Homework 2

Web Information Retrieval

Deadline: **25** May 2016

Exactly Two students for each group

Data and software are available at:

http://www.diag.uniroma1.it/~fazzone/Teaching/WebIR_2016_2017/WebIR_2016_2017.html

Description:

In this homework you are going to solve recommendation problems using link-analysis techniques. In particular, you have to use Topic-Specific-PageRank on a graph representing movies and relations among them, to obtain recommendations. This graph is called "Movie-Graph".
This graph has been obtained using the MovieLens 100K Dataset. This dataset contains 100000 ratings of 943 users on 1682 movies. Each rating is a natural number in [1, 5].

The homework is splitted in three parts:

- First part: implement a method for the computation of the Topic-Specific-PageRank able to work with a sparse matrix representation of a weighted graph.
- In the second part, you have to implement a movie recommendation method using a particular link-analysis procedure based on Topic-Specific-PageRank (explained below).
- In the third part, you have to create another movie recommendation method, always based on Topic-Specific-PageRank, that is able to recommend movies without computing PageRank at recommendation time (explained below).

DataSets description:

In this homework you will use the following datasets:

Movie-Graph:
The Movie-Graph is a weighted and undirected graph in which nodes represent movies_ids and edges between nodes represent a particular relation between movies. For the aim of the homework, it is not important to know the nature of this relation. Of course, if you want to know more about that, you are invited to use piazza.
The file "`movie_graph.txt`" represents the Movie-Graph one edge per line according with the following format: "`movie_id \t movie_id \t weight`"
The lines are sorted by ascending order of the first column and then by the second column.

User-Movie-Rating-Data:
The file "`user_movie_rating.txt`" contains real data about ratings from users on movies.
The file contains 100000 ratings, one rating per line, in the following format:
"`user_id \t movie_id \t rating`"
The rows of the file are grouped by user_id.

Movie-Categories-Data:
The file "category_movies.txt" contains data about movies categories: Drama, Sci-Fi, Animation, Documentary...
Each line of this file represents the set of movie identifiers, separated by \t, belonging to the category with identifier equals to the line number.
In total you have five categories, with id in [1, 5].
You will use "category_movies.txt" file **only in part three**.

Zoom On Part One:

In this part of the homework you have to implement a method for the computation of the Topic-Specific-PageRank able to work with a sparse matrix representation of the input graph (undirected and weighted graph). For this purpose, you must modify the algorithm presented in Lab_2.
You will use this implementation, in a black box way, to solve the other two parts of the homework ;)

Zoom On Part Two:

The goal of this part is to implement a method for recommending movies to users contained in the file "user_movie_rating.txt".
The recommendation must be performed by applying Topic-Specific-PageRank on Movie-Graph, by considering as nodes of the topic all movies rated by the input user. To increase the personalization of the recommendation, the teleporting probability distribution among all nodes in the virtual topic must be biased using the rating values that the input user has given to each rated movies.
For example, let's consider a Movie-Graph of only four nodes: movie_1, movie_2, movie_3 and movie_4.
If user_1 has rated movie_1 with 4, movie_2 with 2 and no more ratings are present, the teleporting probability distribution for user_1 on the nodes of the Movie-Graph is the following: {movie_1: 4/(4+2), movie_2: 2/(4+2), movie_3: 0, movie_4: 0}.

Your recommendation method must be embedded in a software called WebIR_HW_2_part_2.
This software must run by command line in the following way:

```
python WebIR_HW_2_part_2.py ./datasets/movie_graph.txt ./datasets/user_movie_rating.txt INPUT_USER_ID
```

In output this software must print on screen a single sorted list of couples movies_ids-PR_score in **descending** order of PR value.
Here you have an example of the output format:

```
153, 0.42
191, 0.21
426, 0.17
815, 0.02
…
```

Of course, the output list must not contain movies_id associated to movies already rated by the input user.

