CAIM –FIB

Pràctica 1

ElasticSearch and Zipf's and Heaps' laws

Pau Bosch Coll

Ramon Ribas Domingo

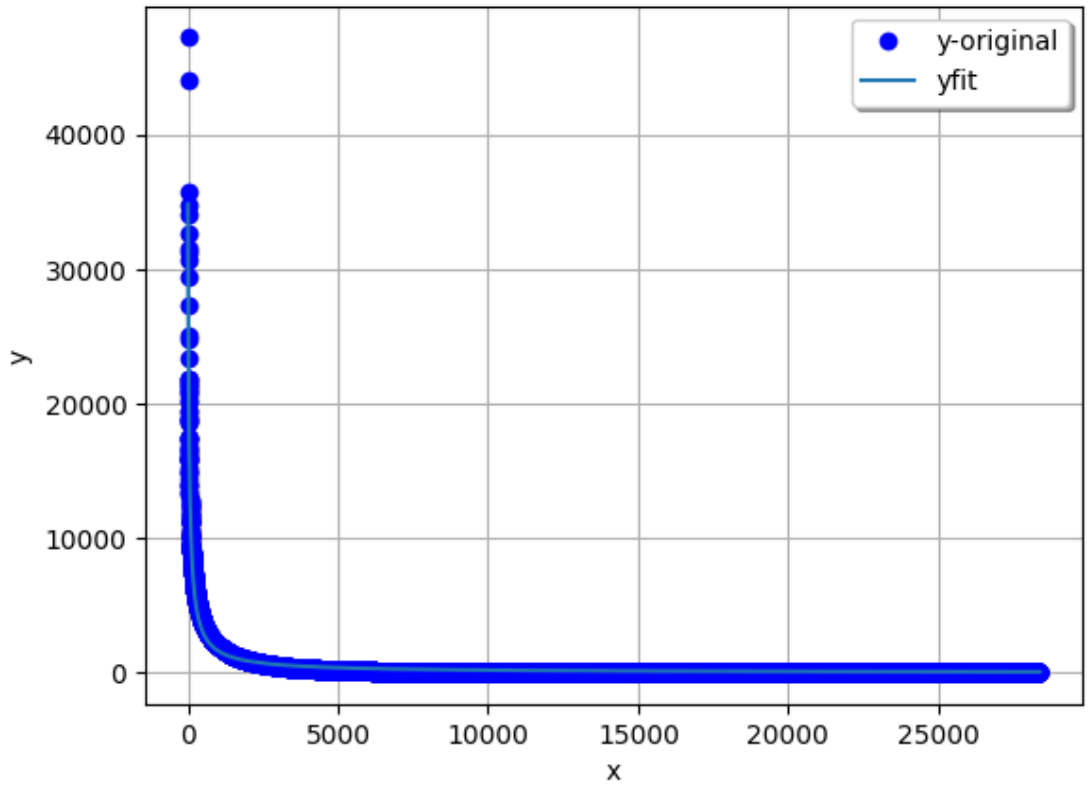
Index

Introducció

En aquesta pràctica utilitzarem el software elasticsearch per trobar les repeticions de paraules de diferents textos per tal d'apreciar dues lleis del llenguatge: la llei de Zipf i la llei de Heap.

