Loi de Zipf - Analyse de Brown Corpus

UFR de sociologie et d'informatique pour les sciences humaines

Programmation de Modèles Linguistiques (I) 2020-21

Carlos González et Gaël Lejeune

Chargement de bibliothèques

In [1]:

```
from nltk.corpus import brown
from functools import reduce
import matplotlib.pyplot as pyplot
import re
```

Définition des fonctions

In [2]:

```
def load_brown_corpus():
    p = re.compile('\W')
    texte = [token.lower() for token in brown.words()]
    texte = [token for token in texte if not p.match(token)]

    return texte


def texte_to_dict(texte):
    texte_dict = {}

    for token in texte:
        if token in texte_dict:
            texte_dict[token] += 1
        else:
            texte_dict[token] = 1

    return texte_dict


def dict_to_list(texte_dict):
    texte_list=[]

    for mot in texte_dict.keys():
        texte_list.append([texte_dict[mot], mot])

    texte_list.sort(reverse=True)
    return texte_list


def afficher_n(texte_list, n):

    cumul = 0
    print("rang\tmot\tfrequence\tfrequence(Zipf)")
    print("-"*50)
    for _ in range(n):
        cumul += texte_list[_][0]
        print("{}\t{}\t{}\t\t{:.0f}".format(_+1, texte_list[_][1], texte_list[_]
[0], texte_list[0][0]/(_+1)))

    total = reduce(lambda x, y: x+y, [_[0] for _ in texte_list])
    prop = cumul/total*100

    print("-"*50)
    print("Ces {} mots représentent le {:0.2f}% du corpus".format(n, prop))


def plot_zipf(texte_list, log=False):
    pyplot.rcParams['figure.figsize'] = [15, 10]

    y = [_[0] for _ in texte_list]

    y_ = []
    for _ in range(len(texte_list)):
        y_.append(int(texte_list[0][0]/(_+1)))

    pyplot.plot(y, "-", label="Réelle")
    pyplot.plot(y_, "--", label="Approximation (Zipf)")
```

```
    if log:
        pyplot.yscale("log")
        pyplot.xscale("log")

    pyplot.legend()
    pyplot.title("Loi de Zipf (Brown Corpus)")
    pyplot.xlabel("Rang")
    pyplot.ylabel("Fréquence")
    pyplot.show()
```

Analyse du corpus

In [3]:

```
texte = load_brown_corpus()
print("Quantité des mots (tokens) :", len(texte))
print("Quantité des mots differentes (types) :", len(set(texte)))
```

```
Quantité des mots (tokens) : 1012528
Quantité des mots differentes (types) : 49398
```

In [4]:

```
print(texte[:50])
```

```
['the', 'fulton', 'county', 'grand', 'jury', 'said', 'friday', 'an',
'investigation', 'of', "atlanta's", 'recent', 'primary', 'election',
'produced', 'no', 'evidence', 'that', 'any', 'irregularities', 'too
k', 'place', 'the', 'jury', 'further', 'said', 'in', 'term-end', 'pr
esentments', 'that', 'the', 'city', 'executive', 'committee', 'whic
h', 'had', 'over-all', 'charge', 'of', 'the', 'election', 'deserve
s', 'the', 'praise', 'and', 'thanks', 'of', 'the', 'city', 'of']
```

In [5]:

```
texte_dict = texte_to_dict(texte)
```

In [6]:

```
mots = ["the", "of", "and", "i"]
print("mot\tfrequence")
for mot in mots:
    print("{}\t{}".format(mot, texte_dict[mot]))
```

```
mot     frequence
the     69971
of      36412
and     28853
i       5164
```

In [7]:

```
texte_list = dict_to_list(texte_dict)
```

In [8]:

```
afficher_n(texte_list, 20)
```

```
rang    mot     frequence       frequence(Zipf)
--------------------------------------------------
1       the     69971           69971
2       of      36412           34986
3       and     28853           23324
4       to      26158           17493
5       a       23195           13994
6       in      21337           11662
7       that    10594           9996
8       is      10109           8746
9       was     9815            7775
10      he      9548            6997
11      for     9489            6361
12      it      8760            5831
13      with    7289            5382
14      as      7253            4998
15      his     6996            4665
16      on      6741            4373
17      be      6377            4116
18      at      5372            3887
19      by      5306            3683
20      i       5164            3499
--------------------------------------------------
Ces 20 mots représentent le 31.08% du corpus
```

In [9]:

```
plot_zipf(texte_list[:135], log=False)
```