Relatório

Tudo entendimento do Algoritmo foi realizado utilizando o chatGPT e o endereço: https://en.wikipedia.org/wiki/Okapi_BM25

Através da pesquisa: “Algoritmo BM25”

Resposta:
O algoritmo BM25 (Best Matching 25) é um método de ranking utilizado em sistemas de busca de texto, como motores de busca na web. O BM25 utiliza o cálculo da probabilidade de relevância de um documento para um conjunto de termos de busca especificados pelo usuário.
O algoritmo BM25 foi desenvolvido para melhorar a eficácia do algoritmo BM11. Ele usa uma fórmula mais sofisticada para calcular a relevância do documento em relação aos termos de pesquisa. A fórmula inclui três principais elementos: a frequência do termo na consulta, a frequência do termo no documento e o comprimento do documento.

A fórmula geral do BM25 é a seguinte:

score(D,Q) = ∑i=1..n IDF(qi) * ((f(qi, D) * (k1 + 1)) / (f(qi, D) + k1 * (1 - b + b * (|D| / avgdl))))

Onde:

D: é o documento sendo avaliado.
Q: é a consulta (conjunto de termos de busca).
n: é o número de termos na consulta.
IDF(qi): é o inverso da frequência do termo qi em todo o corpus de documentos.
f(qi, D): é a frequência do termo qi no documento D.
k1 e b: são parâmetros que ajustam a contribuição dos termos de busca e do comprimento do documento. k1 e *b* são parâmetros livres, geralmente escolhidos, na ausência de uma otimização avançada, como k1 ∈ [1.2,2.0] e *b*=0,75.
|D|: é o comprimento do documento em termos de palavras.
avgdl: é a média do comprimento de todos os documentos do corpus.

O BM25 é amplamente utilizado em sistemas de busca na web, pois é eficiente e eficaz em lidar com grandes conjuntos de documentos. Ele permite que os resultados da pesquisa sejam classificados de acordo com a relevância, ajudando os usuários a encontrar informações relevantes de forma rápida e fácil.

A implementação no algoritmo

Através da pesquisa: "Building a Simple Information Retrieval System using BM25"

Resposta:

```python
import math

class BM25:
    def __init__(self, corpus):
        self.corpus_size = len(corpus)
        self.avgdl = sum([len(doc) for doc in corpus]) / self.corpus_size
        self.corpus = corpus
        self.f = []
        self.df = {}
        self.idf = {}
        self.k1 = 1.5
        self.b = 0.75
        self.epsilon = 0.25
        self.build()

    def build(self):
        for document in self.corpus:
            frequencies = {}
            for word in document:
                if word not in frequencies:
                    frequencies[word] = 0
                frequencies[word] += 1
            self.f.append(frequencies)
            for word, freq in frequencies.items():
```

Algumas das etapas de construção de algoritmo de recuperação de informações (IR) não foram realizadas: Como por exemplo na parte de pré-processar os dados: remover stop words, stemming e etc.

Como utilizar

Copiar o arquivo "Exercício Aluno Especial.ipynb" para Google Colab Notebook;

1. Copiar o arquivo "cisi.all" disponibilizado no endereço http://ir.dcs.gla.ac.uk/resources/test_collections/cisi/ e importar no Google Colab Notebook do tópico anterior;
2. Executar as células de códigos;
    a. Leitura do arquivo CISI e transformando em um DataFrame;
    b. Implementação BM25 com ajuda do chatGPT;
    c. Busca de documentos relevantes;