UNIVERSIDAD
DE GRANADA

Práctica 1

Preprocesado de documentos

Parser de documentos con Tika

Integrantes

David Estévez Martínez

Mauricio Luque Jiménez

5 de Octubre de 2023

Recuperación de Información

Primer Apartado

Metadatos del fichero

Para este ejercicio, en el que se nos pide mostrar los metadatos de cada fichero (nombre, tipo, codificación e idioma), vamos a utilizar tres métodos distintos, uno para cada uno de los tres últimos datos mencionados. Para esto, vamos a utilizar la clase Parser, de la cual inicializamos un objeto *parser* con la función *AutoDetectParser().* Una vez tenemos el objeto *parser*, aplicamos la función principal de la clase, a la que le pasamos como argumento un fichero concreto (mediante un *InputStream*), el handler (mediante *BodyContentHandler*), el atributo *metadata* (que va a devolver el valor del metadato concreto que queramos obtener en función del método en el que se le llame) y el atributo *ParseContext.*

Esta estructura la aplicamos de forma idéntica para extraer el tipo y la codificación del archivo que se pase inicialmente como argumento, cambiando únicamente el valor obtenido de *metadata*.

Por otra parte, para conocer el idioma del archivo, vamos a hacer uso del paquete *LanguageDetector*, con el que vamos a crear un objeto *languageDetector* con el que vamos a leer el archivo pasado como argumento.

Extracción de enlaces

Para la extracción de enlaces del fichero, vamos a aplicar parte del ejercicio anterior, concretamente la función *AutoDetectParser()* para identificar el tipo de archivo. En este caso, la principal diferencia viene en el uso del handler, ya que en vez de crear un *BodyContentHandler* creamos un *LinkContentHandler*, que extrae todos los enlaces del archivo, que son almacenados en una lista.

Ocurrencia de palabras

Por último, para este ejercicio, vamos a hacer un recuento de las apariciones de cada palabra en el fichero pasado como argumento. Como en los anteriores ejercicios, hacemos uso del *AutoDetectParser* para identificar el tipo de archivo, y en este caso utilizamos el *BodyContentHandler* para poder leer el contenido del fichero y pasarlo a un *String*, del cual separamos las palabras por los espacios que hay entre ellas.

Para hacer el recuento de apariciones de cada palabra, creamos un *Map* en el que la clave es la propia palabra y el valor es el número de apariciones. Se pasan todas las palabras a minúsculas, evitando problemas con tildes, y se introduce cada palabra en el *Map*: se llama a la función *getOrDefault()* para que devuelva el valor de la clave pasada como argumento. Si no la encuentra, quiere decir que es la primera aparición de esa palabra, por lo que la añade al Map y le da valor 1. Si la encuentra, simplemente le suma 1 a su valor actual. Finalmente, se ordenan las palabras según su frecuencia en el archivo y se genera el CSV correspondiente.

A continuación, un par de muestras de las distintas nubes de palabras obtenidas de unos ficheros concretos.

Segundo Apartado

ley de Zipf

Para este segundo apartado, vamos a tomar como referencia documentos de mayor tamaño. Concretamente, vamos a tratar libros en diferentes idiomas. Estos libros van a ser El Ingenioso Hidalgo Don Quijote de la Mancha, de Miguel de Cervantes, en español y en su traducción al húngaro, y El Príncipe, de Nicolás Maquiavelo, en inglés.

