

# Aula 03 - Minicurso Python

Property Ítalo Lima

## Introdução

Na última aula, verificamos algumas descrições estatísticas e fizemos observações sobre as suas medidas, mas não visualizamos os dados através de gráficos, então, o intuito desta aula é dar continuidade ao estudo. Veremos formas fáceis de gerar diferentes tipos de gráfico e inclusive configurá-los para uma exibição mais limpa e representativa.

**Para isso, utilizaremos uma nova biblioteca, a Matplotlib.**



A Matplotlib é uma biblioteca do Python para visualização de dados e criação de gráficos.



Para importar a biblioteca no nosso Notebook, adicionaremos essa linha: `import matplotlib.pyplot as plt`

## Prática

### Importando as Bibliotecas e os Dados

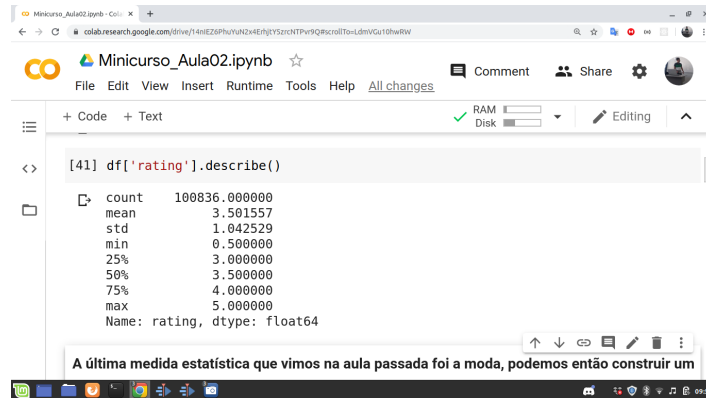
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
df = pd.read_csv('https://raw.githubusercontent.com/alura-cursos/introducao-a-data-science/master/aula0/ml-latest-small/ratings.csv')
df.head(10)
```

A screenshot of a Jupyter Notebook interface. The top bar shows the file name "Minicurso\_Aula02.ipynb". The notebook has a menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". Below the menu bar, there are tabs for "Code" and "Text". The "Code" tab is active, showing two code cells. The first cell contains the import statement: `import matplotlib.pyplot as plt`. The second cell contains the code to read a CSV file and display its first 10 rows: `df = pd.read_csv('https://raw.githubusercontent.com/alura-cursos/introducao-a-data-science/master/aula0/ml-latest-small/ratings.csv')` and `df.head(10)`. The output of the second cell is a table with 10 rows and 4 columns: "userId", "movieId", "rating", and "timestamp". The table shows the first 10 rows of the ratings dataset.

	userId	movieId	rating	timestamp
0	1	1	4.0	964982703
1	1	3	4.0	964981247
2	1	6	4.0	964982224
3	1	47	5.0	964983815
4	1	50	5.0	964982931
5	1	70	3.0	964982400
6	1	101	5.0	964980868
7	1	110	4.0	964982176
8	1	151	5.0	964984041
9	1	157	5.0	964984100

Como Calcular as Medidas obtidas no método Describe( ) ?



