Lattice:

O pacote lattice do R é extremamente útil para se construir painéis com gráficos.
No pacote lattice pode-se trabalhar com dados multivariados.

O pacote possui uma ampla variedade de funções que permitem criar gráficos básicos do pacote R.

Os comandos básicos do pacote lattice são:
Xyplot, barchart, bwplot, densityplot, dotplot, histogram.
Instale o pacote no RStudio: install.packages('lattice') e depois execute library(lattice).

IN

Pacote Lattice:

Para se construir gráficos de dispersão no lattice usa-se a função xyplot(fórmula, data), na qual:
O primeiro argumento é uma fórmula e o segundo argumento é um data frame.

Na fórmula, as variáveis são os nomes das colunas do data frame, por exemplo xyplot(altura ~ peso, data)

A variável à esquerda do sinal de til (~) é plotada no eixo-Y, enquanto que a variável à direita é plotada no eixo-X.

IN

Pacote Lattice:

Para a realização deste estudo vamos precisar instalar a biblioteca lattice e ver quais banco de dados estão instalados.

Para isto usa-se:

```
library(lattice)
data()
```

A partir dai pode-se ver o banco de dados instalado.
Vamos usar o dataframe íris do banco de dados. Este dataframe fornece características de pétalas de orquídeas.

IN

Gráfico de dispersão no pacote Lattice:

Para criar um gráfico com o dataframe íris, usa-se:

```
data("iris")
xyplot(Sepal.Length ~ Petal.Length, data = iris)
```

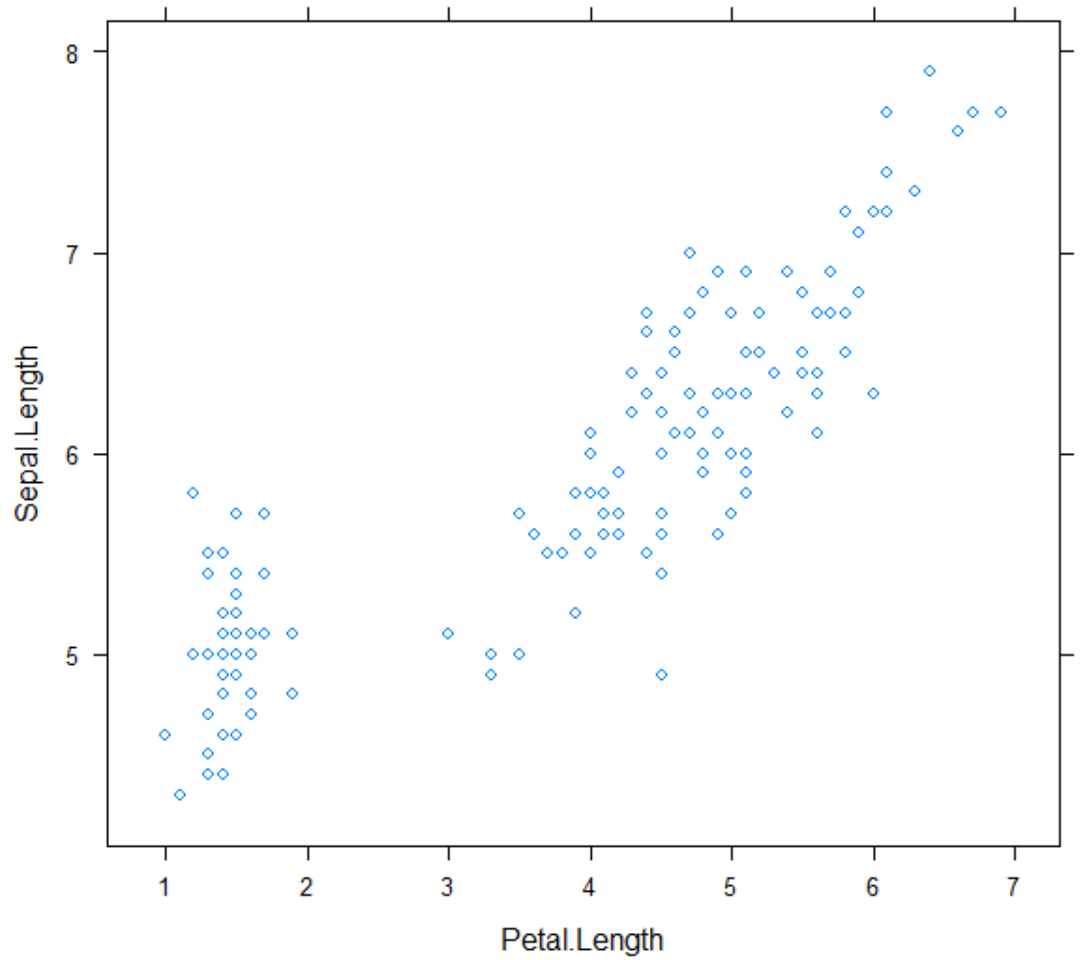
```
109
110
111 library(lattice)
112 data()
113 data(iris)
114 head(iris)
115 xyplot(Sepal.Length ~ Petal.Length, data = iris)
116
117
118
```

