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.**

Delta Group Members

Oana Dumitrasc
odumitrasc@uni-koblenz.de

Alisa Becker
alisabecker@uni-koblenz.de

Omar Aly
oaly@uni-koblenz.de

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.

```
1: import pandas as pd
2: import math
3: import numpy as np
4: import matplotlib.pyplot as plt
5:
6: def avgUserEntropyPerDay(users):
7:     total = len(users)
8:     sum = 0
9:     for user in users:
10:        sum += userEntropy(users[user])
11:    return sum/total
12:
13: def userEntropy(user):
14:    sum = 0
15:    entropy = 0
16:    for tag in user:
17:        sum += len(tag)
18:    for tag in user:
19:        entropy -= (len(tag)/sum) * math.log2(len(tag)/sum)
20:    return entropy
```

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

```
21:
22:
23: def systemEntropy(date):
24:     sum = 0
25:     entropy = 0
26:     for tag in date:
27:         sum += date[tag]
28:     for tag in date:
29:         entropy -= (date[tag]/sum) * math.log2(date[tag]/sum)
30:     return entropy
31:
32: data = pd.read_table("onlyhash.data", names=["user","date","hashtag"])
33: data.head()
34:
35: datesUsersHashtags = dict()
36: datesSystemHashtag = dict()
37:
38: for date in data["date"]:
39:     datesUsersHashtags[date] = dict()
40:     datesSystemHashtag[date] = dict()
41:
42: for i in range(len(data.values)):
43:     user = data.values[i][0]
44:     date = data.values[i][1]
45:     if user not in datesUsersHashtags[date]:
46:         datesUsersHashtags[date][user] = dict()
47:     hashtags = data.values[i][2].split(" ")
48:     for hashtag in hashtags:
49:         # for users
50:         if hashtag not in datesUsersHashtags[date][user]:
51:             datesUsersHashtags[date][user][hashtag] = 0
52:         datesUsersHashtags[date][user][hashtag] += 1
53:         # for system
54:         if hashtag not in datesSystemHashtag[date]:
55:             datesSystemHashtag[date][hashtag] = 0
56:         datesSystemHashtag[date][hashtag] += 1
57:
58: entropyPerDayUser = dict()
59: entropyPerDaySystem = dict()
60:
61: for date in datesUsersHashtags:
62:     entropyPerDayUser[date] = avgUserEntropyPerDay(datesUsersHashtags[date])
63:     entropyPerDaySystem[date] = systemEntropy(datesSystemHashtag[date])
64:
65: x = [x for x in range(0, len(entropyPerDayUser))]
66: yUser = list(entropyPerDayUser.values())
67:
68: ySystem = sorted(list(entropyPerDaySystem.values()))
69:
```

```
70: plt.title(" Average User Entropy Per Day and System Entropy Sorted")
71: plt.xticks(np.arange(0, max(x), 40))
72: plt.yticks(range(0,int(max(ySystem)+1)))
73: plt.xlabel("Days Rank")
74: plt.ylabel("Entropy")
75: plt.scatter(x, yUser, color = 'r')
76: plt.scatter(x, ySystem, color = 'b')
77: plt.show()
```

