IMPLEMENTACION BÁSICA DEL MODELO VECTORIAL

Recuperación y Acceso a la Información
Actividad Dirigida

Grupo L05
Miguel Xoel García Balsa – 100291036
Carlos Contreras Sanz – 100303562

Estructura

El proyecto entregado como solución a la práctica está estructurada de la siguiente manera:

- **Clase Cálculos**: Clase que contiene los métodos que se encargan de realizar los cálculos necesarios para obtener los resultados que se buscan obtener con la práctica (Producto escalar TF, Producto escalar TFIDF, cosenoTF y cosenoTFIDF).
- **Clase Diccionario**: Clase encargada de crear el HashMap que se utilizara como diccionario que contendrá los términos que se encuentran en cada uno de los documentos. Además también se encarga de evitar que se añadan StopWords al diccionario.
- **Clase Documentos**: Clase cuya función es la de listar los documentos con formato .html que se encuentren dentro de una ruta dada. Además también se encarga de obtener el texto del fichero para que la clase diccionario pueda trabajar con los datos.
- **Clase Limpieza**: Clase que se encarga de limpiar el contenido del documento html y separar con StringTokenizer las palabras con las que trabajaremos en el diccionario.
- **Clase Modelos**: Es la clase principal en la que se encuentra el método main. En ella se crean los objetos necesarios para conseguir los resultados que se buscan para cada una de las tres consultas.

Funcionamiento y explicación de las decisiones tomadas

El workflow del proyecto es el siguiente:

En primer lugar, se obtienen todos los documentos .html que se encuentran dentro de la ruta del proyecto que se le pasa por parámetros (Por defecto se encuentran en ./Documentos/) y se les asigna un número identificador único, esto es para poder identificar al documento con el número con un HashMap<Identificador, Nombre Documento>. Siempre suponiendo que todos los documentos tendrán un nombre distinto. Además hemos añadido la funcionalidad de obtener documentos que se encuentran dentro de carpetas que se encuentren en la ruta pasada por parámetros.

Una vez tenemos cada uno de los ficheros .html que se encuentran en la ruta identificados se comienza a generar el Diccionario en el que se almacenaran cada una de las palabras que contienen los documentos.

Para ello primero necesitamos limpiar el contenido de caracteres extraños como &@?(... Es aquí donde la clase Limpieza se encarga de dejar el contenido de los documentos separado en palabras que únicamente contienen caracteres del abecedario.

Tras realizar esta limpieza se pasa a insertar cada una de las palabras al Diccionario. El Diccionario que hemos decidido crear está formado por un HashMap de HashMaps, es decir en este diccionario guardaremos lo siguiente:

< Palabra, < Documento, Numero veces que aparece la palabra > >

Siendo Documento el numero identificador del que hablábamos anteriormente. Siempre asegurándonos que de no se insertan palabras repetidas en el diccionario y de que se suprimen las conocidas como StopWords que no aportan información que se pueda tener en cuenta a la hora de realizar los cálculos que buscamos. Además tras ver que la separación de palabras realizada por el StringTokenizer dejaba varios caracteres solos, hemos decidido suprimir cada una de las letras del abecedario que apareciera sola, y hemos pasado todo el texto extraído de los documentos a minúsculas.

Este paso se realiza con cada uno de los documentos encontrados en el primer paso y una vez terminado ya disponemos del diccionario de los documentos, por lo que se procede a realizar los cálculos.

Es ahora cuando ya tenemos que considerar también los datos de las consultas, y como son palabras externas a los documentos no las añadimos al diccionario, ya que podrían afectar a los resultados del resto de consultas. Es por esto por lo que creamos lo que llamamos diccionarioConsulta que es un diccionario que incluye únicamente las palabras de la consulta junto al número de veces que aparece cada palabra.

El primer calculo que realizamos es el del IDF, que lo obtenemos por cada palabra del diccionario y consulta (pueden existir palabras que aparezcan en la consulta y no en los documentos) haciendo el logaritmo en base 10 del número total de documentos entre el número de documentos en el que aparece la palabra (incluyendo la consulta). Esto hace que las palabras que aparecen en todos los documentos tengan un valor negativo y dándole una mayor importancia a las palabras que aparecen en un único documento. Este resultado se utilizara a la hora de calcular el productoEscalarTFIDF y cosenoTFIDF.

Tras esto se pasa a realizar el cálculo del productoEscalarTF con el que tenemos que tener en cuenta el contenido del Diccionario de Documentos y el de la consulta, ya que si una palabra se repite en la consulta obtiene un mayor peso en el caso de que esa palabra exista en el diccionario, tras hacer las operaciones se obtiene un HashMap < Numero_ID_Documento, Valor ProductoEscalarTF > sobre la consulta que se ha pasado por parámetros. Al igual ocurre con el resto de cálculos a pesar de que se utilizan datos distintos.

Para la realización de estos métodos que calculan los valores que se buscan con la práctica se han utilizado una serie de métodos auxiliares para hacer que el código fuera más limpio y evitar que los métodos realizaran un gran número de operaciones.

Tabla de Resultados

```
* RELEVANCIA: ProductoEscalarTF
Nombre del doc  Q1              Q2              Q3
2010-22-100     0.000000        0.000000        0.000000
2010-42-103     210.000000      0.000000        0.000000
2010-58-044     3.000000        15.000000       1.000000
2010-76-088     50.000000       0.000000        2.000000
2010-99-086     5.000000        1.000000        34.000000

* RELEVANCIA: ProductoEscalarTFIDF
Nombre del doc  Q1              Q2              Q3
2010-22-100     0.000000        0.000000        0.000000
2010-42-103     11.950439       0.000000        0.000000
2010-58-044     0.177139        0.956532        0.158356
2010-76-088     1.743988        0.000000        0.207573
2010-99-086     0.046958        0.049217        4.292719

* RELEVANCIA: CosenoTF
Nombre del doc  Q1              Q2              Q3
2010-22-100     0.000000        0.000000        0.000000
2010-42-103     0.474624        0.000000        0.000000
2010-58-044     0.037689        0.312500        0.018634
2010-76-088     0.133276        0.000000        0.007907
2010-99-086     0.028369        0.009409        0.286129

* RELEVANCIA: CosenoTFIDF
Nombre del doc  Q1              Q2              Q3
2010-22-100     0.000000        0.000000        0.000000
2010-42-103     0.195749        0.000000        0.000000
2010-58-044     0.016311        0.070778        0.014547
2010-76-088     0.030561        0.000000        0.003599
2010-99-086     0.001790        0.001421        0.173088
```

Ademas de los metodos creados para la realizacion de los calculos tambien se han creado unos metodos para mostrar los datos en forma de tabla como se pedia en el guion de la practica.

Los Resultados obtenidos muestran que en realidad funciona como debe, ya que si vemos la consulta Q1 trata sobre videojuegos de conduccion, y el documento 2010-42-103 es una pagina de una wiki que habla de un videojuego de conduccion.

Ademas de esto, el documento 1 que es una pagina que apenas tiene contenido vemos como siempre obtiene el valor de 0 en todos los calculos al no estar relacionado con las consultas.

Con el resto de consultas ocurre lo mismo.

Ademas podemos asegurar su funcionamiento al realizar pruebas con los mismos datos aportados en la hoja de calculo del modelo vectorial obteniendo los mismos resultados que en la hoja de calculo.