# 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.**

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

# 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.

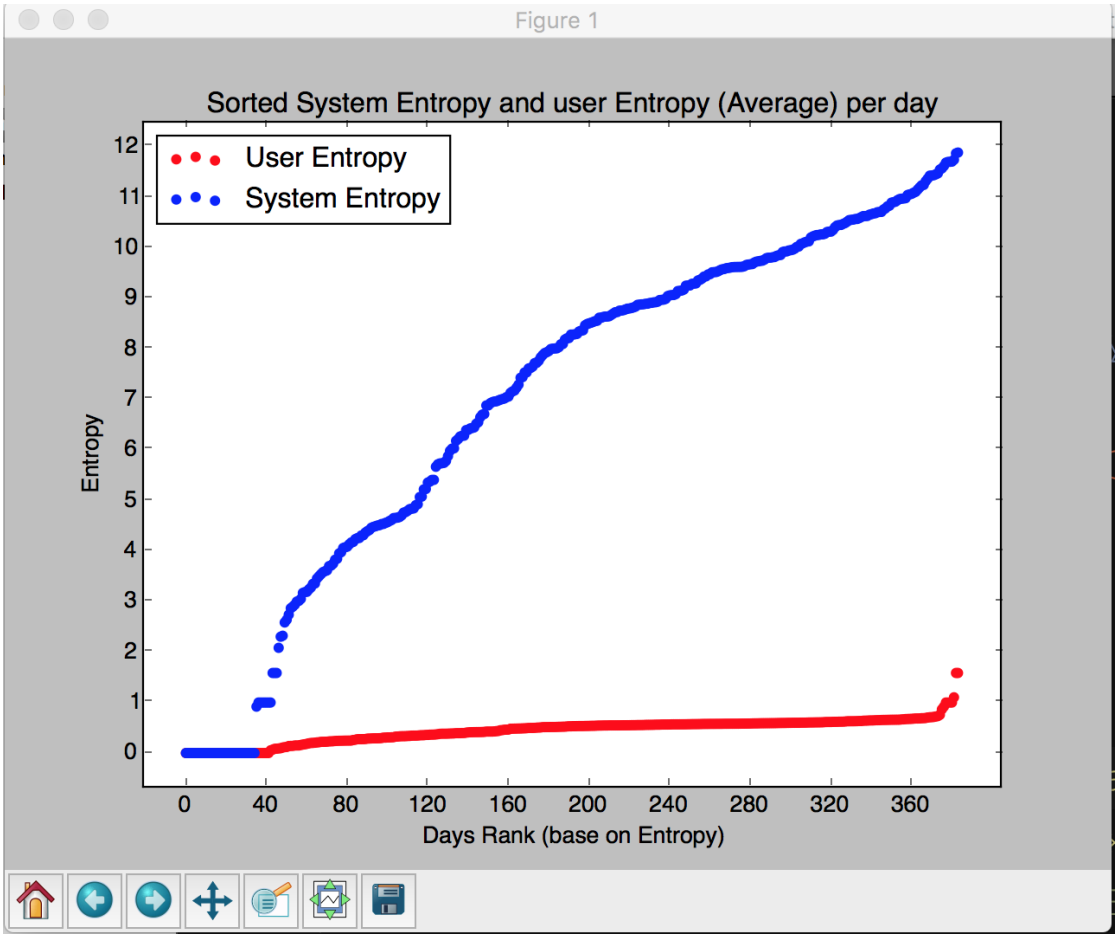
**Answers**

```
from collections import Counter
from math import log2
import time
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt


def is_valid_date(date_str):
    """ returns True iff givin date is valid"""
    try:
        valid_date = time.strptime(date_str, '%Y-%m-%d')
    except ValueError:
        return False
    else:
        return True

def draw_plot(user_entropy, system_entropy):
    i = 0
    x = len(user_entropy) * [0]
    while i < len(x):
        x[i] = i+1
        i += 1
    plt.xticks(np.arange(0, max(user_entropy),40))
    plt.yticks(range(0,int(max(system_entropy) + 2)))
    plt.title("Sorted System Entropy and user Entropy (Average) per day")
    plt.xlabel("Days Rank (base on Entropy)")
    plt.ylabel("Entropy")
```

