

**Educação
Profissional
Paulista**

Técnico em
**Ciência de
Dados**

Introdução ao Jupyter Notebook e à configuração de ambiente

Introdução ao Jupyter Notebook

Funcionalidades avançadas do Jupyter Notebook

Código da aula: [DADOS]ANO1C1B4S28A2

Mapa da unidade 6 Componente 1

Introdução ao
Jupyter Notebook e à
configuração de
ambiente

semana

28

Você está aqui!
Introdução ao
Jupyter Notebook

semana

30

Principais
funcionalidades do
Jupyter Notebook

semana

31

Configuração de
ambiente para o
Jupyter Notebook

**Introdução ao
Jupyter Notebook e à
configuração de
ambiente**

Mapa da unidade 6 Componente 1

Você está aqui!

Introdução ao Jupyter Notebook

Aula 2

Código da aula: [DADOS]ANO1C1B4S28A2

28



Objetivos da aula

- Introduzir, configurar e utilizar o Jupyter Notebook.



Recursos didáticos

- Recurso audiovisual para exibição de vídeos e imagens;
- Acesso ao laboratório de informática e/ou à internet.



Duração da aula

50 minutos.



Competências técnicas

- Usar ferramentas de desenvolvimento de software como o Jupyter Notebook;
- Compreender e dominar técnicas de manipulação de dados.



Competências socioemocionais

- Analisar informações de forma crítica e tomar decisões;
- Avaliar a qualidade dos dados e a eficácia dos modelos e das técnicas de análise;
- Colaborar efetivamente com outros profissionais, como cientistas de dados e engenheiros de dados;
- Trabalhar em equipes multifuncionais colaborando com colegas, gestores e clientes.

Construindo
o **conceito**

Uso de células mágicas (*magic commands*)

As células mágicas são comandos especiais que oferecem funcionalidades adicionais ao Jupyter Notebook e começam com:

- ▶ % – Para comandos de linha única.
- ▶ %% – Para comandos de múltiplas linhas.



Curiosidade

As células mágicas são especialmente úteis para desenvolvedores que trabalham com prototipagem rápida e análises exploratórias de dados.

Construindo o **conceito**

Uso de células mágicas (*magic commands*)

Alguns comandos permitem a execução de tarefas, como:

- ▶ `%timeit` - É usado para medir o tempo de execução de uma célula.
- ▶ `%load` - Carrega código de um arquivo.
- ▶ `%matplotlib inline` - Permite a exibição de gráficos diretamente nas células do notebook.



Tome nota

O uso de células mágicas pode otimizar o fluxo de trabalho, permitindo a execução de comandos do sistema, a importação de módulos e a configuração de variáveis ambientais sem sair do ambiente interativo do notebook.

Construindo o **conceito**

Visualização de dados com bibliotecas gráficas

O Jupyter Notebook suporta várias bibliotecas gráficas que permitem criar visualizações de dados diretamente nas células. As bibliotecas mais populares incluem:



Matplotlib

É uma biblioteca de plotagem versátil e amplamente utilizada, que pode gerar gráficos lineares, histogramas, *scatter plots* e muito mais.



Seaborn

É construído sobre o Matplotlib e oferece uma interface de alto nível para criar visualizações estatísticas atraentes.



Plotly

Permite a criação de gráficos que podem ser manipulados dinamicamente no navegador.



Bokeh

Fornecer ferramentas para construir dashboards interativos que podem ser utilizados em aplicações web.

Construindo o conceito

Integração com bases de dados

O Jupyter Notebook pode se conectar a diversas bases de dados, como SQL, NoSQL e APIs de dados, utilizando bibliotecas como SQLAlchemy, Pandas e pymongo. Isso permite carregar, manipular e visualizar dados diretamente no notebook.

SQLAlchemy, por exemplo, é uma biblioteca Python que facilita a conexão com bases de dados SQL. Ela oferece uma interface de alto nível para criar, modificar e consultar bases de dados.

Bibliotecas como **pymongo** são utilizadas para bases de dados NoSQL, como MongoDB.