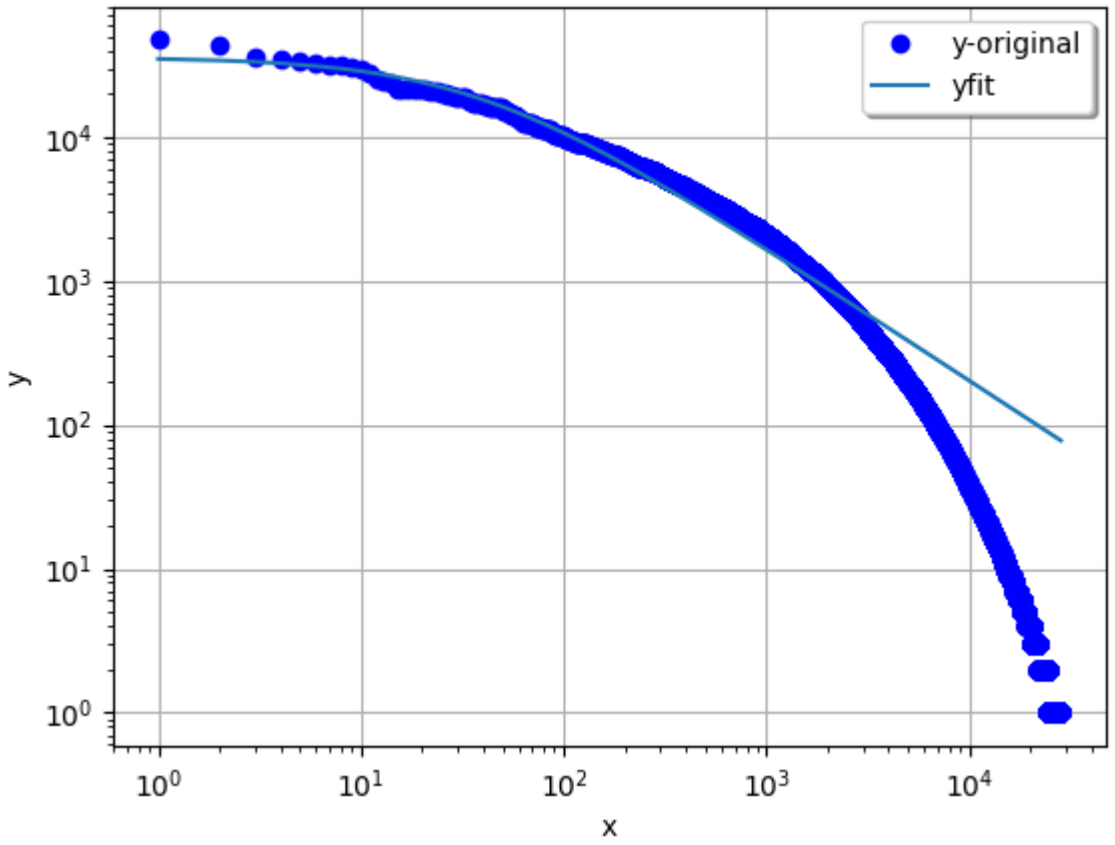
Llei de Zipf

Per apreciar la llei de Zipf necessitarem crear una gràfica logarítmica de les paraules ordenades pel seu rang de forma decreixent. Abans d'això, hem decidit realitzar un preprocessing on esborrarem stop word,i paraules amb faltes d'ortografia i nombres. Les stopwords seran pronoms personals, articles, i en general paraules que no aporten informació al text. Començarem amb la base de dades "news".



