Introduction to Web Science

Assignment 10

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

René Pickhardt
rpickhardt@uni-koblenz.de

Korok Sengupta
koroksengupta@uni-koblenz.de

Olga Zagovora
zagovora@uni-koblenz.de

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

Submission until: January 25, 2016, 10:00 a.m.
Tutorial on: January 27, 2016, 12:00 p.m.

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: Echo

1 Modeling Twitter data (10 points)

In the meme paper[1] by Weng et al., in Figure 2[2] you find a plot, comparing the system entropy with the average user entropy. Your task is to reproduce the plot and corresponding calculations.

1. We provide you with the file 'onlyhashtag.data', containing a collection of hashtags from tweets. Use this data to reproduce the plot from the paper. Once you have the values for average user entropy and system entropy calculated per day create a scatter plot to display the values.
2. Interpret the scatter plot and compare it with the authors interpretation from the graph showed in the paper. Will the interpretations be compatible to each other or will they contradict each other? Do not write more than 5 sentences.

1.1 Hints

1. Use formulas from the lecture to calculate the entropy for one user and the system entropy.
2. Do not forget to give proper names of plot axes.

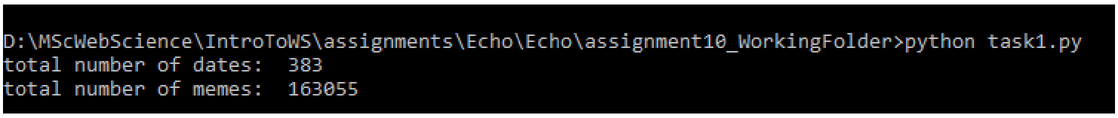
Answer (1)

```
# -*- coding: utf-8 -*-
"""
Created on Tue Jan 31 12:09:30 2017

@author: Hanadi
"""
import re
import csv
date_sys = dict()
hashtags = []
c = dict()
with open("onlyhash.data") as tsv:
    for line in csv.reader(tsv, dialect="excel", delimiter='\t'):
        tmp = []
        temp = line[2].strip().split('#')
        for tem in temp:
            x = re.sub('[^A-Za-z1-9]+', '', tem)
            if (x != ''):
                hashtags.append(x)
                tmp.append(x)
                if line[1] in date_sys.keys():
```

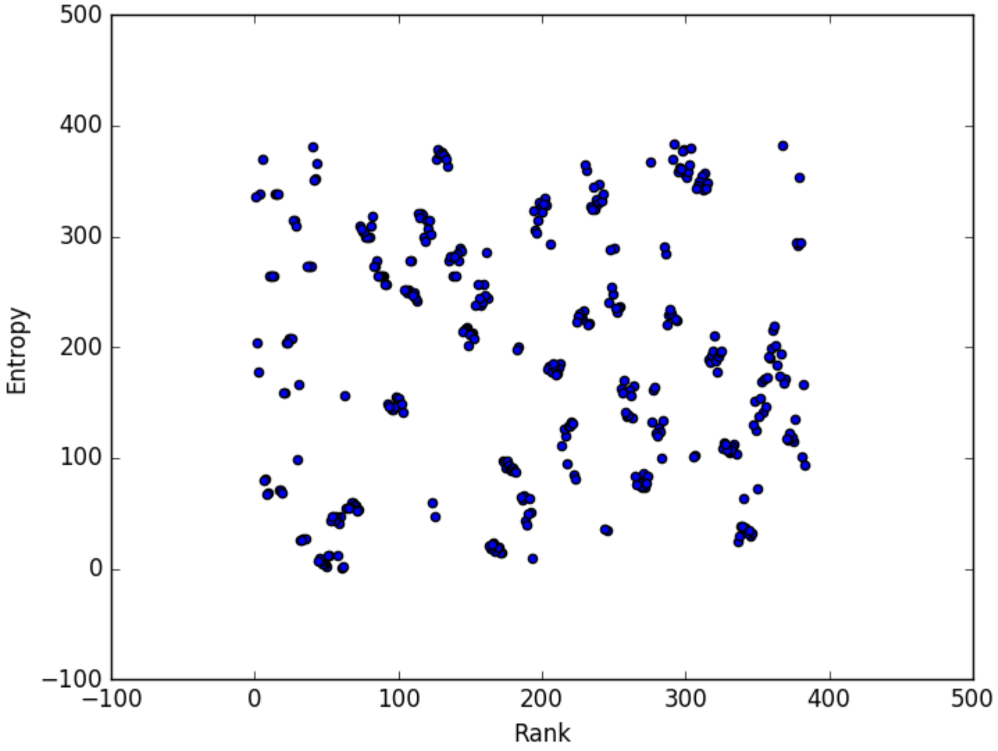
[1] http://www.nature.com/articles/srep00335

