Introduction to Web Science

Assignment 6

Prof. Dr. Steffen Staab
staab@uni-koblenz.de

René Pickhardt
rpickhardt@uni-koblenz.de

Korok Sengupta
koroksengupta@uni-koblenz.de

Institute of Web Science and Technologies
Department of Computer Science
University of Koblenz-Landau

Submission until: December 6, 2016, 10:00 a.m.
Tutorial on: December 9, 2016, 12:00 p.m.

Please look at the lessons 1) **Simple descriptive text models** & 2) **Advanced descriptive text models**

For all the assignment questions that require you to write code, make sure to include the code in the answer sheet, along with a separate python file. Where screen shots are required, please add them in the answers directly and not as separate files.

Team Name: papa
Members: Brigitte Aznar
Bonasmitha Behura
Ilia Tugushi

1 Digging deeper into Norms (10 points)

You have been introduced to the concept of a norm and have seen that the uniform norm $||\cdot||_\infty$ fullfills all three axioms of a norm which are:

1. Positiv definite
2. Homogeneous
3. Triangle inequality

Recall that for a function $f : M \longrightarrow \mathbb{R}$ with M being a finite set[1] we have defined the L_1-norm of f as:

$$||f||_1 := \sum_{x \in M} |f(x)| \tag{1}$$

In this exercise you should

1. calculate $||f-g||_1$ and $||f-g||_\infty$ for the functions f and g that are defined as
   - $f(0) = 2, f(1) = -4, f(2) = 8, f(3) = -4$ and
   - $g(0) = 5, f(1) = 1, g(2) = 7, g(3) = -3$

   for: $||f-g||_\infty$
   $||2-0|| = 2$
   $||-4-1|| = 5$
   $||8-7|| = 1$
   $||-4+3|| = 1$

   for: $||f-g||_1$
   $\sigma||f-g||_\infty$
   $2+5+1+1 = 9$

2. proof that all three axioms for norms hold for the L_1-norm.

   Positive definitive:
   $||f||_1 = \Sigma|f(x)|$
   $||f||_1 = 0 <=> \Sigma|f(x)| = 0$
   $|f(x)| = 0 \forall x => f = 0$

   Homogeneous
   $||\alpha f||_1 = \alpha||f||_1$
   $\alpha||f||_1 = \Sigma|\alpha f|$
   $\Sigma|\alpha f|_1 = \Sigma|\alpha||f|$

[1] You could for example think of the function measuring the frequency of a word depening on its rank.

$\Sigma|\alpha||f| = |\alpha|\Sigma|f|$
$|\alpha|\Sigma|f| = \alpha||f||_1$

Triangle inequity
$||f+g||_1 \leq ||f||_1 + ||g||_1$
$||f+g||_1 \leq \Sigma|f+g|$
$||f+g||_1 \leq \Sigma|f| + |g|$
$||f+g||_1 \leq \Sigma|f| + \Sigma|g|$
$||f+g||_1 \leq ||f||_1 + ||g||_1$
$||f+g||_1 \leq ||f+g||_1$

1.1 Hints:

1. The proofs work in a very similar fashion to those from the uniform norm that was depicted in the videos.
2. You can expect that the proofs for each property also will be "three-liners".
3. Both parts of this exercise are meant to practice proper and clean mathematical notation as this is very helpfull when reading and understanding research papers. Discuss in your study group not only the logics of the calculation and the proof (before submission) but try to emphasize on the question whether your submission is able to communicate exactly what you are doing.

2 Coming up with a research hypothesis (12 points)

You can find all the text of the articles from Simple English Wikipedia at `http://141.26.208.82/simple-20160801-1-article-per-line.zip` each line contains one single article.

In this task we want you to be creative and do some research on this data set. The ultimate goal for this exercise is to practice the way of coming up with a research hypothesis and testable predictions.

