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: India
Group Member : Jasvinder Kaur, Jalpa Patel, Amani Gaddamedi

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 1.1

```
import numpy as np
import matplotlib.pyplot as plt
import csv
file_name = "onlyhash.data"
def read_data():
  user_tweets = {}
  system_tweets = {}
  with open(file_name, newline = '') as csvfile:
    reader = csv.reader(csvfile, delimiter='\t')
    for tweet_info in reader:
      name = tweet_info[0]
      date = tweet_info[1]
      hash_tag = tweet_info[2]
      if date not in system_tweets:
        system_tweets[date] = {'total': 0}
      system_tweets[date][hash_tag] = system_tweets[date].get(hash_tag, 0) + 1
      system_tweets[date]['total'] += 1
      if date not in  user_tweets:
        user_tweets[date] = {}
      if name not in user_tweets[date]:
```

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

```
21:         user_tweets[date][name] = {"total": 0}
22:       user_tweets[date][name][hash_tag] = user_tweets[date][name].get(hash_tag, 0)
23:       user_tweets[date][name]["total"] += 1
24:   return (user_tweets, system_tweets)
25:
26: def calculate_average_user_entropy(user_tweets):
27:   entropy = {}
28:   daily_average_entropy = {}
29:   probability = 0
30:   no_of_user = 0
31:   for date, user_hash_tags in user_tweets.items():
32:     daily_average_entropy[date] = 0
33:     for user, hash_tags in user_hash_tags.items():
34:       entropy[user] = 0
35:       for hash_tag, tweet_count in hash_tags.items():
36:         probability = tweet_count/hash_tags['total']
37:         entropy[user] += probability * np.log(probability)
38:       entropy[user] *= -1
39:     for user, user_entropy in entropy.items():
40:       daily_average_entropy[date] += user_entropy
41:     no_of_user = len(entropy.keys())
42:     daily_average_entropy[date] /= no_of_user
43:   return  daily_average_entropy
44:
45: def calculate_average_system_entropy(system_tweets):
46:   probability = 0
47:   daily_entropy = {}
48:   probability = 0
49:
50:   for date, hash_tags in system_tweets.items():
51:     daily_entropy[date] = 0
52:     for hash_tag, tweet_count in hash_tags.items():
53:       probability = tweet_count/ hash_tags['total']
54:       daily_entropy[date] += probability * np.log(probability)
55:     daily_entropy[date] *= -1
56:   return daily_entropy
57: def plot_entropy():
58:   data = read_data()
59:   user_entropy = calculate_average_user_entropy(data[0])
60:   system_entropy = calculate_average_system_entropy(data[1])
61:   dates = sorted(user_entropy.keys())
62:   x = [user_entropy[d] for d in dates]
63:   y = [system_entropy[d] for d in dates]
64:   fig = plt.figure()
65:   plt.plot(x, y, 'o', c='blue', alpha=0.8, markeredgecolor='none')
66:   plt.legend()
67:   plt.title("Entropy scatter plot")
68:   plt.xlabel("User Entropy")
69:   plt.ylabel("System Entropy")
```

```
70:     fig.savefig("entropy.png", transparent=True, bbox_inches='tight', pad_inches=0)
71:     plt.show()
72:
73: plot_entropy()
```