```
[41] df['rating'].describe()
count    100836.000000
mean      3.501557
std       1.042529
min       0.500000
25%       3.000000
50%       3.500000
75%       4.000000
max       5.000000
Name: rating, dtype: float64
```

A última medida estatística que vimos na aula passada foi a moda, podemos então construir um

#### ▼ Média

A média é calculada assim:

$\Sigma \text{Valores} / \text{Quantidade de valores}$



A média é uma boa representação para os dados, sendo bastante utilizada em estudos, no entanto é bastante sensível a outliers.

#### ▼ Desvio Padrão

O Desvio Padrão é uma medida de dispersão e é obtido a partir de:

Raiz Quadrada da Variância

A Variância por sua vez, é obtida dessa forma:

$(\Sigma (x_i - u)^2) / N$  para i começando em 1 e indo até N.



Esse Cálculo acima é para o Desvio Padrão Populacional, para calcular o amostral a divisão deve ser feita por n-1.



A divisão por n-1 é uma forma de melhor estimar a variância amostral em comparação com a populacional, visto que a divisão por n-1 origina um resultado maior que por n, concorda ?



Em um gráfico com diferentes amostras e cálculos para a variância é possível observar que a variância amostral converge para a populacional, devido essa nova forma de se fazer a divisão, a divisão por n acabava subestimando os valores, e uma possível divisão por n-2 superestima esses, assim o n-1 foi definido para representar uma amostra não viciada.

#### ▼ Quartis

Os quartis separam os dados ordenados em 4 partes. O segundo quartil é o primeiro a ser calculado, e representa também a mediana. O 1º e 3º quartis são calculados com base no segundo, dividindo os dados.

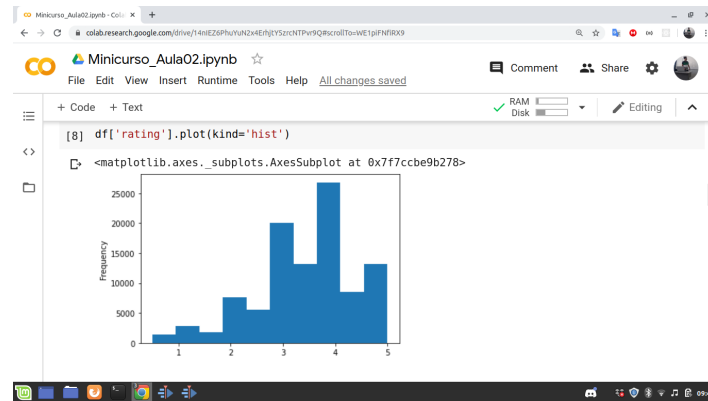


O gráfico de caixa e bigodes (box-plot) representa a concentração de 50% dos dados.

## Visualizando os Dados Através de Gráficos

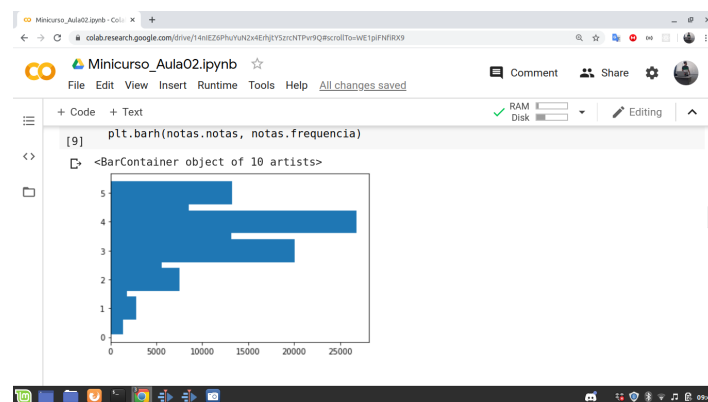
## Histograma

```
df['rating'].plot(kind='hist')
```



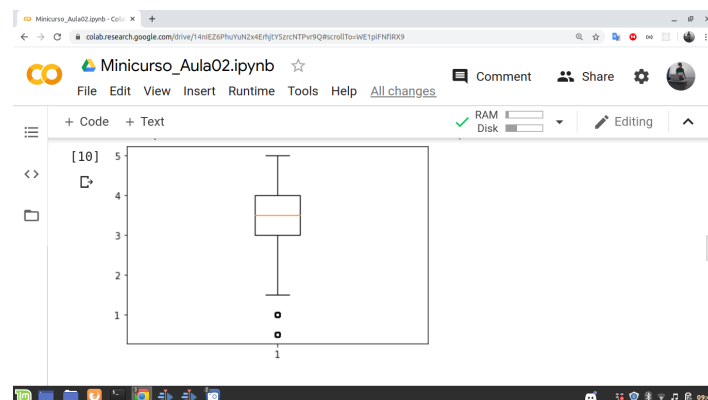
## Gráfico de Barras Horizontal

