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 : Yankee
Candidates :

1. Sabin Bhattarai - 216203590
   sbhattarai@uni-koblenz.de
2. Biplov K.C. - 216203865
   biplov@uni-koblenz.de
3. Syed Salman Ali. - 216203923
   salmanali@uni-koblenz.de

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. Positive 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.

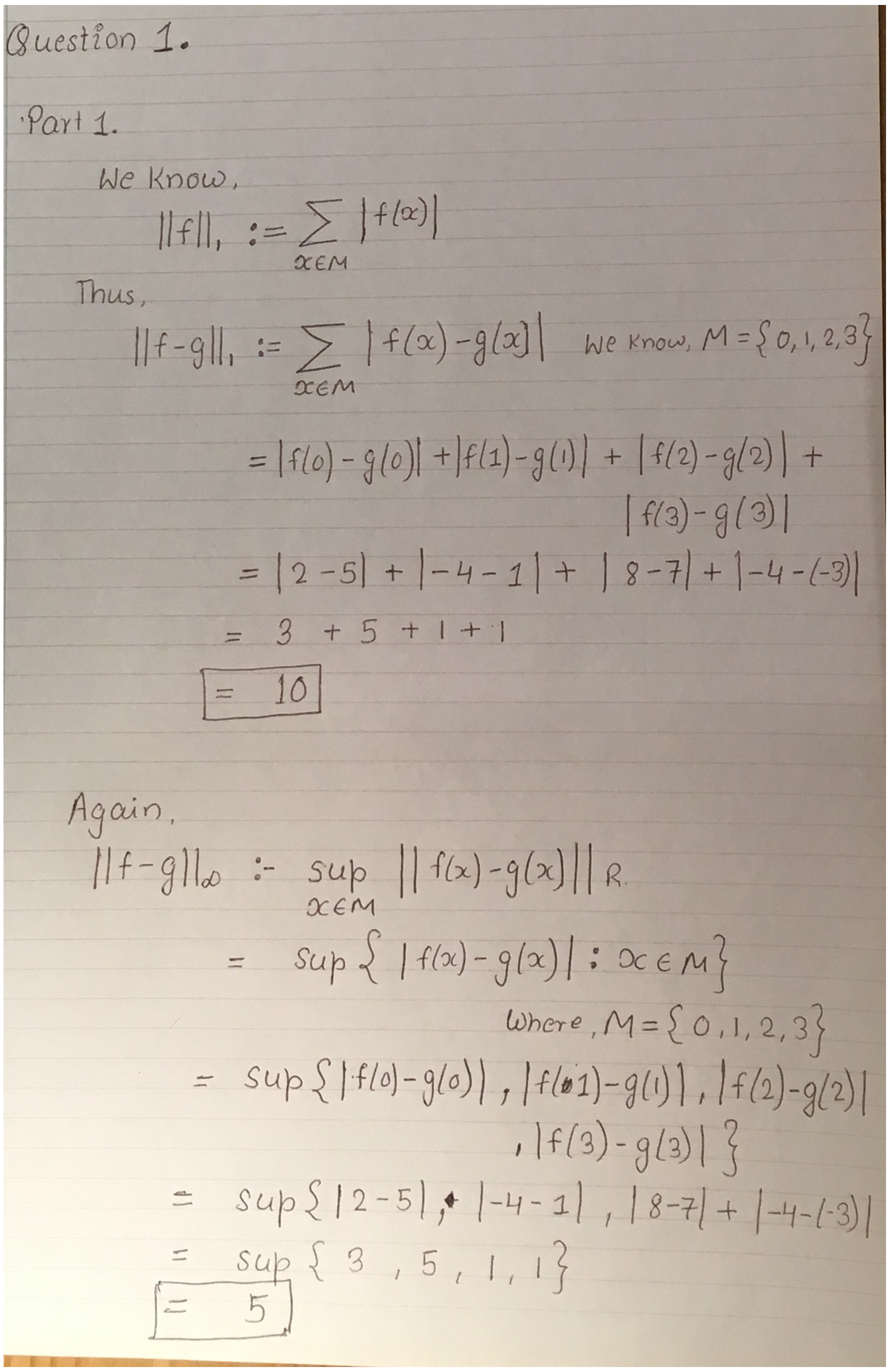
1.1 Hints:

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.

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

Question 1

Part 1



Question 1.

Part 1.