VERY IMPORTANT RULE:
Your software must print on screen only the output of the recommendation process, nothing more and nothing less than that.

Zoom On Part Three:

In this part you have to implement a recommendation method different from the one explained in the previous part. In particular, this method must be able to recommend movies, by taking in input a user preferences-vector. This preferences-vector can be considered as a representation of the user in the space of the movie categories.
The output of the recommendation **must be** a sorted list of movies_id in descending order of Topic-Specific-PageRank computed by biasing the teleporting procedure according to the content of the input preference-vector. Unfortunately, in this part of the homework, it is not allowed to compute the PageRank at recommendation time.

In the datasets, we have five categories with identifiers in [1, 5] and we are representing a preferences-vector as a string with this format:

```
WeightInCategory1_WeightInCategory2_WeightInCategory3_WeightInCategory4_WeightInCategory5
```

For example, the string `3_4_0_1_1` represents a preference_vector with coordinates 3 in the first category, 4 in the second, 0 in the third, 1 in the fourth and 1 in the fifth category.

As said before, this movie recommendation method must be based on Topic-Specific-PageRank and must also be able to recommend movies without computing PageRank at recommendation time.
For this reasons, it is requested to develop two softwares: `WebIR_HW_2_part_3_offline` and `WebIR_HW_2_part_3_online`.

The software `WebIR_HW_2_part_3_offline` is responsible of the generation of input-independent data that will be used by `WebIR_HW_2_part_3_online` software at recommendation time.
It is requested that `WebIR_HW_2_part_3_offline` must store the input-independent data inside the directory "`datasets`" using text files.

The software `WebIR_HW_2_part_3_online` must use the input-independent data generated by `WebIR_HW_2_part_3_offline`, to perform a movie recommendation for the input user represented by the input preferences-vector.
This software must be run by command line in the following way:

```
python WebIR_HW_2_part_3_online.py USER_preferences-vector
```

For example, the software can be run in this way:

```
python WebIR_HW_2_part_3_online.py 3_4_0_1_1
```

In output this software must print on screen a single sorted list of couples movies_ids-PR_score in **descending** order of PR value.
Here you have an example of the output format:

```
153, 0.42
191, 0.21
426, 0.17
815, 0.02
...
```

VERY IMPORTANT RULE:
Your software must print on screen only the output of the recommendation process, nothing more and nothing less than that.

To complete the homework, you have to complete these steps:

1. Download and decompress the zip file "WebIR_HW_2_datasets.zip" on your machine.
   You can find what you need at this link
   http://www.diag.uniroma1.it/~fazzone/Teaching/WebIR_2016_2017/WebIR_2016_2017.html in section "Homework_2".
2. Complete part 1, 2 and 3.
3. Create a final report **in PDF**.
4. Create a zip file with this content:
   1) `WebIR_HW_2_part_2.py , WebIR_HW_2_part_3_offline.py and WebIR_HW_2_part_3_online.py`
   2) The original version of `Network_Based_Recommendation_System.py`.
   3) A single human-readable file containing the final report **in PDF**.
   4) An empty directory named "`datasets`"
   4) Nothing less, nothing more.
5. The file name must have this format:
   `WebIR_Homework_2__STUDENTID_NAME_SURNAME__STUDENTID_NAME_SURNAME.zip`
6. Finally, you must to send this zip file to fazzone@diag.uniroma1.it with this email subject: `WebIR_Homework_2__STUDENTID_NAME_SURNAME__STUDENTID_NAME_SURNAME`.

What To Put in the Report:

The final report must contain:

1. Data about the used datasets:
   Number of nodes and number of edges in Movie-Graph.
   Number of users, number of movies and number of reviews contained in the `user_movie_rating.txt`.

2. A short, but complete, description of the changes made to the algorithm for PageRank presented in Lab_2 for addressing what requested in the first part.

3. A short, but complete, description of the approach you followed to address what requested in the third part.

For any problem/doubt/question, you must use Piazza.
Please, don't be shy :)