113:11 (Top Level) R Script

Console Terminal Background Jobs

R 4.2.2 · ~/

```
> library(lattice)
> data()
> data(iris)
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> xyplot(Sepal.Length ~ Petal.Length, data = iris)
>
```



IN

Melhorando o aspecto do gráfico:

Pode-se colorir os pontos do gráfico de dispersão com base nas categorias.

Para isto use:
xyplot(Sepal.Length ~ Petal.Length, data = iris, group = Species, auto.key = TRUE)

IN

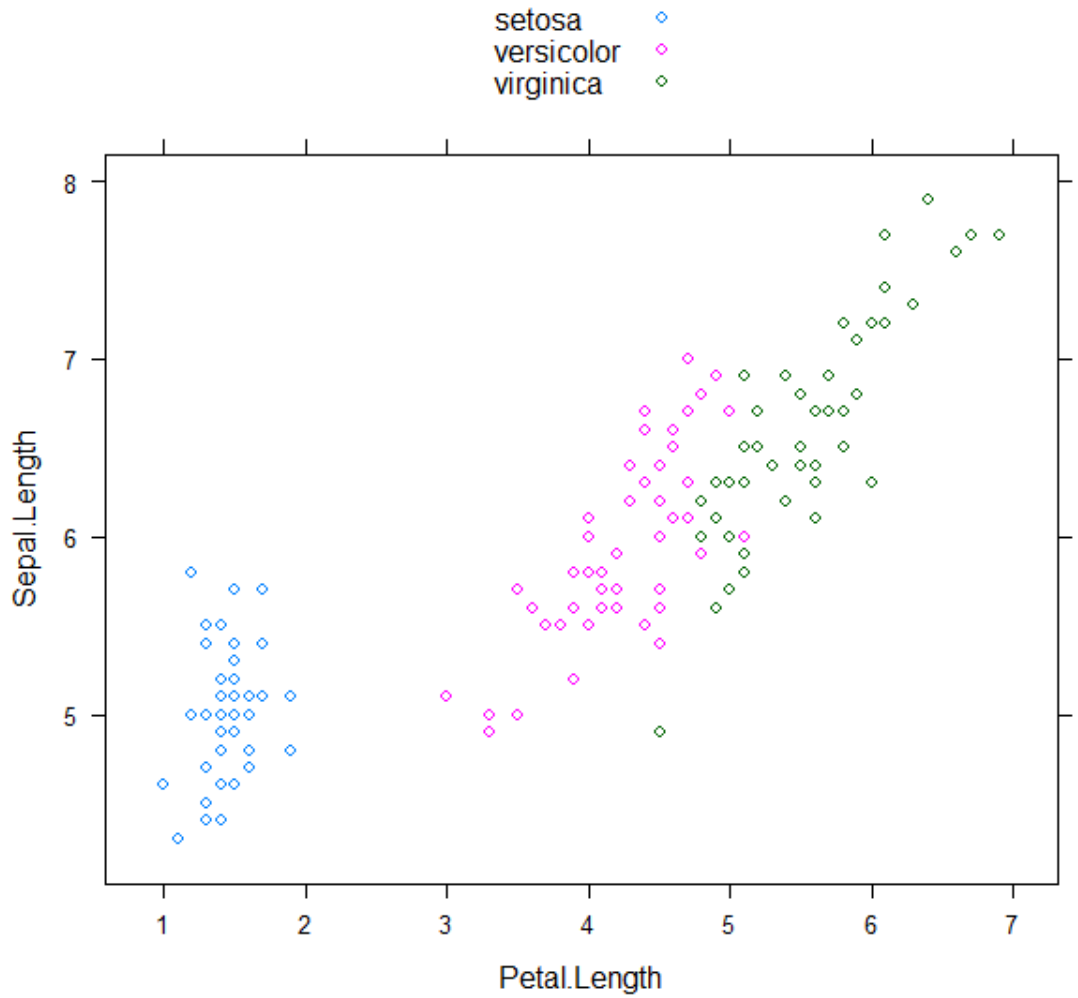
```
108
109
110
111 library(lattice)
112 data()
113 data(iris)
114 head(iris)
115 xyplot(Sepal.Length ~ Petal.Length, data = iris)
116 xyplot(Sepal.Length ~ Petal.Length, data = iris, group = Species, auto.key = TRUE
117
118
```

