UNIVERSITAT POLITÈCNICA DE CATALUNYA

BACHELOR DEGREE ON INFORMATICS ENGINEERING

CERCA I ANÀLISI D'INFORMACIÓ MASSIVA

Q1 Course 2020-2021

La ley de Zipf y Heaps

GROUP 21

Autores:
Daniel CANO CARRASCOSA
Samuel SALVADOR ROCA

INDEX

Objetivos **2**

Limpieza de documentos **2**

Implementación 2

Análisis de resultados de la ley de Zipf **3**

Resultados 3

Análisis de resultados de la ley de Heaps **3**

1. Objetivos

En esta práctica nuestro objetivo es dado unos conjuntos de texto, analizar si estos siguen las leyes de Zipf y Heaps las cuales vienen definidas por las siguientes fórmulas:

Zipf = $\frac{c}{(rank + b)^a}$ Que nos indica la frecuencia esperada de la palabra con posición *rank* de forma decreciente(cuanto más cercana a uno más rango) dado unos parámetros c, b y a	Heaps = $k * N$ Que nos indica para una documento de N palabras en total, el número esperado de palabras diferentes, es decir únicas, que no se repiten

2. Limpieza de documentos

La primera tarea que hemos tenido que hacer ha sido la limpieza de los 3 documentos proporcionados, aunque en nuestro caso hemos implementado un método de limpieza único para cada uno de los tipos de fichero que se nos proporcionaban, los cuales eran los ficheros de tipo **novelas** (novels), **noticias** (news) o **documentos** (arxiv) ya que tal como se nos informo el más "preparado" para ser analizado o mejor dicho el documento que estaba de base más pulido, con ficheros de una longitud aceptable, y por ende de los cuales se podría esperar inicialmente un mejor ajuste a las funciones mencionadas previamente eran los ficheros de novelas, decidimos por nuestra parte analizar los 3 para así poder ver las posibles diferencias entre ellos y sacar más conclusiones que analizando un único conjunto de ficheros.

2.1 Implementación

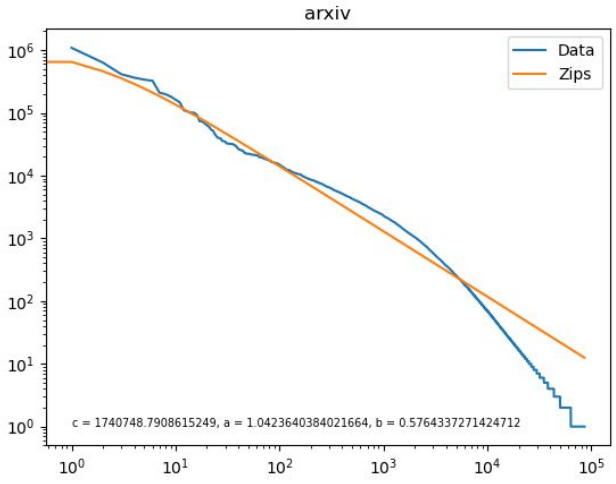
Para poder aplicar la limpieza de los documentos en nuestro caso creamos tres clases, una por cada tipo de fichero, todas declaradas dentro del fichero **Clean.py** donde algunos de los criterios que usamos fueron algunos como eliminar palabras negativas en forma negada, es decir cualquier palabra que tuviera la combinación de caracteres n't eliminar esta parte, eliminar las barras como "_" o eliminar caracteres especiales como "Ô", todo depende del tipo de documento al cual nos estábamos dedicando, aunque hay que mencionar que algunas características eran comunes en los 3 documentos por lo que se mantenían en todos.

