CS685: Data Mining

Assignment 2 (150 marks)

Due on: 20th November, 2020, 6:00pm

Visit the website `http://snap.stanford.edu/data/wikispeedia.html`. We will work with the `wikispeedia_paths-and-graph.tar.gz` listed there. There are 6 *tab-separated* `.tsv` files in it.

Submit all the necessary components, files and data of your assignment as a single `zip` file named `rollno-assign2.zip` in the `hello.iitk.ac.in` portal within the deadline.

While the programs/scripts should be named `.sh` files, you can invoke any program from within the shell file. The programs should run in the Linux operating system.

All the output files should be sorted by the first field. If the output file is `.csv`, it should be in *comma-separated* format.

1. (5 marks) Augment the names of the articles in `articles.tsv` in the order given to have ids from `A0001` to `A4604`.

   Subsequently, use *only* these modified article names.

   Output this file as `article-ids.csv`.

2. (20 marks) Convert the categories of the articles from `categories.tsv` into a hierarchy. Starting from `subject`, assign them ids starting from `C0001` till the end. Use breadth-first ordering with the children ordered in alphabetical order.

   Every line should have a category and the corresponding id.

   Output this as `category-ids.csv`.

3. (10 marks) Augment the `article-ids.csv` file to include all the categories the corresponding article is in. If an article has no category, assign it the highest category `subject`.

   Every article should be in one and only one line.

   Output this as `article-categories.csv`.

4. (10 marks) Using the shortest paths given in `shortest-path-distance-matrix.txt`, construct the directed graph of articles in an edge adjacency list format.

   Output this as `edges.csv`. Every edge should be in a separate line.

5. (15 marks) How many connected components are there in the graph?

   For each component, find the following values:

   1. Number of nodes, n
   2. Number of edges, m
   3. Diameter of the graph (ignoring directions of edges), d: diameter is the longest value of shortest path length between any two vertices

   Output this as `graph-components.csv`. Every component should be in a separate line.

6. (10 marks) Explore the finished paths in `paths_finished.tsv`. For every path, report the following.

   1. Length of path traversed by human
   2. Length of shortest path
   3. Ratio of human path to shortest path

   Prepare two versions, one that does not count the back clicks, and the one that does not.

   Output these as `finished-paths-no-back.csv` and `finished-paths-back.csv`.

7. (10 marks) For each of these files, find the percentage of human paths that have:

   1. Exactly the same path length as the shortest path
   2. Path length is 1 to 10 more than the shortest path (each separately)
   3. Path length is 11 or more than the shortest path

   Output these as `percentage-paths-no-back.csv` and `percentage-paths-back.csv`.

8. (10 marks) For each finished human path (without back links) and its corresponding shortest path, find the following for each category.

   1. Number of paths this category is traversed.
   2. Number of times this category is traversed (count multiple times if a category is visited multiple times in a path).

   Output this as `category-paths.csv`.

9. (10 marks) Repeat the above by considering all the categories under a category's subtree as its counts. Thus, *subject.Geography.African_Geography* should also include *subject.Geography.African_Geography.African_Countries*, etc.

   Output this as `category-subtree-paths.csv`.

10. (15 marks) For every unfinished path in `paths_unfinished.tsv`, find the source and destination category pair taking into account all the sub-categories under it (as earlier).

    Repeat the same for every finished path.

    For every source-destination category pair, find the percentage of finished and unfinished paths.

    Output this as `category-pairs.csv`.

11. (10 marks) For every source-destination category pair, find the ratio of human (without back clicks) to shortest paths.

    Output this as `category-ratios.csv`.

12. (5 marks) Write a manual that describes how to use your code. Include all the programs, their plugins, and dependencies needed to run the program. Include a top-level script `assign2.sh` that runs the entire assignment.

    Call this manual `README.txt`.

13. (20 marks) Include a small report in LaTeX where you analyze the results, draw overall conclusions, etc. In the analysis, please do not keep repeating the numbers or the derived values. The analysis should be brief and should be easily explainable to the general audience (not mathematically oriented computer scientists).

    Call this report `report.tex`. Include all the necessary accessory files needed for its successful compilation.