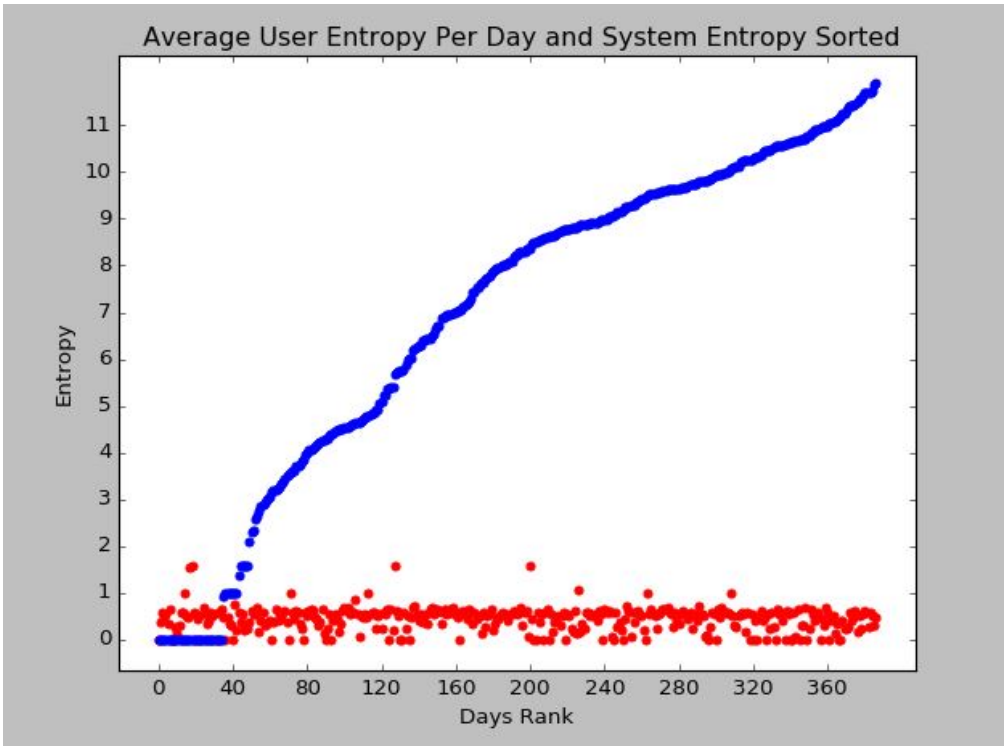

Figure 1: Average User Entropy Per Day and System Entropy Sorted

2. After comparing the two scatter plots, the one from the paper with the one present in the assignment, we can make the fair statement that the attention of an user is limited. That can be seen easily in the above graph. The entropy for the average user is mostly between 0 and 1, with some exceptions that tend to entropy 2, while the entropy of the system has a value range from 0 to almost 12. As the entropy of the system is sorted, we can not make further comments with regard to the distribution. The system entropy presented in the paper has also a bigger range, from approximately 10 to 13, compared to the user entropy, which is constantly there between 0 and We can make the same assumption as the one stated in the paper, that new memes survive only if others die.

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 random
2: import json
3: import matplotlib.pyplot as plt
4: import numpy as np
5:
6: def calcGini(ls):
7:     numerator  = np.sum(np.abs(np.subtract.outer(ls, ls)))
8:     denominator = 2 * len(ls) * np.sum(ls)
9:
10:    gini = numerator/denominator
11:
12:    return gini
13:
14: def generateChineseRestaurant(customers):
15:    # First customer always sits at the first table
16:    tables = [1]
17:
18:    #customerChances = [1]
19:    giniCoefficients = []
20:
21:    #for all other customers do
22:    for cust in range(2, customers+1):
```

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

```
23:             # rand between 0 and 1
24:             rand = random.random()
25:             # Total probability to sit at a table
26:             prob = 0
27:             # No table found yet
28:             table_found = False
29:             # Iterate over tables
30:             for table, guests in enumerate(tables):
31:                 # calc probability for actual table and add it to total probabili
32:                 prob += guests / (cust)
33:                 # If rand is smaller than the current total prob., customer will
34:                 if rand < prob:
35:                     #customerChances.append(prob)
36:                     # incr. #customers for that table
37:                     tables[table] += 1
38:                     # customer has found table
39:                     table_found = True
40:                     # no more tables need to be iterated, break out for loop
41:                     break
42:
43:             # If table iteration is over and no table was found, open new table
44:             if not table_found:
45:                 #customerChances.append(1)
46:                 tables.append(1)
47:
48:             giniCoefficients.append(calcGini(tables))
49:
50:     return tables, giniCoefficients
51:
52: restaurants = 1000
53:
54: giniCoef = list()
55:
56: giniCoefs = list()
57: networks = list()
58:
59: for i in range(5):
60:     network, giniCoef = generateChineseRestaurant(restaurants)
61:
62:     networks.append(network)
63:     giniCoefs.append(giniCoef)
64:
65:
66: with open('network_' + str(restaurants) + '.json', 'w') as out:
67:     json.dump(network, out)
68:
69:
70: plt.plot(range(np.size(giniCoefs[0])), giniCoefs[0], label='1st run')
71: plt.plot(range(np.size(giniCoefs[0])), giniCoefs[1], label='2nd run')
```