[2] Slide 27, Lecture Meme spreading on the Web

```
                    # a dictionary of dates as keys and values of list of hashtag
                    date_sys[line[1]].append(x)
                else:
                    date_sys[line[1]] = tmp
dates = len(date_sys)
N = len(set(hashtags))
print "total number of dates: ", dates
print "total number of memes: ", N

from collections import Counter
import math
f = dict()
sys = dict()
for key, val in date_sys.items():
    counts = Counter(val)
    for val1 in counts:
        y = counts[val1]/float(N)
        f[val1] = round((y*math.log(y, 2)),3)
    # system entropy of hashtags on each date
    sys[key] = round( sum(f.values()) * (-1), 3)

import matplotlib.pylab as plt
from scipy.stats import rankdata
plt.scatter(rankdata(sys.keys()),rankdata(sys.values()))
plt.ylabel('Entropy')
plt.xlabel('Rank')
plt.show()
```



```
D:\MScWebScience\IntroToWS\assignments\Echo\Echo\assignment10_WorkingFolder>python task1.py
total number of dates:  383
total number of memes:  163055
```



(2) We have made the system entropy per each day by finding the hashtags that were tweeted on that date and then used the entropy function on them, and each summation is the daily entropy of the tweeted hashtags. The results are different because we have used a different entropy methodology than what is used in the author's plot.

2 Measuring inequality (10 points)

We provide you with a sample implementation of the Chinese Restaurant Process[3].

Assume there is a restaurant with an infinite number of tables. When a new customer enters a restaurant he chooses an occupied table or the next empty table with some probabilities.

According to the process first customer always sits at the first table. Probability of the next customer to sit down at an occupied table i equals ratio of guests sitting at the table (c_i/n), where n is the number of guests in the restaurant and c_i is the number of guests sitting at table i.
Probability of customer to choose an empty table equals : $1 - \sum_{i=1}^{S} p_i$, where S is the number of occupied tables and $p_i = c_i/n$.

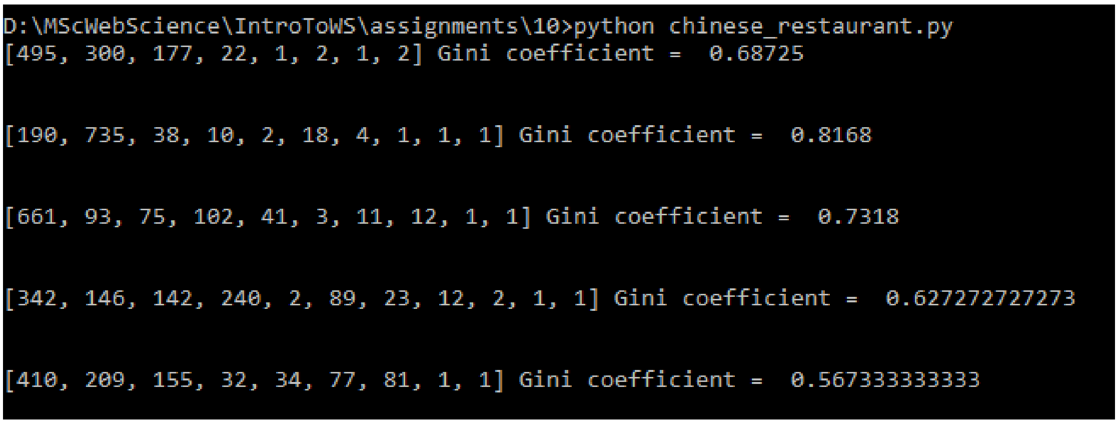
Provided script simulates the process and returns number of people sitting at each table. We will study restaurants for 1000 customers. Now you should modify the code and evaluate how unequal were the customers' choices of tables.