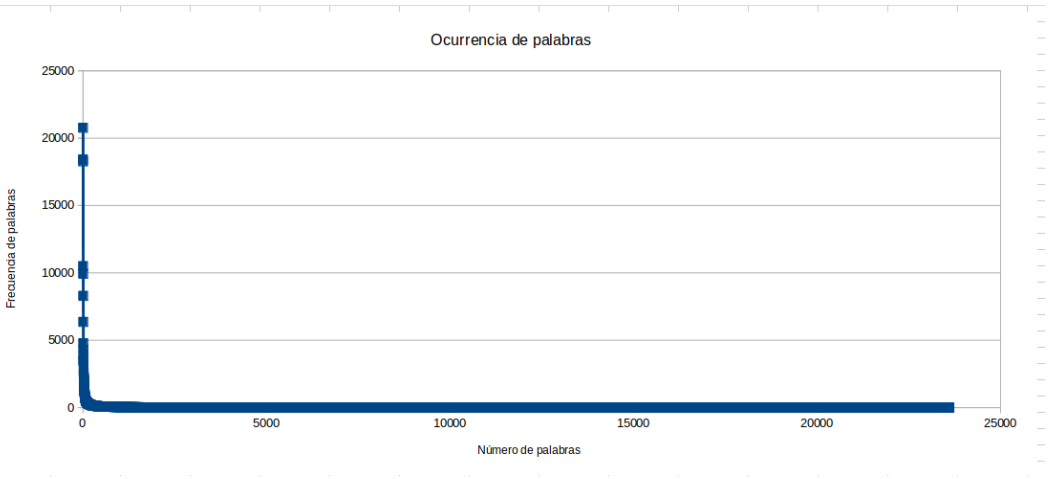
Aplicando el algoritmo explicado en el último ejercicio del apartado anterior y pasando los resultados a una gráfica, vemos la distribución absoluta de cada uno de los tres libros. Vemos que hay un pico muy pronunciado en los valores más bajos del eje X (que indica el número de palabras con una determinada frecuencia) y en los valores más altos del eje Y (que indica el número de veces que aparece un número concreto de palabras). En todos los casos vemos cómo se cumple el patrón de que las palabras más repetidas son minoría, como es lógico.

Por otra parte, si aplicamos escala logarítmica a los dos ejes de cada gráfica, vemos que en los tres casos se mantiene bastante fiel a la ley de Zipf, tomando *f(x)* como la frecuencia de una palabra, el primer valor numérico como una constante, *x* como el ranking de dicha palabra y el exponente de *x* como otra constante. Además, observamos que donde más falla es en frecuencias altas y bajas, tal y como se explica en la teoría.

A continuación, las gráficas, tanto de las frecuencias absolutas como de las escalas logarítmicas (y su relación con la ley de Zipf) de los tres libros escogidos.

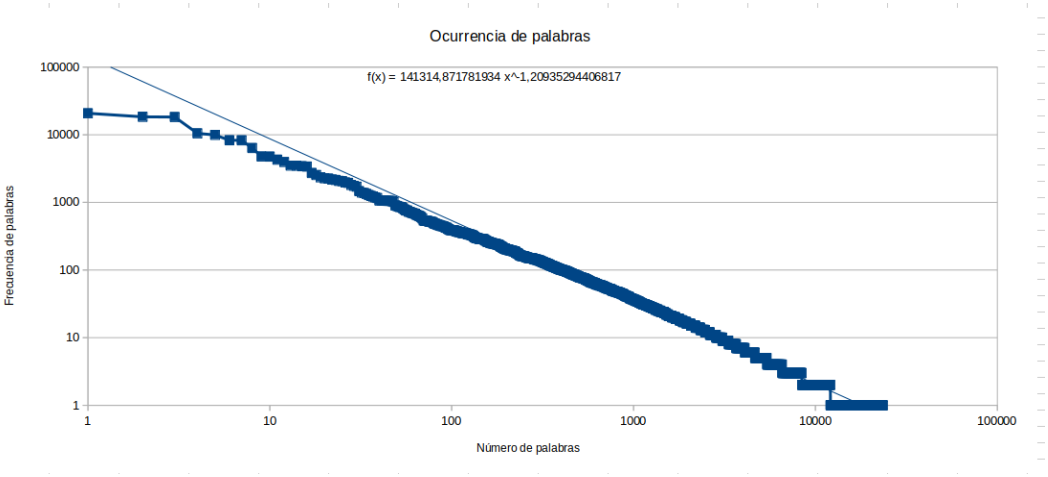
El Ingenioso Hidalgo Don Quijote de la Mancha, Miguel de Cervantes

que	20767
de	18412
y	18271
la	10492
a	9933
en	8284
el	8265
no	6356
los	4769
se	4751
con	4275
por	3945
lo	3492
las	3486
le	3420
su	3388
don	2718
del	2536
me	2345
como	2269
quijote	2245
sancho	2174
es	2145
yo	2077
más	2055
si	1968
un	1943
dijo	1808
al	1754



Frecuencia absoluta

que	20767
de	18412
y	18271
la	10492
a	9933
en	8284
el	8265
no	6356
los	4769
se	4751
con	4275
por	3945
lo	3492
las	3486
le	3420
su	3388
don	2718
del	2536
me	2345
como	2269
quijote	2245
sancho	2174
es	2145
yo	2077
más	2055
si	1968
un	1943
dijo	1808
al	1754