```
72: plt.plot(range(np.size(giniCoefs[0])), giniCoefs[2], label='3rd run')
73: plt.plot(range(np.size(giniCoefs[0])), giniCoefs[3], label='4th run')
74: plt.plot(range(np.size(giniCoefs[0])), giniCoefs[4], label='5th run')
75: plt.xlabel("Number Of Customers")
76: plt.ylabel("Gini Coefficient G")
77: plt.legend()
78: plt.show()
```

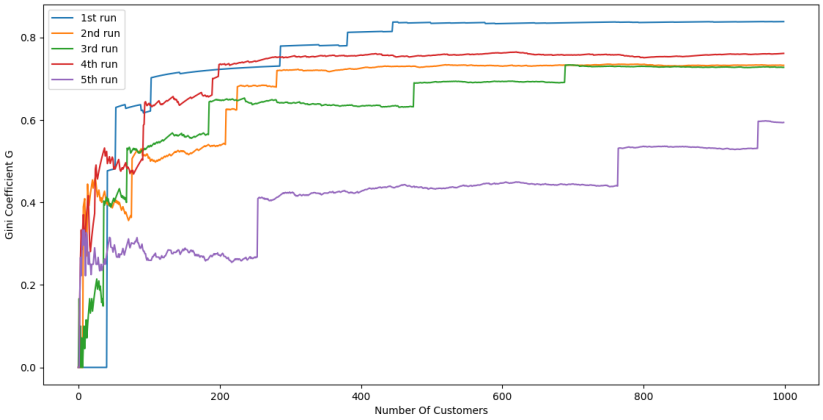

Figure 2: Gini-Coefficient

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.

Answer Part 1 : The Secret

	Alisa	Oana	Omar
Ehrenbreitstein Fortress	150m	300m	174m

Alisa: I tried to look at pictures of the fortress and compared its' height to the surrounding buildings.

Oana: Knowing that Koblenz is at a 74 m altitude and placed in a depression, I assume that the fortress is around 300 m high as it stands on small hill.

Omar: From images and from my memory when visited it and looked down to the Deutsches Eck and how people were far and the people size to the monument size from up above at the fortress.

Calculations

Mean	208
Standard Deviation	65.787
Variance	4328

Answer Part 2 : The Discussion

	Alisa	Oana	Omar
Fernmeldeturm	300m	300m	274m

Alisa: I googled for similar towers and the one in Mannheim is around 220m added to the 74m of Koblennz i guessed it's around 300.

Oana: After the discussion within the team I assume the Fernmeldeturm's height is around 300 because Alisa told us that in Mannheim there is a similar tower and that one is not higher than 250m. Knowing that Mannheim has about the same height above the see level as Koblenz it is safe to believe that the height's tower does not exceed 300.

Omar: After the discussion I assume the Fernmeldeturm's height is around 274m because from the pictures I saw it seems 200m and adding the 74m for Koblenz so its 274 from sea level.

Calculations

Mean	291.3
Standard Deviation	12.256
Variance	150.2

The conclusion of this task is pretty obvious, in the first part, where we did not talk to one another our ways of thinking were quite different, as soon as we discussed our way of thinking started to have a similar pattern. The same behavioral patterns happen also online and this is helpful, sharing knowledge pushes us forward, the question is until which point is this really useful, some might argue that individuality is lost little by little.

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`.