Pandas, uma das bibliotecas mais populares em data science, também oferece funcionalidades para ler e escrever dados de/para bases de dados SQL diretamente. Isso permite a manipulação e a análise de grandes conjuntos de dados de forma eficiente.

Construindo o **conceito**

Uso de widgets interativos

- ▶ São fornecidos pela biblioteca ipywidgets.
- ▶ Permitem criar controles interativos, como *sliders*, caixas de seleção e botões, e podem ser usados para criar dashboards interativos e ferramentas de visualização de dados.
- ▶ Tem capacidade de criar interfaces interativas diretamente no notebook, facilitando a experimentação e tornando a apresentação dos resultados mais envolvente e acessível.



Tome nota

Por exemplo, um *slider* pode ser usado para ajustar parâmetros de um modelo de machine learning em tempo real, enquanto gráficos atualizam automaticamente para refletir as mudanças.

Construindo
o **conceito**

Exportação e publicação de notebooks

- ▶ Os notebooks podem ser **exportados** para vários formatos como HTML, PDF e slides, permitindo a fácil publicação e o compartilhamento.
- ▶ Os notebooks podem ser **convertidos** para HTML, PDF, slides e outros formatos, usando a ferramenta nbconvert. Isso facilita a distribuição de análises, relatórios e apresentações.



Tome nota

Plataformas como GitHub e NBViewer permitem que notebooks sejam hospedados e visualizados online, facilitando o compartilhamento com colegas e colaboradores. Essa capacidade de exportar e publicar notebooks amplia o alcance do trabalho realizado no Jupyter Notebook, tornando-o acessível para uma audiência maior.

Construindo
o **conceito**

Exemplos práticos

1. Uso de células mágicas (magic commands)

Exemplo 1 – Medindo a performance de duas implementações diferentes de um algoritmo:

```
import numpy as np

# Primeira implementação
%timeit np.sum(np.arange(100000))

# Segunda implementação
%timeit sum(np.arange(100000))
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o conceito

Exemplos práticos

1. Uso de células mágicas (*magic commands*)

Exemplo 2 – Carregando a extensão de SQL e executando queries SQL:

```
# Instalar a extensão SQL
!pip install ipython-sql

# Carregar a extensão SQL
%load_ext sql

# Conectar a um banco de dados SQLite em memória e executar comandos SQL
%sql sqlite://

%%sql
CREATE TABLE exemplo (id INT, nome VARCHAR);
INSERT INTO exemplo VALUES (1, 'Ana');
INSERT INTO exemplo VALUES (2, 'João');
INSERT INTO exemplo VALUES (3, 'Maria');
SELECT * FROM exemplo;
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo
o **conceito**

Exemplos práticos

1. Uso de células mágicas (magic commands)

Exemplo 3 – Salvando uma célula de código em um arquivo Python:

```
%%writefile meu_script.py  
def saudacao(nome):  
    return f"Olá, {nome}!"  
  
print(saudacao("Mundo"))
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o **conceito**

Exemplos práticos

2. Visualização de dados com bibliotecas gráficas

Exemplo 1 – Criando um gráfico de barras com Matplotlib:

```
import matplotlib.pyplot as plt

meses = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun']
vendas = [1500, 1800, 1200, 1700, 1900, 2200]

plt.bar(meses, vendas)
plt.xlabel('Meses')
plt.ylabel('Vendas')
plt.title('Vendas Mensais')
plt.show()
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo
o **conceito**

Exemplos práticos

2. Visualização de dados com bibliotecas gráficas

Exemplo 2 – Criando um heatmap com Seaborn:

```
import seaborn as sns
import numpy as np

dados = np.random.rand(10, 12)
sns.heatmap(dados, annot=True)
plt.title('Heatmap de Correlação')
plt.show()
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o **conceito**

Exemplos práticos

2. Visualização de dados com bibliotecas gráficas

Exemplo 3 – Criando um gráfico de dispersão interativo com Plotly:

```
import plotly.express as px
import pandas as pd

df = pd.DataFrame({
    'x': [1, 2, 3, 4, 5],
    'y': [10, 11, 12, 13, 14],
    'tamanho': [100, 200, 300, 400, 500]
})

fig = px.scatter(df, x='x', y='y', size='tamanho', title='Gráfico de Dispersão Interativo')
fig.show()
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o **conceito**

Exemplos práticos

3. Integração com bases de dados

Exemplo 1 – Acessando uma API pública de dados meteorológicos:

```
import requests
import pandas as pd

# Adicionar um cabeçalho User-Agent à solicitação
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3'
}

# Fazer a solicitação GET com o cabeçalho User-Agent
resposta = requests.get('https://api.met.no/weatherapi/locationforecast/2.0/compact?lat=60.10&lon=9.58', headers=headers)

# Verificar se a resposta foi bem-sucedida
if resposta.status_code == 200:
    dados = resposta.json()
    previsoes = dados['properties']['timeseries']
    df_previsoes = pd.DataFrame(previsoes)
    display(df_previsoes.head())
else:
    print(f"Erro ao acessar a API: {resposta.status_code}")
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o **conceito**

Exemplos práticos

4. Uso de widgets interativos

Exemplo 2 – Criando um dashboard interativo:

```
import ipywidgets as widgets
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, 10, 100)
y = np.sin(x)

slider = widgets.FloatSlider(value=1, min=0.1, max=2, step=0.1, description='Amplitude:')

def update_plot(amplitude):
    plt.plot(x, amplitude * y)
    plt.xlabel('X')
    plt.ylabel('Y')
    plt.title(f'Seno de X com Amplitude {amplitude}')
    plt.show()

widgets.interactive(update_plot, amplitude=slider)
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo
o **conceito**

Exemplos práticos

5. Exportação e publicação de notebooks

Exemplo – Convertendo um notebook em slides para uma apresentação:

```
!jupyter nbconvert --to slides meu_notebook.ipynb
```

Substitua meu_notebook
pelo nome do seu notebook!

Fonte: JUPYTER, [s.d].
Elaborado especialmente para o curso.