We know,

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

Thus,

$$\|f-g\|_1 := \sum_{x \in M} |f(x)-g(x)| \quad \text{We know, } M=\{0,1,2,3\}$$

$$= |f(0)-g(0)| + |f(1)-g(1)| + |f(2)-g(2)| + |f(3)-g(3)|$$

$$= |2-5| + |-4-1| + |8-7| + |-4-(-3)|$$

$$= 3 + 5 + 1 + 1$$

$$\boxed{= 10}$$

Again,

$$\|f-g\|_\infty :- \sup_{x \in M} \|f(x)-g(x)\|_R$$

$$= \sup\{|f(x)-g(x)| : x \in M\}$$

Where, $M=\{0,1,2,3\}$

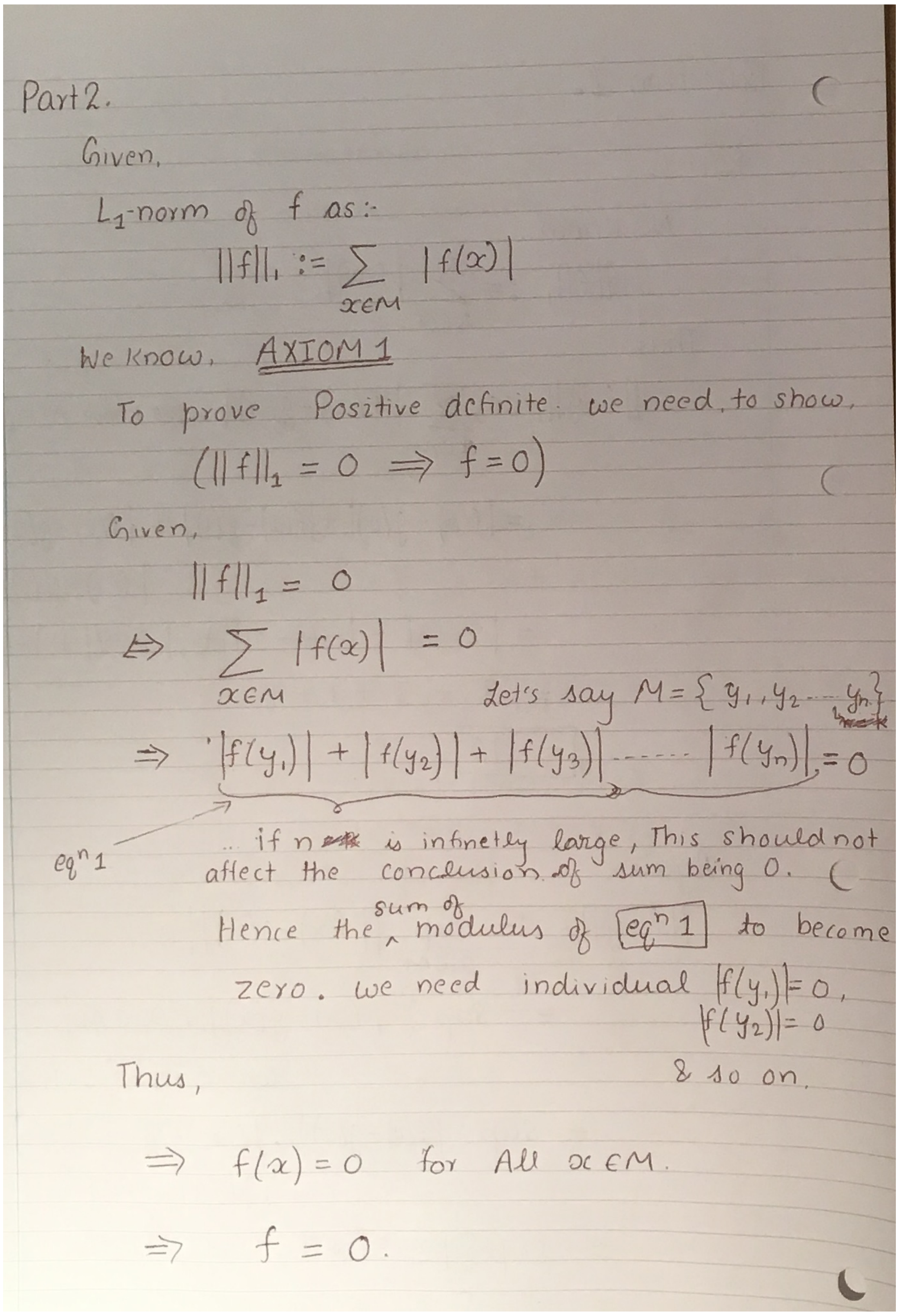
$$= \sup\{|f(0)-g(0)|, |f(1)-g(1)|, |f(2)-g(2)|, |f(3)-g(3)|\}$$

$$= \sup\{|2-5|, |-4-1|, |8-7| + |-4-(-3)|$$

$$= \sup\{3, 5, 1, 1\}$$

$$\boxed{= 5}$$

Part 2



Part 2.

Given,

L_1-norm of f as:-

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

We know, AXIOM 1

To prove Positive definite. we need to show,

$$(\|f\|_1 = 0 \Rightarrow f = 0)$$

Given,

$$\|f\|_1 = 0$$

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

Let's say $M = \{y_1, y_2 \cdots y_n\}$

$$\Rightarrow \underbrace{|f(y_1)| + |f(y_2)| + |f(y_3)| \cdots\cdots |f(y_n)|}_{eq^n 1} = 0$$