Calculate the Gini- coefficient measuring the inequality between the tables, until the coefficient stabilizes. Do five different runs and plot your results in a similar way that plots in the lecture slides are done, cf. Slide 32 and Slide 33. **Answer**

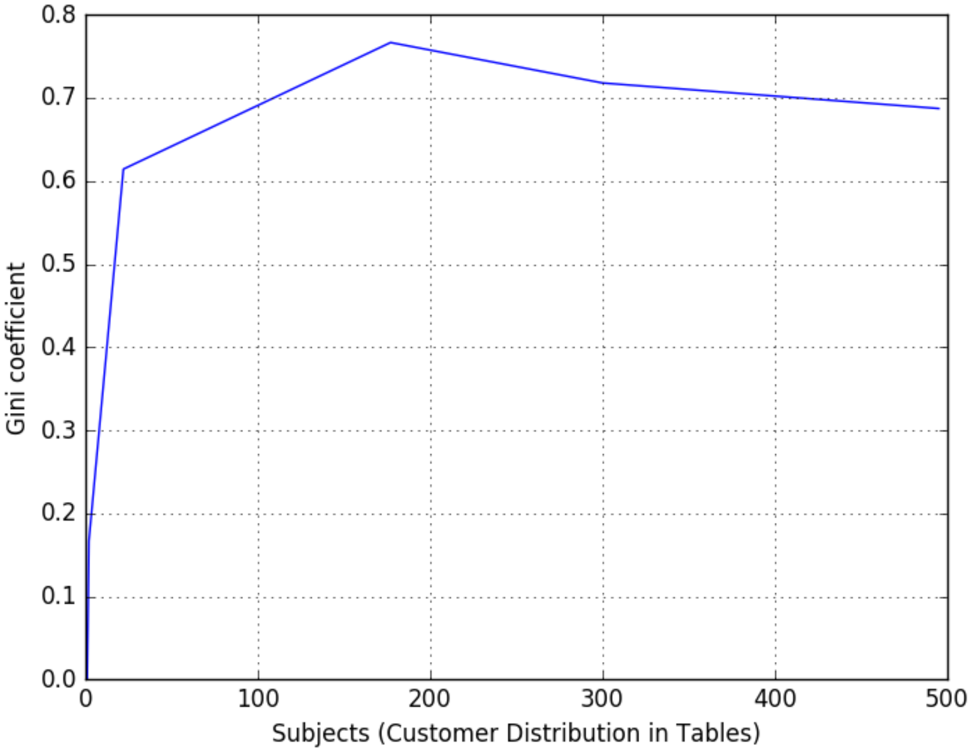
```
import random
import json
import matplotlib.pyplot as plt

def generateChineseRestaurant(customers):
    # First customer always sits at the first table
    tables = [1]
    #for all other customers do
    for cust in range(2, customers+1):
            # rand between 0 and 1
            rand = random.random()
            # Total probability to sit at a table
            prob = 0
            # No table found yet
            table_found = False
            # Iterate over tables
            for table, guests in enumerate(tables):
                # calc probability for actual table an add it to total probabilit
                prob += float(guests) / float(cust)
                # If rand is smaller than the current total prob., customer will
                if rand < prob:
                    # incr. #customers for that table
```

