INF5830, 2017, Obligatory assignment 1

Deadline: Sept. 22, 18.00

To be delivered in devilry

Exercise 1 – Probabilities and Python (20%)

When we apply probabilities and statistics, we will mainly use packages with built-in functions. But to get a better understanding of what we are doing, it is good training to implement some of it ourselves. We will in this exercise use basic Python and not packages like math, numpy or scipy. We will then later on compare our implementations with these packages.

a) Implement a function factor(n) which returns the factorial of n when n is an integer ≥ 0. (i.e. f(n) = n! = 1*2*…*n, and f(0)=1.)
b) Implement a function binom(m, n) which to two integers where $n \geq m \geq 0$ returns $\binom{n}{m}$.
c) Implement a function binom_pmf(k, n, p) where k and n are integers, where $n \geq k \geq 0$, and p is a real $1 \geq p \geq 0$. The function returns the probability mass function at k for the binomial distribution of n individuals with probability p, i.e. b(k; n, p).
d) Implement a function binom_cdf(k, n, p) with the same arguments which returns the value of the cumulative distribution function at k.
e) Fix n=8 and p=1/2 and calculate the pmf and cdf for k=0, 1, 2, …, 8. This corresponds to flipping a fair coin 8 times. Report the numbers in a table and as a bar chart.
f) Repeat similarly for n=5 and p=1/6. This corresponds to throwing a fair dice and counting 6s as success.

Deliveries: A python file with the functions from points (a-d). The tables and charts asked for in (e) and (f).

Exercise 2 – Python library: random (15%)

Computers are deterministic. They do not act randomly. However, they can simulate randomness and act so-called pseudo randomly. There is a module *random* in the standard library with several useful functions.

a. In particular, the function random.random() returns a real number between 0 and 1 with uniform probability. We can use this e.g. as follows.

```
def bernoulli(p):
    if random.random() < p:
        return 1
    else:
        return 0
```

What does this function simulate? Run bernoulli(0.5) 10 times and record the results.

b. We will then see what happens when we perform n Bernoulli trials (flip the coin or throw the dice n times). Make a function bin_exper(n, p) which performs n random Bernoulli experiments with probability p and return the number of successes. Run bin_exper(10, 0.5) ten times and report the results.
c. We will inspect the effect of running an experiment many times and taking the averages. Make a function bin_freqs(m, n, p) which runs bin_exper(n, p) m many times and returns the relative frequencies of k successes for k = 0, 1, …, n.
d. Fix p=0.5 and n=8 and see what happens when m varies. Run bin_freqs for m= 4, 10, 100, 1000, 10000. Report the results in a (6*9) table where you also include the values for the theoretical distribution binom_pmf(k, n, p) from exercise (1)
e. To familiarize yourself a little more with random try the following

```
>>> a = range(100)
>>> random.choice(a)
>>> random.choice(a)
>>> random.sample(a,10)
>>> random.sample(a,10)
>>> random.shuffle(a)
>>> a
```

Make sure you understand the commands. (No deliveries at this point).

Deliveries: A file with the functions from (b) and (c). Answer the question in (a). The results of the computations in (a), (b) and (d), where the results from (d) should be presented in a table as explained.

Exercise 3 – Conditional frequency distributions (35%)

The NLTK book, chapter 2, has an example in section 2.1 in the paragraph Brown Corpus where they compare the use of modals across different genres. We will conduct a similar experiment, but we will instead inspect the differences in gender. We are in particular interested in to which degree the different genres use the masculine pronouns (he, him) or the feminine pronouns (she, her).

a. Conduct a similar experiment as the one mentioned above with the genres: *news, religion, government, fiction, romance* as conditions, and the words: *he, she, her, him*. Make a table and deliver code and table.

Maybe not so surprisingly, the masculine forms are more frequent than the feminine forms. But we also observe another pattern. The relative frequency of *her* compared to *she* seems higher than the relative frequency of *him* compared to *he*. We want to explore this further and make a hypothesis which we can test.

Ha: The relative frequency of the object form, *her*, of the feminine personal pronoun (*she* or *her*) is higher than the relative frequency of the object form, *him*, of the masculine personal pronoun, (*he* or *him*).

b. First, consider the complete Brown corpus. Construct a conditional frequency distribution, which uses gender as condition, and for each gender count the occurrences of subjective and objective forms. Report the results in a two by two table. Then calculate the relative frequency of *her* from *she* or *her*, and compare to the relative frequency of *him* from *he* or *him*. Report the numbers. Submit table, numbers and code you used.

c. It is tempting to conclude from this that the object form of the feminine pronoun is relatively more frequent than the object form of the male pronoun. But beware, *her* is not only the feminine equivalent of *him*, but also of *his.* So what can we do? The simplest thing we may do is also to count the occurrences of *his* to get a better idea of the distribution. Deliver: The absolute frequency of the six words *she, he, her, him, his, hers* in the Brown corpus.

d. We could do a similar calculation as in point (b), comparing the relative frequency of *her* not to the relative frequency of *him* but to *him* + *his.* But that does not help us in checking the hypothesis, Ha. We can check the hypothesis with the help of a tagged corpus if the corpus tags *her* as a personal pronoun differently from *her* as a possessive pronoun. The tagged Brown corpus with the full tag set does that. Use this to count the occurrences of *she, he, her, him* as personal pronouns and *her, his, hers* as possessive pronouns. See NLTK book, Ch. 5, Sec. 2, for the tagged Brown corpus. Report in a two-ways table.

e. We can now correct the numbers from point (b) above. How large percentage of the feminine personal pronoun occurs in subject form and in object form? What are the comparable percentages for the masculine personal pronoun? (We will later on discuss whether these numbers are statistically significant and how we can check for that.)

f. Illustrate the numbers from (e) with a bar chart.

g. Answer in 4-10 sentences: What do you think this reveals about language and culture?

h. Answer in 6-10 sentences: Do you see any ethical problems in constructing NLP systems based on data from actual language use?

