

DATA ANALYTICS

VISUALIZAÇÃO DE DADOS

AULA 03

SUMÁRIO

O QUE VEM POR AÍ?	3
CONHEÇA SOBRE O ASSUNTO	4
HANDS ON	8
O QUE VOCÊ VIU NESTA AULA	9
REFERÊNCIAS	10

EMSE

O QUE VEM POR AÍ?

Até aqui você aprendeu a como carregar fontes de dados de maneiras distintas, vimos algumas manipulações base para sua jornada e também foi apresentado à biblioteca Seaborn do Python.

Para te ajudar a complementar suas análises, essa aula mostrará como melhorar visualizações gráficas, utilizando funções em conjunto com o Matplotlib.

A base de dados você encontra aqui <<https://github.com/alura-tech/pos-datascience-introducao-a-visualizacao/archive/refs/heads/dados.zip>>.

Agora vamos desbravar esse maravilhoso mundo dos dados!

CONHEÇA SOBRE O ASSUNTO

Complementando o assunto da aula anterior, podemos pegar aquele mesmo notebook e implementar algumas mudanças.

Ficou cansado de olhar aquele fundo branco e sentiu saudades da época de fazer gráficos em papel quadriculado (uma relíquia para quem lembra!)?

E se for possível ativar esse recurso na sua visualização no Python?

Experimente pegar qualquer uma das visualizações e adicionar uma linha:

```
plt.grid()
```

Essa ação pode resultar em uma mágica:

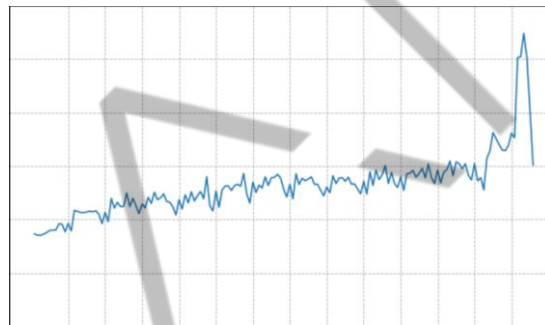


Figura 1 – grid
Fonte: Elaborado pelo autor (2023)

Basicamente, o `grid()` é uma função do Matplotlib que serve para desenhar uma grade no gráfico. A grade pode ser usada para ajudar na leitura dele, tornando mais fácil para o usuário identificar valores específicos em um eixo.

Por exemplo:

```
import matplotlib.pyplot as plt

x = [1, 2, 3, 4, 5]
y = [2, 4, 6, 8, 10]

plt.plot(x, y)
plt.grid(True)
plt.show()
```

Cujo resultado é:

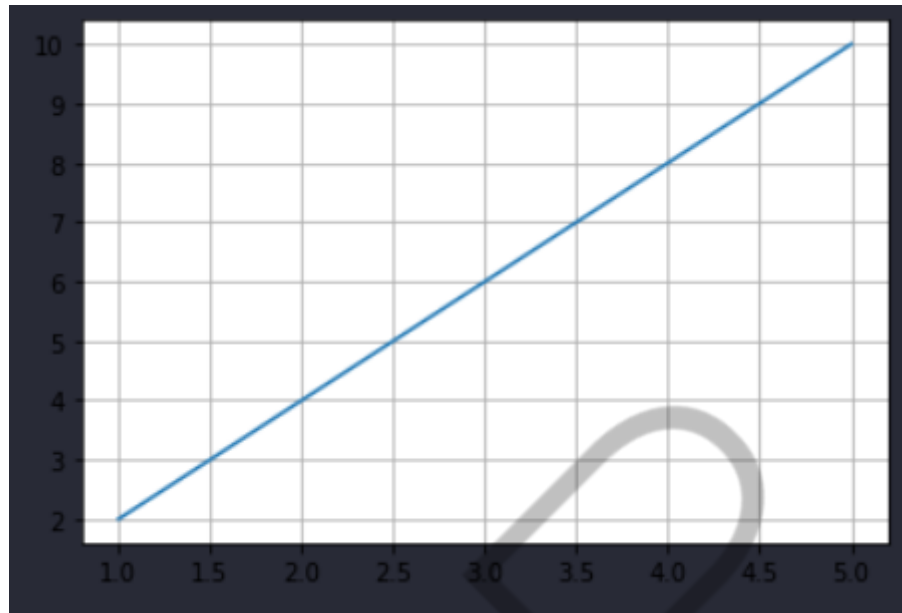


Figura 2 – Grid com código
Fonte: Elaborado pelo autor (2023)

Este exemplo desenha uma grade sobre o gráfico de linha plotado. A opção `True` passada como argumento para `grid()` significa que a grade deve ser exibida no gráfico. Se você passar `False` como argumento, a grade não será exibida.