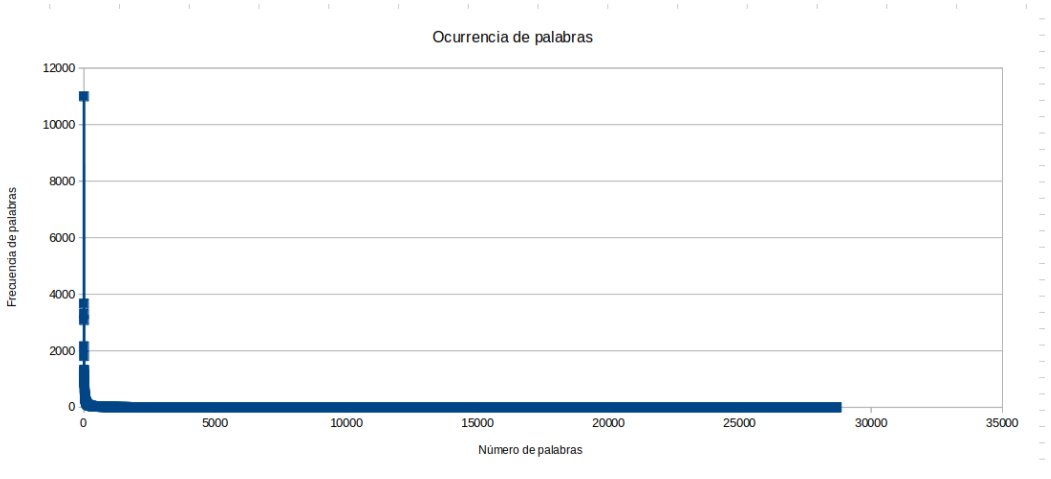


Frecuencia logarítmica

El Ingenioso Hidalgo Don Quijote de la Mancha, Miguel de Cervantes

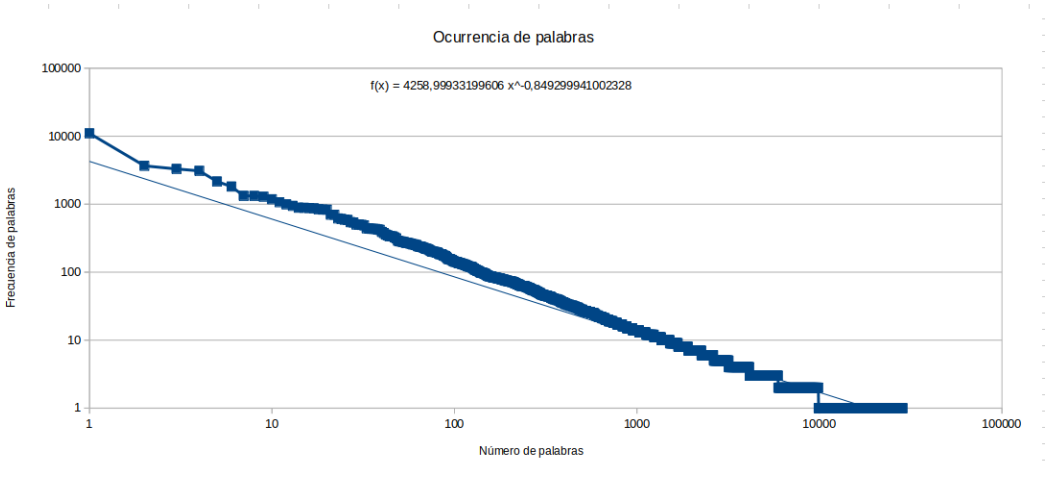
Traducción al Húngaro

a	11000
s	3665
az	3298
hogy	3089
is	2155
nem	1816
don	1325
meg	1323
és	1286
sancho	1180
quijote	1068
egy	992
ki	944
mint	886
ha	884
de	876
még	868
azt	842
csak	832
én	830
mert	699
már	699
volt	617
úgy	606
e	591
mi	586
ez	543
pedig	537
el	501



