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: Bravo

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:
2: # coding: utf-8
3:
4: # In[9]:
5:
6: import pandas as pd
7: import math
8: from collections import Counter
9: import pylab as plt
10:
11:
12:
13: # In[2]:
14:
15: #data=pd.read_excel("C:\\Users\\ShreeH\\Desktop\\WebScience\\Introduction to web s
16: data=pd.read_excel("onlyhash.xlsx")
17:
18:
19: # In[24]:
20:
21: # calculates the average user entropy for a date
22: #input date
23: #output average user entropy for that date
```

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

```
24:
25: def UserEntropyDateWise(date):
26:     tempData=data[data['date']==date] # gives subset of data for that date
27:     singleUserEntropy={}
28:     userData=set(tempData.user)
29:     for user in userData:
30:         hashtagData=tempData['hashtag'][tempData['user']==user]
31:         #print((hashtagData),user)
32:         hashtagList=[]
33:         for hashtag in hashtagData:
34:             hashtag=str(hashtag)
35:             # if the hashtag contains multiple hashtags then split it on space
36:             if hashtag.count('#') > 1:
37:                 hashtagList=hashtagList+hashtag.split(' ')
38:             else:
39:                 hashtagList.append(hashtag)
40:         # gives the count of the hashtags in for a user
41:         c=Counter((hashtagList))
42:         #print(c)
43:         n=(sum(c.values()))
44:         userEntropy=0
45:         # calculates the user entropy
46:         for item,value in c.items():
47:             proportion=value/n
48:             userEntropy=userEntropy-(proportion*math.log2(proportion))
49:             #print("ent",userEntropy)
50:         singleUserEntropy[user]=(userEntropy)
51:
52:     return (sum(singleUserEntropy.values())/len(singleUserEntropy))
53:
54:
55:
56:
57: # In[25]:
58:
59: # calculates the system entropy for a date
60: #input date
61: #output system entropy for that date
62:
63:
64: def systemEntropy(date):
65:     #systemEntropyForDay={}
66:     hashtagData=data['hashtag'][data['date']==date]
67:     hashtagList=[]
68:     for hashtag in hashtagData:
69:         hashtag=str(hashtag)
70:         if hashtag.count('#') > 1:
71:             hashtagList=hashtagList+hashtag.split(' ')
72:         else:
```

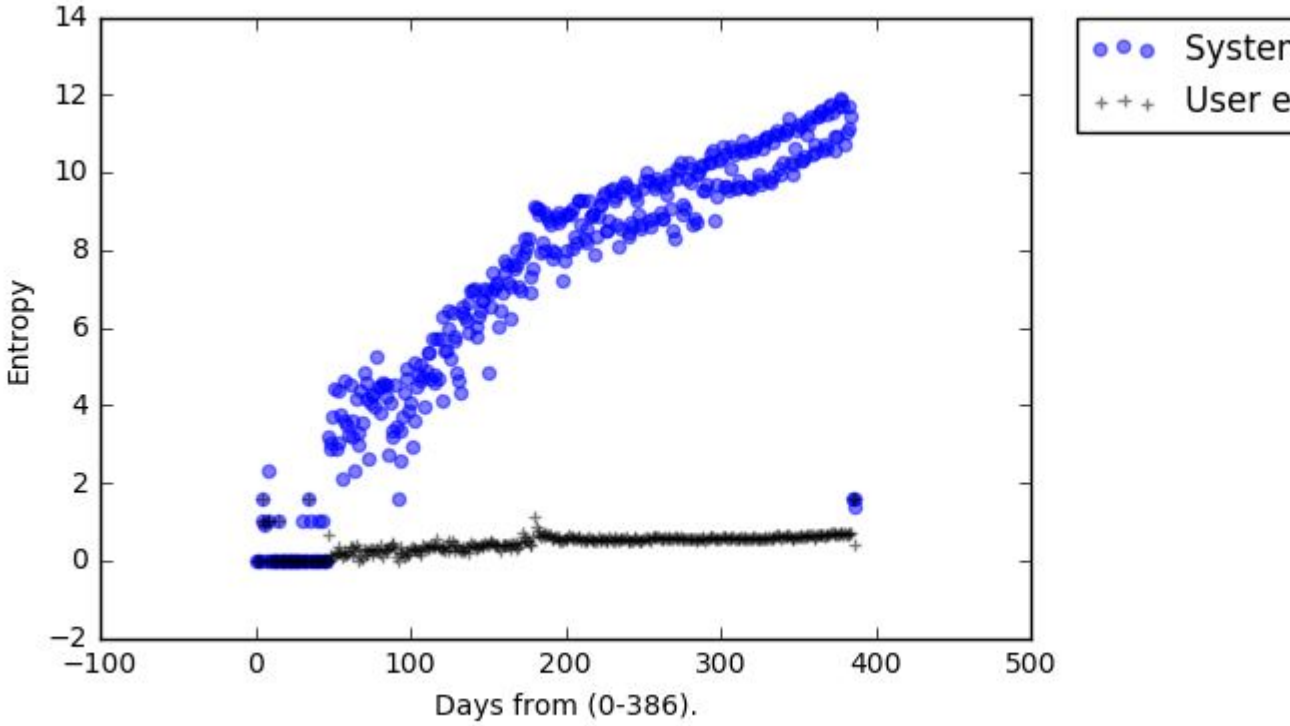
```
            hashtagList.append(hashtag)
    c=Counter((hashtagList))
    n=(sum(c.values()))
    systemEntropy=0
    for item,value in c.items():
        prop=value/n
        systemEntropy=systemEntropy-(prop*math.log2(prop))
    #systemEntropyForDay[str(date)]=systemEntropy
    return systemEntropy
```