```
plt.barh(notas.notas, notas.frequencia)
```



## Box-Plot

```
plt.boxplot(df['rating'])
```



# Configurando os Gráficos

Como pode-se observar, é fácil gerar gráficos com a biblioteca matplotlib, no entanto, os gráficos básicos são muito simples, pequenos, e visualmente "feios".



**Podemos melhorar a exibição dos mesmos através de propriedades e métodos da própria biblioteca.**

## Configurando o Histograma

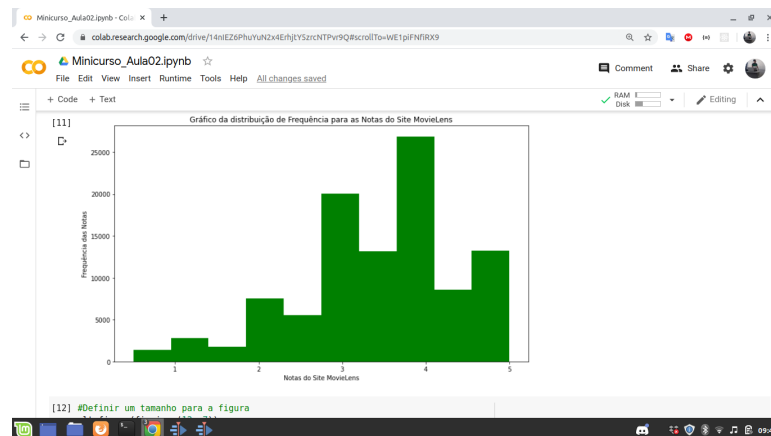
```
#Definir um tamanho para a figura
plt.figure(figsize=(12, 7))

#Criar o gráfico e alterando a cor padrão
df['rating'].plot(kind='hist', color='green')

#Atribuir um Título para o Gráfico
plt.title('Gráfico da distribuição de Frequência para as Notas do Site MovieLens')

#Configurar os labels do eixo X e Y, respectivamente
plt.xlabel('Notas do Site MovieLens')
plt.ylabel('Frequência das Notas')

#Exibir o Gráfico
plt.show()
```



## Configurando o Gráfico de Barras

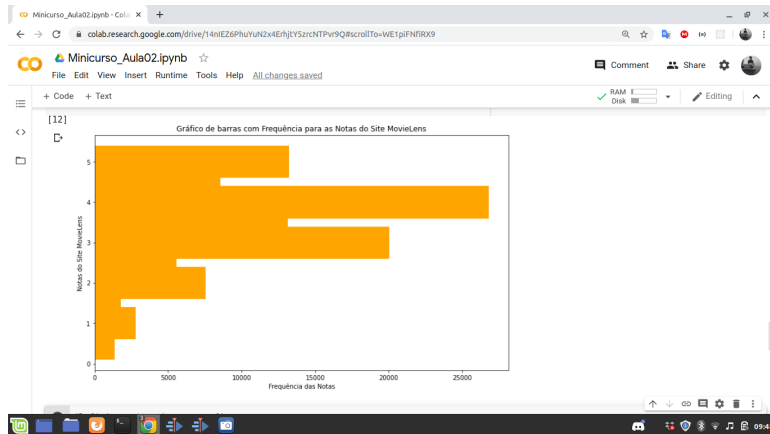
```
#Definir um tamanho para a figura
plt.figure(figsize=(12, 7))

#Criar o gráfico de barras e alterar a sua cor padrão
plt.barh(notas.notas, notas.frequencia, color='orange')

#Atribuir um Título para o Gráfico
plt.title('Gráfico de barras com Frequência para as Notas do Site MovieLens')

#Configurar os labels do eixo X e Y, respectivamente
plt.xlabel('Frequência das Notas')
plt.ylabel('Notas do Site MovieLens')

#Exibir o Gráfico
plt.show()
```



## Configurando o Box-Plot

```
#Definir um tamanho para a figura
plt.figure(figsize=(12, 7))

#Criar o gráfico box-plot
plt.boxplot(df['rating'])

#Atribuir um Título para o Gráfico
plt.title('Box-Plot para as Notas do Site MovieLens')

#Configurar o label do eixo Y
plt.ylabel('Notas do Site MovieLens')

#Exibir o Gráfico
plt.show()
```