Colocando
em **prática**

Exercício: utilizando o Jupyter



20 minutos



Em duplas



Código de programação

1 Crie um Jupyter Notebook que execute as seguintes tarefas:

2 Importar bibliotecas: importe as bibliotecas Numpy e Matplotlib.

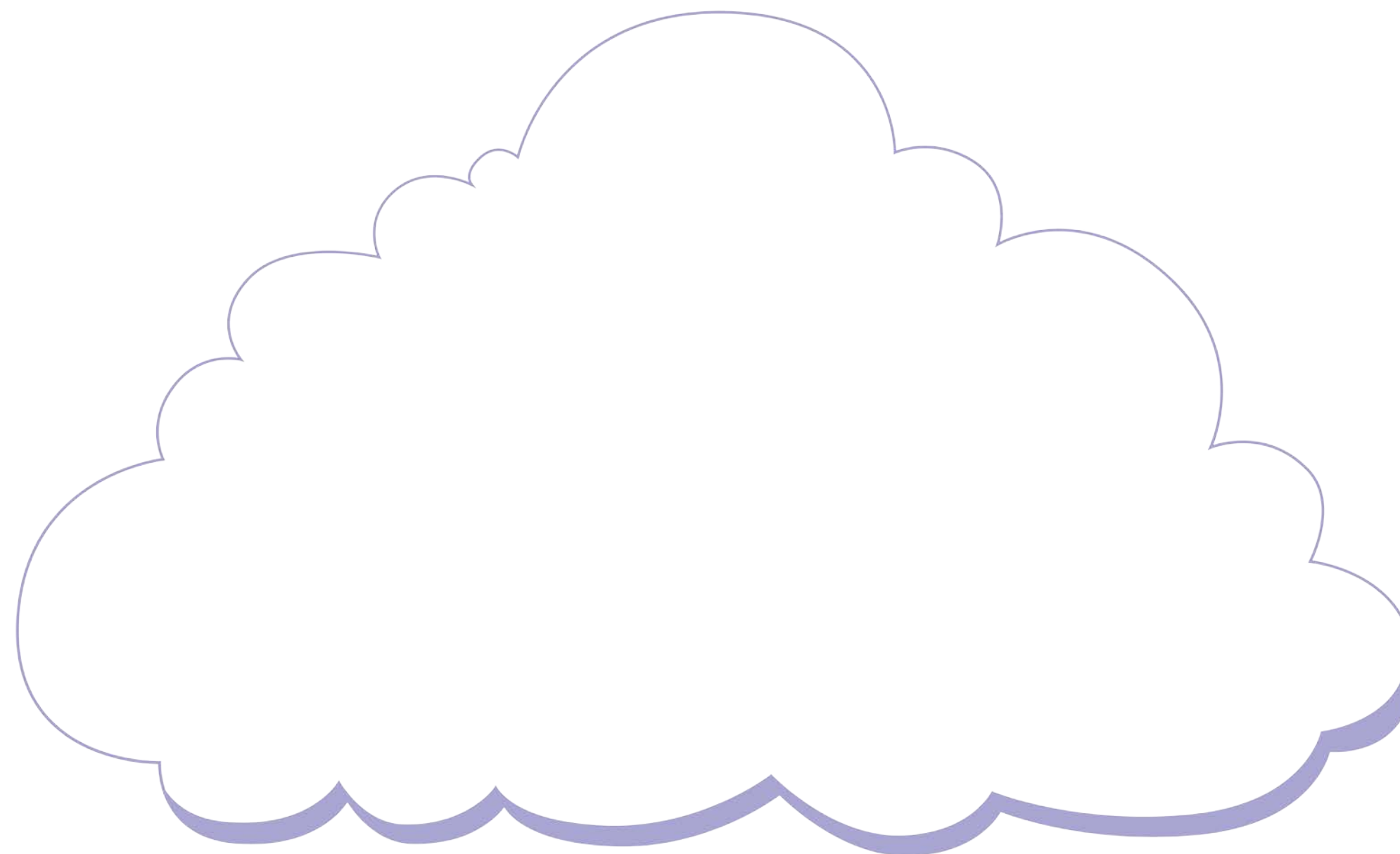
Plotar um gráfico de linhas: use Matplotlib para criar um gráfico de linhas desses números aleatórios.

3 Gerar dados aleatórios: gere um *array* de 100 números aleatórios entre 0 e 1 usando Numpy.

4 Adicionar título e rótulos: adicione um título ao gráfico e rótulos aos eixos X e Y.

5 Salvar o gráfico: salve o gráfico como uma imagem PNG no mesmo diretório do notebook.

Nuvem de palavras



© Getty Images

O que nós
**aprendemos
hoje?**



© Getty Images

O que nós
aprendemos
hoje?

Então ficamos assim...

- 1** Exploramos funcionalidades avançadas do Jupyter Notebook, incluindo células mágicas que simplificam tarefas comuns, como medir o tempo de execução de código e carregar módulos.
- 2** Discutimos visualização de dados com bibliotecas, como Matplotlib, Seaborn, Plotly e Bokeh, que permitem criar gráficos estáticos e interativos para análises detalhadas.
- 3** Abordamos a integração do Jupyter Notebook com bases de dados usando SQLAlchemy e a utilização de widgets interativos com ipywidgets para criar interfaces dinâmicas, além da exportação e publicação de notebooks em diversos formatos.

Saiba mais

Você sabia que a forma tradicional de escrever código pode ser um obstáculo para projetos de ciência de dados?

Descubra as vantagens dos notebooks, como o Jupyter, e como eles podem revolucionar seu código.

MIRANDA, J. V. Jupyter Notebook: Exemplos de Códigos e Como Usar. Alura, 16 ago. 2023. Disponível em: <https://www.alura.com.br/artigos/conhecendo-o-jupyter-notebook>. Acesso em: 24 jul. 2024.

Referências da aula

JUPYTER. *Página inicial*, [s.d.]. Disponível em: <https://jupyter.org>. Acesso em: 24 jul. 2024.

Identidade visual: imagens © Getty Images

**Educação
Profissional
Paulista**

Técnico em
**Ciência de
Dados**