116:1 (Top Level) R Script

Console Terminal Background Jobs

R 4.2.2 · ~/

```
> library(lattice)
> data()
> data(iris)
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> xyplot(Sepal.Length ~ Petal.Length, data = iris)
> xyplot(Sepal.Length ~ Petal.Length, data = iris, group = Species, auto.key = TRUE)
>
```



IN

Montagem de um painel:

Pode também se criar gráficos em vários painéis com base em grupos.
Para isto use:
xyplot(Sepal.Length ~ Petal.Length | Species, group = Species, data = iris, scales = "free")
O símbolo "|" separa em painéis.
A opção group separa usando cores.
A opção auto.key adiciona a legenda.

IN

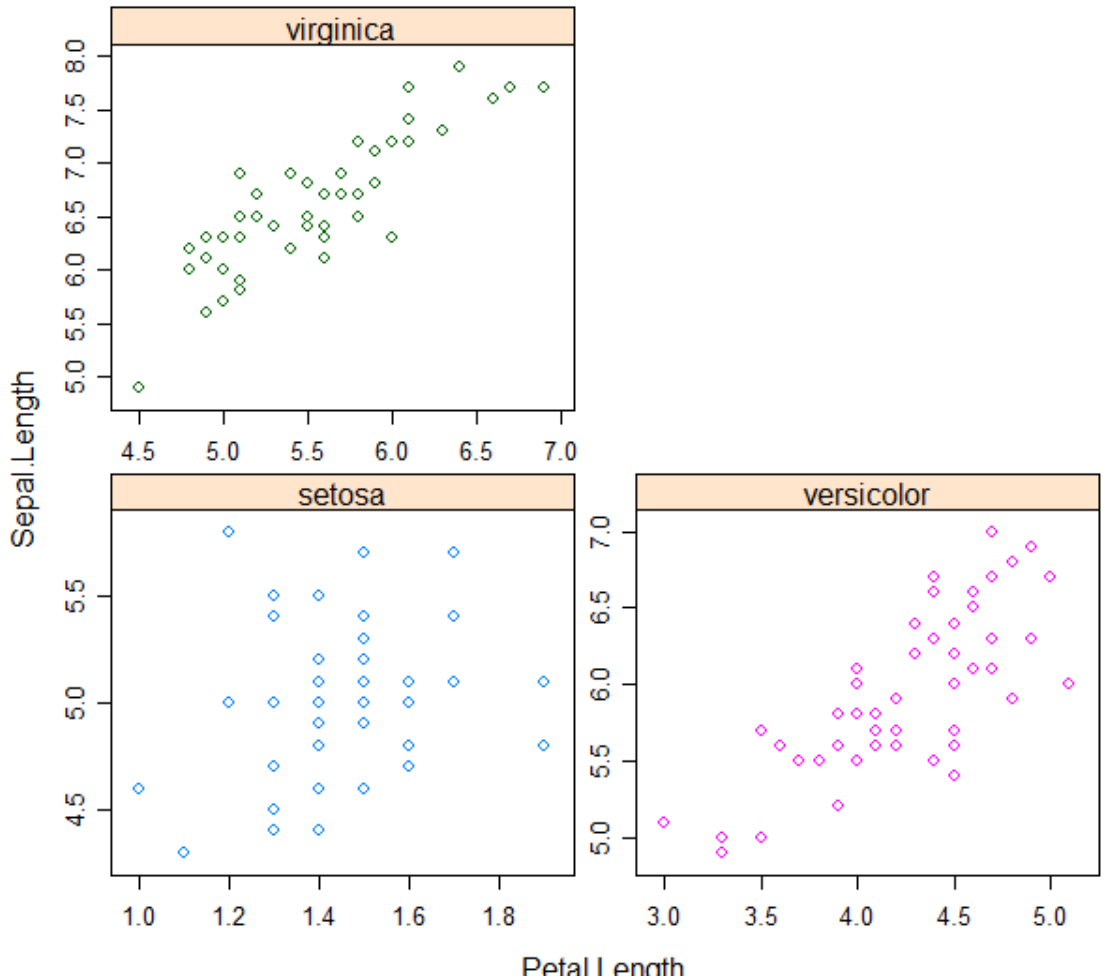
```
111  library(lattice)
112  data()
113  data(iris)
114  head(iris)
115  xyplot(Sepal.Length ~ Petal.Length, data = iris)
116  xyplot(Sepal.Length ~ Petal.Length, data = iris, group = Species, auto.key = TRUE
117  xyplot(Sepal.Length ~ Petal.Length | Species, group = Species, data = iris, scale
118
119
120
```