Essa é uma das várias possibilidades bacanas que podemos fazer utilizando o melhor do Matplotlib com o Seaborn.

Além disso, podemos fazer outras coisas, como alterar rótulos utilizando `set_major_locator()`.

A função `set_major_locator` do Matplotlib é usada para definir a localização dos marcadores principais em um eixo. Os marcadores principais são os valores numerais exibidos ao longo do eixo, que ajudam a identificar as posições dos dados no gráfico.

Por exemplo, você pode usar a função `set_major_locator` para exibir marcadores principais em intervalos regulares de tempo, como dias, meses ou anos, ou em valores numéricos regulares, como 1, 2, 3 etc.

Vejamos um caso utilizando o Python:

```

import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import numpy as np
import datetime

dates = [datetime.datetime(2021, 1, 1),
         datetime.datetime(2021, 2, 1),
         datetime.datetime(2021, 3, 1),
         datetime.datetime(2021, 4, 1),
         datetime.datetime(2021, 5, 1),
         datetime.datetime(2021, 6, 1),
         datetime.datetime(2021, 7, 1)]
y = np.array([1, 4, 9, 16, 25, 36, 49])

plt.plot(dates, y)

# Define a localização dos marcadores principais como intervalos de 1 mês
ax = plt.gca()
ax.xaxis.set_major_locator(mdates.MonthLocator())

# Formata os marcadores principais como meses e anos
ax.xaxis.set_major_formatter(mdates.DateFormatter('%b %Y'))

plt.show()

```

Cujo resultado é:

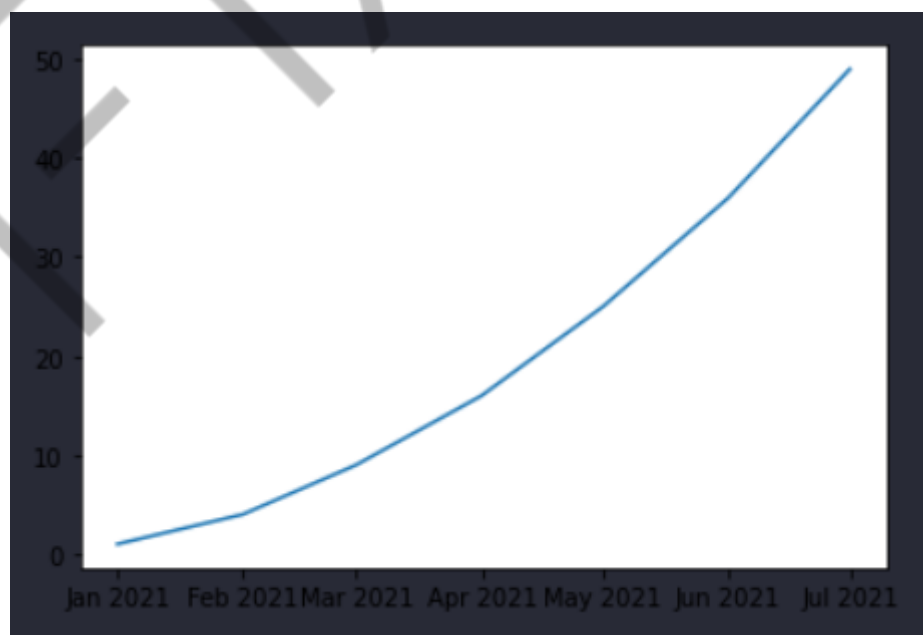


Figura 3 – Gráfico com ajuste de datas
Fonte: Elaborado pelo autor (2023)

Neste exemplo, a função `set_major_locator` é usada para definir a localização dos marcadores principais, como intervalos de 1 mês, usando o `MonthLocator` do Matplotlib. Além disso, a função `set_major_formatter` é usada para formatar os marcadores principais como meses e anos.