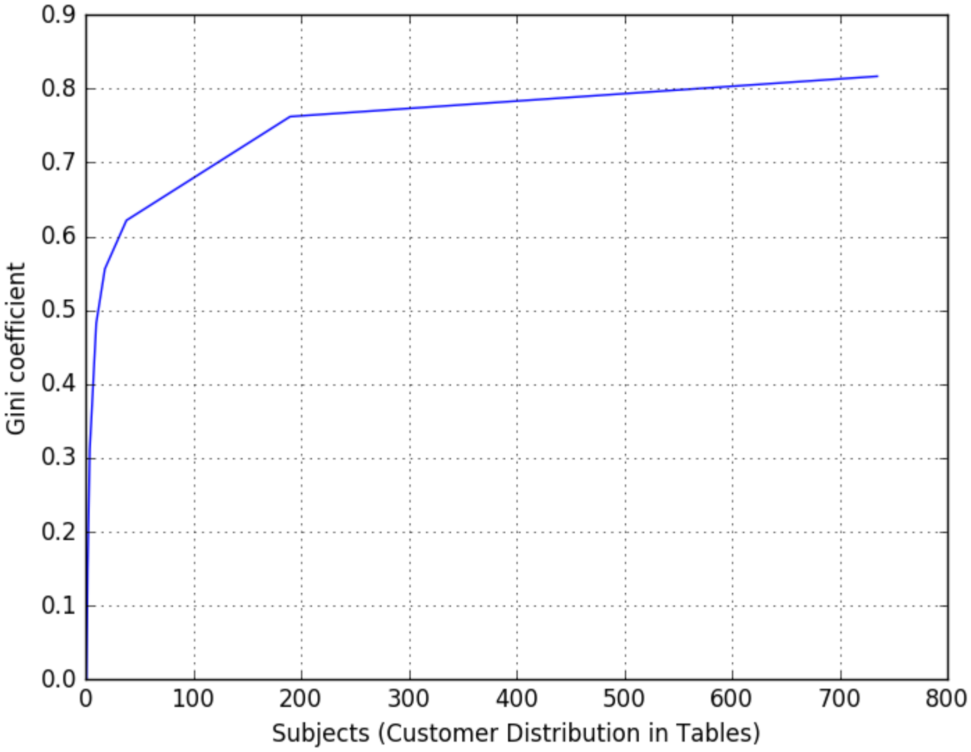
[3] File "chinese_restaurant.py"; Additional information can be found here: https://en.wikipedia.org/wiki/Chinese_restaurant_process

