Text manipulations

Christophe@pallier.org

Sept. 2013

Contents

Download Alice in Wonderland.

(1) Write a program that prints the lines that contains the string 'Alice' (tip: you can use the find function from the module string). Then, test the same program with the strings 'Rabbit', 'rabbit', 'stone', 'office'.

---

(2) Here is a program that converts the text file into a list of words, removing the punctuation marks and converting everything in lower case. Run it.

```
import string
def remove_punctuation(text):
    punct = string.punctuation + chr(10)
    return text.translate(string.maketrans(punct, " " * len(punct)))

textori = file('alice.txt').read().lower()
text = remove_punctuation(textori)
words = text.split()
print(words)
```

Now write a script that counts the number of occurences of 'Alice', 'Rabbit' or 'office' in the list of words.

---

(3) Read about Python's Dictionnaries http://docs.python.org/2/tutorial/datastructures.html#dictionaries and use a dictonnary to store the number of occurrences of each word in Alice in Wonderland (the keys are the words, and the values and the number of occurrences; if word= ['a', 'a', 'b']; dico={'a':2, 'b':1}).

---

(4) Use numpy and matplotlib to plot the word log(frequencies) as a function of the rank of words on the abscissae (the most frequence word being ranked #1)

You can skim through http://matplotlib.org/users/pyplot_tutorial.html.

Remark: The product rank X frequency is roughly constant. This 'law' was discovered by Estoup and popularized by Zipf. See http://en.wikipedia.org/wiki/Zipf%27s_law.

(5) (advanced) Plot the relationship between word length and word frequency.

---

(6) Generate random text (each letter from a-z being equiprobable, and the spacecharacter being 8 times more probable) of 1 million characters. Compute the frequencies of each 'pseudowords' and plot the rank/frequency diagram.

---

(7) (advanced) compute the table of transition frequencies between words in Alice and generate random text following this pattern.

---

(8) Read about the MU Puzzle (http://en.wikipedia.org/wiki/MU_puzzle). Write a program that generates sequences of strings based on the following production rules and the initial state 'MI'

1. `xI -> xIU`
2. `Mx -> Mxx`
3. `xIIIy -> xUy`
4. `xUUy -> xy`

(Tip: use the function string.replace)