Vamos à um exemplo alternativo:

```
import matplotlib.pyplot as plt
import numpy as np
import matplotlib.ticker as ticker

x = np.arange(0, 10, 0.1)
y = x ** 2

plt.plot(x, y)

# Define a localização dos marcadores principais como intervalos de 2
ax = plt.gca()
ax.xaxis.set_major_locator(ticker.MultipleLocator(2))

plt.show()
```

Onde o resultado é:

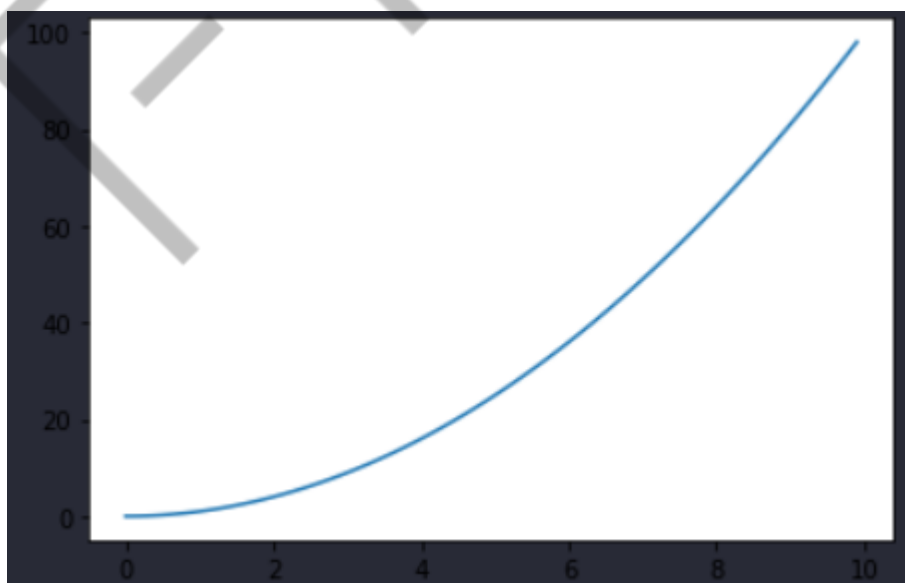


Figura 4 – Outro exemplo de gráfico
Fonte: Elaborado pelo autor (2023)

Neste caso, a função `set_major_locator` é usada para definir a localização dos marcadores principais como intervalos de 2, usando o `MultipleLocator` do Matplotlib. Isso significa que os marcadores principais serão exibidos a cada 2 unidades no eixo x.

HANDS ON

Agora chegou o momento de vermos, na prática, como começar a importar nossos dados e trabalhar com eles via programação. A ideia é não nos limitarmos apenas ao código explícito no hands on, então é sempre bom procurar a documentação das bibliotecas, explorar novas funcionalidades e muito mais!

Aqui foram abordadas novas funcionalidades na manipulação de gráficos, e a prática disso fará com que você crie visualizações melhores. Isso não significa que de fato existe uma biblioteca melhor que a outra, mas sim que todas se complementam.

Confira a aula e entenda um pouco melhor na prática o fundamento da utilização desses recursos importantes!

O QUE VOCÊ VIU NESTA AULA

Como criar um Scatterplot com o Seaborn; como formatar ticks ajustando sua localização da forma mais adequada possível, e como criar um grid em suas visualizações para facilitar a análise.

IMPORTANTE: não esqueça de praticar com o desafio da disciplina, para que assim você possa aprimorar os seus conhecimentos!

Você não está sozinho ou sozinha nesta jornada! Te esperamos no Discord e nas lives com os nossos especialistas, onde você poderá tirar dúvidas, compartilhar conhecimentos e estabelecer conexões!

REFERÊNCIAS

DOCUMENTAÇÃO PANDAS. <<https://pandas.pydata.org/>>. Acesso em: 09 fev 2023.

DOCUMENTAÇÃO SEABORN. <<https://seaborn.pydata.org/>>. Acesso em 09 fev. 2023.

GOOGLE COLAB. <<https://colab.research.google.com/>>. Acesso em: 09 fev. 2023.

EMENDAS

PALAVRAS-CHAVE

Python. Pandas. Dataframe.

EMASP

The background is a dark blue field filled with numerous small, light blue dots, resembling a starry sky. Overlaid on this are several large, flowing, wavy lines in shades of teal, blue, and yellow. These lines create a sense of movement and depth. Scattered throughout the composition are various geometric shapes: a circle with the number '7' inside, a circle with an 'X' inside, a circle with an 'O' inside, and a hexagon. The overall aesthetic is futuristic and technological.

POSTECH