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.

Group name: uniform
Group members: Pradip Giri, Jalak Arvind Kumar Pansuriya, Madhu Rakhal Magar

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
   - $g(0) = 5, f(1) = 1, g(2) = 7, g(3) = -3$
2. proof that all three axioms for norms hold for the L_1-norm.

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.

Answer:
1.1 We know that for a function $f : M \longrightarrow \mathbb{R}$ with M being a finite set, we have defined the L_1-norm of f as:

$$||f||_1 := \sum_{x \in M} |f(x)| \tag{2}$$

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

So, in our case:

$$||f-g||_1 := \sum_{x \in M} |f(x) - g(x)| \tag{3}$$

= |f(0) - g(0)| + |f(1) - g(1)| + |f(2) - g(2)| + |f(3) - g(3)|
= |2 - 5| + |-4 - 1| + |8 -7| + |-4 - -3|
= 3 + 5 + 1 + 1
= 10

Also,

$$||f-g||_\infty := \sum_{x \in M} max(|f(x) - g(x)| : 0...M) \tag{4}$$

= max(|f(0) - g(0)| , |f(1) - g(1)| , |f(2) - g(2)| , |f(3) - g(3)|)
= max(3, 5, 1, 1)
= 5

Hence, we got

$$||f-g||_1 := 10 \tag{5}$$

and

$$||f-g||_\infty := 5 \tag{6}$$

1.2 Here is the solution:

1.2

For a function $f: M \rightarrow R$ with M being a finite set, we can defined L_1-norm of f as:

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

To prove all three axioms for norms holds for L_1-norm, Firstly take:

(1) Positive Definite $(||f||_1 = 0 \Rightarrow f = 0)$

Proof:

$$||f||_1 = 0 \Longleftrightarrow \sum_{x \in M} |f(x)| = 0$$

$$\Rightarrow |f(x)| = 0 \,\forall x$$

$$\Rightarrow f(x) = 0 \,\forall$$

$$\Rightarrow f = 0$$

proved//

(2) Homogeneous $(||\alpha f||_1 = \alpha ||f||_1 \,;\, \alpha \in R)$

Proof:

$$||\alpha f||_1 = \sum_{x \in M} |\alpha f(x)|$$

$$= |\alpha| \sum_{x \in M} |f(x)|$$

$$= |\alpha| \, ||f||_1 \quad \because \alpha \in R$$

proved

③ Triangle Inequality $(||f+g||_1 \leq ||f||_1 + ||g||_1)$

Proof:

$$||f+g||_1 = \sum_{x \in M} | f(x) + g(x) |_1$$

$$\leq \sum_{x \in M} |f(x)| + \sum_{x \in M} |g(x)|$$

$$\leq ||f||_1 + ||g||_1$$

$$= ||f||_1 + ||g||_1$$

Proved

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 obervations about the data set that you can make? State at least three obervations.
2. Which of these observations make you curious and awaken your interest? Ask a question about why this pattern could occur.
3. Formulate up to three potentiel research hypothesis.
4. Take the most promesing hypothesis and develop testable predictions.
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.

   (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.

Answer:
2.1 Here is some observations that we made:

1. Each article in simple wiki is separated by newline.
2. Each article in simple wiki is from diverse field and describes about some particular unique topic.

[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.

3. Each article in simple wiki is informative.
   (Assumption: An informative article informs the reader by explaining and giving details on a given topic. The informative article is not an argumentative article that tries to persuade the reader to one side or the other. It covers all the pertinent details: who, what, when, where and why.)

4. Simple wiki consist of certain number of foreign(loan) words.
   (Foreign(loan) words: is a word used directly from another language with little or no translation.)

2.2 ”Simple wiki consist of certain number of foreign(loan) words.” This observation make us curious.
Is this really true?
Each article tells us about one particular topic and gives the background of it.
The description of article provides specific information, data, facts, definition of terms, date, origin of words which are organized and presented within same context.
So there must be use of some foreign words in the article

2.3 Here are three potential research hypothesis:

1. The simple wiki consists of at least 10% of foreign words(or loan words).
2. Each article in simple wiki is informative.
3. Each article in simple wiki consists of at least 200 words.

Here, we can say that hypothesis is falsifiable if we got the counter example or counter facts for it.
For the first hypothesis, It will be falsifiable if we got less than 10% of foreign words no foreign words after doing the research.
For second hypothesis it will be falsifiable if there is no any kind of data, facts,date etc in article. Also, If it contains dummy text.
For third hypothesis we can say that it is falsifiable if any article in simple wiki is less than 200 words.