Frecuencia absoluta

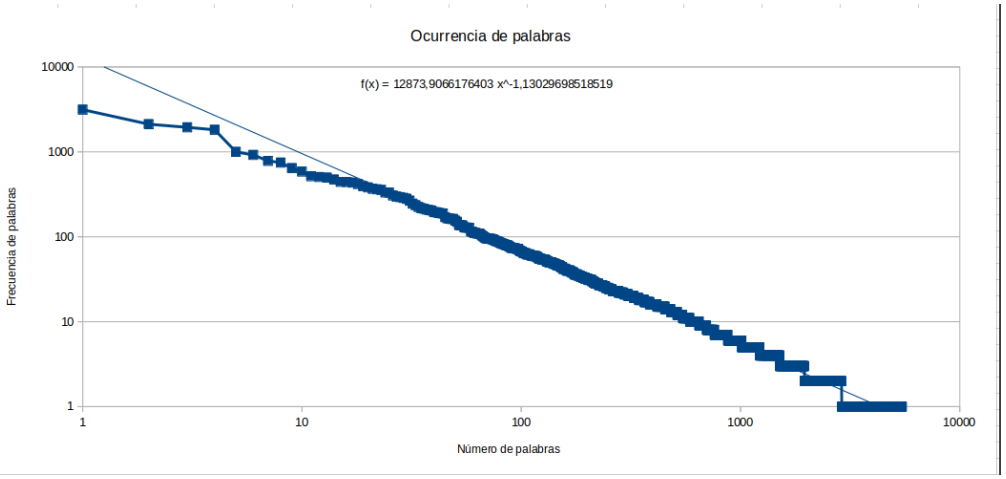
a	11000
s	3665
az	3298
hogy	3089
is	2155
nem	1816
don	1325
meg	1323
és	1286
sancho	1180
quijote	1068
egy	992
ki	944
mint	886
ha	884
de	876
még	868
azt	842
csak	832
én	830
mert	699
már	699
volt	617
úgy	606
e	591
mi	586
ez	543
pedig	537
el	501



Frecuencia logarítmica

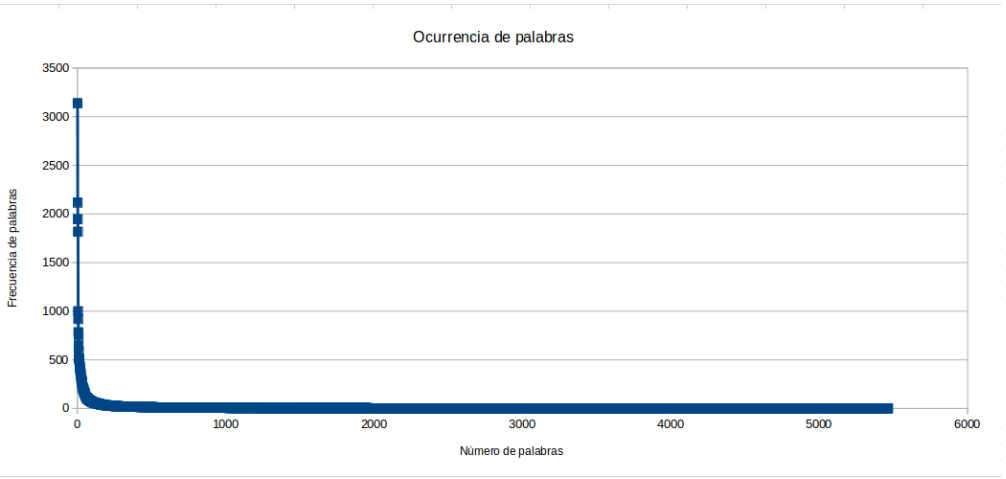
El Príncipe, Nicolás Maquiavelo

the	3138
to	2116
and	1945
of	1817
in	999
he	919
a	783
that	747
his	642
it	585
by	516
not	505
with	497
be	472
for	443
they	441
is	434
him	417
have	394
was	381
who	367
this	362
which	356
them	332
as	331
are	306
one	297
had	293
but	287



Frecuencia absoluta

the	3138
to	2116
and	1945
of	1817
in	999
he	919
a	783
that	747
his	642
it	585
by	516
not	505
with	497
be	472
for	443
they	441
is	434
him	417
have	394
was	381
who	367
this	362
which	356
them	332
as	331
are	306
one	297
had	293
but	287



Frecuencia logarítmica

Trabajo en grupo

Reparto de tareas

- Primer apartado
    - Primer ejercicio
        - Obtención de metadatos: David Estévez Martínez
    - Segundo ejercicio
        - Obtención de enlaces: David Estévez Martínez
    - Tercer ejercicio
        - Ocurrencia de palabras: David Estévez Martínez
        - Nubes de palabras: Mauricio Luque Jiménez
- Segundo apartado
    - Obtención de gráficas
        - Ocurrencias absolutas: Mauricio Luque Jiménez
        - Escala logarítmica (Ley de Zipf): Mauricio Luque Jiménez
- Documentación: Mauricio Luque Jiménez