

## Terceiro trabalho de Organização e Recuperação da Informação 2020-01

### Descrição

Este trabalho consiste na implementação de cálculo da ponderação TF-IDF, usando a fórmula dos exemplos em aula (usando logaritmo na base 10);

$$TF-IDF_{ij} = (1 + \log f_{ij}) \log \left( \frac{N}{n_i} \right), \text{ se } f_{ij} > 0$$

$$TF-IDF_{ij} = 0, \text{ se } f_{ij} = 0$$

Deve ser entregue apenas um programa desenvolvido em Python 3 que realize a tarefa descrita. O programa deve usar apenas as bibliotecas padrão Python 3, isto é, as bibliotecas que já vem com a instalação padrão do interpretador da linguagem, com exceção da biblioteca nltk, que deve ser utilizada para remoção de *stopwords* e extração de radicais.

O trabalho deve ser feito de modo individual e o código gerado deve ser entregue por e-mail (através de arquivo anexado) ao professor (wendelmelo@ufu.br) até a data 13/05/2021.

**Aviso importante:** se for detectado cópia ou qualquer tipo de trapaça entre diferentes grupos, todos os grupos serão punidos com a nota zero. Portanto, pense bem antes de pedir para copiar o trabalho do seu coleguinha, pois ele poderá ser punido também!

É obrigatório o uso do pacote nltk para a extração de radicais dos termos do vocabulário e obtenção de uma lista válida de *stopwords*. Os detalhes sobre a ponderação e o modelo são descritos a seguir. **É importante ler com atenção e seguir todos os detalhes da especificação sob pena de perda de pontos na nota do trabalho!**

### As *stopwords*

Seu programação deve remover stopwords conforme especificado no trabalho 1.

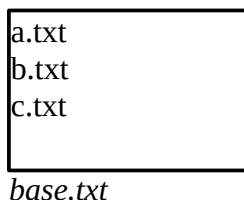
### A entrada do programa

Seu programa deverá receber um argumentos como entrada **pela linha de comando** especificando o caminho de um arquivo texto que contém os caminhos de todos os arquivos que compõem a base, cada um em uma linha (conforme trabalhos anteriores).

**Exemplo:** Vamos supor que nossa base é composta pelos arquivos *a.txt*, *b.txt* e *c.txt*. Vamos supor também que nosso programa se chama *tfidf.py*. Assim, chamaríamos nosso programa pela linha de comando fazendo:

```
> python tfidf.py base.txt
```

onde o arquivo *base.txt* contém os caminhos para os arquivos que compõe a base de documentos, conforme a seguir:



## A saída do programa

O programa deverá gerar um arquivo de saída, com nome e conteúdo exatamente como a seguir:

- *pesos.txt* : arquivo que contém ponderação TF-IDF de cada documento

### O arquivo pesos.txt

O programa deve gerar um arquivo texto chamado *pesos.txt* que contém os pesos de cada termo em cada documento segundo a ponderação TF-IDF. Cada linha desse arquivo deve conter os pesos não nulos dos termos de um dos documentos da base. Por exemplo, considere o exemplo visto em aula, onde supomos que W, X e Y são os termos do nosso vocabulário, já após a remoção de *stopwords* e extração de radicais.

A base de documentos possui o seguinte conteúdo:

Documento	Conteúdo
doc1	W W W X
doc2	W W Y
doc3	W W
doc4	X X

Conforme calculado em aula, os vetores de pesos TF-IDF dos documentos são dados por:

Documento	Vetor de pesos
doc1.txt	(0.1845, 0.3010, 0)
doc2.txt	(0.1625, 0, 0.6021)
doc3.txt	(0.1625, 0, 0)
doc4.txt	(0, 0.3916, 0)

Assim, o conteúdo do arquivo *pesos.txt* será dado por (observe novamente que apenas os pesos diferentes de zero devem ser representados):

doc1.txt:	W, 0.1845	X, 0.3010
doc2.txt:	W, 0.1625	Y, 0.6021
doc3.txt:	W, 0.1625	
doc4.txt:	X, 0.3916	

*pesos.txt*

Note que, para cada documento da base, temos uma lista de pares  $t,q$  onde  $t$  é o termo, e  $q$  é o seu respectivo peso segundo a ponderação adotada. Assim, para o doc2.txt, temos o par W,0.1625. indicando que o termo W tem peso 0.1625 nesse documento, e o par Y,0.6021 indicando que o termo Y tem peso 0.6021 nesse documento. Recomenda-se o uso de dicionários aninhados para o armazenamento dos pesos e o uso do índice invertido construído no trabalho 1. **Ressalta-se que apenas os pesos diferentes de zero devem ser representado no arquivo *pesos.txt*. Os logaritmos devem ser calculados na base 10.**

## Exemplos

Observe os exemplos com termos reais na seção de arquivos da disciplina. Não deixe de usar o corretor automático para conferir seu código. Assumindo que seu arquivo fonte se chama *tfidf.py*, basta usar na linha de comando:

```
> python3 waxm_corretor_tfidf.pyc base.txt tfidf.py
```

ou

```
> py waxm_corretor_tfidf.pyc base.txt tfidf.py
```