2.4 We found that ”The simple wiki consists of at least 10% of foreign words(or loan words).” is the most promising hypothesis.

Testable Predictions:

1. Each article comes from diverse field and talks about unique particular topic.
2. The articles cover older information, usually from the beginning to a historic ancient

times.Which are from different parts of the world. Our assumption is ”to describe the diversified topics simple wiki use some foreign words.”

This means we have to find out the foreign words in English and compare these words with the words in the simple wiki.
-Example of counting the words.

2.5 Firstly, We are going to get the all foreign words in English from Wikipedia. After that we compare each words in an article with foreign words that we have. We will save number of foreign words in an article. After that we repeat the comparison process for all article.
We expect to see two diagrams. the first one is Cumulative Frequency Diagram(CDF) which has % of foreign words in y-axis and no.of article in x-axis. If this diagram shows the number of foreign words is greater than 10% then our hypothesis will be proved.
The second diagram will be a scatter plot where we plot the foreign words and total other words in the article.
We will also calculate the mean, median and percentage of foreign words in simple wiki.

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.

Answer:
Observation:
After reading first few articles we observe that some of the word are from medieval period and early ages with historical background. And found some of the word used by Roman period which suggest us that there must be foreign English word which are borrowed from other languages. We observed there is at least 10% words which are loaned to English language from other language.

Thinking of Interesting Questions:
We are interested in finding how other language help(influence) in development of English language? Are they directly used in English Language or with some modifications?.What is the percentage of foreign words in simple wiki?

Hypothesis Formulation:
The simple wiki consists of at least 10% of foreign words(or loan words).

Development of Testable Predictions:
If our hypothesis is correct, We will get at least 10% of loanwords in simple wiki.

Data Collection:
First we collect all the required loaned word from `https://en.wikipedia.org/wiki/Lists_of_English_words_by_country_or_language_of_origin` Wikipedia and clean it and make it unique.

Test Prediction:
We load all the articles and select only article having more than 15 words because we think an article must have minimum 15 words. And we check each and every word of the given article against our set of English loaned word. During comparison we trim the word so that space will not effect our comparison.

Test Result:
In the process we use 12513 unique loan(foreign) words. After calculation we found that the mean of the loaned word in the given articles is 18.439. Which means every article have 18 loaned word or There is 18.43% loanwords in the given simple wiki file. The median of the calculation is 8.0, which means 50% of the article have less or equal to 8 loaned word and 50 % of the article have more than 8 loan words.

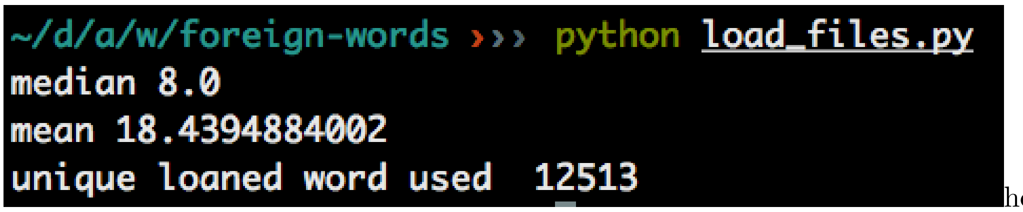
Listing 1 download.py

```python
# pylint: disable-msg=C0103
""" Downloads all the list form given url"""
from urllib.request import urlopen
from urllib.error import HTTPError
from urllib.parse import urlparse
from bs4 import BeautifulSoup

soup = None
restricted_names = ["place names", "references",
    "see also",
    "Common in South African English",
    "External links"]

def save_content(file_name, content):
    """ save the file with given name and givn content """
    with open(file_name, 'w+') as f:
        f.write(content)
    print('{} written sucessfully'.format(file_name))


def download_url(input_url):
    """ returns html content if request is successful and None for other
    responses"""
    try:
        html = urlopen(input_url)
    except HTTPError:
        return None
    return html


def find_inside_given_id(soup, selector):
    """ find inside the given class"""
    uls = soup.select(selector)
    borrowed_word = set()
    for ul in uls:
        lis = ul.select("li")
        print(lis)
        for li in lis:
            a = li.select("a")[0]
            borrowed_word.add(a.text)

    return borrowed_word


def find_next_to_selector(soup, selector):
    """ find next to the given class name """
    spans = soup.select(selector)
    borrowed_word = set()
    for span in spans:
        if span.string.lower() not in restricted_names:
            uls = span.parent.findNext("ul")
            lis = uls.select("li")
            for li in lis:
                found_workd = li.find("a").text
                borrowed_word.add(found_workd)

            dls = span.parent.findNext("dl")
            if dls is not None:
                dts = dls.select("dt")
                for dt in dts:
                    atags = dt.select("a")
                    for a in atags:
                        borrowed_word.add(a.text)
    return borrowed_word


def process_html(input_html):
    """ process the downloaded html"""
    soup = BeautifulSoup(input_html, 'html.parser')
    spans = soup.select('span.mw-headline')
    length = len(spans)
    print(length)
    if length > 2:
        return find_next_to_selector(soup, 'span.mw-headline')
    else:
        return find_inside_given_id(soup, '#mw-content-text')


def extract_file_name(input_url):
    """returns the required file name extracting from url"""
    path = urlparse(input_url).path
    return path.split('/')[-1].strip() + '.txt'


def main(input_url):
    """ Entry of the programme """
    global restricted_names
    restricted_names = list(map(lambda x: x.lower().strip(), restricted_names))
    html = download_url(input_url)
    if html is not None:
        print("Processing {}".format(input_url))
        html_content = html.read().decode()
        words = list(process_html(html_content))
        file_name = extract_file_name(input_url)
        save_content(file_name, str(words))