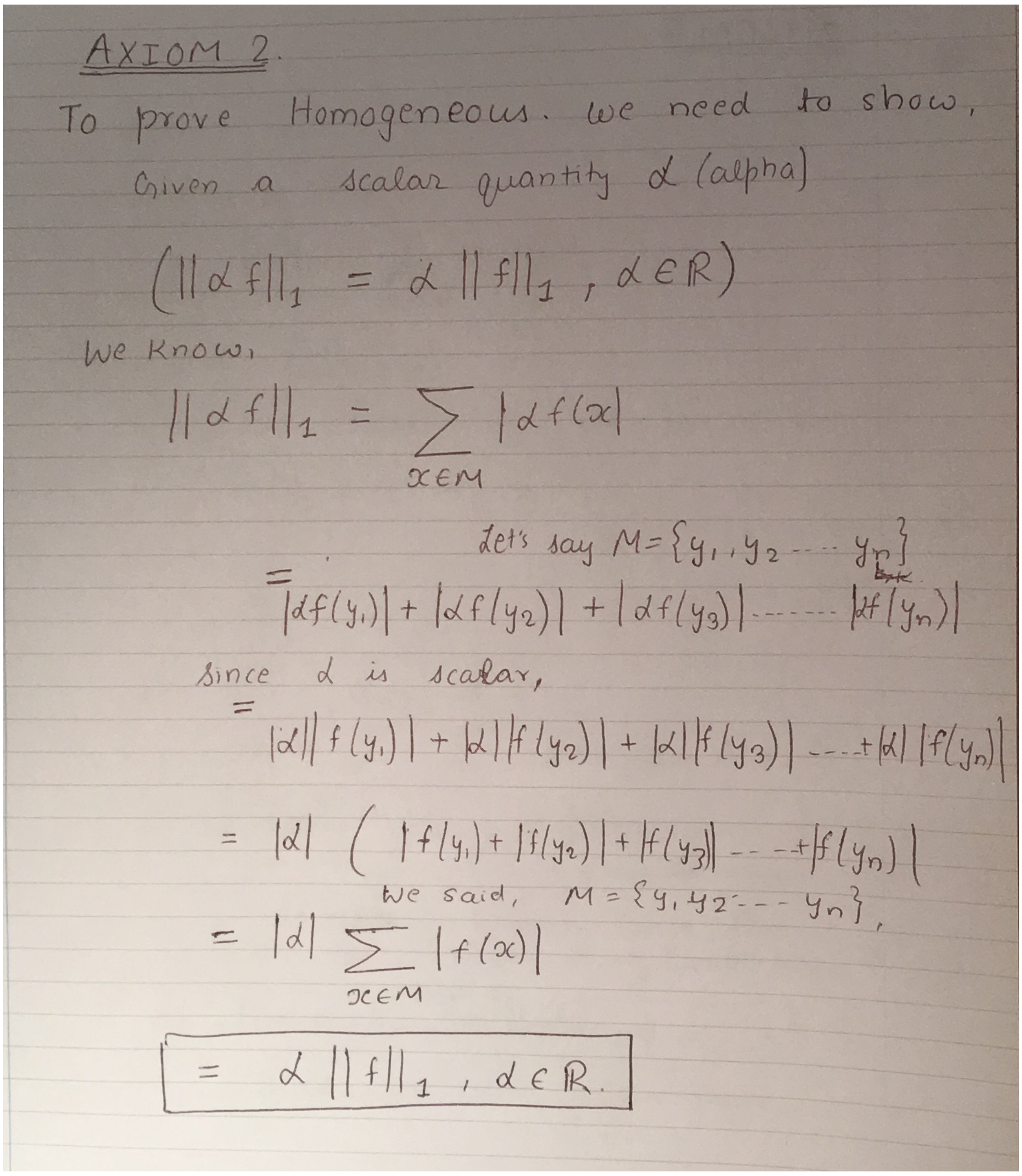
∴ if n is infinetly large, This should not affect the conclusion of sum being 0.

Hence the sum of modulus of $\boxed{eq^n 1}$ to become zero. we need individual $|f(y_1)| = 0$, $|f(y_2)| = 0$ & so on.

Thus,

$$\Rightarrow f(x) = 0 \text{ for All } x \in M.$$

$$\Rightarrow f = 0.$$



AXIOM 2.

To prove Homogeneous. we need to show,

Given a scalar quantity α (alpha)

$$\left(\|\alpha f\|_1 = \alpha \|f\|_1 , \alpha \in \mathbb{R}\right)$$

We know,

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

Let's say $M = \{y_1, y_2 \cdots y_n\}$

$$= |\alpha f(y_1)| + |\alpha f(y_2)| + |\alpha f(y_3)| \cdots\cdots |\alpha f(y_n)|$$

Since α is scalar,

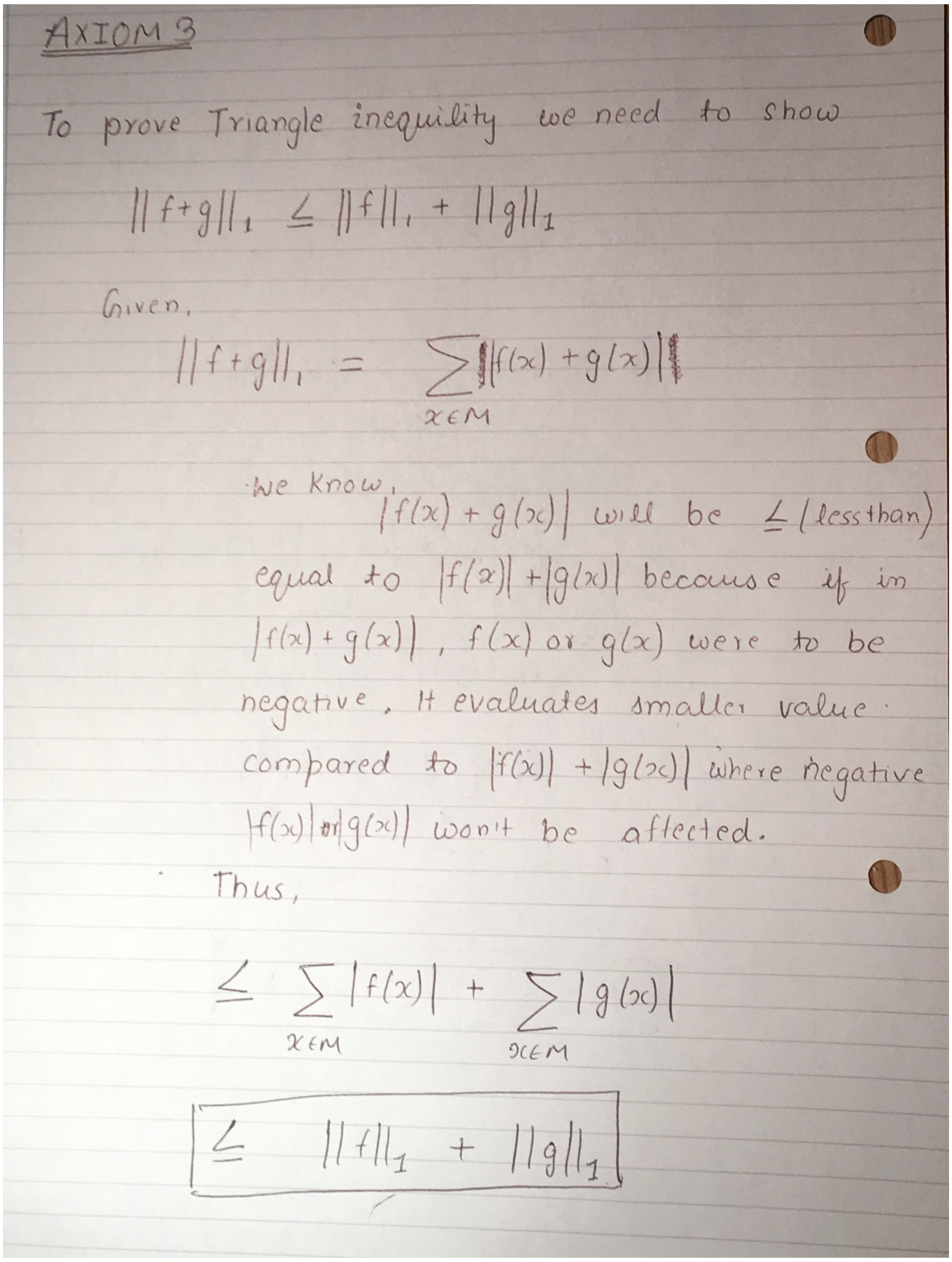
$$= |\alpha||f(y_1)| + |\alpha||f(y_2)| + |\alpha||f(y_3)| \cdots + |\alpha|\,|f(y_n)|$$