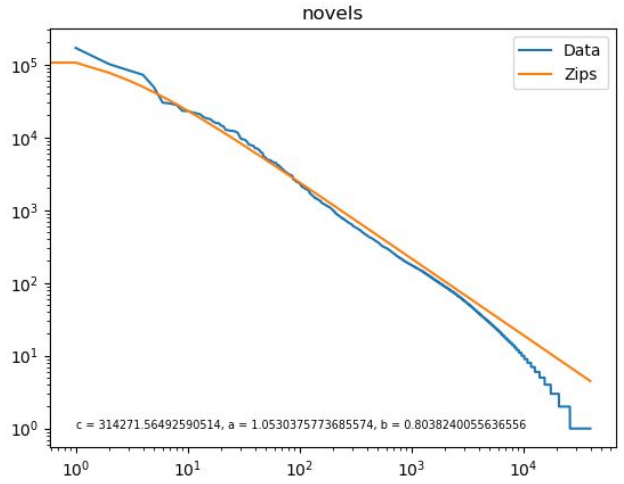
3. Análisis de resultados de la ley de Zipf

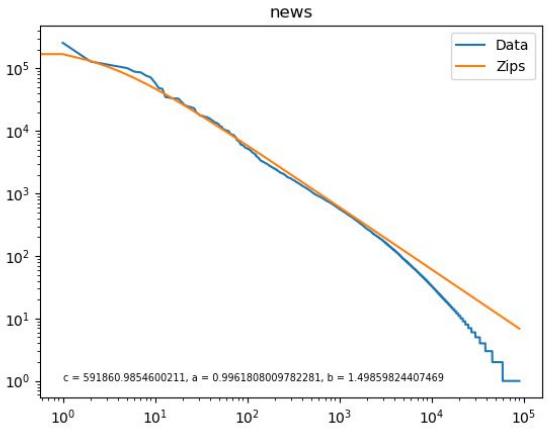
Una vez ejecutada la limpieza de todos los textos era el momento de analizar si estos seguían la ley de Zipf, es decir si se podían ajustar los parámetros para poder llegar a predecir, para un texto, la frecuencia de la enessima palabra con más frecuencia, por lo que se puede entender más como una especie de estimador o "esperanza" de la frecuencia más que una ley exacta como podremos ver.

3.1 Resultados

Una vez ejecutado elasticsearch usando el CountWords.py proporcionado por los profesores pero modificado obtenemos los siguientes resultados:







Como podemos ver a primera vista no hay diferencias exageradas, no obstante si nos fijamos bien podemos ver como de todas las pruebas hechas la que más nos ha dado un buen resultado ha sido la de novelas, la cual se acerca más a los valores teóricos

donde se dice que a = 1 y b = 1, en cambio se pueden ver en las otras gráficas como esto no es así, al igual que la recta se desvía más de el ajuste que en el caso de novelas. Así como también podemos ver como en el caso de los "Axiv" al principio de todo parecen haber una recta, que repentinamente cambia su curso hacia una trayectoria más diagonal, lo que nos puede dar a entender que la gráfica está compuesta de dos rectas.

3.2 Ajuste de la recta

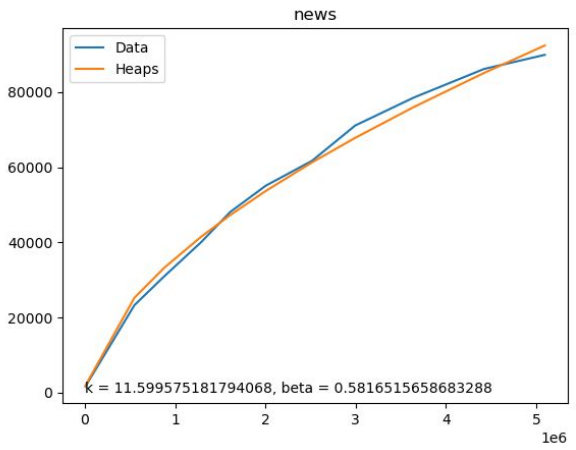
En este caso (y en el siguiente apartado) lo que hemos hecho ha sido, en vez de malgastar tiempo intentando obtener el valor adecuado para cada uno de los parámetros usar la función curve fit de scipy el cual nos hace el trabajo de ajustarnos lo máximo posible la función a nuestra recta original usando como método de error el MSE (mean square error), método bastante conocido y eficaz para este tipo de problemas, aunque todo se tiene que decir que inicialmente hicimos las pruebas manualmente, cosa que nos sirvió para ayudar al algoritmo a establecer unos valores iniciales para los parámetros ya que si no nos daba error ya que al tener un límite de iteraciones de entrenamiento no lograba ajustar los valores. Al igual que el uso de Matplotlib para mostrar la gráfica en escala logarítmica, librería que por nuestra parte ha resultado bastante sencilla de usar sin ningún tipo de problema.