```

```
if __name__ == "__main__":
    url = 'https://en.wikipedia.org/wiki/List_of_English_words_of_French_origin_(
    main(url)
```

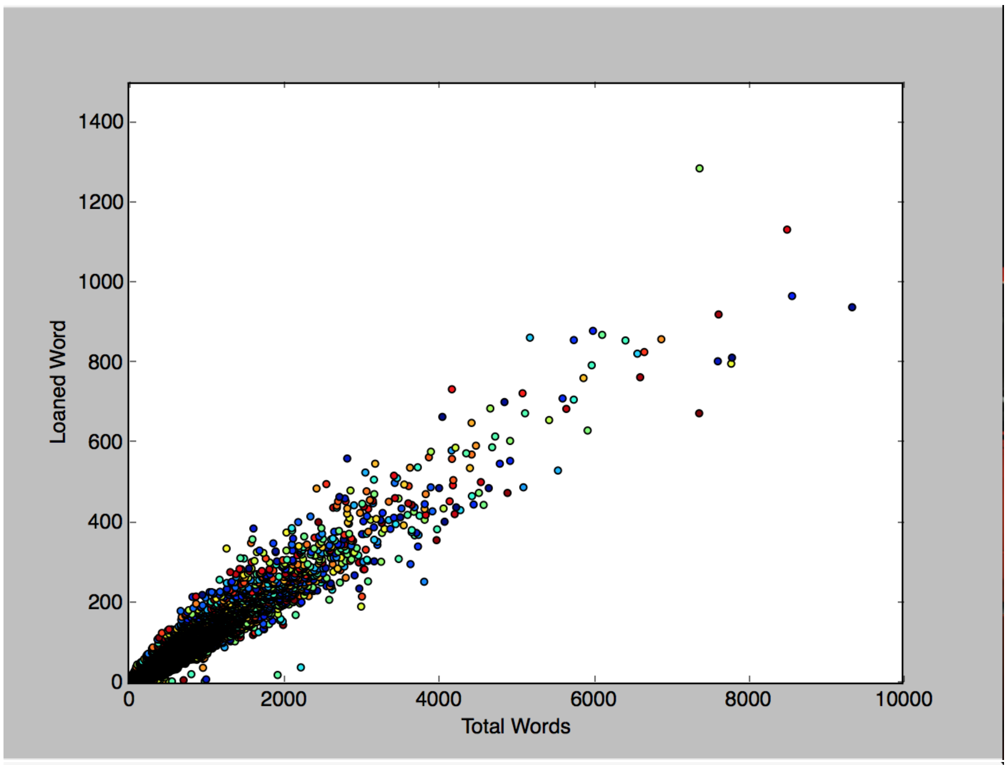
Listing 2 assignment6.py

```
"""
How to use it
first run the script
python3 assignment6.py
if you have already run script before you can only plot by using
python3 assignment6.py plot
"""
import os
import sys
import glob
import numpy as np
import matplotlib.pyplot as plt


path = os.getcwd() + '/' + 'foreign-words'
article_file = os.getcwd() + '/' + 'simple-20160801-1-article-per-line'
all_foreign_words = set([])

# this is list wich will contain number of foreign words
artile_infos_with_foreign_words = []

def open_file(file_name):
    file_content = None
    with open(file_name, 'r+') as f:
        file_content = f.read()
    return file_content

def process_foreign_text_file(file_name):
    file_content = open_file(file_name)
    list_of_words = eval(file_content)
    for word in list_of_words:
        all_foreign_words.add(word.strip())

def process_articles(articles):
    all_articles = articles.split('\n')
    for id, article in enumerate(all_articles):
        words_arr = article.split(" ")
        #article having less than 15 words are discarded
        if len(words_arr) > 15:
            found_world = 0
            for word in words_arr:
                if word.strip() in all_foreign_words:
                    found_world += 1
            artile_infos_with_foreign_words.append({"id": id, "loan_word": found_

def mean(arr):
    return np.mean(arr)

def median(arr):
    return np.median(arr)


def plot_graph(data):
    num_bins = 100
    counts, bin_edges = np.histogram(data, bins=num_bins, normed=True)
    cdf = np.cumsum(counts)
    plt.plot(bin_edges[1:], cdf)
    plt.xlabel('Article')
    plt.ylabel('Loaned World %')
