TECNOLOGIAS SOCIAIS DIGITAIS · AULA 8

Hands-On: API da Câmara + Pandas

Acessando e analisando dados públicos na prática

AGENDA

30 min · conceitos + tour da API

50 min · hands-on no Colab (5 exercícios)

15 min · compartilhamento + dúvidas

5 min · fechamento

Na aula anterior...

Os 4 pontos que precisamos lembrar da aula anterior

1

Dados abertos são matéria-prima

Disponíveis em formato estruturado, com licença livre, prontos para reuso.

2

Governança ≠ e-Gov

Não basta digitalizar serviço. É preciso habilitar fiscalização e participação.

3

Pipeline tem 6 etapas

Coleta · Limpeza · Persistência · API · Interface · Manutenção.

4

Dado público ≠ dado sem risco

Qualidade, ética, LGPD, grupos vulneráveis — toda solução cívica precisa avaliar.

Hoje vamos sair do conceito e tocar no dado real, em Python, com a API da Câmara.

Objetivos da aula

Ao final, o grupo deve ter conseguido:

1. Fazer requisições HTTP a uma API REST pública (sem autenticação) e entender a resposta JSON.
2. Transformar JSON em pandas.DataFrame e aplicar filtros, agrupamentos e ordenações.
3. Lidar com paginação — entender quando uma API esconde dados em várias páginas.
4. Construir uma visualização simples (bar chart) que comunica um insight sobre dado público.
5. Reconhecer outliers e justificar quando vale (ou não vale) chamá-los de 'suspeitos'.

API ou download de CSV?

Cada um serve para uma coisa diferente

DOWNLOAD CSV

Foto do passado

- ✓ Funciona offline depois de baixar
- ✓ Bom para análises grandes e reprodutíveis
- ✓ Não depende de servidor estar no ar
- ✗ Pode estar desatualizado
- ✗ Sem filtro: baixa o universo inteiro
- ✗ Difícil automatizar atualizações

Use quando: análise pontual com dado fechado (ENEM 2023, Censo).

API REST

Filme em tempo real

- ✓ Dados sempre atualizados
- ✓ Filtros via query params (puxa só o que precisa)
- ✓ Fácil automatizar em produção
- ✗ Depende do servidor estar online
- ✗ Rate limit / paginação às vezes complica
- ✗ Schema pode mudar sem aviso

Use quando: dashboards vivos, alertas, integração com outros sistemas.

Anatomia de uma API REST

Os 4 componentes que você vê em qualquer endpoint

```
GET https://dadosabertos.camara.leg.br/api/v2/deputados?siglaUf=SP&itens=10
```

verbo · endereço do servidor · recurso · filtros

Verbo HTTP

GET para ler. POST/PUT/DELETE para escrever. Em APIs públicas, quase tudo é GET.

Endpoint

Caminho que identifica o recurso. /deputados, /partidos, /votacoes.

Query params

Filtros depois do '?'. Ex.: ?ano=2024&siglaUf=SP &itens=100

Resposta

Normalmente JSON com status code (200 = OK, 404 = não achou, 429 = devagar).

A API da Câmara dos Deputados

Por que escolhemos ela para a aula

`dadosabertos.camara.leg.br/api/v2`

Cobertura: legislaturas desde 1991 · deputados, votações, despesas (CEAP), partidos, frentes, eventos.

Zero auth

Não precisa de API key, cadastro ou OAuth. Você abre o link e funciona.

JSON limpo

Resposta consistente: { dados: [...], links: [...] }. Parse direto com .json().

Paginação

Use ?pagina=N&itens=K. Em alguns endpoints há cabeçalhos de navegação.

Sem rate strict

Funciona bem para uso pedagógico. Ainda assim, dê time.sleep() em loops.

Endpoints que vamos usar (e outros úteis)

Cada endpoint = uma pergunta possível

ENDPOINT	O QUE RETORNA	PERGUNTA EXEMPLO
`/deputados`	Deputados em exercício	Quantos deputados meu estado tem hoje?
`/deputados/{id}`	Perfil de um deputado	Qual a escolaridade da bancada de SP?
`/deputados/{id}/despesas`	Despesas CEAP (cota parlamentar)	Em que mais gastam? Quem é outlier?
`/deputados/{id}/discursos`	Discursos em plenário	Qual deputado fala mais sobre saúde?
`/votacoes`	Votações nominais	Como cada partido votou no PL X?
`/partidos`	Lista de partidos ativos	Quantos parlamentares por partido?
`/frentes`	Frentes parlamentares	Quem participa da frente Y?