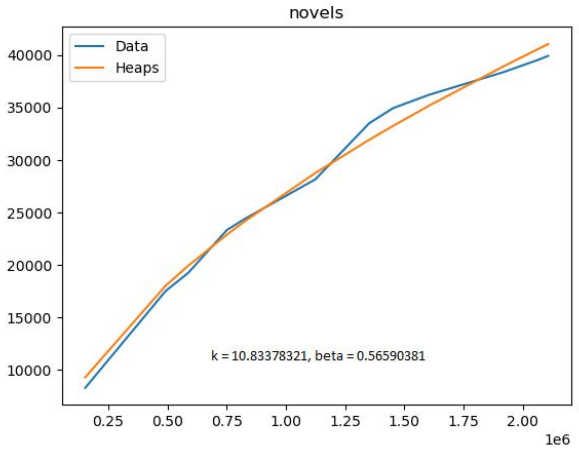
4. Análisis de resultados de la ley de Heaps

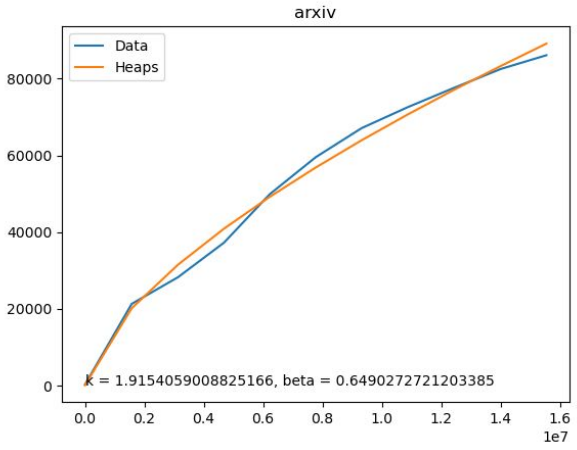
Seguidamente de analizar los resultados de Zipf nos disponemos a mostrar los resultados obtenidos por parte de la ley de Heaps, que intenta predecir para un documento de N palabras cuántas palabras diferentes hay en ese documento, nuevamente se puede ver como un estimador más que como una función exacta.

4.1 Resultados

Una vez ejecutado Heaps para los tres tipos de documentos hemos obtenido los siguientes resultados que se muestran a continuación.







Como se puede observar en las tres gráficas la ley de Heaps se ajusta bastante bien, no obstante hay que destacar que para el conjunto de ficheros arxiv el valor de k es significativamente diferente al de los siendo este significativamente menor, lo cual teniendo en cuenta que la ley de Heaps nos habla del número diferentes de palabras que hay en un texto de N caracteres, al ser k un valor tan pequeño podemos entenderlo como que el número de palabras distintas no aumenta significativamente a medida que aumentamos el tamaño del corpus que estamos analizando, por lo que se podría entender como que en este caso arxiv tiene una riqueza lèxica mucho menor a sus dos compañeros, ya que da el crecimiento de palabras variadas realmente es bastante pobre respecto a los dos anteriores

5. Conclusiones

Una vez analizados los tres textos con los das leyes correspondientes hemos podido apreciar, la gran sorpresa de los valores de la ley de heaps y como estos indican según nuestros análisis la poca riqueza léxica del texto, en cambio por la ley de Zipf hemos podido enlazar este hecho con el repentino descenso inicial que se puede apreciar en la gráfica. También hemos podido apreciar como interesantemente al final de todas las gráficas de Zipf las caídas se ven escalonadas indicando que hay una enorme cantidad de valores que comparten bajas frecuencias mientras no es así con las altas frecuencias y así como hemos podido percatarnos de esto hemos podido ver como de todos los corpus el que mejor se adapta era el de novelas ya que entendemos que era el más pulido de todos inicialmente con menos caracteres erróneos.