    plt.show()


def main():
    all_txt_files = glob.glob(path + '/*.txt')
    for text_file in all_txt_files:
        process_foreign_text_file(text_file)
    artile_content = open_file(article_file)
    process_articles(artile_content)
    with open("result.txt", "w+") as f:
        sorted_list = sorted(artile_infos_with_foreign_words, key=lambda info: in
        f.write(str(sorted_list))
    arr = list(map(lambda x: (x['total_word'], x['loan_word']), artile_infos_with_
    arr = list(map(lambda x: x[1], arr))
    print("median", median(arr))
    print("mean", mean(arr))
    print("unique loaned word used ", len(all_foreign_words))
    process_diagram()


def draw_scatter_plot(arr):
    total_word = [x[0] for x in arr]
    foreign_word = [x[1] for x in arr]
    N = len(total_word)
    colors = np.random.rand(N)
    plt.scatter(total_word, foreign_word, s=15, c=colors, alpha=1)
    plt.ylabel('Loaned Word')
    plt.xlabel('Total Words')
    plt.xlim(0, 10000)
    plt.ylim(0, 1500)
    plt.show()

def draw_histogram(data):
    plt.hist(data, bins= np.arange(0, 1300, 5), normed=True)
    plt.ylabel('Loaned Word Frequency')
    plt.xlabel('Number of Loaned Word')
    plt.show()

def process_diagram():
    raw_data = None
    with open("result.txt", "r+") as f:
        raw_data = eval(f.read())
    if raw_data:
        arr = list(map(lambda x: (x['total_word'], x['loan_word']), raw_data))
        plot_graph(arr)
        draw_scatter_plot(arr)
        mod_arr = list(map(lambda x: x[1], arr))
        draw_histogram(mod_arr)

if __name__ == "__main__":
    plot = None
    try:
        plot = sys.argv[1]
    except:
        pass
    if plot == "plot":
        process_diagram()
    else:
        main()
```



```
~/d/a/w/foreign-words ››› python load_files.py
median 8.0
mean 18.4394884002
unique loaned word used  12513
```