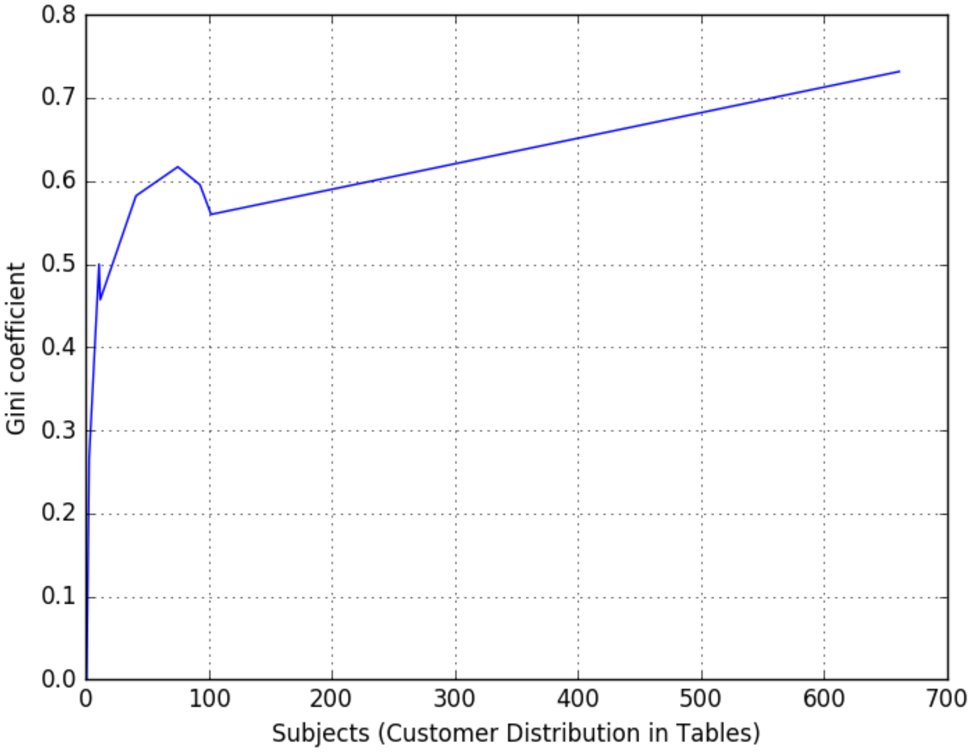
```
                    tables[table] += 1
                    # customer has found table
                    table_found = True
                    # no more tables need to be iterated, break out for loop
                    break
            # If table iteration is over and no table was found, open new table
            if not table_found:
                tables.append(1)
    return tables

def giniIt(list_of_values):
    sorted_list = sorted(list_of_values)
    height, area = 0, 0
    for value in sorted_list:
        height += value
        area += height - value / 2.
    fair_area = height * len(list_of_values) / 2.
    return (fair_area - area) / fair_area

def giniPlot(dist_list):
    dist_list = sorted(dist_list)
    var_dist_list = []
    var_gini_list = []
    for x in dist_list:
        var_dist_list.append(x)
        g = giniIt(var_dist_list)
        var_gini_list.append(g)
    plt.plot(var_dist_list, var_gini_list)
    plt.title('')
    plt.ylabel('Gini coefficient')
    plt.xlabel('Subjects (Customer Distribution in Tables)')
    plt.grid('on')
    plt.show()

restaurants = 1000
for i in range(5):
    dist_list = generateChineseRestaurant(restaurants)
    g = giniIt(dist_list)
    print dist_list, 'Gini coefficient = ', g
    giniPlot(dist_list)
    print '\n'

# network = generateChineseRestaurant(restaurants)
# with open('network_' + str(restaurants) + '.json', 'w') as out:
#     json.dump(network, out)
```

