Métodos Probabilísticos Para Engenharia Informática

Procura de notícias repetidas

Eduardo Santos – 93107
Pedro Bastos - 93150

Módulos

BloomFilter.java

Construtor:

Na classe Bloom Filter temos como parâmetros do construtor o número de elementos do array, o tamanho do array e o número de Hash Functions, sendo estes valores escolhidos conforme a situação pretendida.

Métodos:

- initBloomFilter: este método inicializa o array do Bloom Filter.
- isPrime: este método recebe como parâmetro um numero (int) ao qual será feita a verificação para ver se é primo, se sim retorna true.
- hashFunction: este método recebe como parâmetro uma string e um numero k (int), sendo este k o número de Hash Functions, e retorna o Hash value dessa mesma string.
- InitHashFunction: este método recebe como parâmetros a length (int) da Hash Function a ser criada e calcula valores aleatórios, os quais coloca na matriz.
- insertElement: este método recebe como parâmetros uma string, a qual vai inserir no Bloom Filter.
- isMember: este método recebe como parâmetros uma string, retornando true se esta está contida no Bloom Filter.

HashFunctions.java

Esta Hash Function é universal e tem como propósito ser usada no MinHash.

Construtor:

O construtor recebe como parâmetros o número de Hash Functions que queremos criar.

Métodos:

- isPrime: este método é igual ao método com o mesmo nome do **BloomFilter.java**
- initHashFunction: este método é igual ao método com o mesmo nome do **BloomFilter.java**
- gh: este método calcula e retorna os hash values.

Shingles.java

Construtor:

O construtor recebe como parâmetros uma string com o texto todo, e um número k (int) que é o comprimento de cada shingle.

Métodos:

- addShingles: este método recebe como parâmetro uma string que é o texto, retornando o array de shingles desse mesmo texto.
- printSh: este método imprime cada shingle do array de shingles.

MinHash.java

Construtor:
O construtor recebe como parâmetros um número k (int), sendo este introduzido como parâmetro da HashFuntion criada dentro do próprio construtor.

Métodos:
- createMinHash: este método recebe como parâmetros um ArrayList<String> de shingles, retornando um vetor com as assinaturas do texto.

- printVetor: este método recebe como parâmetros um ArrayList<Integer> de hashes, imprimindo o vetor com estes mesmo hashes.
- compareHashes: este método recebe como parâmetros dois ArrayList<Integer> de hashes, retornando o número de vezes nas quais os valores são iguais.
- printSimilaresTitles: este método recebe como parâmetros um HashMap<Integer, ArrayList<Integer>>, double percentagem, ArrayList<News> news, retornando as notícias que têm títulos semelhantes.
- printSimilaresContents: este método recebe como parâmetros um HashMap<Integer, ArrayList<Integer>>, double percentagem, ArrayList<News> news, retornando os números das notícias cujos conteúdos são semelhantes.

News.java
Foi criada esta classe para facilitar a manipulação das notícias (títulos - conteúdo).

Construtor:
O construtor recebe como parâmetros uma string com o título da notícia, e outra string com o seu conteúdo.

Testes

Testes individuais ao bloom filter:

Primeiro utilizamos um array de chars com o alfabeto para gerar um array de strings aleatórias, que vão ser usadas nos testes ao bloom filter. Seguidamente, para 1000 elementos, adicionamos cada string aleatória ao bloom filter, e verificamos se foi inserido.

- Função FalsePositives:

Função que recebe um bloom filter e a length das strings. Criando outro array de strings aleatórias, verifica se cada string desse novo array está contida no bloom filter passado como argumento e, se estiver, é um falso positivo. Devolve o número de falsos positivos.

- Função testKFunctions:

Função que recebe um array de strings e a length. Para cada k de 1 a 15, número de hash functions, cria um bloom filter novo e verifica os falsos positivos. Podemos verificar com esta função que o valor ideal de hash functions para o nosso problema é entre 3 e 10, pois é onde encontramos menos falsos positivos.

- Função testSizeBF:

Função que recebe também um array de strings e a length. Cria bloom filters com vários tamanhos em relação ao número de elementos e verifica os falsos positivos. Podemos concluir assim que quanto menor o tamanho do bloom filter, naturalmente mais falsos positivos vai ter e, por isso, o ideal é ter o maior tamanho possível.

Testes individuais ao min hash:

São criadas 2 strings para teste, onde se calcula os arrays das assinaturas de cada uma. Depois, compara-se as assinaturas e verifica-se qual a similaridade entre as duas strings.

Final test:

Visto que todos os dataSets encontrados têm os parâmetros separados por vírgulas, o que no nosso caso não dá, pois o conteúdo de cada notícia contém vírgulas, tivemos que realizar vários splits e uma série de manipulações do dataSet para conseguir separar os parâmetros necessários. Assim, apesar do dataSet utilizado apenas conter 2300 notícias, foi o único que conseguimos manipular corretamente. Dito isto, a primeira parte do código desta classe serve para ler o ficheiro e corretamente separar cada notícia no seu título e conteúdo. Criamos um array de elementos da classe 'News' para uma mais fácil utilização.
Entrando agora na aplicação, criámos um menu para o utilizador poder escolher o que pretende fazer.

- No primeiro caso, o objetivo é encontrar notícias com títulos iguais. O programa vai criar o bloom filter e percorrer as notícias. Para cada uma, verifica se está contida no bloom filter. Se não estiver, insere-a, e se estiver, adiciona a um array de notícias repetidas. Depois dá print no terminal das mesmas.

- No segundo caso, o objetivo é encontrar notícias com títulos similares. Assim, o programa pede ao utilizador a percentagem de similaridade a partir da qual considerar que é parecida. Calcula (usando o módulo MinHash) e coloca as assinaturas num mapa e, utilizando a função 'printSimilaresTitles', calcula e dá print no terminal quais as notícias correspondentes.

- No terceiro caso, o objetivo é encontrar nóticias com conteúdos similares. Assim, o programa faz o mesmo que no segundo caso mas para os conteúdos das notícias.