$$= |\alpha| \left(|f(y_1) + |f(y_2)| + |f(y_3)| \cdots + |f(y_n)| \right|$$

We said, $M = \{y_1, y_2 \cdots y_n\}$,

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

$$\boxed{= \alpha \|f\|_1 , \alpha \in \mathbb{R}.}$$



AXIOM 3

To prove Triangle inequility we need to show

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

Given,

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

We know, $|f(x) + g(x)|$ will be $\leq$ (less than) equal to $|f(x)| + |g(x)|$ because if in $|f(x) + g(x)|$, $f(x)$ or $g(x)$ were to be negative, It evaluates smaller value compared to $|f(x)| + |g(x)|$ where negative $|f(x)|$ or $|g(x)|$ won't be affected.

Thus,

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

$$\boxed{\leq \|f\|_1 + \|g\|_1}$$

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.

[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.

Question 2

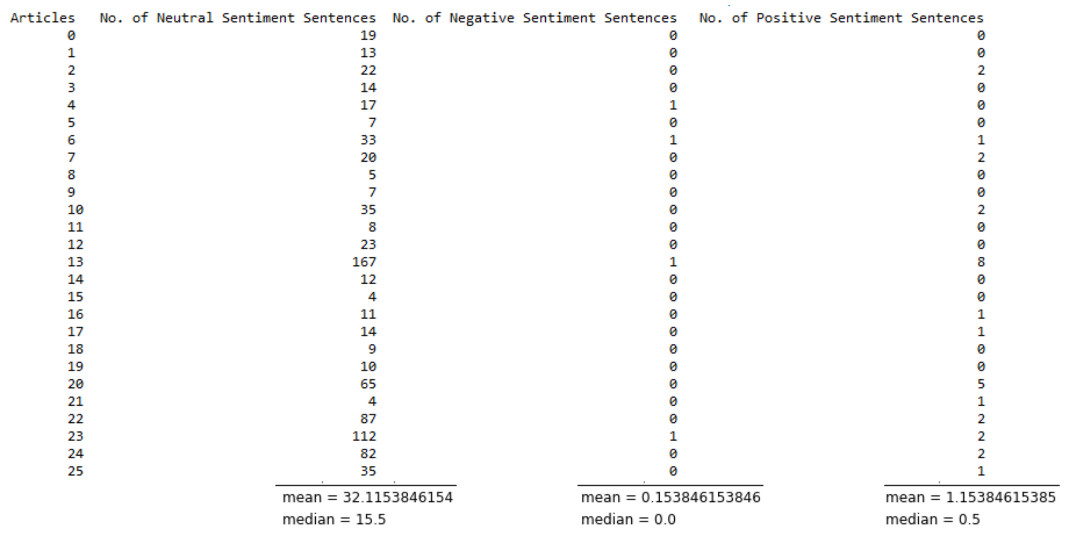
Answer 1

Three different observations (a, b, c) made about the data set are described below.

a.
Sentiment reflected by articles are 'Neutral' in nature

Define - Sentiment: We refer sentiment analysis based on definition provided in https://en.wikipedia.org/wikiSentiment_analysis

The below diagram shows how we formulated the above observation. The table shows the number of 'Negative' 'Positive' and 'Neutral' sentiments for first 25 articles and also states their average value.



Articles	No. of Neutral Sentiment Sentences	No. of Negative Sentiment Sentences	No. of Positive Sentiment Sentences
0	19	0	0
1	13	0	0
2	22	0	2
3	14	0	0
4	17	1	0
5	7	0	0
6	33	1	1
7	20	0	2
8	5	0	0
9	7	0	0
10	35	0	2
11	8	0	0
12	23	0	0
13	167	1	8
14	12	0	0
15	4	0	0
16	11	0	1
17	14	0	1
18	9	0	0
19	10	0	0
20	65	0	5
21	4	0	1
22	87	0	2
23	112	1	2
24	82	0	2
25	35	0	1
	mean = 32.1153846154	mean = 0.153846153846	mean = 1.15384615385
	median = 15.5	median = 0.0	median = 0.5

Figure 1: Observed result for first 25 articles

b.
90% of the articles do not include any numerical value.

Define - Numerical Value as numbers which includes 0-9 value.

c.
Every 5 out of 100 articles contains a word that cannot be typed from English Keyboard.