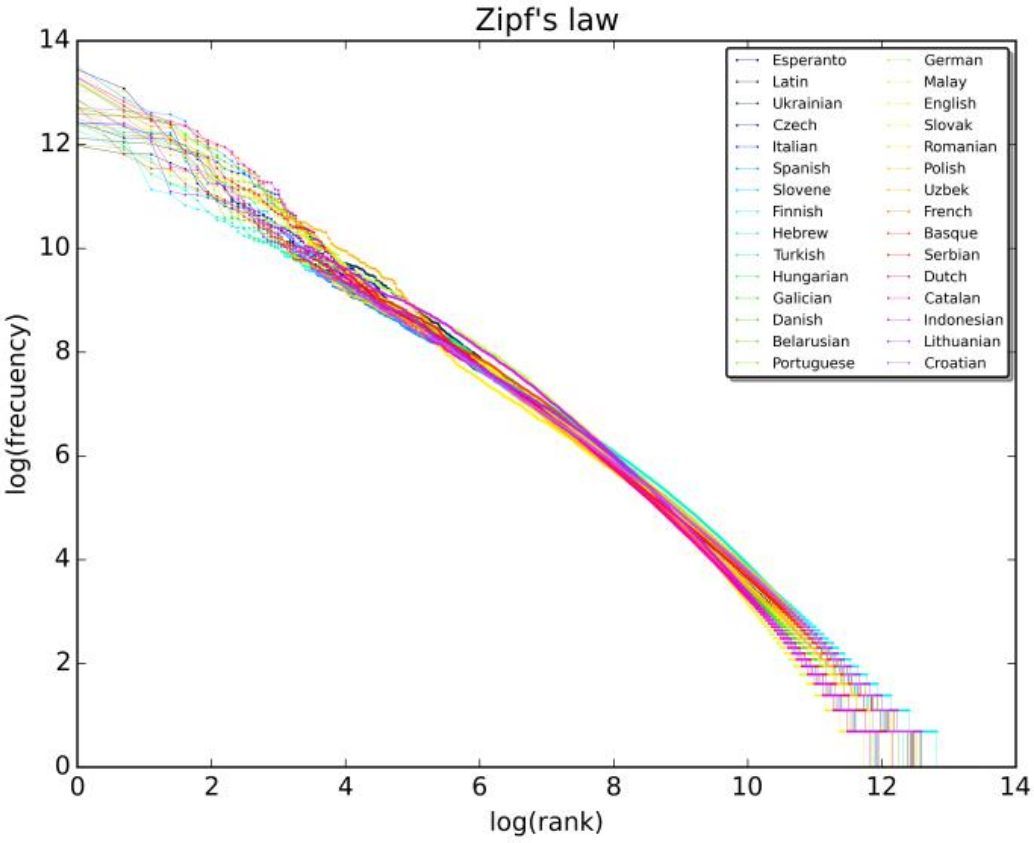
Exercise 4 – Downloading texts and Zipf's law (30%)

In this exercise we will consider Zipf's law which is explained in exercise 23 in NLTK chapter 2, and more thoroughly in the Wikipedia article: *Zipf's law*, which you are advised to read. We will use the text *Tom Sawyer* as that is used also in the book *Foundations of Statistical Natural Language Processing* for studying Zipf's law.

a. First, you need to get hold of the text. It can be downloaded from project Gutenberg as explained in section 1 in chapter 3 in the NLTK book.
b. Then you have to do some clean up. For example, there might be additional headers in the text which are not part of the text itself.
c. You can then extract the words. Explain the steps you take here and in point (b) above. (You may use nltk.word_tokenize().)

d. Use the nltk.FreqDist() to count the words. Report the 20 most frequent words in a table with their absolute frequencies.
e. Consider the frequencies of frequencies. How many words occur only 1 time? How many words occur n times, etc. for n = 1, 2, …, 10; how many words have between 11 and 50 occurrences; how many have 51-100 occurrences; and how many have more than 100 occurrences? Report in a table!
f. Let r be the frequency rank for each word and n its frequency. According to Zipf's law r*n should be nearly constant. Calculate r*n for the 20 most frequent words and report in a table? How well does this fit Zipf's law?
g. Try to plot the frequency of frequencies. First use the actual numbers on the axis, i.e. not logarithmic scale. Then try to make a plot similarly to the Wikipedia figure below. They have used logarithmic scale at both axis. Logarithms are available in numpy, using np.log().

Deliveries: Explanation of the steps you took in (b) and (c). The tables asked for in (d) and (e). The table in (f), the plots in (g) and try to answer the question in (f) in words.



(source: Wikipedia)

The END