Tanja Crijns s4204999 - Exercises week 6: Query log data

Exercise 1:

For this exercise, I used the python package 'pandas' to read and represent the data and to extract the statistics. Pandas is a data analysis library.

Extract and show the following basic statistics from the data:

- *The number of unique queries*

There are 21806 unique queries.

- *The top-10 most frequent queries*

Query	Frequency
seks	801
forum	781
anaal	765
sex	755
zwanger	673
trio	536
vreemdgaan	511
pijpen	510
vakantie	357
sauna	322

- *The top-10 most clicked URLs*

URL	Frequency
http://forum.viva.nl/?utm_medium=cpc&utm_source=startpagina&utm_campaign=20140702_viva_startpaginabarterdeal2014&utm_content=tekstlink&utm_term=forumleesmeer	47
http://forum.viva.nl/forum/relaties/hersenbloeding-bij-vriend/list_messages/269993	36
http://forum.viva.nl/forum/seks/on-topic-hij-wil-zo-vaak-anaal/list_messages/259609	34
http://forum.viva.nl/forum/zwanger/kind-in-je-uppie-deel-6/list_messages/247697	31
http://forum.viva.nl/forum/list_messages/244326	29
http://forum.viva.nl/forum/psyche/hier-schrijf-ik-graag-verder-van-mij-af/list_messages/249131	29

http://forum.viva.nl/forum/kinderen/kind-misbruikt-hoe-nu-verder/list_messages/248992	27
http://forum.viva.nl/forum/overig/bantopic/list_messages/235701	26
http://forum.viva.nl/forum/seks/beschrijf-je-laatste-neuk/list_messages/71343	26
http://forum.viva.nl/forum/Relaties/list_topics/1?utm_medium=cpc&utm_source=startpagina&utm_campaign=20140702*viva*startpaginabarterdeal2014&utm_content=tekstlink&utm_term=forumrelaties	25

Exercise 2:

- ***Create a vector space representation (a matrix) with as rows each of the unique queries from the top-10, as dimensions (columns) the clicked URLs occurring in the data and as cell values the click count for the query on the URL***
- ***For each query pair in the top-10 most frequent queries, calculate the cosine similarity between the queries using the matrix of click counts***
- ***Show a 10-by-10 matrix with the top-10 most frequent queries as rows and columns and a cosine similarity score in each cell. The diagonal will be all 1s. You can keep the bottom half of the matrix empty (because it is symmetric)***

For this exercise, I also used pandas for data extraction. I represented the vector space represenation as a two-dimensional python array (as a matrix).

Cosine similarity matrix:

	Seks	Forum	Anaal	Sex	Zwanger	Trio	Vreemdgaan	Pijpen	Vakantie	Sauna
Seks	1.	0.	0.01	0.15	0.01	0.01	0.	0.01	0.	0.03
Forum	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.
Anaal	0.01	0.	1.	0.01	0.	0.	0.	0.01	0.	0.
Sex	0.15	0.	0.01	1.	0.	0.	0.02	0.	0.	0.02
Zwanger	0.01	0.	0.	0.	1.	0.	0.	0.	0.	0.
Trio	0.01	0.	0.	0.	0.	1.	0.	0.	0.	0.
Vreemdgaan	0.	0.	0.	0.02	0.	0.	1.	0.01	0.	0.
Pijpen	0.01	0.	0.01	0.	0.	0.	0.01	1.	0.	0.
Vakantie	0.	0.	0.	0.	0.	0.	0.	0.	1.	0.
Sauna	0.03	0.	0.	0.02	0.	0.	0.	0.	0.	1.

Code

```
__author__      = "Tanja Crijns"
import pandas as pd, numpy as np
from sklearn.metrics.pairwise import cosine_similarity
from scipy import sparse

if __name__ == "__main__":
    df = pd.read_csv("D:/Users/Tanja/Documents/Master/Information retrieval/
    Assignment2/Querylog.csv", sep=';')
    queries = df['Query']
    urls = df['Clicked link']
    uniqueQueries = queries.value_counts()
    uniqueURLs = urls.value_counts()

    # Number of unique queries
    print "Unique queries = " + str(len(uniqueQueries))

    # Top 10 most frequent queries
    topTenQueries = uniqueQueries[:10]
    print "Most frequent queries = \n" + str(topTenQueries)

    # Top 10 most clicked URLs
    uniqueURLs = uniqueURLs.drop("\N")
    topTenURLs = uniqueURLs[0:10]
    print "Most frequent URLs = \n" + str(topTenURLs)

    # Vector space representation
    URLsTopQueries = df[df.Query.isin(topTenQueries.index)]['Clicked link']
    URLsTopQueries = URLsTopQueries[URLsTopQueries != '\N']
    vectorSpace = np.zeros(shape=(len(topTenQueries),len(URLsTopQueries)))
    logs = df[df.Query.isin(topTenQueries.index)]
    logs = logs[logs['Clicked link'] != '\N']
    queriestmp = topTenQueries.index.tolist()
    URLstmp = URLsTopQueries.values.tolist()

    # Making the matrix
    for index, log in logs.iterrows():
        vectorSpace[queriestmp.index(log['Query'])][URLstmp.index(log
            ['Clicked link'])] += 1

    # Cosine calculation
    vectorSpace_sparse = sparse.csr_matrix(vectorSpace)
    np.set_printoptions(suppress=True)
    similarities = cosine_similarity(vectorSpace_sparse)
    print similarities.round(decimals=2)
```