DADOS, PYTHON E ESTATÍSTICA: CAMINHOS PARA STEM

Ana Júlia Amaro Pereira Rocha

SEMANA 2

DIA 2

RECAPITULANDO...

Na aula passada, analisamos dados no papel. Identificamos erros, dados faltantes e inconsistências. Hoje vamos fazer isso usando Python.

Aprender a limpar dados no papel primeiro foi importante pois, embora o computador execute comandos rapidamente e seja mais eficiente que um humano nessa tarefa, quem decide o que ele vai fazer é uma pessoa. Então, pensar antes de programar evita erros e manipulação de dados.

DADOS NO COMPUTADOR

Importar dados significa ler arquivos no Python. Um formato muito comum é o CSV, cada linha representa uma observação e cada coluna uma variável. Antes de qualquer limpeza, precisamos olhar os dados. Quantas linhas existem? Que tipos de informação aparecem em cada coluna?

Os mesmos problemas do papel aparecem no computador, a diferença é que agora podemos tratar eles com código. Como já vimos, limpar dados não é apagar o que não gostamos, é tornar os dados utilizáveis e coerentes justificando cada decisão a respeito deles. **Quando limpamos dados em Python, criamos regras que são aplicadas a todo o conjunto.**

GITHUB

Vamos discutir alguns conceitos importantes do GitHub hoje também. Já sabemos que trata-se de uma plataforma para guardar e compartilhar código e nela podemos acompanhar mudanças ao longo do tempo nos projetos. Agora vamos a alguns conceitos:

branch: Um branch é uma linha paralela de desenvolvimento. Ele permite testar mudanças sem afetar o código principal. É como fazer rascunhos sem apagar o original. Usamos branches para evitar bagunça no projeto principal. Cada pessoa pode trabalhar em uma branch e depois as mudanças podem ser revisadas.

GITHUB

Pull Request: um pedido para juntar mudanças. Ele permite revisar código antes de aceitar e é muito comum em projetos reais.

Fork: uma cópia de um repositório de outra pessoa. Ele permite contribuir sem alterar o original diretamente e é muito usado em projetos abertos.

Conflitos de merge: Um conflito acontece quando duas mudanças afetam o mesmo trecho. O Git não sabe qual escolher, então cabe à pessoa decidir qual versão manter. Acontece bastante quando se trabalha em equipe. O fluxo típico de projetos reais é **criar branch → alterar código → commit → pull request. Revisar → resolver conflitos → merge.**