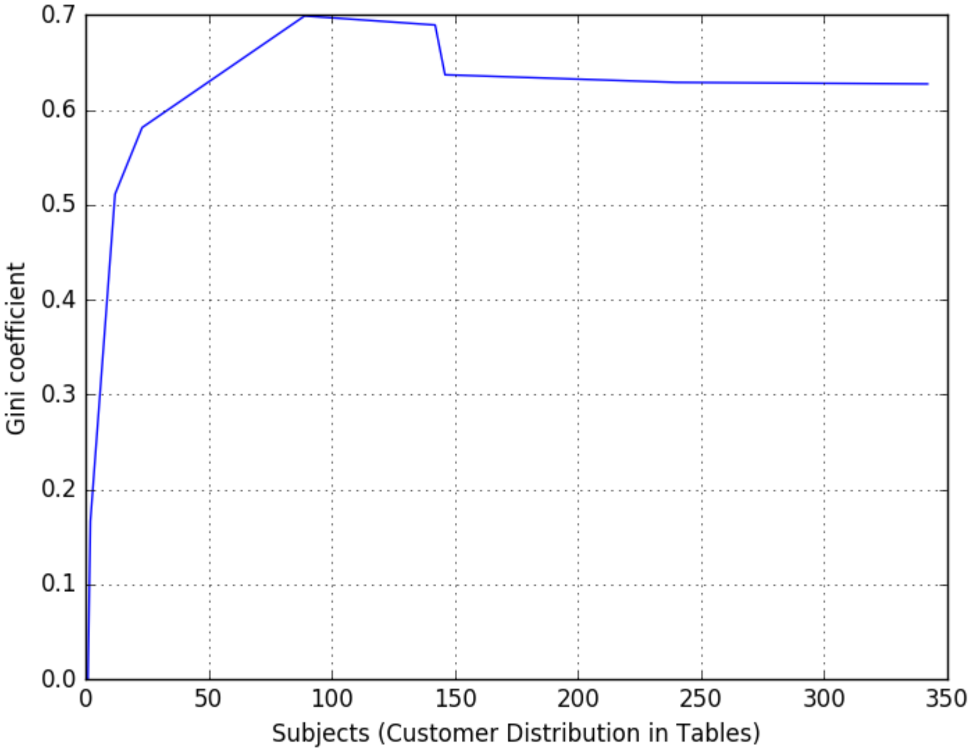


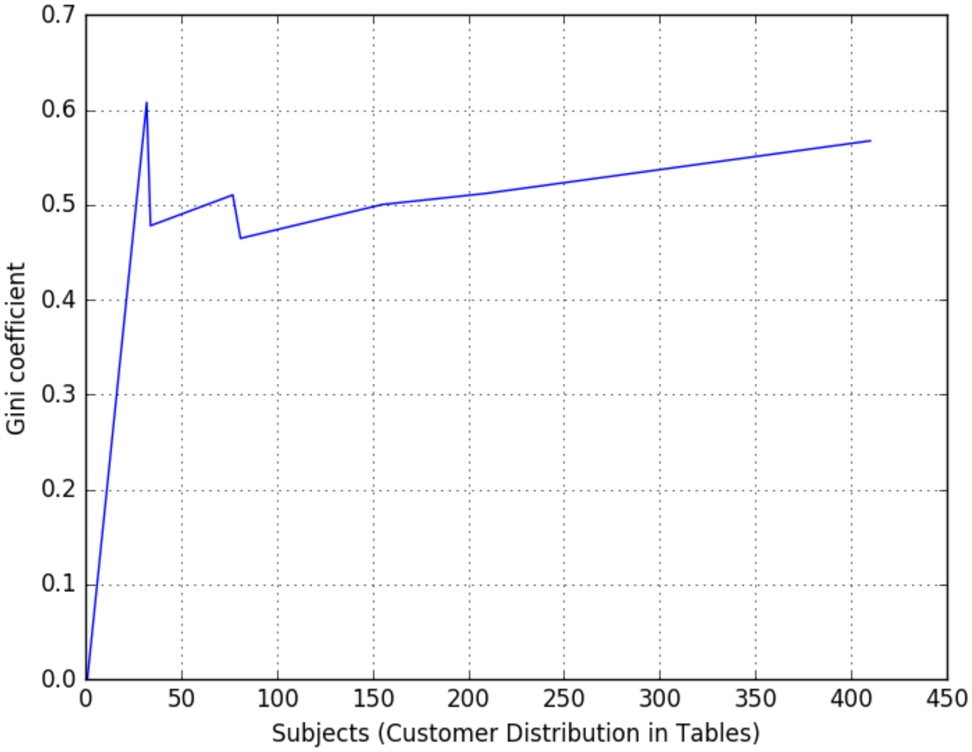
```
D:\MScWebScience\IntroToWS\assignments\10>python chinese_restaurant.py
[495, 300, 177, 22, 1, 2, 1, 2] Gini coefficient =  0.68725


[190, 735, 38, 10, 2, 18, 4, 1, 1, 1] Gini coefficient =  0.8168


[661, 93, 75, 102, 41, 3, 11, 12, 1, 1] Gini coefficient =  0.7318


[342, 146, 142, 240, 2, 89, 23, 12, 2, 1, 1] Gini coefficient =  0.627272727273


[410, 209, 155, 32, 34, 77, 81, 1, 1] Gini coefficient =  0.567333333333
```




Gini coefficient
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
0
100
200
300
400
500
600
700
800
Subjects (Customer Distribution in Tables)


Gini coefficient
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
100
200
300
400
500
600
700
Subjects (Customer Distribution in Tables)


Gini coefficient
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
50
100
150
200
250
300
350
Subjects (Customer Distribution in Tables)


0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Gini coefficient
0
50
100
150
200
250
300
350
400
450
Subjects (Customer Distribution in Tables)