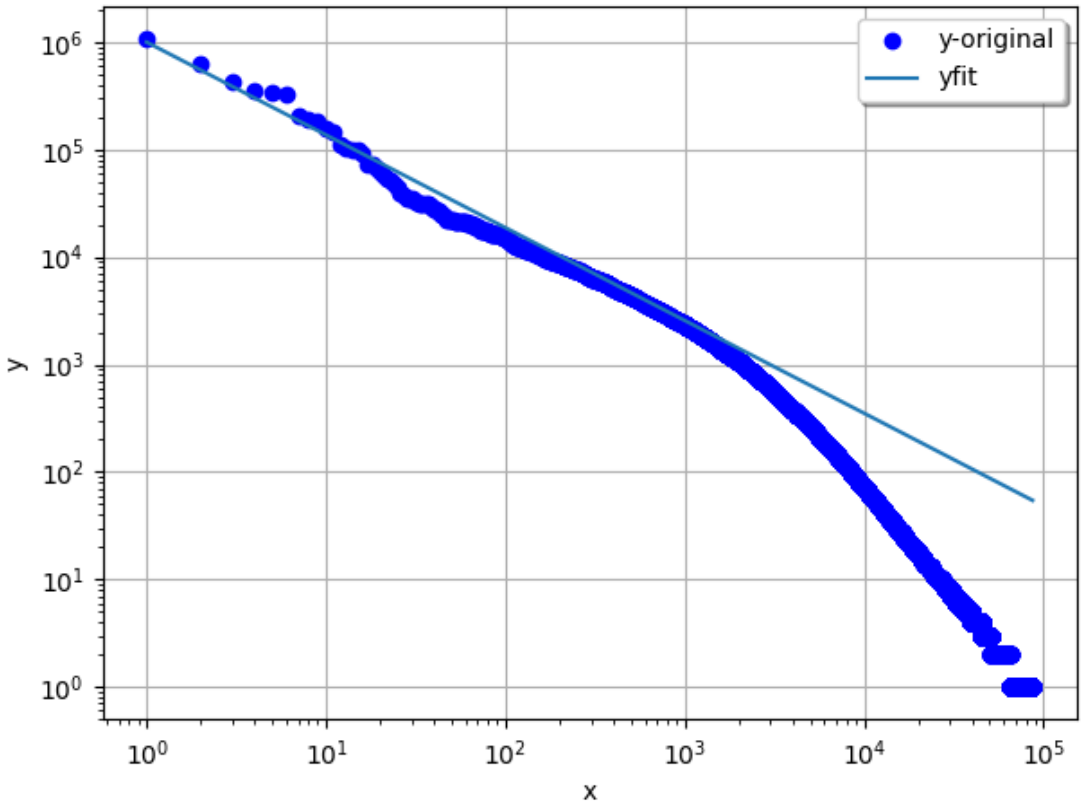
Imatge1: Escala normal amb stopwords i amb spellcheck de news

El resultat descriu una gràfica exponencial, pero per poder observar la llei de zipf necessitarem escalar a escala logaritmica els eixos X i Y i intentar trobar una recta que ho defineixi.



Imatge2: Escala log log + spellcheck + eliminar stopwords de news

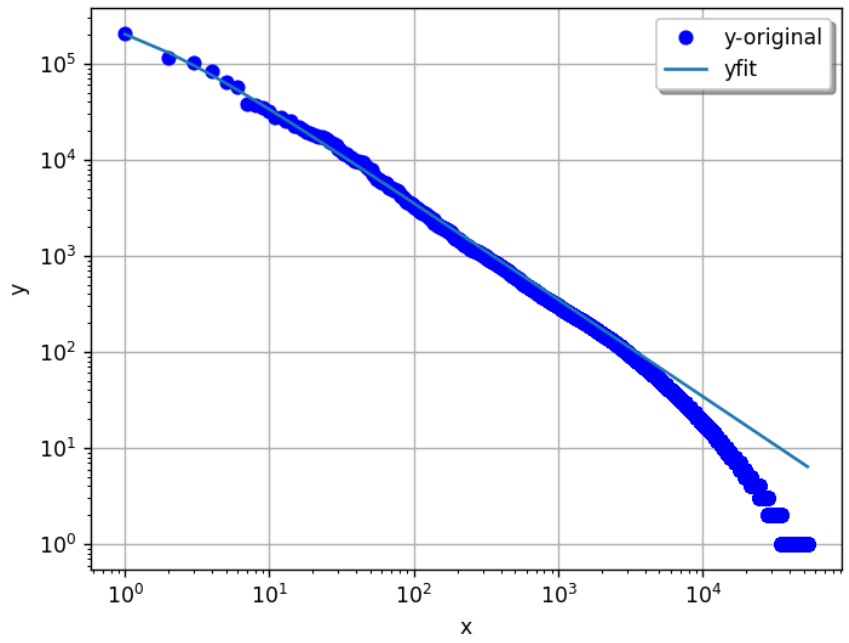
En la gràfica resultant es nota una tendència a definir-se per una funció decreixent però encara és lluny de mostrar la llei de Zipf, ja que seria d'esperar una recta amb pendent -1. Això es pot deure a que la llei de zipf té en compte les repeticions de totes les paraules, incloses les stop words. L'spellcheck també pot ser causa d'error, ja que no funciona perfectament i pot causar desbalanceig. Així que ho executarem un altre cop sense aquests preprocessings.



