02

Praticando: Desvio Padrão

Olá! Então, acabamos de ver um pouquinho sobre **variância** e **desvio padrão**.

Vamos aplicar isso no **R** agora.

E você, provavelmente, induziu que é fácil. O **R** tem tudo isso pronto. Está vendo como foi uma boa ideia usar o **R**?

Vamos lá!

Criando uma lista de números quaisquer novamente:

```
> numeros <- c(1, 2, 3, 5, 6, 7, 8, 11, 2, 3, 44, 55, 67, 12, 34, 56)
```

Se eu quiser calcular a **variância**:

```
> var(numeros)
```

E está aqui a nossa variância: 531.

Mas, a gente sabe que a **variância** sozinha não fala muita coisa para gente. Então eu vou guardar isso em uma variável que eu vou chamar de *variância*.

```
> variancia <- var(numeros)
```

E aí, eu vou calcular a **raiz quadrada da variância**:

```
> sqrt(variancia)
```

Isso me dá o meu **desvio padrão**. A gente sabe que o **desvio padrão** é a **raiz quadrada da variância**.

Mas eu preciso fazer em dois passos? Claro que não. O **R** tem pronto:

```
> sd(numeros)
```

Que me dá o mesmo valor; o resultado tem que bater.

Então calcular o **desvio padrão** é fácil: use `> sd` (*standard deviation*), passe a lista e ele calcula para mim.

Para deixar mais legal aqui, o que eu vou mostrar agora para você é como ler dados de um arquivo. Porque até então nós estamos criando esses números na mão, mas eu quero ler de um arquivo. Então, imagina que o meu teste estatístico, todo esse meu código que leu com Estatística, ele usa dados que vêm de um lugar. Esse algum lugar pode ser um arquivo **.csv**, que é aquele formato bastante comum na Computação, que são coisas separadas por **vírgulas**.

Estou aqui em um diretório qualquer, e eu vou criar um arquivo que eu vou chamar de **numeros.csv**.

Eu usei o *vi as hex* que é um editor de texto; você pode usar o *bloco de notas*, o que você quiser.

E aqui eu vou colocar alguns valores, por exemplo:

```
|1|2| |2|4| |3|6| |4|8|
```

Vou salvar esse arquivo e, dê uma olhada, se eu imprimir esse arquivo está lá:

```
Alura:- aluraS vi numeros.csv
Alura:- aluraS cat numeros.csv
1,2
2,4
3,6
4,8

Alura:- aluraS
```

Agora, eu estou no **R**:

```
Alura:- aluraS R
```

Eu quero ler esses números. Como eu faço? Use *read*, csv - que é o formato -, abro parênteses e vou passar o parâmetro para ele - *file* -, /Users, /alura - que é o diretório em que estou, eu vou até dar um pwd para confirmar, /numeros.csv.

```
> read.csv(file="/Users/alura/numeros.csv")
```

Se você estiver no Windows é: c:/...diretório...nome do arquivo. É o caminho completo.

Essa linha aqui: `> read.csv(file="/Users/alura/numeros.csv")` vai ler todo o conteúdo desse numeros.csv. Só que eu tenho que guardar isso em algum lugar. Vou guardar em uma variável. O método csv retorna para gente uma estrutura de dados com essas coisas.

Então se eu fizer `> nums` e der um ENTER, dá uma olhada:

```
> nums
```

```
|---| X1 | X2 | | 1 | 2 | 4 | | 2 | 3 | 6 | | 3 | 4 | 8 |
```

Em X1, ele colocou a primeira coluna que era 2, 3 e 4; em X2, a segunda coluna, 4, 6 e 8.

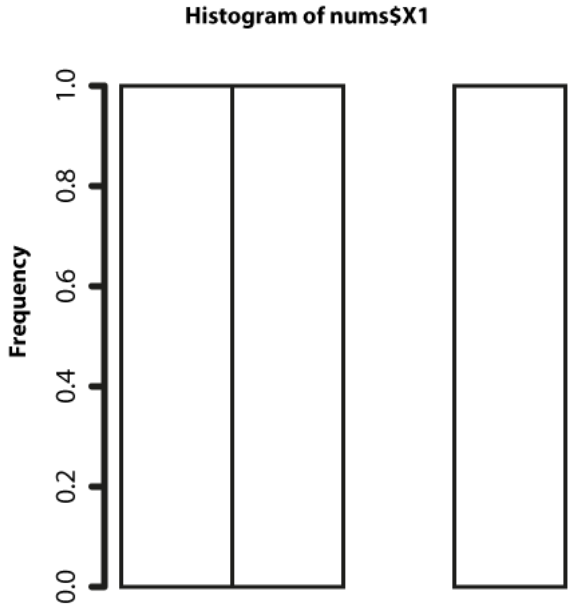
E eu posso, agora, por exemplo, exibir só uma coluna:

```
> nums$X1 [1] 2 3 4
```

Veja só, uma coisa nova no **R**. O **$** (dólar) quando eu tenho essa estrutura que é um *dataset*, eu consigo pegar uma coluna usando o **$**.

Posso também traçar o **histograma** de uma coluna só:

```
> hist(nums$X1)
```



Ele me desenhou o histograma e assim por diante.

Então assim que a gente lê de arquivos .csv: read.csv. Eu consigo ler de um monte de lugares diferentes, e aí você olha no manual do **R**.

Nessa aula eu mostrei para vocês **variância** e **desvio padrão**. Simples! O **R** faz toda a mágica para gente.

Obrigado!