[1] http://www.nature.com/articles/srep00335
[2] Slide 27, Lecture Meme spreading on the Web

```
29:     plt.scatter(x, user_entropy, label='User Entropy', color="r")
30:     plt.scatter(x, system_entropy, label='System Entropy', color="b")
31:     plt.legend(loc=2)
32:     plt.show()
33:
34: def info_according_to_date(date, datas):
35:     dic = dict()
36:     dic['date'] = date
37:     dic['tweets'] = Counter()
38:     dic['names'] = dict()
39:     for data in datas:
40:         if data[1] == date:
41:             name = data[0]
42:             tweet = data[2]
43:             if name not in dic['names']:
44:                 dic['names'][name] = Counter(tweet.split(" "))
45:             else:
46:                 dic['names'][name] += Counter(tweet.split(" "))
47:             dic['tweets'] += Counter(tweet.split(" "))
48:     return dic
49:
50:
51: def cal_entropy(arr_twee_num):
52:     entropy = 0.0
53:     total = sum(arr_twee_num)
54:     for val in arr_twee_num:
55:         prob =  val / total
56:         entropy = entropy + prob * log2(prob)
57:     return abs(entropy)
58:
59: def extract_counter(conter):
60:     count_values = []
61:     for _, val in dict(conter).items():
62:         count_values.append(val)
63:     return count_values
64:
65: def cal_user_entropy(user_info):
66:     return cal_entropy(extract_counter(user_info))
67:
68:
69: def cal_sys_entropy(tweets_info):
70:     return cal_entropy(extract_counter(tweets_info))
71:
72: def main():
73:     """ entry point of the application"""
74:     system_entropies = []
75:     average_entropies = []
76:
77:     df = pd.read_csv("onlyhash.data", sep="\t", header = None)
```

```
78:     df.columns = ["user", "date", "#tag"]
79:     df.groupby('date').head()
80:     df = df[df['date'].map(is_valid_date) != False]
81:     unique_dates = df['date'].unique()
82:     for date in unique_dates:
83:         print(" Processing ", date)
84:         raw_data = df[df['date'] == date]
85:         all_data = raw_data.values.tolist()
86:         info = info_according_to_date(date, all_data)
87:         tweets = info['tweets']
88:         names = info['names']
89:         user_entropies = []
90:
91:         for name, info_user in names.items():
92:             user_entropies.append(cal_user_entropy(info_user))
93:         sum_of = sum(user_entropies)
94:         try:
95:             average_entropy = sum_of / len(user_entropies)
96:         except Exception:
97:             print("Died on ",  date)
98:             average_entropy = 0
99:
100:        average_entropies.append(average_entropy)
101:        system_entropies.append(cal_user_entropy(tweets))
102:
103:    # lets sort it
104:    average_entropies = sorted(average_entropies)
105:    system_entropies = sorted(system_entropies)
106:    draw_plot(average_entropies, system_entropies)
107:
108: if __name__ == '__main__':
109:     main()
```




From the above diagram we can see that user breadth of attention is almost constant and independent of system entropy. In the lecture slide, the system entropy starts from about 10.5 and goes by slightly increasing up to 13.Whereas in our plot the system entropy starts from 0 and goes up to 12 by slightly increasing.

## 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.

## 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:**

```
 1: import math
 2: import random
 3: import numpy as np
 4: import matplotlib.pyplot as plt
 5:
 6: def gini_coefficient(tables, guests):
 7:     """ calculates the gini coefficient"""
 8:     length = len(tables)
 9:     shared_table = dict()
10:     for table in tables:
11:         shared_table[table] = tables[table] / guests
12:     denom = 0
13:     for table in shared_table:
14:         denom += shared_table[table]
15:     denom = denom / length
16:     denom = 2 * denom
17:     temp = 0
18:     for i in shared_table:
19:         for j in shared_table:
20:             temp += abs(shared_table[i] - shared_table[j])
```

