Relatório do Projeto: Sistema de Busca de Voos (Passagens Aéreas)

1. Introdução

Este relatório descreve o desenvolvimento de um sistema de busca de voos que utiliza técnicas de Recuperação de Informação (RI) para indexar e recuperar dados de passagens aéreas. O sistema foi desenvolvido como parte de um projeto acadêmico e implementa um modelo vetorial com ponderação TF-IDF para classificar os resultados de busca.

2. Interface com o Usuário

O sistema foi desenvolvido como uma aplicação web utilizando o framework Flask (Python) e possui as seguintes características:

- **Interface gráfica intuitiva** com campos de busca para origem, destino, companhia aérea, preço e número de escalas
- **Responsividade** para diferentes tamanhos de tela
- **Visualização detalhada** dos resultados de busca
- **Estatísticas avançadas** para cada rota específica
- **Filtros dinâmicos** para refinar os resultados

A interface foi desenvolvida utilizando:

- Bootstrap 5 para layout e componentes UI
- Font Awesome para ícones
- JavaScript para interatividade

3. Descrição da Solução Final

3.1. Arquitetura do Sistema

O sistema é composto por três componentes principais:

1. **Pré-processamento e Indexação**: Processa os dados brutos de voos e constrói um índice invertido por campo
2. **Módulo de Busca**: O módulo buscador.py implementa em python a lógica central do sistema de recuperação de voos, utilizando o **Modelo Vetorial** com ponderação **TF-IDF**.
3. **Interface Web**: Permite a interação do usuário com o sistema

3.2. Modelo Vetorial e TF-IDF

O sistema implementa o Modelo Vetorial, onde:

- Cada documento (rota aérea) e consulta é representado como um vetor no espaço vetorial
- Os termos são ponderados usando TF-IDF (Term Frequency-Inverse Document Frequency)

3.3. Implementação do Ranking

O ranking dos resultados é calculado considerando:

1. **Pesos por campo**: Diferentes campos (origem, destino, companhia, etc.) têm pesos distintos
2. **Normalização TF**: Frequência de termos normalizada pelo máximo no documento
3. **IDF suavizado**: Para evitar divisão por zero

```
# Cálculo do score no código
peso_campo = self.pesos.get(campo, 1.0)
df = len(postings) idf = math.log((self.doc_count + 1) / (df + 1)) + 1 # IDF com suavização
pontuacoes[doc_id] += (tf / max_tf) * idf * peso_campo
```

3.4. Melhorias Implementadas

1. **Pré-processamento avançado**:
    a. Stemming com RSLPStemmer para termos em português
    b. Remoção de stopwords
    c. Normalização de texto (minúsculas, remoção de caracteres especiais)
2. **Estatísticas avançadas**:
    a. Cálculo de custo-tempo (preço por minuto de voo)
    b. Identificação de voos mais rápidos e mais baratos
    c. Análise por companhia aérea
    d. Identificação de Voos sem escalas
3. **Interface aprimorada**:
    a. Filtros dinâmicos em tempo real
    b. Visualização de dados intuitiva
    c. Detalhamento de resultados

4. Principais Desafios e Soluções

Desafio 1: Lidar com dados heterogêneos (texto, códigos, preços, horários)

- **Solução**: Implementação de campos específicos no índice invertido com tratamento diferenciado para cada tipo de dado

Desafio 2: Melhorar a precisão das buscas por localidade (cidade vs código aeroporto)

- **Solução**: Busca por campos separados (origem/destino como texto e código) com pesos diferentes

Desafio 3: Normalização de valores numéricos (preços, durações) para cálculo de similaridade

- **Solução**: Categorização de faixas de preço e conversão de durações para minutos

Desafio 4: Performance em coleções grandes

- **Solução**: Utilização de estruturas de dados eficientes (dicionários) e pré-computação de valores

5. Bibliotecas Utilizadas

- **Flask**: Framework web para construção da interface
- **NLTK**: Para pré-processamento de texto (tokenização, stemming, stopwords)
- **Bootstrap**: Para interface gráfica responsiva
- **JSON**: Para armazenamento e manipulação dos índices

6. Conclusão

O sistema desenvolvido implementa com sucesso os conceitos de Recuperação de Informação estudados, particularmente o Modelo Vetorial com ponderação TF-IDF. As principais vantagens da solução incluem:

- Recuperação eficiente de voos relevantes
- Interface intuitiva para o usuário final
- Análises estatísticas úteis para tomada de decisão