In order to do this please **shortly**[2] answer the following questions:

1. What are some observations about the data set that you can make? State at least three observations.
   a) 7 out of the 10 first articles have the Word "is" as the 2nd word of the articles.
   b) Simple English Wikipedia has short articles.
   c) Most of the articles contain less than 15 sentences.
2. Which of these observations make you curious and awaken your interest? Ask a question about why this pattern could occur.
   `"Simple English Wikipedia has short articles"`
   Can it be read in a short amount of time?
3. Formulate up to three potential research hypothesis.
   a) `It's possible to read Simple English Wikipedia in 30 days.`
   It may take longer to read, based on the number of words and the avg human reading speed.
   b) `More than 50 % of the articles in Simple English Wikipedia have less than 15 sentences.`
   Count the number of sentences and then calculate the percentage of articles for which this is true.
4. Take the most promising hypothesis and develop testable predictions.
   `It's possible to read Simple English Wikipedia in a 30 days time.`
5. Explain how you would like to use the data set to test the prediction by means of descriptive statistics. Also explain how you would expect your outcome.
   a) Count all the words in the data set by using regular expression (r'\w+'). Being a word (consecutive alphanumeric characters)
   b) Calculate how much time does an average reader needs to read the entire data set

[2] Depending on the question shortly could mean one or two sentences or up to a thousand characters. We don't want to give a harsh limit because we trust in you to be reasonable.

c) Calculate how much time does a speedy reader needs to read the entire data set

(If you realize that the last two steps would not lead anywhere repeat with one of your other research hypothesis.)

2.1 Hints:

- The first question could already include some diagrams (from the lecture or ones that you did yourselves).
- In step 3 explain how each of your hypothesis is falsifiable.
- In the fifth step you could state something like: "We expect to see two diagrams. The first one has ... on the x-axis and ... on the y-axis. The image should look like a ... The second diagram ...". You could even draw a sketch of the diagram and explain how this would support or reject your testable hypothesis.

3 Statistical Validity (8 points)

In the above question, you were asked to formulate your hypothesis. In this one, you should follow your own defined roadmap from task 2 validate (or reject) your hypothesis.

3.1 Hints:

- In case feel uncomfortable to test one of the predictions from task 2 you can "steal" one of the many hypothesis (and with them imlicitly associated testable predictions) or diagrams depicted from the lecture and reproduce it. However in that case you cannot expect to get the total amount of points for task 3.

```
1:
2: # coding: utf-8
3:
4: # # It's possible to read Simple English Wikipedia in 30 days time
5:
6: # Avarage reading speed is 180 Words per minute (in a monitor)
7: # source: https://en.wikipedia.org/wiki/Words_per_minute#Reading_and_comprehensio
8: # Speed readers can read about 700 words per minute
9: # source: https://en.wikipedia.org/wiki/Speed_reading
10: #
11: # Given this: Count the total number of words in Simple English Wikipedia
12: # and calculate how long it will take for each to finish reading it.
13: #
14: # Since the initial observation states that Simple English Wikipedia articles are
15: # short, counting this words, and taking this speeds into consideration.
16: # We calculated that 30 days is a plausible time for doing this task.
17:
18: # In[43]:
19:
20: import os
21: import re
22: import matplotlib.pyplot as plt
23: import matplotlib.patches as mpatches
24:
25:
26: # In[8]:
27:
28: def get_total_words():
29:     total_words = 0
30:     with open("simple-20160801-1-article-per-line\simple_english_wikipedia.txt",
31:         for line in file:
32:             total_words += len(re.findall(r'\w+', line))
33:     return total_words
34:
35:
```

```
# In[9]:

def get_avg_time(words):
    avg_speed = 180
    return ((words/avg_speed)/60)/24



# In[10]:

def get_speedy_time(words):
    speedy_speed = 700
    return ((words/speedy_speed)/60)/24


# In[11]:

def get_wpm(words):
    time = 43200 #30 days in minutes
    return (words/time)


# In[12]:

words       = get_total_words()
avg_time    = get_avg_time(words)
speedy_time = get_speedy_time(words)

print("Total number of words: ", words)
print("Total time (in days) for an average reader: ", round(avg_time,2))
print("Total time (in days) for a speed reader: ", round(speedy_time,2))


# It is not possible to read simple english wikipedia within 30 days (for a normal
# However it is possible to do it for a speed reader.

# In[14]:

wpm = get_wpm(words)
print("In order to be able to read single english wikipedia within 30 days, a pers


# In[54]:

y = [round(avg_time, 2),round(speedy_time,2)]
x = len([round(avg_time, 2),round(speedy_time,2)])
width = 1/1.5
```

```
85: plt.bar(1, round(avg_time, 2), width, color="blue")
86: plt.bar(2, round(speedy_time,2), width, color="red")
87: plt.ylabel('Time in days')
88: plt.xlabel('Readers')
89:
90:
91: avg_reader = mpatches.Patch(color='blue', label='Averge reader')
92: speedy_reader = mpatches.Patch(color='red', label='Speed reader')
93:
94: plt.legend(handles=[avg_reader, speedy_reader],loc=5)
95: plt.show()
```