Define - word is a single distinct meaningful element shown with a space on either side when written or printed.

<u>Answer 2</u>

The observation mentioned in part a. awakens our interest because the sentiments ('Negative' , 'Neutral' and 'Positive') counted in first 25 articles as shown in the diagram supports our observation. And we believe the pattern will follow similarly in bigger data sets.

<u>Answer 3</u>

Assumptions : We define each line in the text-file consists of an article.

Three different Hypothesis from our observation mentioned in Answer 2 are listed below

Hypothesis 1

- *Each article has more than half of 'Neutral' sentiment sentences compared to that of 'Positive' and 'Negative' sentiment sentences combined.*

Falsifiability : As per our assumptions each single line associates to article. However we have observed few empty lines which falsifies our hypothesis since calculating "Positive" "Negative" and "Neutral" sentiments would result in equal number of sentiments.

Hypothesis 2

• *Sentences in each articles consists of fewer number of words which reflects the "Neutral" sentiment explanation of sentences.*

Explanation: We refer fewer words as number less than 20 words in a sentence.

Falsifiability : We have observed sentences with words greater than 30 and have "Neutral" sentiments. The sentence had no use of positive(eg. word like 'love') or negative(eg. word like 'hate') words.

Hypothesis 3

• *Articles contain more than 70% of "Neutral" words in all of its sentences.*

Falsifiability : We have observed sentences for example "They have a negative charge." which consists less than 70% of "Neutral" sentiment words.

<u>Answer 4</u>

We choose Hypothesis **3** i.e.
• *Articles contain more than 70% of "Neutral" words in all of its sentences.*

Since our Hypothesis is concrete, we will only slightly modify our hypothesis and will use it for testable predictions. Thus our Testable prediction for our Hypothesis would be

Each word has either 'Neutral' 'Positive' or 'Negative' sentiment and thus all sentences in Articles contain more than 70% of "Neutral" words.

Note: We define word as a single distinct meaningful element shown with a space on either side when written or printed.

However, after performing few iterations on sentences and observing the sentiment phenomenon in each word, our hypothesis is very difficult to test due to inconsistency in formulation of sentences. i.e. foreign language and numerical data were hard to test for sentiments.

Thus we reconsidered our choice and have now decided to work on **Hypothesis 1** i.e.

• *Each article has more than half of 'Neutral' sentiment sentences compared to that of 'Positive' and 'Negative' sentiment sentences combined.*

Similarly, our testable prediction replicates our hypothesis because of its concrete nature. Testable prediction we will use will be

Each article has more than half of 'Neutral' sentiment sentences in comparison of the sum of all 'Positive' and 'Negative' sentiment sentences.

Answer 5

We will perform analysis on data set by writing a python code which will make use of various sentiment analyses libraries.

We will obtain sentiment details for each sentence in an article. This means a sentence will be categorized in on the three sentiments ('Neutral' ,'Positive' ,'Negative'). This will be further extended for all the articles.

We will try and obtain the following statistics from the data set.

- Obtain a List which contains number of 'Negative' sentiment sentences in each article across all the articles
- Obtain a List which contains number of 'Positive' sentiment sentences in each article across all the articles
- Obtain a List which contains number of 'Neutral' sentiment sentences in each article across all the articles
- Since our hypothesis suggests we compare 'Neutral' sentiment to 'Positive' and 'Negative' sentiment combined, We will add the positive and negative sentiments obtained in each article.

Once the above is obtained ”We expect to see two different histograms. One for representing the ”Neutral” sentiment and the other for representing the ’Positive and Negative’ merged sentiment. Similarly we will construct Cumulative Distribution function graph in addition to the histogram.

The first histogram i.e. the histogram representing ”Neutral” will have *Number of ’Neutral’ sentiment sentences in article* on the x-axis and *absolute number of articles* on the y-axis. The histogram will have no definite pattern as we expect random distribution of sentences across articles. We will however be able to observe the median and mean within the graph. Furthermore we will observer mean and median also in our cumulative distribution function graph and see if our hypothesis can be supported

Similarly, The second histogram will follow the similar trend only that it now represents ”Positive + Negative” sentiments. Cumulative distribution Function Graph will also be drawn.

We will then compare and contrast our figures described above 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 implicitly 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.

Answer

In order to validate we have firstly implemented a process to calculate whether a sentence in an article has one of ’Positive’ ’Negative’ or ’Neutral’ sentiment i.e. a Python code was written. The code snippet can be seen below after Figure 6. We follow the steps in Question 2 part 5 of our answer whereby we mentioned of plotting the histogram and cumulative Distribution plot graph for our observations. We will now go through each diagram and study the significance in supporting our hypothesis.

Initially we observed the following histogram for Neutral Sentiments against the article frequency. However the graph seemed unreadable

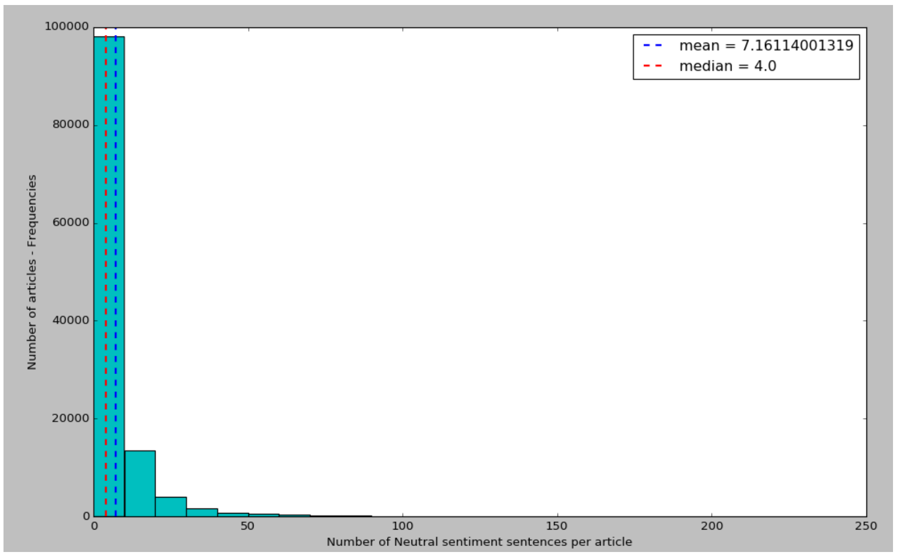


Figure 2: Bad Histogram plot for Neutral sentiment representation

Thus we changed the y-axis to logarithmic plot and obtained the rather readable histogram. We also draw the lines representing the mean and median.