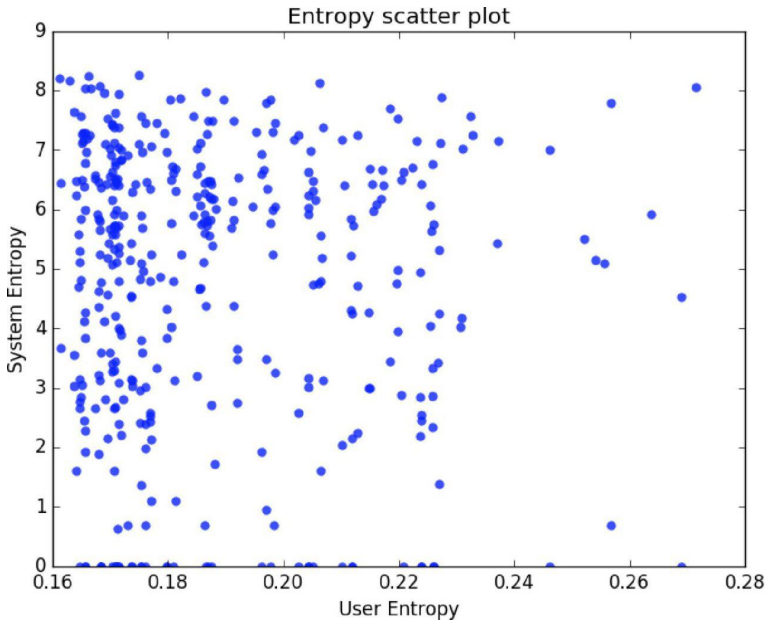


Figure 1: Entropy

Answer 1.2

It shows from the scatter plot that user average entropy is getting increase mainly in range of 0.16 to 0.22 and then becoming constant while system entropy is increasing from 0 to 8. It is same as the author interpretation but it contradict on Line plot which clearly shows system entropy but not user average entropy.Because this graph have two different scale plotted in line graph(In system entropy it is from 0 to 11 while user entropy from 0 to 1).Multiple scales can be used in the scatter plot, when you want to compare several markers with significantly different value ranges.

Reference : https://docs.tibco.com/pub/spotfire/6.5.2/doc/html/scat/scat_what_is_a_scatter_plot.htm

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 2

1. chinese_restaurant.py

```
1: import random
2: import json
3:
4: def generateChineseRestaurant(customers):
5:     # First customer always sits at the first table
6:     tables = [1]
7:     #for all other customers do
8:     for cust in range(2, customers+1):
9:             # rand between 0 and 1
10:            rand = random.random()
11:            # Total probability to sit at a table
12:            prob = 0
13:            # No table found yet
14:            table_found = False
15:            # Iterate over tables
16:            for table, guests in enumerate(tables):
17:                # calc probability for actual table an add it to total probab
18:                prob += guests / (cust)
19:                # If rand is smaller than the current total prob., customer u
20:                if rand < prob:
```

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

```
                    # incr. #customers for that table
                    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 tab
            if not table_found:
                tables.append(1)
    return tables


restaurants = [200,400,600,800,1000]
#print(restaurants)

single_simulation_data = []
final_simulation_data = []
for i in range(5):
    for customers in restaurants:
        network = generateChineseRestaurant(customers)
        single_simulation_data.append(network)
    #with open('network_' + str(customers) + '.json', 'w') as out:
    final_simulation_data.append(single_simulation_data)
    single_simulation_data = []

with open('network_' + str(customers) + '.json', 'w') as out:
    json.dump(final_simulation_data, out)
```

2. Output of Chinese Restaurant obtained in network_1000.json

3. India_assignment10_giniCoefficient.py

```
import matplotlib.pyplot as plot
import numpy as np
import pandas as pd
import json
import matplotlib.pyplot as plt

def cum_sum(given_array):
  array_after_cumsum = []
  array_after_cumsum = np.cumsum(given_array)
  return array_after_cumsum

# x = given array
# y = cumsum array of array x
def gini_coefficient(x, y):
  diff = 0
  for i in x:
    for j in x:
      diff+= np.abs(np.subtract(i,j))
```

```
19:   denominator = 2*len(y)*y[-1]
20:   gini_value = diff/denominator
21:   return gini_value
22:
23: if __name__ == "__main__":
24:   with open('network_1000.json') as json_data:
25:     sample_array = json.load(json_data)
26:   gini_array = []
27:   gini_arrays = []
28:   fig = plt.figure()
29:   for array_in_array in sample_array:
30:     for current_array in array_in_array:
31:       cumsum_dataset = cum_sum(current_array)
32:       gini_output = gini_coefficient(current_array, cumsum_dataset)
33:       gini_array.append(gini_output)
34:     gini_arrays.append(gini_array)
35:
36:     gini_array = []
37:   print(gini_arrays[0])
38:   plt.plot([200,400,600,800,1000], gini_arrays[0], 'b-')
39:   plt.plot([200,400,600,800,1000], gini_arrays[1], 'b--')
40:   plt.plot([200,400,600,800,1000], gini_arrays[2], 'r-')
41:   plt.plot([200,400,600,800,1000], gini_arrays[3], 'r--')
42:   plt.plot([200,400,600,800,1000], gini_arrays[4], 'g-')
43:   plt.legend()
44:   plt.title("Gini coefficient plot")
45:   plt.xlabel("Customers")
46:   plt.ylabel("Gini coefficient G")
47:   fig.savefig("gini.png", transparent=True, bbox_inches='tight', pad_inches=0
48:   plt.show()
49:
50:   print('Gini coefficient for 200,400,600,800,1000 customers respectively are
```