According to our findings, it is not possible to read Simple English Wikipedia in 30 days for an average reader. However it is possible for a speed reader to accomplish this task In the following plot, we can see how the two types of reader compare to each other.

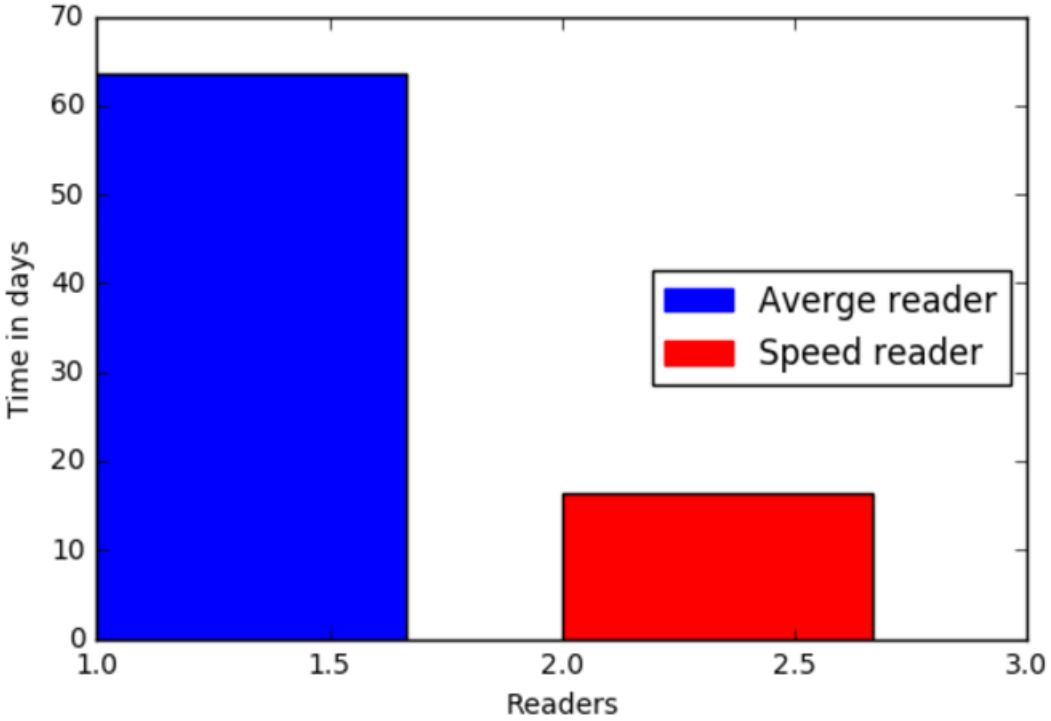

Figure 1: Shows difference in time between average reader and speed reader

Important Notes

Submission

- Solutions have to be checked into the github repository. Use the directory name `groupname/assignment6/` in your group's repository.
- The name of the group and the names of all participating students must be listed on each submission.
- Solution format: all solutions as *one* PDF document. Programming code has to be submitted as Python code to the github repository. Upload *all* `.py` files of your program! Use `UTF-8` as the file encoding. *Other encodings will not be taken into account!*
- Check that your code compiles without errors.
- Make sure your code is formatted to be easy to read.
  - Make sure you code has consistent indentation.
  - Make sure you comment and document your code adequately in English.
  - Choose consistent and intuitive names for your identifiers.
- Do *not* use any accents, spaces or special characters in your filenames.

Acknowledgment

This latex template was created by Lukas Schmelzeisen for the tutorials of "Web Information Retrieval".

LaTeX

Currently the code can only be build using LuaLaTeX, so make sure you have that installed. If on Overleaf, there's an error, go to settings and change the LaTeXengine to `LuaLaTeX`.