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

```
21:     temp = temp / math.pow(length, 2)
22:     gini_coe = temp / denom
23:     return gini_coe
24:
25: def draw(list1, list2, list3, list4, list5):
26:     length = len(list1)
27:     x = length * [0]
28:     for i in range(length):
29:         x[i] = i + 1
30:     plt.xticks(np.arange(0,1001,100))
31:     plt.ylim(0,1)
32:     plt.title("Gini Coeficient")
33:     plt.xlabel("Subjects")
34:     plt.ylabel("Gini Coefficient")
35:     plt.plot(x, list1, label='Simulation 1',color="b")
36:     plt.plot(x, list2, label='Simulation 2',color="r")
37:     plt.plot(x, list3, label='Simulation 3',color="g")
38:     plt.plot(x, list4, label='Simulation 4',color="y")
39:     plt.plot(x, list5, label='Simulation 5',color="k")
40:     plt.legend(loc=0,fontsize='small')
41:     plt.show()
42:
43: def sitting_at_table(customer_count):
44:     tables = dict()
45:     all_gini_coe = 1000 * [0]
46:     tables[1] = 1
47:     number_of_tables = 1
48:     number_of_all_guests = 1
49:     re = gini_coefficient(tables, number_of_all_guests)
50:     all_gini_coe[number_of_all_guests - 1] = re
51:
52:     for i in range(2, customer_count + 1):
53:         number_of_all_guests += 1
54:         found = 0
55:         rand = random.random()
56:         for table in tables:
57:             prob = tables[table] / number_of_all_guests
58:             if rand < prob:
59:                 tables[table] += 1
60:                 found = 1
61:                 re = gini_coefficient(tables, number_of_all_guests)
62:                 all_gini_coe[number_of_all_guests - 1] = re
63:                 break
64:         if found == 0:
65:             number_of_tables += 1
66:             tables[number_of_tables] = 1
67:             re = gini_coefficient(tables, number_of_all_guests)
68:             all_gini_coe[number_of_all_guests - 1] = re
69:     return all_gini_coe
```

```
70:
71:
72: def main():
73:     """ main """
74:     customers_num = 1000
75:     print("Simulating1... ")
76:     all_gini_coeff1 = sitting_at_table(customers_num)
77:     print("Simulating2... ")
78:     all_gini_coeff2 = sitting_at_table(customers_num)
79:     print("Simulating3... ")
80:     all_gini_coeff3 = sitting_at_table(customers_num)
81:     print("Simulating4... ")
82:     all_gini_coeff4 = sitting_at_table(customers_num)
83:     print("Simulating5... ")
84:     all_gini_coeff5 = sitting_at_table(customers_num)
85:     draw(all_gini_coeff1, all_gini_coeff2, all_gini_coeff3, all_gini_coeff4, all_g
86:
87:
88: if __name__ == '__main__':
89:     main()
```