Step 4 : Final output obtained and saved in gini.png which will be different every time when program will run.(Because of Chinese restaurant is random process)

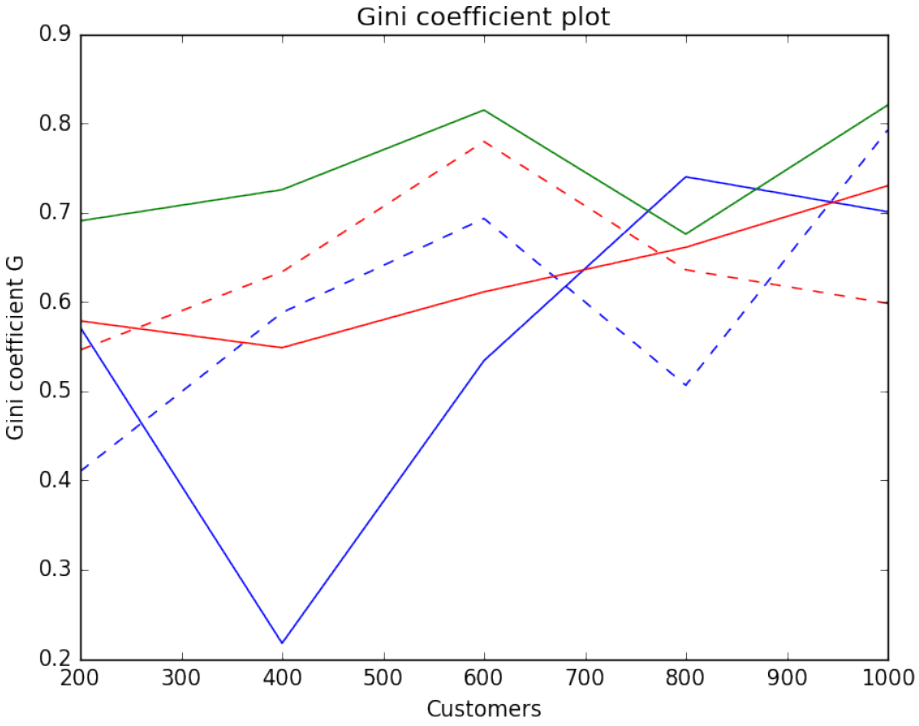


Figure 2: Gini-Coefficient

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.

