Entregable Laboratorio 1

CAIM

28/09/2018

Alejandro Domínguez Besserer
José Juan Aguilar Cerrejón

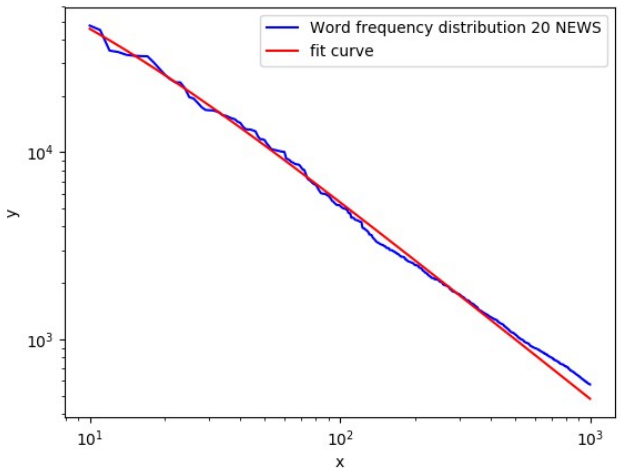
Ley de zipf

Metodología:

Hemos escrito un script en python (**ses1.py**) que recoge los resultados de CountWords.py, los distribuye en un plot según su frecuencia y después busca los valores de a,b y c que se ajusten mejor a los datos obtenidos siguiendo la powerlaw Zipf-Mandelbrot, para esto usamos la clase **curve_fit** de la libreria **scipy** de python. En las gráficas log-log que vemos a continuación podemos apreciar que efectivamente la distribución de palabras aproxima una powerlaw. Para eliminar los "outliers", cogemos a partir del término 10 hasta el 1000.

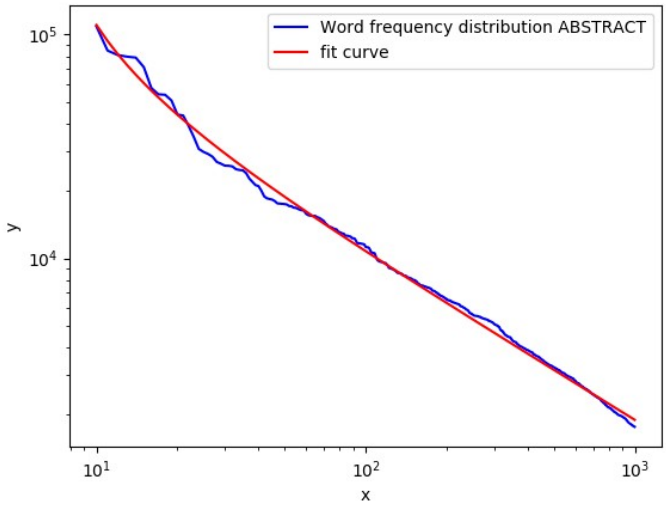
20 newsgroups:

a = 1.067
b = 4.124
c = 770581.293



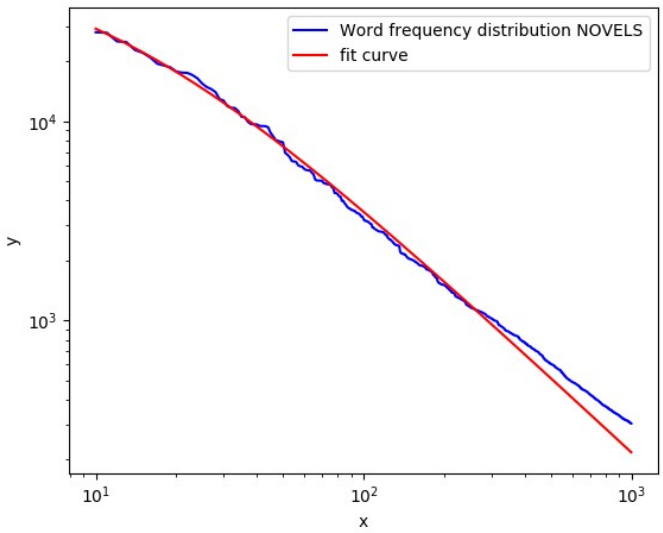
Abstract:

a = 0.736
b = -6.01
c = 306262.486



Novels:

a = 1.256
b = 10.535
c = 1299751.978



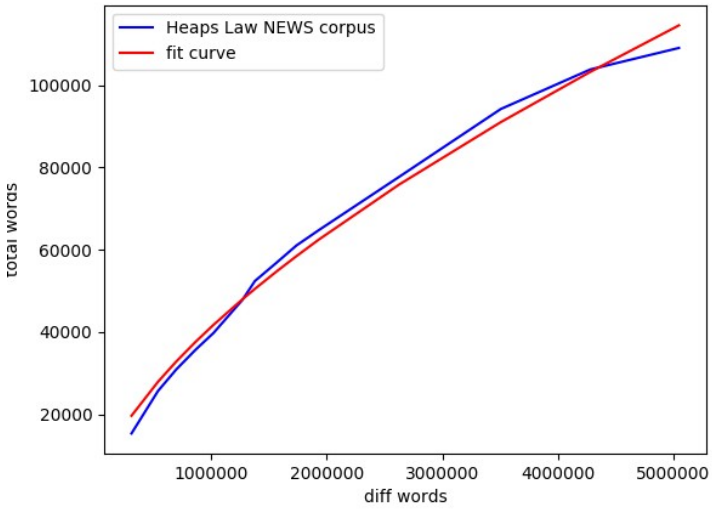
Ley de Heap

Metodología:

Hemos creado un script llamado **CountAllWords.py** que para un corpus dado nos proporciona el número de palabras diferentes y el número de palabras totales. Con este resultado, el script **plotting.py** nos crea una gráfica en la cual vemos la relación entre el número de palabras diferentes y el de palabras totales. Para ver si esta distribución sigue la ley de heap, volvemos a usar el **curve_fit** para que nos encuentre los valores que aproximan nuestros datos la curva k × N^β.

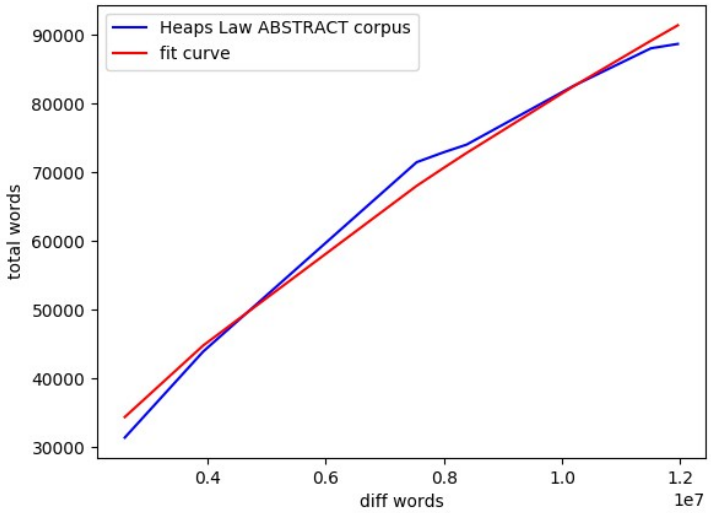
20 newsgroups:

k = 6.785
β = 0.63



Abstract: (8 samples)

k = 2.742
β = 0.639



Novels: (14 samples)

k = 4.817
β = 0.628