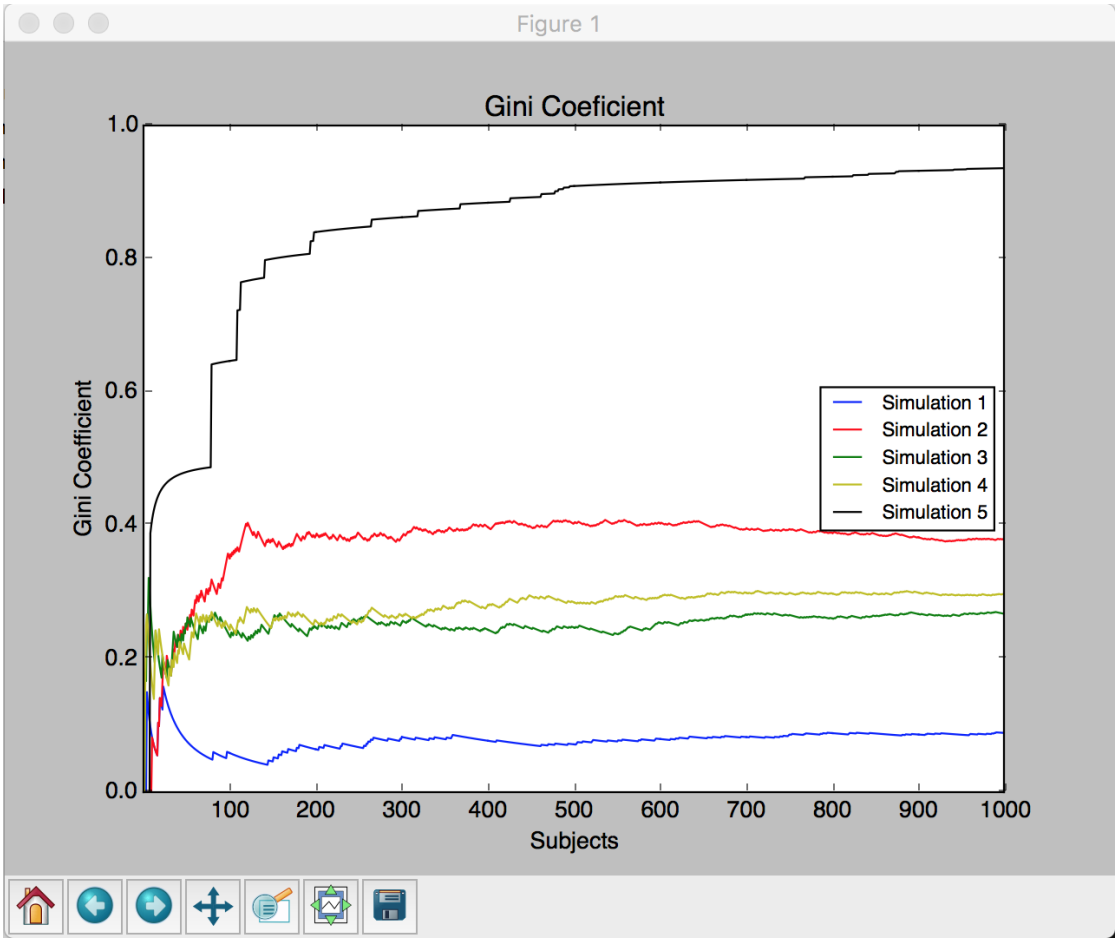


Figure 1
Gini Coeficient
Gini Coefficient
Subjects
1.0
0.8
0.6
0.4
0.2
0.0
100
200
300
400
500
600
700
800
900
1000
Simulation 1
Simulation 2
Simulation 3
Simulation 4
Simulation 5

## 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.

Answers:
Part 1 : The Secret
Estimated height of Ehrenbreitstein Fortress

| Members | Height assumption | Reason |
|---|---|---|
| 1 | 85 | Ehrenbreitstein Fortress is a castle located on the mountain of the town of Koblenz. If the height of Koblenz is 74 m above the sea level, the height of the Ehrenbreitstein Fortress should be higher than town. |
| 2 | 78 | As the Ehrenbreitstein Fortress is situated on the Rhine river bank, it can be affected by flood anytime. So that, the height of Ehrenbreitstein Fortress must be slightly higher than the town. |
| 3 | 110 | Ehrenbreitstein Fortress should have more height than Koblenz due to two reasons: 1) If it is on the height, it can provide protection from the war to the people who are living in that castle. 2) People can see whole view of the town from the castle. |

Part II : The Discussion

Estimated height of Fernmeldeturm Koblenz

| Members | Height assumption | Reason |
|---|---|---|
| 1 | 74 | Fernmeldeturm Koblenz lies on the town of koblenz. So, it has same height as of Koblenz. |
| 2 | 80 | Fernmeldeturm Koblenz is a telecommunications tower. It should be on the mountain from where it can communicate with antennas and many other signals easily. |
| 3 | 100 | Fernmeldeturn Koblenz must have a tall structure which can help in broadcasting and telecommunications. |