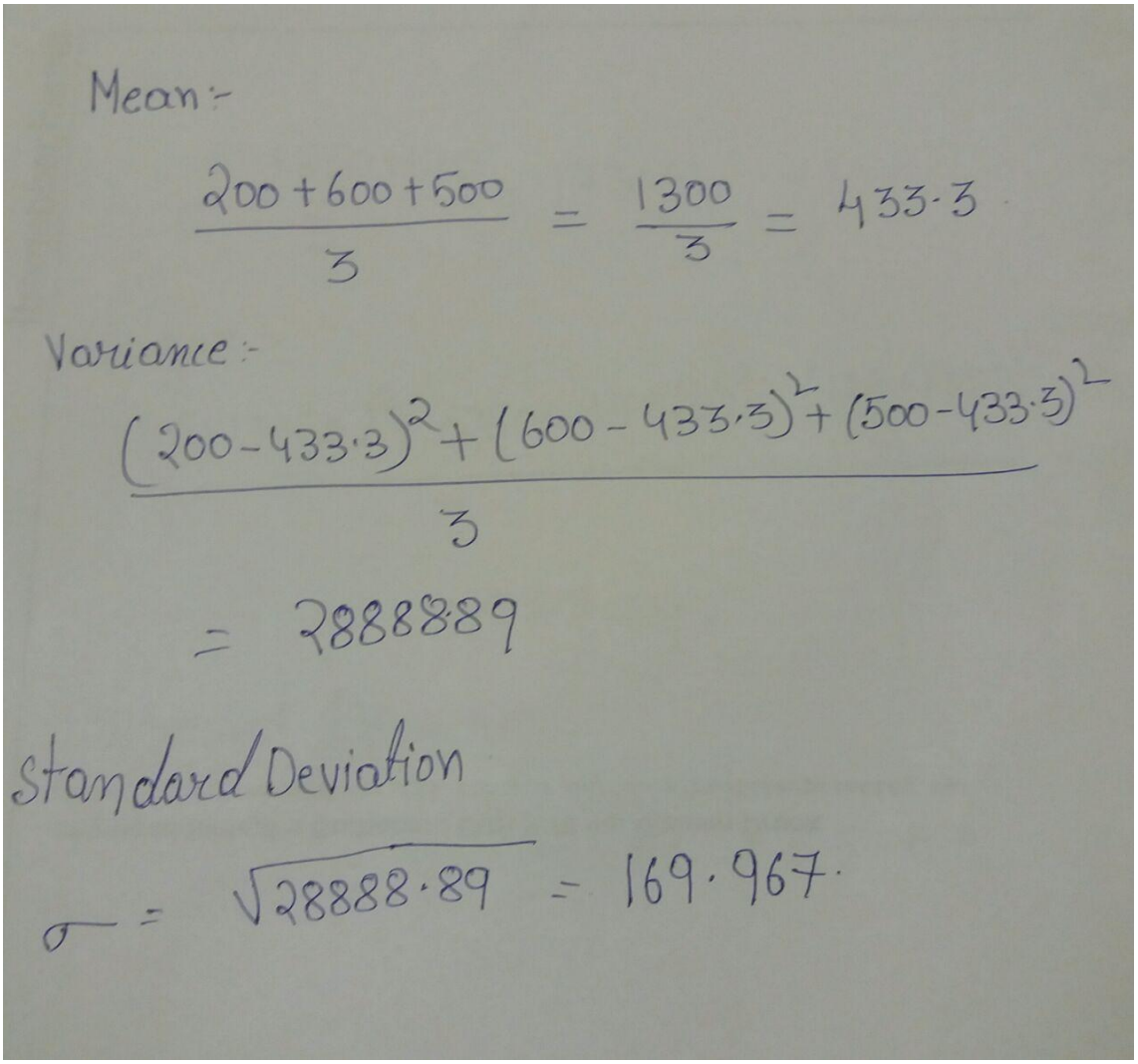
Part II : The Discussion
Discuss amongst yourself with valid reasoning what could be the height of the Fernmeldeturm Koblenz. Only after discussing, each member of the group is asked to arrive at a value and present this value in a tabulated format as was done in Part I.

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.

