Grupo 2: Pythonautas

Isabela, Alice, Gabriela Lopes, Rafael Silva, Mychelle Ketlen, William Lopes

Visão Geral

Este documento apresenta uma análise crítica e detalhada do projeto submetido no módulo 1 da formação da Globo. O objetivo é avaliar a implementação do sistema de análise de interações com conteúdos, considerando boas práticas de programação, clareza lógica, uso adequado dos recursos da linguagem Python e cumprimento dos requisitos propostos no enunciado.

2. Organização do Código e Estrutura Modular

A equipe estruturou o projeto em múltiplos arquivos, promovendo uma separação lógica clara entre as etapas do pipeline: leitura, limpeza, estruturação, cálculo de métricas e salvamento. Esta modularização é uma boa prática, facilitando a manutenção, testes e reuso de código.

3. Análise das Etapas do Pipeline

3.1 Leitura dos Dados

A função `carregar_dados_de_arquivo_csv` trata corretamente a leitura do CSV e implementa um mecanismo para tratar vírgulas dentro dos comentários, o que demonstra atenção a problemas comuns de parsing. A verificação de integridade e o feedback por exceções são positivos para depuração e robustez.

3.2 Limpeza dos Dados

A equipe implementou uma sequência bem estruturada de funções para remover espaços, tratar valores nulos e converter campos numéricos. O pipeline de limpeza considera situações de erro e converte campos estratégicos para inteiros, permitindo análises estatísticas. Essa etapa mostra domínio técnico e entendimento dos dados.

3.3 Estruturação dos Dados

A função `agrupar_por_conteudo` organiza as interações por `id_conteudo` de maneira eficiente. A estruturação facilita a agregação e visualização posterior, sendo fundamental para as métricas implementadas. A lógica é clara e bem executada.

3.4 Cálculo das Métricas

As métricas foram todas implementadas conforme o solicitado: total de interações por conteúdo, contagem por tipo, tempo total e médio de visualização, listagem de comentários e top 5 por tempo. A separação em funções específicas para cada métrica favorece a reutilização e testes unitários. Além disso, a conversão de tempo para o formato HH:MM:SS é um refinamento que agrega valor à apresentação dos dados.

3.5 Interface Interativa

O `main.py` traz um menu bem estruturado, que cobre todas as métricas e funcionalidades exigidas, incluindo salvamento em CSV. A utilização de emojis melhora a legibilidade no terminal e torna a interação mais agradável. O uso de `try-except` no tratamento de entrada do usuário é adequado e evita falhas na execução.

4. Comparação com a Solução Exemplo

Comparando com o exemplo `analise_globo_fase1.py`, a solução da equipe adota uma abordagem mais distribuída e modularizada, o que é positivo. Ambas abordagens são válidas, mas a da equipe se destaca por permitir crescimento do projeto em etapas futuras, como testes, novas visualizações e integrações com APIs.

5. Pontos de Melhoria e Recomendações

- Incluir testes unitários automatizados para funções críticas (ex.: limpeza, métricas).

- Criar mensagens de log (em vez de apenas `print`) para controle mais profissional dos processos.

- Adotar type hints para funções principais, facilitando a leitura e uso por outros programadores.

6. Conclusão

Vocês apresentaram um trabalho bem estruturado, funcional e tecnicamente sólido. O código revela domínio dos conceitos de listas, dicionários, funções, tratamento de erros e organização modular. A clareza e qualidade da lógica demonstram excelente assimilação dos conteúdos trabalhados no módulo.

Parabens!!!!