Imatge3: Escala log log amb stop words i sense spellcheck de news

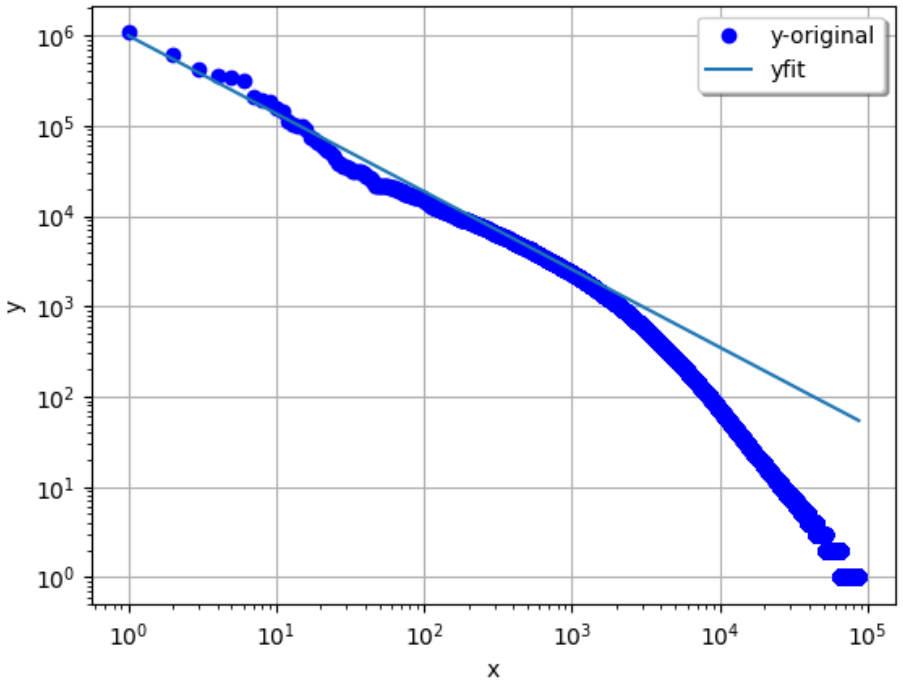
params: [9.22569352e-01 3.70138105e+01 1.00000000e+06] a,b i c respectivament

El nou resultat descriu millor la llei de zipf, tot i que demostra dues tendències. Això es pot deure a diferents registres de llenguatge. Els parametres de la funció descrita els hem trobat gràcies a traves de la funció curve_fit de la llibreria scipy.optimize.