hello


Loaned Word
Total Words
0
200
400
600
800
1000
1200
1400
0
2000
4000
6000
8000
10000

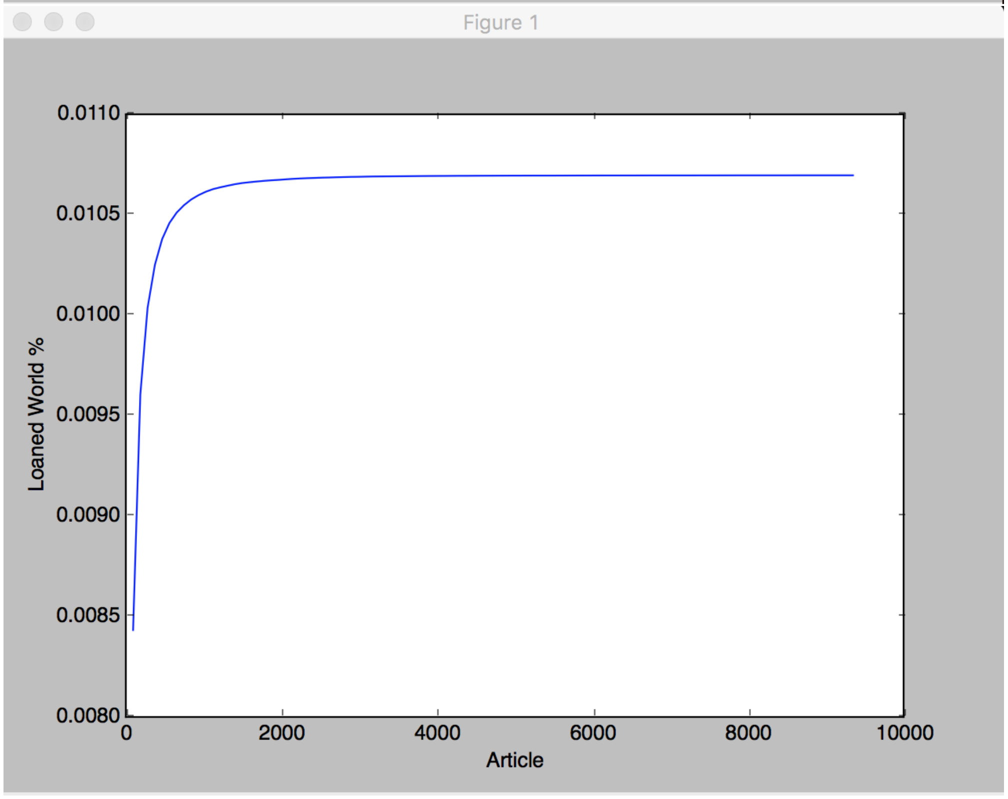
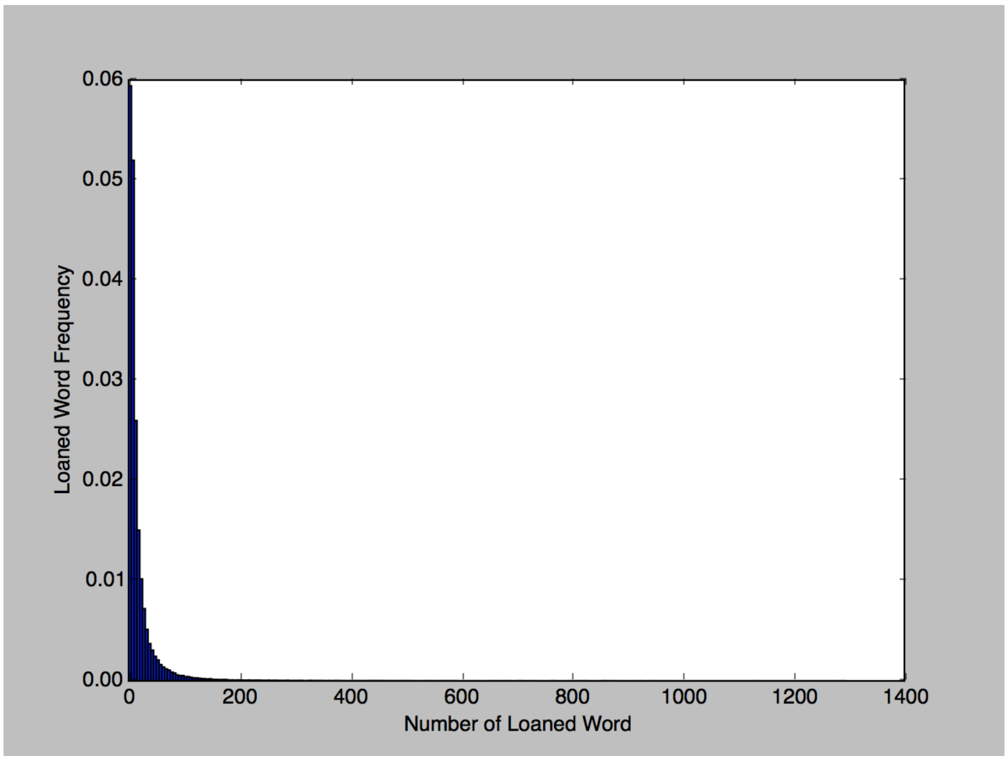


Figure 1
Loaned World %
Article
0.0080
0.0085
0.0090
0.0095
0.0100
0.0105
0.0110
0
2000
4000
6000
8000
10000



Conclusion
From calculated mean, median and the graph(CDF, histogram and scatter plot) we found that simple wiki consist of at least 10% of the loaned English word.

Problem faced

1. Data collection
   During collection the list of loaned word. We crawl the wikidedia page but the pages have different HTML structure which take a lot of modification in the downloading script for specific page.

2. Length of Paragraph
   We assume the separation of the articles are by
   n and while processing the articles some are only blank lines and some of the articles have only few words like 3 and 7, 9 etc so we remove the blank lines and only takes the articles having more than 15 words.

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.

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`.