Documentação desafio data engineering codeesh

O desafio de engenheiro de dados da Coodesh tem como objetivo criar um processo de ETL que migra um banco de dados legado para um Data Lake (S3 no caso do teste).

1. Configuração do Ambiente

Primeiramente, é necessário criar um ambiente Python utilizando conda, que acompanhará todas as lógicas do desafio. A versão recomendada é o Python 3.8.

Comandos para criar e ativar o ambiente:

Comandos:
Gerando env que acompanhará todas as lógicas.

> conda create -n env-teste python==3.8

> conda activate env_teste

> pip install -r requirements.txt

2. Gerando o Banco de Dados Legado

O script sqlite_data_generator.py é responsável por gerar um banco de dados SQLite (coodesh-teste.db), contendo uma tabela de vendas com 500 registros, que são gerados por uma lógica randomizada.

Comando para gerar o banco de dados:

> python3 scripts/sqlite_data_generator.py

3. Processo de ETL (Migração Completa)

Com o banco de dados legado gerado, é possível rodar o processo de ETL que migra esses dados para o Data Lake (S3). A migração completa pode ser realizada com o script get_total_sales_full_migration.py.

Comando para realizar a migração completa:

> python3 scripts/get_total_sales_full_migration.py

Estratégias de Migração Incremental:

O script atual realiza apenas migração completa. Para implementar migração incremental, existem duas abordagens possíveis:

1. **Tabela de Metadados**: Criar uma tabela que armazena o último ponto de migração e usar essa informação para migrar apenas os dados a partir daquele ponto.
2. **Tarefa Agendada**: Implementar uma rotina que busque apenas os dados dos últimos n dias.

Ambas as abordagens utilizam as funções já implementadas na pasta etls/. O que mudaria seria passar os parâmetros begin e end na função extract_vendas. Esse encapsulamento segue os princípios da metodologia SOLID, tornando o código mais reutilizável e robusto.

4. Criação do Modelo de Machine Learning

Além do processo de ETL, o desafio inclui a criação de um modelo de aprendizado de máquina para prever a quantidade de vendas diárias nos meses futuros. O script generating_model.py é responsável por gerar um modelo de árvore de decisão XGBoost, em formato pickle, que pode ser utilizado para previsões.

Passos realizados no script `generating_model.py`:

- Limpeza e pré-processamento dos dados
- Feature engineering
- Separação dos dados para treino e teste
- Treinamento do modelo
- Criação do modelo para ser produtizado

O modelo gerado pode ser implantado em serviços como Amazon SageMaker.

Comando para criar o modelo de Machine Learning:

> python3 scripts/generating_model.py

Criação do modelo de Machine Learning:

O script generating_model.py gera um modelo de árvore de decisão Xgboost em formato pickle que pode ser usado para prever a quantidade de vendas diária dos meses futuros. Nesse script fazemos todo o processo de limpeza, pré-processamento, feature engineering, separação de dados de treino e teste, treinamento do modelo e criação de modelo para ser produtizado (esse pode ser produtizado no Amazon SageMaker).

Criar modelo de ML:
python3 script/generating_model.py

Resumo:

- ETL: O script get_total_sales_full_migration.py realiza uma migração completa do banco de dados legado para o Data Lake (S3). Para migração incremental, sugere-se a criação de uma tabela de metadados ou a implementação de tarefas agendadas.
- ML: O script generating_model.py treina e gera um modelo XGBoost que pode ser usado para prever vendas futuras, sendo possível sua implantação no Amazon SageMaker.