119:1 (Top Level) R Script

Console Terminal Background Jobs

R 4.2.2 · ~/

```
> library(lattice)
> data()
> data(iris)
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> xyplot(Sepal.Length ~ Petal.Length, data = iris)
> xyplot(Sepal.Length ~ Petal.Length, data = iris, group = Species, auto.key = TRUE)
> xyplot(Sepal.Length ~ Petal.Length | Species, group = Species, data = iris, type = c
("p", "smooth"), scales = "free")
There were 15 warnings (use warnings() to see them)
> xyplot(Sepal.Length ~ Petal.Length | Species, group = Species, data = iris, scales = "f
ree")
>
```

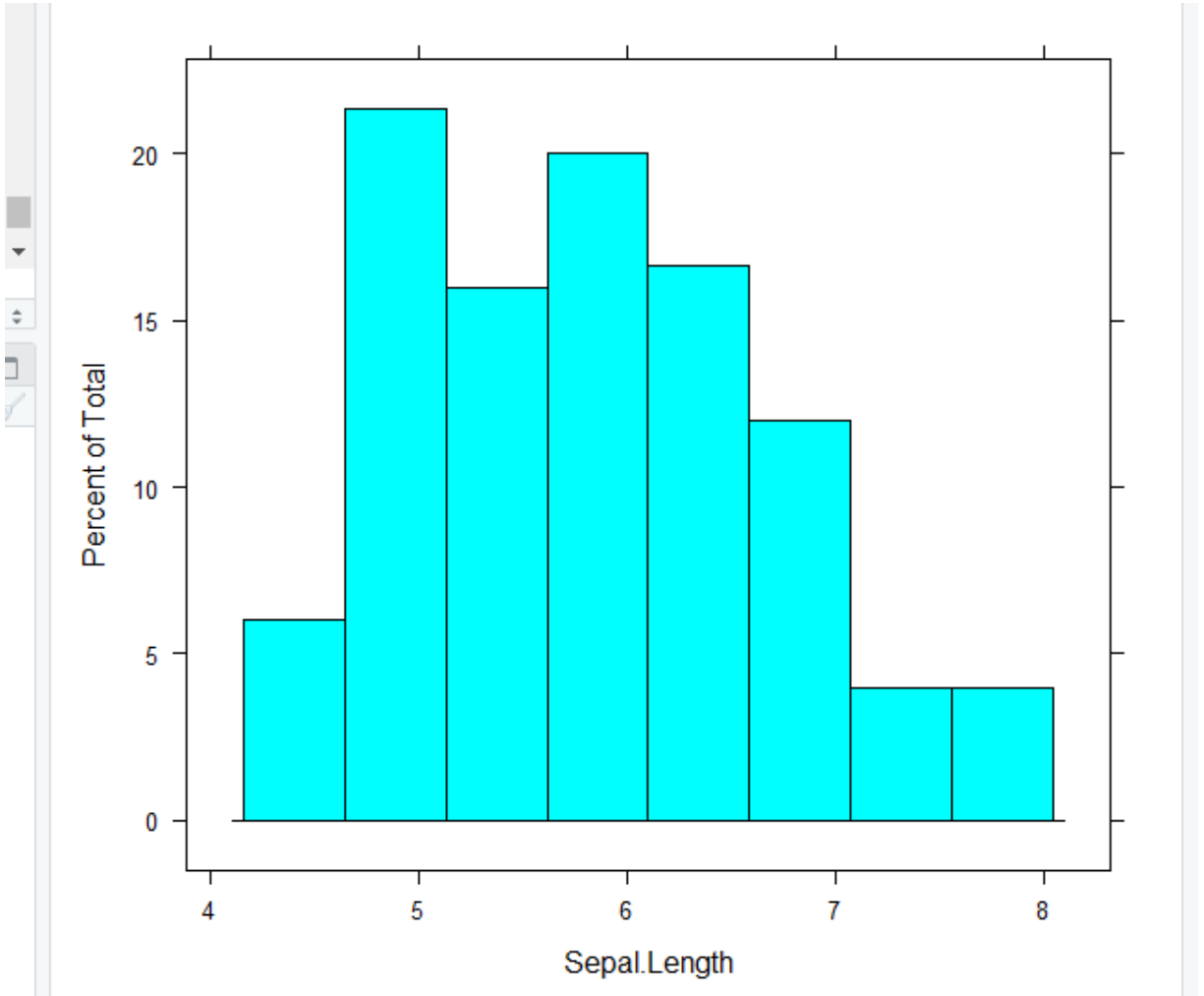


IN

Histograma:

Pode-se usar o dataframe iris do pacote lattice para se construir um histograma.
Para isto usa-se:
histogram(~ Sepal.Length, data = iris)

Observação: Um histograma é um gráfico de frequência portanto, só tem uma única variável em x.


Percent of Total
0
5
10
15
20
4
5
6
7
8
Sepal.Length

IN

Pacote Tables:

Um pacote muito usado em R é o tables. Com este pacote é possível se escrever tabelas em diversos formatos.
Para usar este pacote instale-o:

```
install.packages('tables')
library(tables)
```

IN

Exemplo de uso do pacote Tables:

Uma maneira de imprimir uma tabela com o pacote tables é da seguinte forma:

```
tabular( variável independente ~
        All(data_frame)*(mean + sd), data=data_frame )