Imatge4: Escala log log amb stop words i sense spellcheck de Novel·les

params: [1.00782486e+00 8.08270196e-01 3.69210463e+05] a,b i c respectivament



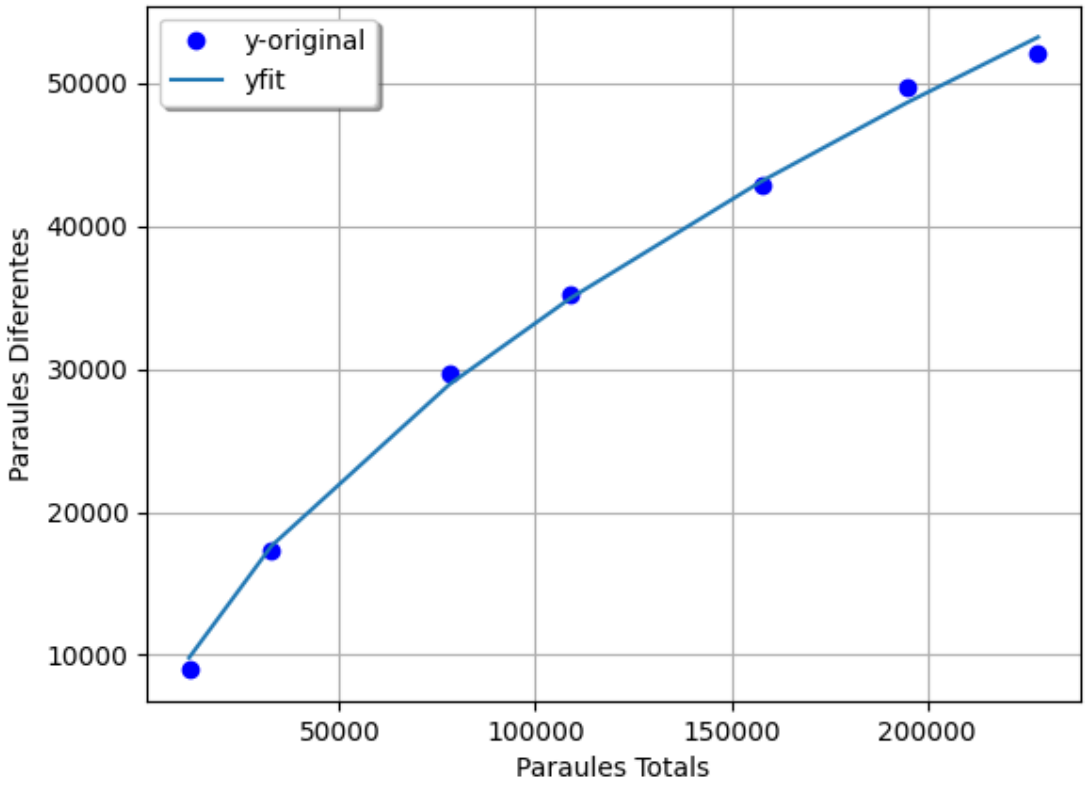
Imatge5: Escala log log amb stop words i sense spellcheck de Arxiv_abs

params: [1.16031561e+00 3.38134866e+03 1.00000000e+06] a, b i c respectivament

Hem calculat el mateix en les databases de Noveles i Arxiv_abs i podem apreciar millor com la distribució de freqüències descriu una funció exponencial tal com indica la llei de Zipf, i on observem petites variacions en les diferents mostres, com es d'esperar, degut a que s'utilitzen diferents registres, i en les novel·les on s'utilitza un lèxic més ric i variat es on segueix més la llei de Zipf.

Llei de Heaps

Per tal de comprovar si el nombre de termes diferents que apareixen en un text de mida N segueix la fórmula $k \times N^{\beta}$ per alguns paràmetres K i β, hem modificat els fitxers indexFiles.py per tal de crear índex amb un cert nombre de novel·les i CountWordsHeap que compta el nombre de paraules diferents i totals que té cada index que hem creat, i un cop ho tenim emprem la funció de curve_fit per tal d'obtenir els paràmetres que millor s'ajusta i plotejem la gràfica, tot obtinguent els resultats mostrats a la imatge 6 i que com podem apreciar la funció que representa la llei de Heap's s'ajusta força bé als punts que hem obtingut i el resultat que esperàvem. Si tinguéssim una col·lecció de novel·les major i continuessim augmentat el nombre de paraules totals el pendent seguiria creixent cada cop més lentament fins arribar a un límit.



Imatge 6: Gràfica de la llei de Heap's amb parametres : [48.01169085 0.56819151] k i β respectivament