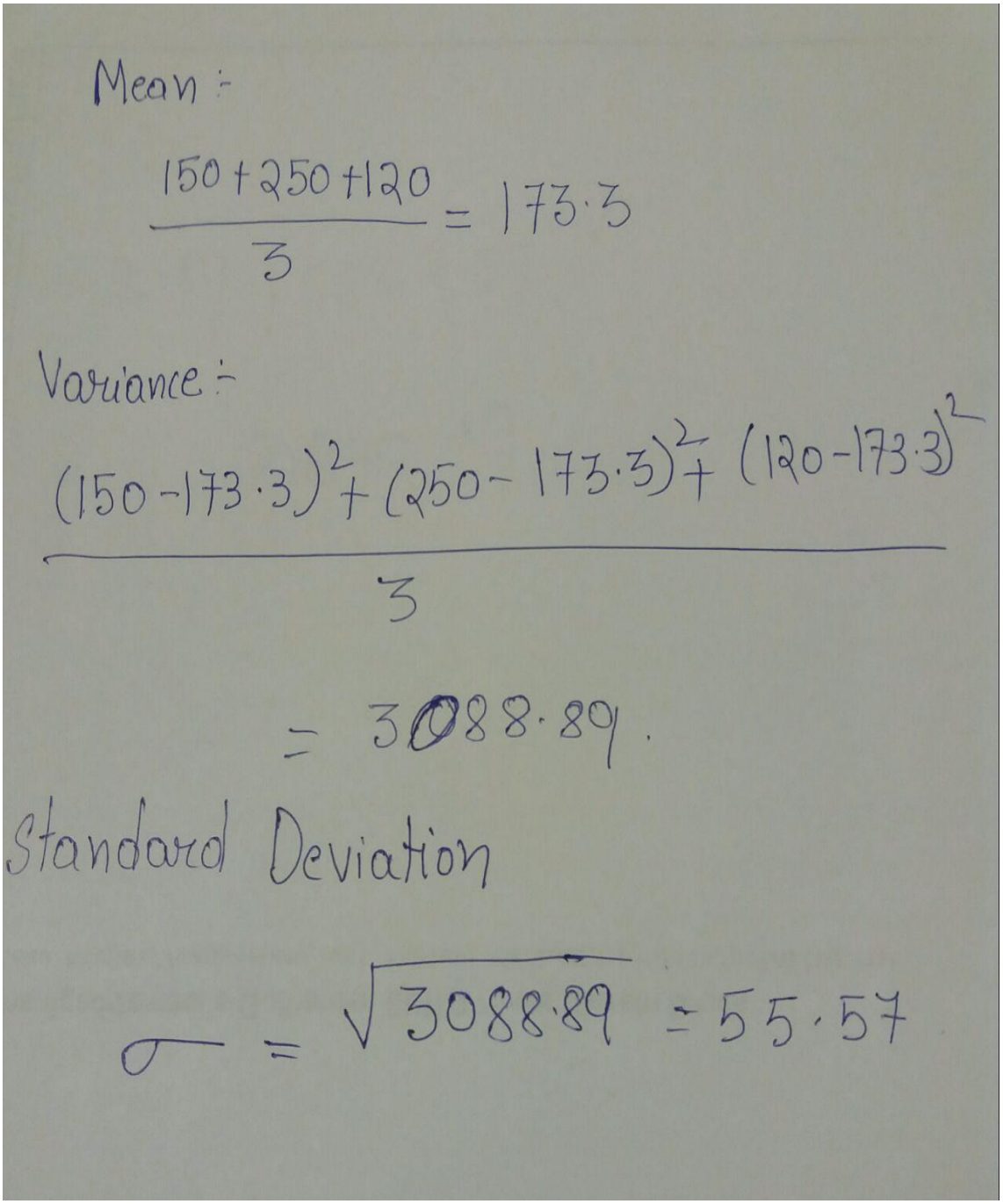
Answers Part-I

Members	value	Reasons
Amani	200m	the distance from the sea level to the top of Fortress
Jalpa	600m	THe height from sea level and apart from that it takes around 30 mins of walk on slope on hill to reach there
Jass	500m	ssuming height of one room to be 3 meters, fort seems to be equal to 170 roofs approx



Part-II

Members	value	Reasons
Amani	150	The tip of tower is seen with the distance away from te foot and is more than the tall building in city
Jalpa	250	From the sea level ,it is tower whose top is visible from anywhere in city so according to me it is not on that high level.
Jass	120	the sea level as it is at a height much more above than the fort and can be seen easily from anywhere from whole koblenz



In part 1 where the calculations for mean,variance and stander deviation are higher in value the stander deviation is double the mean in both the cases and where as in part 2 we had an discussion in the group regarding the height of tower and the values are near to the mean and no outliar can be observed in small set of data were effected by the discussion deviation of part 2 is less than the stander deviation calculated in part 1 this show that the discussion could effect behavior of the chooser.

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`.