Hoje vamos focar em /deputados e /deputados/{id}/despesas

Toolkit do dia

Três libs!

`requests`

Fazer requisições HTTP em uma linha. Já vem instalada no Colab.

```
import requests

r = requests.get(url,
    params={'uf': 'SP'})
data = r.json()
```

`pandas`

DataFrame é a estrutura central. Filtrar, agrupar, ordenar — tudo em uma linha.

```
import pandas as pd

df = pd.DataFrame(
    data['dados'])
df.groupby('partido').si
ze()
```

`matplotlib`

Visualizações simples. Para um pitch acadêmico, bar/line/scatter já bastam.

```
import matplotlib.pyplot
as plt

df.plot(kind='bar',
    x='cat', y='vlr')
plt.show()
```

O case de hoje: gastos da CEAP

Uma versão miniatura do que a Rosie faz

CEAP

Cota para o Exercício da Atividade Parlamentar

Verba pública que cada deputado usa para custear o mandato: combustível, passagens, escritório, telefone, divulgação, alimentação em viagem etc. Cada reembolso vira uma linha pública na API.

AS PERGUNTAS DE HOJE

1. Quantos deputados meu estado tem?
2. Quais as categorias de gasto mais comuns?
3. Qual teve o maior gasto em 2024?
4. Há reembolsos anômalos? Quantos?
5. Como visualizar de forma honesta?

LEMBRA DA AULA PASSADA?

Operação Serenata de Amor

A Rosie, robô do projeto, analisa exatamente esse mesmo dado da CEAP. Em escala: 1,6 milhão de reembolsos por hora, 17.700+ suspeitas em 5 anos.

Hoje vocês vão construir uma versão minúscula disso — em ~50 minutos, com Python + Pandas, sem framework, sem servidor.

O notebook tem 3 partes

Abra no Colab e siga a ordem

1

Setup + Demo

Você executa as células prontas e entende como uma requisição funciona. Sem precisar codar ainda — só ler e rodar (Shift + Enter).

2

Exercícios com TODO

5 exercícios. Em cada um, você completa uma função. A célula seguinte tem a correção.

3

Desafio + investigação livre

Z-score para detectar outliers, e uma seção aberta para a dupla testar perguntas próprias.

Os 5 exercícios

Progressão do mais simples ao mais analítico

1

Deputados por UF

OBJETIVO

Filtrar /deputados pelo estado.

VALIDA

Sua função retorna lista, tamanho razoável, todos da UF correta.

2

Despesas + paginação

OBJETIVO

Implementar loop de páginas até esgotar.

VALIDA

DataFrame não-vazio, colunas esperadas, ano filtrado correto.

3

Top categorias de gasto

OBJETIVO

groupby + sum + sort.

VALIDA

Shape correto, ordenado descendente, soma bate com o total.

4

Bar chart honesto

OBJETIVO

matplotlib com título, eixo, legenda.

VALIDA

Plot tem título, label de eixos e ao menos 5 barras.

5

Detector de outliers

OBJETIVO

z-score > 3 sobre os reembolsos.

VALIDA

Função retorna DataFrame com z-score; alguns identificados.

Três cuidados para hoje

Antes de sair atirando requests

1 Rate limit e gentileza

Em loops longos, dê time.sleep(0.2). A API é pública mas tem limite. Em produção, cache local.

2 Reprodutibilidade

Fixe ano=2024 nos exercícios. Dados de anos abertos mudam diariamente — seu colega não terá o mesmo resultado.

3 Outlier ≠ irregularidade

Z-score alto significa gasto incomum, não ilegal. Usamos para levantar hipóteses, não acusar. Cuidado com o vocabulário ao apresentar.

FECHAMENTO

Vocês acabaram de fazer civic tech.

Em 50 minutos, com uma stack mínima, vocês acessaram dado público real, analisaram, encontraram padrões e visualizaram. Isso é o que os times do g0v, do Code for America, da Open Knowledge Brasil fazem todos os dias.

ONDE BUSCAR AJUDA

Docs oficiais · dadosabertos.camara.leg.br/swagger/api.html

Pandas docs · pandas.pydata.org/docs

Stack Overflow / ChatGPT · cite a fonte no relatório se usar IA

Professor · até a próxima aula