Here we can see that mean is 7.16 suggesting the average 'Neutral' sentiment sentences in an article. Similarly we observe the median of 4.0 suggesting the centre of a dataset. The graph itself shows that there is a very high occurrences of Neutral sentiment sentences.

This however can not show if our hypothesis is supported. We will try and compare later with the histogram obtained from 'Negative + Positive' sentiments to get the feel of how high the 'neutral sentiment sentences dominate the articles.

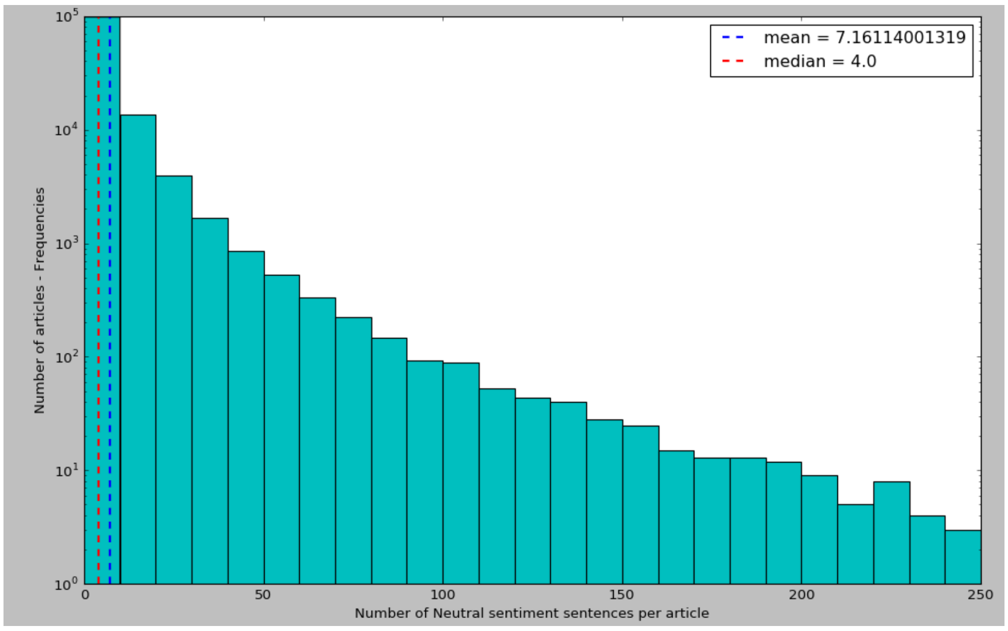


Figure 3: Histogram plot for Neutral sentiment representation using logarithmic y-axis

Let us now create a Cumulative Distribution Function graph as can be seen in Figure 4.

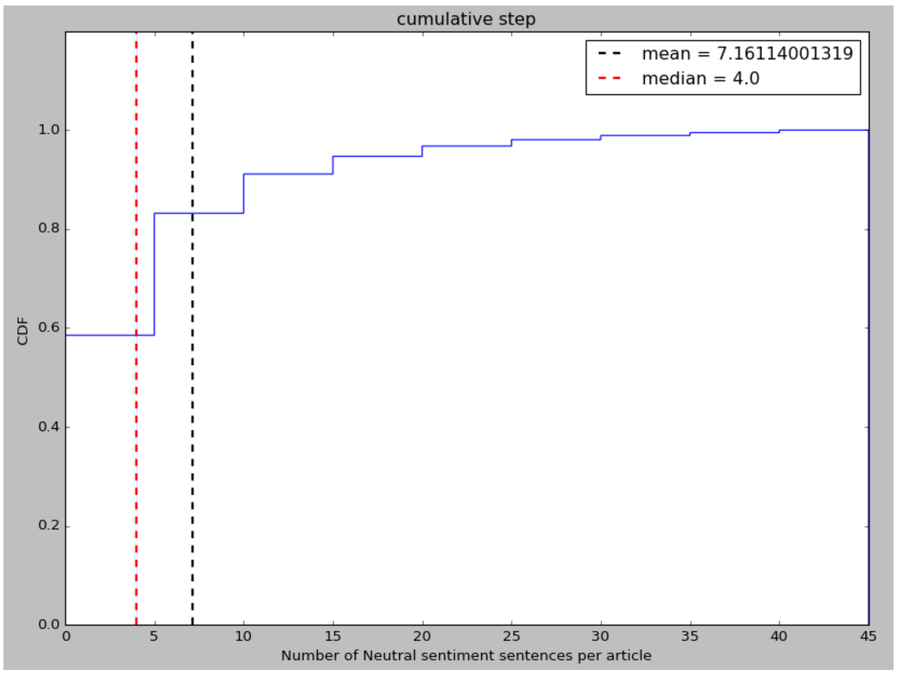


Figure 4: Cumulative Distribution function plot for Neutral sentiment representation

With this plot it is very clear that median with 4.0 crosses the CDF axis at 0.58. This therefore supports our hypothesis which stated

- *Each article has more than half of 'Neutral' sentiment sentences compared to that of 'Positive' and 'Negative' sentiment sentences combined.*

We will further see the diagrams for 'Negative + Positive' plots. This will allow us to observe how 'Neutral' Sentiment sentences dominates the article.

