Exercício 2

Suponha que você tenha uma base de dados contendo textos jurídicos, como decisões judiciais, petições e documentos legais. A base de dados inclui informações sobre o conteúdo do texto, data, jurisdição e outras informações relevantes. Seu objetivo é criar um sistema de recomendação que sugira textos jurídicos semelhantes a um texto de referência.

Para todos os itens: Informe as bibliotecas usadas, se necessário, o motivo de cada decisão, explore as possibilidades.

1. Descreva como você desenvolveria o sistema de recomendação que recebe um texto de referência e sugere os textos mais semelhantes a ele na base de dados.

Para desenvolver o sistema de recomendação de textos jurídicos podemos seguir alguns passos, iniciando Pré-processamento dos dados, Representação dos textos, Cálculo de Similaridade e a criação do Sistema de Recomendação em si, integrando os passos anteriores. Cada etapa é descrita abaixo:

1. **Pré-processamento dos Dados**:
   * **Bibliotecas**: nltk, spacy, scikit-learn, pandas.
   * **Passos**:
     + **Tokenização**: Quebrar os textos em palavras ou frases.
     + **Remoção de Stopwords**: Eliminar palavras comuns que não agregam significado (ex.: "o", "e", "a").
     + **Stemming/Lematização**: Reduzir as palavras às suas raízes.
     + **Normalização**: Transformar todos os textos para letras minúsculas, remover pontuações, etc.
   * **Objetivo**: Facilitar a análise textual ao reduzir a complexidade e padronizar o formato.
2. **Representação dos Textos**:
   * **Bibliotecas**: scikit-learn, gensim, transformers.
   * **Técnicas**:
     + **TF-IDF (Term Frequency-Inverse Document Frequency)**: Quantifica a importância de uma palavra em um documento em relação ao corpus inteiro.
     + **Embeddings (ex.: Word2Vec, BERT)**: Representa cada palavra como um vetor denso de alta dimensão.
     + **Bag of Words (BoW)**: Cria uma representação dos textos baseada na contagem de palavras.
   * **Objetivo**: Transformar o texto em uma forma numérica que pode ser usada para cálculos de similaridade.
3. **Cálculo de Similaridade**:
   * **Bibliotecas**: scikit-learn, scipy.
   * **Métodos**:
     + **Cosseno de Similaridade**: Mede o ângulo entre os vetores de dois textos. Usado para TF-IDF ou embeddings.
     + **Distância Euclidiana**: Mede a diferença entre dois vetores.
     + **Jaccard**: Mede a similaridade entre dois conjuntos de palavras.
   * **Objetivo:** Encontrar o grau de similaridade entre o texto de referência e os textos na base de dados.
4. **Sistema de Recomendação**:
   * **Pipeline**: Crie um pipeline que inclua o pré-processamento, a transformação para vetores, e o cálculo da similaridade.
   * **Retorno**: Para um texto de referência, retorne os textos com maior similaridade calculada.
   * **Interface de Usuário**: Implementar uma interface simples (ex.: via Flask ou Streamlit) para entrada do texto de referência e exibição das recomendações.
5. Como você avaliaria esse sistema de recomendação?

Para avaliarmos o Sistema de Recomendação, podemos aplicar as métricas e métodos descritos abaixo:

1. **Métricas de Avaliação**:
   * **Precisión@k**: Mede a precisão entre os k textos recomendados.
   * **Recall@k**: Mede a proporção de documentos relevantes recuperados nos k textos recomendados.
   * **F1 Score**: Combina precisão e recall.
   * **MAP (Mean Average Precision)**: Média das precisões em diferentes níveis de recall.
   * **NDCG (Normalized Discounted Cumulative Gain)**: Avalia a qualidade da ordenação das recomendações.
2. **Método de Avaliação**:
   * **Validação Cruzada**: Dividir os dados em partes e testar o modelo em diferentes combinações.
   * **Base de Dados Anotada**: Caso tenha acesso a dados anotados com relevância, use-os para validar as recomendações.
   * **Avaliação Humana**: Especialistas jurídicos podem validar a relevância das recomendações.
3. Suponha que novos textos jurídicos sejam adicionados diariamente. Como você manteria o sistema de recomendação atualizado e garantiria que ele continue a fornece recomendações relevantes?

Para manter o Sistema de Recomendação atualizado e garantir que ele continue fornecendo recomendações relevantes com a adição diária de novos textos, podemos aplicar as seguintes abordagens:

1. **Atualização Contínua dos Dados**:
   * **Pipeline de Ingestão de Dados**: Criar um pipeline automatizado para integrar novos textos ao banco de dados existente. Incluir o pré-processamento dos novos dados e a inclusão no conjunto de dados de treinamento.
   * **Bibliotecas**: Airflow
2. **Incorporação de Novos Textos**:
   * **Re-treino Incremental**: Defina uma frequência para re-treinamento do modelo, como semanal ou mensal. Atualizar os modelos de embeddings ou recalcular TF-IDF periodicamente para incluir novos textos.
   * **Batch Processing**: Adicionar novos textos em lote, por exemplo, diariamente.
3. **Monitoramento de Desempenho**:
   * **Bibliotecas**: mlflow, prometheus.
   * **Monitoramento**: Acompanhe métricas de desempenho ao longo do tempo para identificar quedas na qualidade.
   * **Feedback Loop**: Incorporar feedback dos usuários para ajustar as recomendações.
4. **Ajustes de Modelagem**:
   * **Fine-Tuning**: Ajustar os parâmetros dos modelos conforme novos dados são incorporados.
   * **Atualização do Pipeline**: Revisar e otimizar as etapas do pipeline conforme necessário.