Query Processing

Retrieval

```
public HashMap<Smartphone, Double> query(Smartphone queryPhone) {
    HashMap<Smartphone, Double> results;
    Query query = new Query(queryPhone);
    if(DOCUMENT_AT_ATIME) {
        results = documentRetrieval(query);
    } else {
        results = termRetrieval(query);
    }
    return results;
}
```

Retrieval

```
public QueryEvaluation() {
    BuildInvertedList bil = null;
    try {
        FileWriter resultFile = new FileWriter(new File("invertedIndex/saida.csv"));
        List<File> files = new ArrayList<>();

        for (int i = 0; i < 10; i++) {
            String fileName = "database/" + i + ".csv";
            files.add(new File(fileName));
        }

        bil = new BuildInvertedList(resultFile, files);
        bil.build();
    } catch(IOException e) {
        e.printStackTrace();
    } finally {
        this.invertedIndex = bil.getInvertedIndex();
        this.numFiles = bil.getNumFiles();
        this.tamCSVs = bil.getTamCSVs();
    }
}
```

Retrieval

```
procedure TermAtATimeRetrieval(Q, I, f, g k)
    A ← HashTable()
    L ← Array()
    R ← PriorityQueue(k)
    for all terms w_i in Q do
        l_i ← InvertedList(w_i, I)
        L.add( l_i )
    end for
    for all lists l_i ∈ L do
        while l_i is not finished do
            d ← l_i.getCurrentDocument()
            A_d ← A_d + g_i(Q) f(l_i)
            l_i.moveToNextDocument()
        end while
    end for
    for all accumulators A_d in A do
        s_d ← A_d                          ▷ Accumulator contains the document score
        R.add( s_d, d )
    end for
    return the top k results from R
end procedure
```

Retrieval

```
procedure DocumentAtATimeRetrieval(Q, I, f, g, k)
    L ← Array()
    R ← PriorityQueue(k)
    for all terms w_i in Q do
        l_i ← InvertedList(w_i, I)
        L.add( l_i )
    end for
    for all documents d ∈ I do
        s_d ← 0
        for all inverted lists l_i in L do
            if l_i.getCurrentDocument() = d then
                s_d ← s_d + g_i(Q) f_i(l_i)          ▷ Update the document score
            end if
            l_i.movePastDocument( d )
        end for
        R.add( s_d, d )
    end for
    return the top k results from R
end procedure
```

Retrieval

```
public static double[][] pairs(double[] a){
    double[] aux = new double[a.length];
    System.arraycopy(a, 0, aux, 0, a.length);

    double[][] sorted = sortArray(a);
    double[][] pairs = new double[(a.length*(a.length-1))/2][2];
    int index = 0;

    for(int i = 0; i < sorted.length; i++){
        for(int j = 0; j < sorted.length; j++){
            if(i != j){
                if(sorted[i][1] < sorted[j][1]){
                    pairs[index][0] = sorted[i][0] + 1;
                    pairs[index][1] = sorted[j][0] + 1;
                    index++;
                }
            }
        }
    }

    System.arraycopy(aux, 0, a, 0, aux.length);
    return pairs;
}
```

```
public static double kendaltau(double[] a, double[] b){
    int d = discordant(pairs(a), pairs(b)) + discordant(pairs(b), pairs(a));
    int k = a.length;
    return 1 - ((double)2*d)/((double)k*(k-1));
}
```

Retrieval

```
public static double ssd(double [] a, double[] b){
    double result = 0;
    for(int i = 0; i < a.length; i++){
        result = result + Math.pow((a[i] - b[i]), 2);
    }
    return result;
 }

public static double spearman(double [] a, double[] b){
    double result = 0;
    double k = a.length;
    result = 1 - ((6 * ssd(a, b)) / (k * (k*k - 1)));
    return result;
 }
```

Retrieval

Busca

Nome

samsung

Info

Preço (R$) 1201-1500 Info

Bateria (mAh) 601-900 Info

Sistema operacional Android

Info

Conectividade Nothing selected Info

Pesquisar

Limpar resultados

Resultado (está retornando no console)

```
nome: smartphone samsung galaxy j3 duos smj320m/ds dourado com dual chip, tela 5.0", câmera 8mp, android 5.1 e processador quad cor
preco: 599.0
so: android
bateria: 5000.0
conectividades: [wi-fi]
nome: smartphone samsung galaxy j3 duos smj320m/ds preto com dual chip, tela 5.0", câmera 8mp, android 5.1 e processador quad core
preco: 625.0
so: android
bateria: 5000.0
conectividades: [wi-fi]
nome: smartphone motorola moto g5 xt1672 ouro com 32gb, tela de 5'', dual chip, android 7.0, 4g, câmera 13mp, processador octacore
preco: 999.0
so: ios
bateria: 5000.0
conectividades: [wi-fi]
nome: smartphone samsung galaxy j5 duos dourado com dual chip, tela 5.0", 4g, câmera 13mp, android 5.1 e processador quad core de 1
preco: 709.0
so: windows_phone
bateria: 5000.0
conectividades: [wi-fi]
nome: smartphone samsung galaxy j7 prime duos preto com 32gb, tela 5.5", dual chip, 4g, câmera 13mp, leitor biométrico, android 6.0
preco: 1249.0
so: android
bateria: 5000.0
conectividades: [wi-fi]
nome: smartphone samsung galaxy j7 prime duos dourado com 32gb, tela 5.5", dual chip, 4g, câmera 13mp, leitor biométrico, android 6.
preco: 1249.0
so: android
bateria: 5000.0
conectividades: [wi-fi]
nome: celular smartphone samsung galaxy j7 duos j700m branco dual chip, 4g, tela 5.5 amoled, câmera 13mp + frontal 5mp com flash, o
preco: 999.0
so: android
bateria: 5000.0
conectividades: [3G]
nome: smartphone lenovo vibe b a2016b30 preto  dual chip, 4g, tela 4.5", câmera 5mp + frontal 2mp, quad core mediatek 1.0ghz, 8gb,
preco: 449.0
so: ios
```

Retrieval

- Alguns celulares não estão com o SO compatível com o modelo
- Ex.: Samsung Galaxy com SO Windows Phone
- Isso aconteceu porque o SO de alguns retornavam nulo, para evitar um NullPointerException, colocamos um SO randômico

Retrieval

```
else if (lowercaseLine.contains("sistema operacional") || lowercaseLine.contains("versão"))
{
    so = this.getOS(lowercaseLine);

    if(so == null) {
        Random rand = new Random();
        int soRand = rand.nextInt(3);
        switch (soRand) {
            case 0:
                so = "android";
                break;
            case 1:
                so = "ios";
                break;
            case 2:
                so = "windows_phone";
                break;
        }
    }

    insertOS(so, fileName, position);
}
```

Retrieval

Comparação com/sem tfidf

```
[0.050313925540785794 0.050313925540785794 0.023998136067101584 0.12835249042145594 0
[0.0 0.0 1.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

SPEARMAN
0.9999946029412977
KENDAL_TAU
0.14960098219766726
```

Retrieval

Obs.: está dando resultados diferentes em computadores diferentes
mesmo usando a mesma versão de projeto do github