As we can observe from below histogram, around 100000 articles had 0 - 5 number of negative and positive sentiment sentences. This graph rapidly goes down suggesting lesser number of Negative and positive sentiment sentences. This compared to the above histogram for 'Neutral' sentiment, we can be certain about 'Neutral' sentiments appears more than half of that of 'positive' and 'Negative' sentences combined.

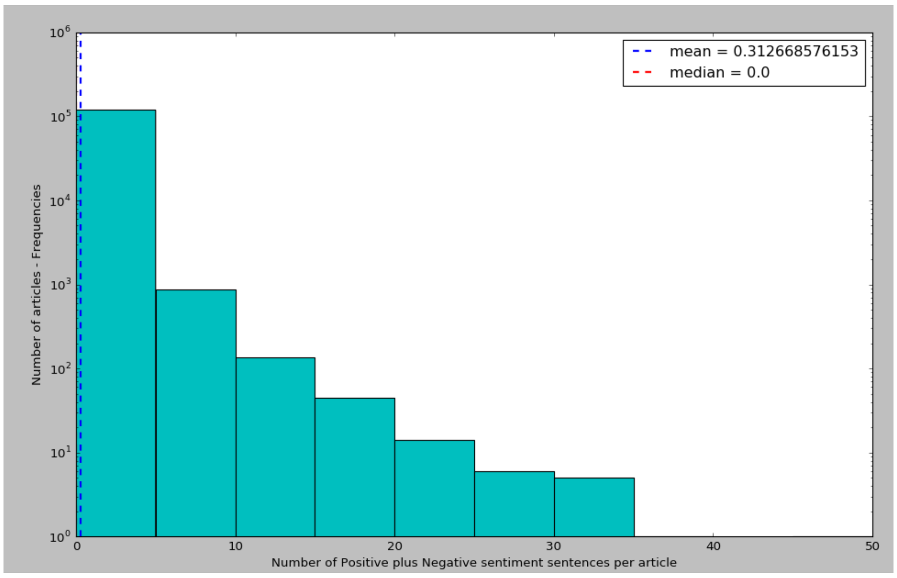


Figure 5: Histogram plot for Negative + Positive sentiment representation using logarithmic y-axis

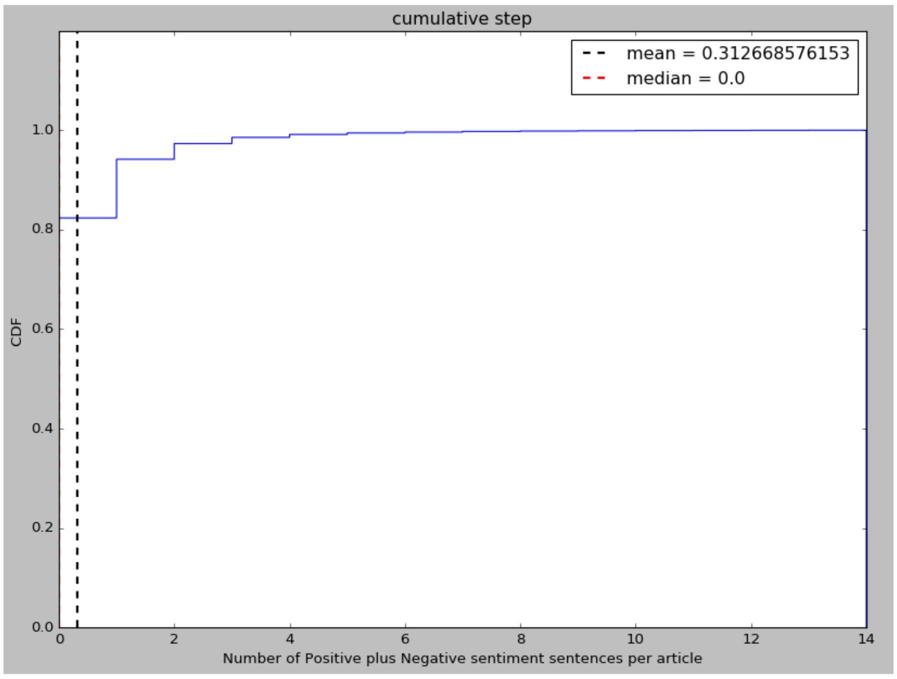


Figure 6: Cumulative Distribution function plot for Negative + Positive sentiment representation

Now again if we view the CDF for 'Negative and Positive' The median is 0.0 suggesting minimal occurrences of these sentiments compared to 'Neutral' which had median of 4.0. Thus we can be strong enough to support the following.

