O que já foi feito:

1. Análise e Mapeamento de Dados na Origem

- **Levantamento Inicial:** O primeiro passo do projeto consistiu em um estudo aprofundado do banco de dados de produção para identificar e mapear todas as variáveis relevantes para a medição do engajamento dos usuários.
- **Definição de Métricas:** Esta análise permitiu definir quais interações, exercícios e outros eventos seriam utilizados como base para a engenharia de features do modelo de classificação.

2. Prototipagem e Validação de Modelos

- **Ambiente de Testes:** Utilizando os dados já disponíveis no dataset `analytics` do BigQuery, foi montado um ambiente de experimentação.
- **Exploração com Jupyter Notebooks:** A primeira fase da prototipagem usou notebooks Python para uma exploração interativa dos dados, permitindo maior flexibilidade na manipulação e visualização.
- **Validação com BigQuery ML:** Em seguida, os modelos de clusterização (K-Means) foram implementados e testados diretamente na plataforma com o BigQuery ML, validando a viabilidade técnica da abordagem.
- **Identificação de Custos:** Esta fase foi crucial para identificar que o processo iterativo de prototipagem, com múltiplas consultas em grandes volumes, gerava altos custos de processamento, justificando a necessidade de um pipeline otimizado.

3. Desenvolvimento do Pipeline de ML Automatizado

- **Arquitetura em SQL:** Como solução para os altos custos e para a operacionalização do modelo, foi projetado e implementado um pipeline de ponta a ponta, escrito inteiramente em SQL no BigQuery.
- **Estrutura Modular:** O pipeline foi dividido em schemas com responsabilidades claras:
    1. **`analytics`**: Contém as tabelas de dados processados, como o `cubo_engajamento`, que serve de entrada para o modelo.
    2. **`config`**: Abriga toda a lógica de execução e parametrização.
- **Automação com Stored Procedures:** Foram criados procedimentos armazenados para cada etapa do processo:
    1. **Treinamento do Modelo:** Executa o `CREATE MODEL` para o K-Means.
    2. **Geração de Labels:** Interpreta os clusters e atribui nomes significativos.
    3. **Classificação:** Aplica o modelo treinado para classificar os usuários.
- **Orquestração e Parametrização:** Um procedimento mestre foi desenvolvido para orquestrar a execução de todas as etapas em sequência, e tabelas de configuração foram criadas para permitir o ajuste de parâmetros sem alterar o código.

4. Documentação e Planejamento Estratégico

- **Scripts de Referência (`Ref`):** Foram criados e documentados os scripts SQL utilizados para extrair as métricas brutas da base do Redu, garantindo a rastreabilidade do processo de ETL.
- **Arquitetura de Dados Futura (`DW`):** Foi elaborada uma documentação estratégica contendo um esquema EER (Diagrama Entidade-Relacionamento Estendido) como proposta para a evolução do Data Warehouse e um roteiro com sugestões de próximos passos para o projeto.

Ideias e Sugestões para o EmpatIA

- **1. Pipeline de Ingestão de Dados (Opções de Arquitetura):**
  - **O quê (Opção A - Centralizado):** Implementar um pipeline de dados que centralize as informações dos vários bancos e servidores dos clientes em um único Data Warehouse. As cargas podem ser semanais (em horários de pouco fluxo) usando uma ferramenta como o **Airbyte**, com um processo de ETL ou ELT.
  - **Por quê:** Para unificar todos os dados em uma única fonte da verdade, simplificando a análise global e a manutenção.
  - **O quê (Opção B - DW por Cliente):** Como alternativa, criar um Data Warehouse local para cada cliente. Cada DW seria independente, mas todos seguiriam um esquema conceitual em comum para padronização (similar à arquitetura usada no Redu).
  - **Por quê:** Para garantir um maior isolamento e segurança dos dados de cada cliente, potencialmente simplificando a lógica de ingestão individual.
- **2. Estrutura do Data Warehouse:**
  - **O quê:** A modelagem do DW deve ser estruturada com duas tabelas fato principais: uma para **interações** (fonte: `statuses`) e outra para **assignments** (fonte: `user_assignments`). *Esta estrutura deve ser o padrão aplicado, seja no DW centralizado (Opção A) ou em cada DW individual (Opção B).*
  - **Por quê:** Para criar um modelo de dados eficiente e padronizado para monitorar o engajamento dos alunos, independentemente da arquitetura de ingestão escolhida.
- **3. Criação e Armazenamento do `cubo_engajamento`:**
  - **O quê:** A partir das tabelas fato, criar o `cubo_engajamento` (via view ou function). O destino final deste cubo, após ser classificado, seria uma estrutura à parte (ex: um schema dedicado) dentro do mesmo banco de dados do DW correspondente.
  - **Por quê:** Para centralizar a arquitetura, simplificar a infraestrutura, reduzir custos e garantir acesso performático aos dados de engajamento já processados.
- **4. Evolução com Inteligência Artificial (Modelos de Classificação):**

    - **O quê (Abordagem de Modelagem):** Para classificar os dados do cubo, treinar múltiplos modelos (um para cada nível hierárquico x agregação de tempo) ou desenvolver um *ensemble* de classificadores.
    - **Por quê:** Para garantir que a classificação do engajamento seja precisa e adaptada às diferentes granularidades de análise.
    - **O quê (Opção - Não Supervisionado):** Implementar um algoritmo de clusterização (K-Means ou similar) para descobrir segmentos e perfis de usuários de forma automática.
    - **Por quê:** Para revelar padrões de comportamento inesperados sem depender de rótulos pré-existentes.
- **5. Repositório de Referência**
    - Além disso, todas as informações do que foi produzido e possíveis mudanças, incluindo esse documento, encontram-se no seguinte repositório
    - https://github.com/matheussilvaviitra/Classificador_engajamento.git