```
# In[33]:

#prepare 2 dictionary for datewise systemEntropy and userEntropy

systemEntropyMap={}
userEntropyMap={}
uniqueDates=set(data.date)# get all the unique dates in the dataset
for date in uniqueDates:  # iterating over the unique date to find the average use
    dayWiseSystemEntropy=systemEntropy(date)
    stringDate=str(date)
    userEntropyMap[stringDate]=UserEntropyDateWise(date)
    systemEntropyMap[stringDate]=dayWiseSystemEntropy
```

```
# In[70]:

# dump the hashmap data to json file for further data processing

import json
with open ('C:\\Users\\ShreeH\\Desktop\\singleuserentropy.json', 'w') as fp:
    json.dump(userEntropyMap,fp)

with open ('C:\\Users\\ShreeH\\Desktop\\systementropy.json', 'w') as fp:
    json.dump(systemEntropyMap,fp)


# In[80]:

##loading the json data to the dictionary
with open('C:\\Users\\ShreeH\\Desktop\\systementropy.json') as data_file:
    systemEntropyMap = json.load(data_file)
```

```
122:
123: with open('C:\\Users\\ShreeH\\Desktop\\singleuserentropy.json') as data_file:
124:     userEntropyMap = json.load(data_file)
125:
126:
127:
128: # In[81]:
129:
130: #prepare the data for plotting
131: # sort the user entropy and the system entropy on the basis of the date
132:
133: sortedUserEntropyList=[]
134: xpoint=[]
135: i=0
136: for key in sorted(userEntropyMap):
137:     xpoint.append(i)
138:     i=i+1
139:
140:     sortedUserEntropyList.append(userEntropyMap[key])
141:
142: sortedSystemEntropyList=[]
143:
144: for key in sorted(systemEntropyMap):
145:     sortedSystemEntropyList.append(systemEntropyMap[key])
146:
147:
148: # In[87]:
149:
150: plt.xlabel("Day(0-384): 0 being the lowest system entropy and 384 the highest.")
151: plt.ylabel("Entropy")
152: systemEntropyPlot = plt.scatter(xpoint, sortedSystemEntropyList, color='b', marke
153: userEntropyPlot = plt.scatter(xpoint, sortedUserEntropyList, color='k', marker='+
154: plt.legend(handles=[systemEntropyPlot, userEntropyPlot],bbox_to_anchor=(1.05, 1),
155: plt.show()
```

Sol 2: as seen in the scatter plot the user entropy remains at constant level irrespective of the variations in the system entropy. which can be interpreted as the average breadth of user's attention remains constant irrespective of the diversity in the system. this observed behavior is compatible with the findings of the author,which also represents that even though the system is made up of diverse memes, the users attention is confined to very few of them.


14
12
10
8
6
4
2
0
−2
Entropy
−100
0
100
200
300
400
500
Days from (0-386).
Syster
User e

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.

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

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.

Sol :Part I

Member	Ehrenbreitstein	Fernmeldeturm	Reason1	Reason2
Daniel	174	274	Distance from Koblenz roughly 100 then plus with 74	200+74
Kandy	20	150	The festung is 20 meter high from ground	estimate
Shriharsh	12	200	The festung is not high from ground	Fernmeldeturm would ideally have an height greater than 100 meter

Above table Before Discussion

Member	Ehrenbreitstein	Fernmeldeturm
Daniel	15	150
Kandy	10	150
Shriharsh	12	120

Above table After Discussion

Case1	Ehrenbreitstein	Fernmeldeturm
Mean	68.66667	208
Standard deviation	91.309	62.3859
Variance	8337.33333	3892

Case2	Ehrenbreitstein	Fernmeldeturm
Mean	12.33333	140
Standard deviation	2.51661	17.32
Variance	6.33333	300

Sol :Part II

We can see that before discussion the Variance and standard deviation were quite high in both cases, but after some discussion and guessing again we can see that both Variance and standard deviation are reduced considerably. But it does not necessary means that the answers are directed in the correct way

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`.