• *Each article has more than half of 'Neutral' sentiment sentences compared to that of 'Positive' and 'Negative' sentiment sentences combined.*

```
#Code for implementing Sentiment observation in all articles

import nltk
from nltk.sentiment.vader import SentimentIntensityAnalyzer
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

sentimentsEachArticles = []
neutralSentimentList = []
positiveSentimentList = []
negativeSentimentList = []

def getSentimentOptimal(textToFindSentimentOf):
    #initialise the SentimentAnalyszer
    sid = SentimentIntensityAnalyzer()
    ss = sid.polarity_scores(textToFindSentimentOf)

    # maximum value of all obtained sentiments
    maxValueKey =  max(ss.keys(), key=(lambda key: ss[key]))
    if ss[maxValueKey] <= 0.0 :
        return None
    return maxValueKey

# We give a sentence and get the noun or verb or adjective from the sentence
def getRelevantWordForSentimentEvaluation(sentence):
    perSentenceWords = nltk.word_tokenize(sentence)
    # Here we get the grammarType for each words we splitted from the sentence.
    getGrammarTypeOfEachWord = nltk.pos_tag(perSentenceWords)
    print (getGrammarTypeOfEachWord)

    #Here we convert to dictionary inorder to obtain the appropriate word for
    #testing.
    grammarTypeReversed = [(b,a) for (a,b) in getGrammarTypeOfEachWord]
    grammarTypeReversed = dict(grammarTypeReversed)

    try:
        try:
            try:
                try:
                    try:
                        try:
                            try:
                                try:
                                    try:
                                        value = grammarTypeReversed['NNS']
                                    except:
                                        value = grammarTypeReversed['NNPS']
                                except:
                                    value = grammarTypeReversed['NN']
                            except:
                                value = grammarTypeReversed['NNP']
                        except:
                            value = grammarTypeReversed['NNS']
                    except:
                        value = grammarTypeReversed['RB']
                except:
                    value = grammarTypeReversed['RBR']
            except:
                value = grammarTypeReversed['RBS']
        except:
            value = grammarTypeReversed['RP']
    except:
        try:
            value = next (iter (dict.values(grammarTypeReversed)))
        except:
            value = None
    return value



def drawHistogram(listElements , xLabel , yLabel, interval):
    x_1 = np.array(listElements)
    plt.figure(figsize=(12,9))
    plt.hist(x_1 , bins= range(0, 100 + interval, interval) , color='c' )
    plt.xlabel(xLabel)
    plt.ylabel(yLabel)
    plt.yscale('log')
    plt.axvline(x_1.mean(), color='b', linestyle='dashed', linewidth=2 , \
                                        label="mean = " + str(np.mean(x_1)))
    plt.axvline(np.median(x_1), color='r', linestyle='dashed', linewidth=2 , \
                                    label = "median = " + str(np.median(x_1)))

    plt.legend(loc='upper right')
    plt.show()


```

```
def drawCDF(listElements , xLabel , yLabel, interval):
    x_1 = np.array(listElements)
    plt.figure(figsize=(12,9))
    plt.xlabel(xLabel)
    plt.ylabel(yLabel)
    plt.axvline(x_1.mean(), color='black', linestyle='dashed', linewidth=2 ,\
                                    label="mean = " + str(np.mean(x_1)))
    plt.axvline(np.median(x_1), color='r', linestyle='dashed', linewidth=2 ,\
                                label = "median = " + str(np.median(x_1)))

    n, bins, patches = plt.hist(x_1, bins= range(0, 90, interval), normed=1,
                            histtype='step', cumulative=True)

    plt.ylim(0, 1.2)
    plt.title('cumulative step')
    plt.legend(loc='upper top')

    plt.show()

#Parses the file
def getEachLinesFromFile(filename):
    #fname = "simple-20160801-1-article-per-line1.txt"
    with open(filename , encoding="utf8") as fp:
        try:
            content = fp.readlines()
        except:
            content = ""
    return content


#Takes filename and starts the process for sentiment observation in that file
def mainFunction(filename):
    # Here we load tokenizers from nltk and obtain sentences.
    tokenizer = nltk.data.load('tokenizers/punkt/english.pickle')
    content = getEachLinesFromFile(filename)

    #Obtain each articles i.e. each line
    listOfSentencesFromArticles = [tokenizer.tokenize(eachArticle) for \
                                   eachArticle in content]

    for sentencesForSentiment in listOfSentencesFromArticles:
        neutralSentiments = 0
        negativeSentiments = 0
        positiveSentiments = 0
        for eachSentence in sentencesForSentiment:
            demoKolagi = eachSentence

            #get Sentiment calculated for eachSentence
            sentiment = getSentimentOptimal(demoKolagi)
            if sentiment is not None:
                if sentiment.lower() == "neu":
                    neutralSentiments = neutralSentiments + 1
                if sentiment.lower() == "neg":
                    negativeSentiments = negativeSentiments + 1
                if sentiment.lower() == "pos":
                    positiveSentiments = positiveSentiments + 1
                if sentiment.lower() == "compound":
                    positiveSentiments = positiveSentiments + 1

        neutralSentimentList.append(neutralSentiments)
        negativeSentimentList.append(negativeSentiments)
        positiveSentimentList.append(positiveSentiments)

        articleSentimentsDict = {}
        articleSentimentsDict["Neutral"] = neutralSentiments
        articleSentimentsDict["Positive"] = positiveSentiments
        articleSentimentsDict["Negative"] = negativeSentiments
        sentimentsEachArticles.append(articleSentimentsDict)



if __name__ == "__main__":
    filename = "simple-20160801-1-article-per-line2.txt"
    mainFunction(filename)
    print("\n Each articles has : " , sentimentsEachArticles)
    print ("\n Each articles has This neutral : " , neutralSentimentList)

    print ("\n Each articles has This positive : " , positiveSentimentList)

    print ("\n Each articles has This negative : " , negativeSentimentList)
    sumNegativePositiveSentimentList = [x + y for x, y in zip(\
                                positiveSentimentList, negativeSentimentList)]

    drawHistogram(neutralSentimentList , "Number of Neutral sentiment sentences\
                    per article" , "Number of articles - Frequencies" , 10)
    drawCDF(neutralSentimentList , "Number of Neutral sentiment sentences per \
                                                        article" , "CDF " , 5)

    drawHistogram(sumNegativePositiveSentimentList , "Number of Positive plus \
                                Negative sentiment sentences per article" ,\
                                "Number of articles - Frequencies" , 5)
    drawCDF(sumNegativePositiveSentimentList , "Number of Positive plus \
                        Negative sentiment sentences per article" , "CDF " , 1)
```

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`.