Calculations

Calculations for Ehrenbreitstein Fortress

Mean : $\mu = \frac{85 + 78 + 110}{3}$

$= 91$

Standard deviation : $\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$

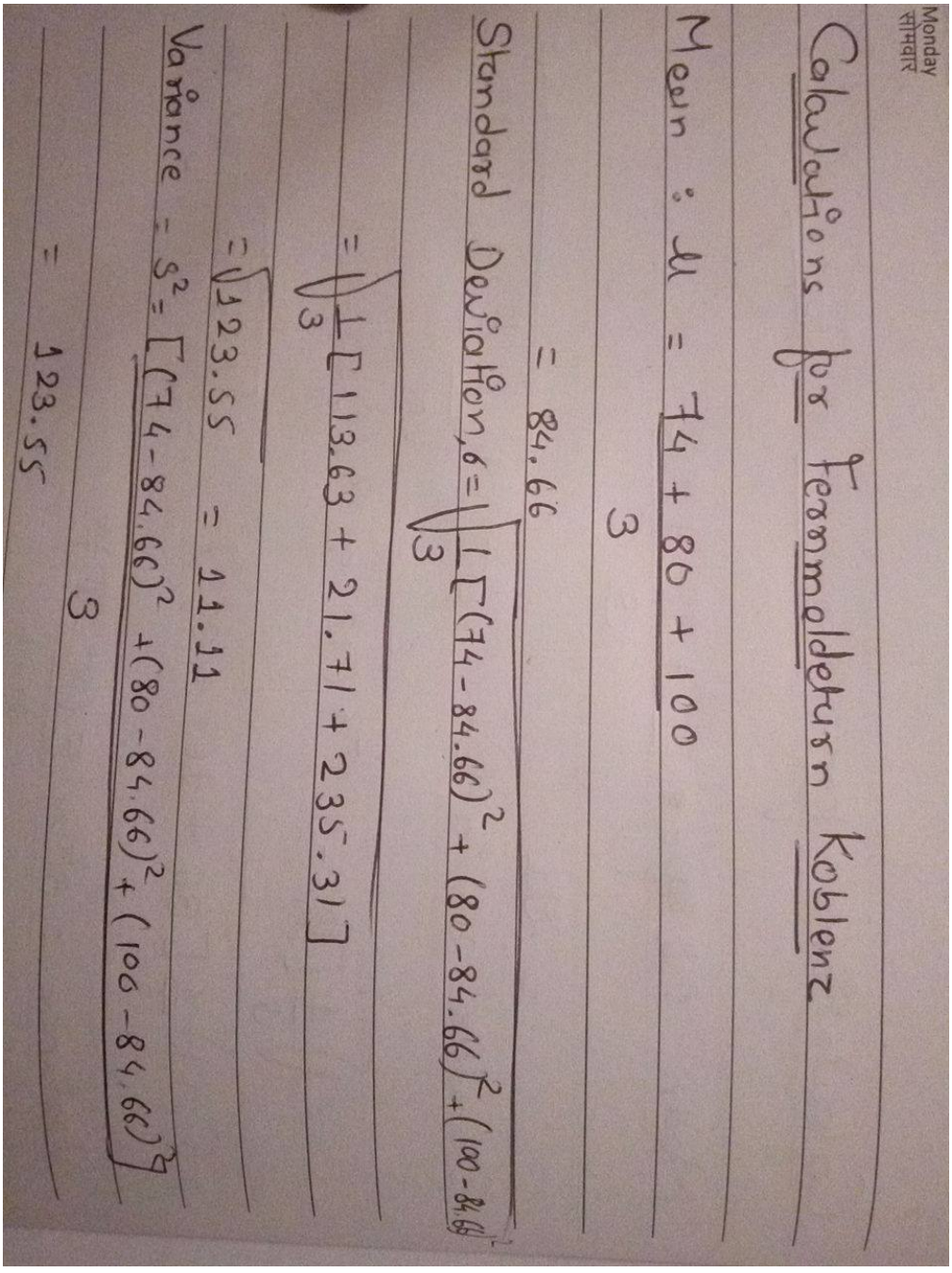
$= \sqrt{\frac{1}{3}[(85-91)^2 + (78-91)^2 + (110-91)^2]}$

$= \sqrt{\frac{1}{3}[36 + 169 + 361]} = \sqrt{188.66} = 13.73$

Variance : $s^2 = \frac{\sum(x_i - \bar{x})^2}{n-1}$

$= \frac{[(85-91)^2 + (78-91)^2 + (110-91)^2]}{2}$

$= \frac{566}{3} = 188.66$



Monday
सोमवार

Calculations for Femmoldetuon Koblenz

Mean : $\mu = \frac{74 + 80 + 100}{3}$

$= 84.66$

Standard Deviation, $\sigma = \sqrt{\frac{1}{3}[(74-84.66)^2 + (80-84.66)^2 + (100-84.66)^2]}$

$= \sqrt{\frac{1}{3}[113.63 + 21.71 + 235.31]}$

$= \sqrt{123.55} = 11.11$

Variance $= S^2 = \frac{[(74-84.66)^2 + (80-84.66)^2 + (100-84.66)^2]}{3}$

$= 123.55$

# 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`.