```

IN

Exemplo de construção de uma tabela:

Use o dataframe íris para determinar o valor médio e o desvio padrão dos dados por espécie.

Para isto usa-se:
tabular(Species ~ All(iris)*(mean + sd), data=iris)

IN

```
131  tabular( Species ~
132             All(iris)*(mean + sd), data=iris )
133
134  head(iris)
135
136
137
138
```

131:1 (Top Level) R Script

Console Terminal Background Jobs

R 4.2.2 · ~/

```
1           5.1         3.5          1.4         0.2  setosa
2           4.9         3.0          1.4         0.2  setosa
3           4.7         3.2          1.3         0.2  setosa
4           4.6         3.1          1.5         0.2  setosa
5           5.0         3.6          1.4         0.2  setosa
6           5.4         3.9          1.7         0.4  setosa
> tabular( Species ~
+             All(iris)*(mean + sd), data=iris )

            Sepal.Length          Sepal.Width          Petal.Length          Petal.Width
 Species    mean          sd      mean          sd     mean          sd      mean
 setosa     5.006         0.3525  3.428         0.3791 1.462         0.1737  0.246
 versicolor 5.936         0.5162  2.770         0.3138 4.260         0.4699  1.326
 virginica  6.588         0.6359  2.974         0.3225 5.552         0.5519  2.026


 sd
 0.1054
 0.1978
 0.2747
```