3 Herding (10 points)

Let us consider the altitude of Koblenz to be 74 m above sea level. You are asked to figure out the height of the Ehrenbreitstein Fortress and the Fernmeldeturm Koblenz without googling.
The exercise is split in two parts:

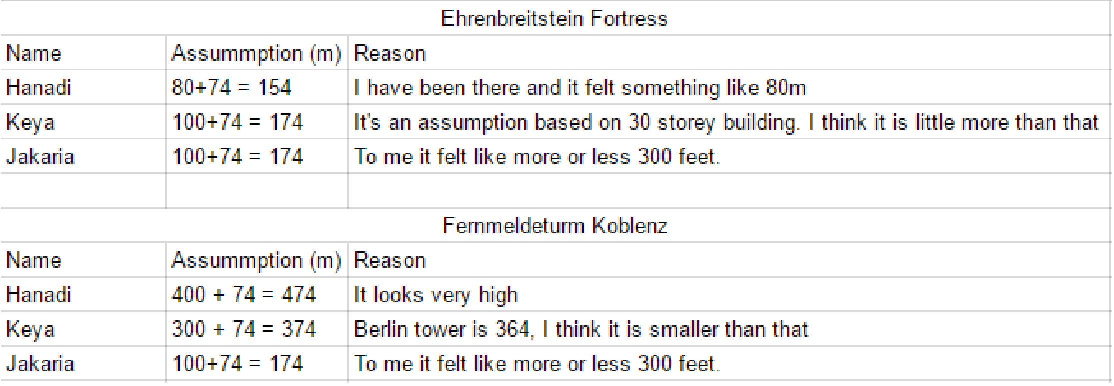
Part 1 : The Secret
In *complete secrecy*, each member of the team will write down their estimated height of the Ehrenbreitstein Fortress without any form of discussion. Please keep in mind that you need to have reasons for your assumption. Once you are done, then openly discuss in the group and present you values in a tabulated format with the reasons each one assumed to arrive at that value.

Part II : The Discussion
Discuss amongst yourself with valid reasoning what could be the height of the Fernmeldeturm Koblenz. Only after discussing, each member of the group is asked to arrive at a value and present this value in a tabulated format as was done in Part I.

Calculate the Mean, Standard Deviation and Variance of your noted results for both the cases and explain briefly what you infer from it.

Note: This exercise is for you to understand the concepts of herding and not to get the perfect height by googling information. There is in fact no point associated with the height but with the complete reasoning that you provide for your answers.

Answer



	Ehrenbreitstein Fortress	
Name	Assummption (m)	Reason
Hanadi	80+74 = 154	I have been there and it felt something like 80m
Keya	100+74 = 174	It's an assumption based on 30 storey building. I think it is little more than that
Jakaria	100+74 = 174	To me it felt like more or less 300 feet.
	Fernmeldeturm Koblenz	
Name	Assummption (m)	Reason
Hanadi	400 + 74 = 474	It looks very high
Keya	300 + 74 = 374	Berlin tower is 364, I think it is smaller than that
Jakaria	100+74 = 174	To me it felt like more or less 300 feet.

Ehrenbreitstein Fortress
Mean: $(154 + 174 + 174)/3 = 167.33$
Variance: $((154 - 167.33)^2+(174 - 167.33)^2+(174 - 167.33)^2)/3 = 88.88$
Standard Deviation: $\sqrt{}(\text{Variance}) = \sqrt{}(88.88) = 9.42$

Fernmeldeturm Koblenz
Mean: $(474 + 374 + 174)/3 = 340.66$

Variance: $((474 - 340.66)^2+(374 - 340.66)^2+(174 - 340.66)^2)/3 = 15{,}555.55$

Standard Deviation: $\sqrt{}$(Variance) = $\sqrt{}$(15,555.55) = 124.72

Important Notes

Submission

- Solutions have to be checked into the github repository. Use the directory name `groupname/assignment10/` 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`.