EECS4415 Big Data Systems

Fall 2019

Assignment 1 (10%): Data Analytics using Python

Due Date: 11:59 pm on Friday, Oct 11, 2019

Objective

In this assignment, you will write python programs/scripts for performing basic analytics on a large dataset. The dataset is a subset of Yelp[1]'s businesses, reviews, and user data. It was originally put together for the Yelp Dataset Challenge which is a chance for students to conduct research or analysis on Yelp's data. The dataset contains seven CSV files including information about businesses across 11 metropolitan areas in four countries and can be accessed here (registration to Kaggle is required):

Yelp dataset: https://www.kaggle.com/yelp-dataset/yelp-dataset/version/6

The first program (`dstats.py`) performs descriptive analytics of the dataset. The second (`dist-stats.py`) computes useful frequency distributions. The third (`yelp-network.py`) constructs a social network of Yelp friends. The fourth (`graph-stats.py`) performs basic network analytics.

Important Notes:

- You must use the *submit* command to electronically submit your solution by the due date.
- Your programs should be tested on the *docker image that we provided* before being submitted.
- All programs are to be written using Python 3 and to get full marks, code must be documented.

What to Submit

When you have completed the assignment, move or copy your python scripts in a directory (e.g., assignment1), and use the following command to electronically submit your files within that directory:

```
% submit 4415 a1 dstats.py dist-stats.py yelp-network.py graph-stats.py team.txt
```

The `team.txt` file includes information about the team members (*first name, last name, student ID, login, yorku email*). You can also submit the files individually after you complete each part of the assignment – simply execute the *submit* command and give the filename that you wish to submit. Make sure you name your files **exactly** as stated (including lower/upper case letters). Failure to do so will result in a mark of 0 being assigned. You may check the status of your submission using the command:

```
% submit -l 4415 a1
```

[1] Yelp. http://www.yelp.com

A. Descriptive Statistics (30%, 5% each)

Write a python program (`dstats.py`) that given a collection of businesses in a file `filename.csv` and a name of a city `city` **computes** and **prints out (in the STDOUT)** the following statistics:

- `numOfBus`: the number of businesses in the `city`
- `avgStars`: the average number of stars of a business in the `city`
- `numOfRestaurants`: the number of restaurants in the `city`
- `avgStarsRestaurants`: the average number of stars of restaurants in the `city`
- `avgNumOfReviews`: the average number of reviews for all businesses in the `city`
- `avgNumOfReviewsBus`: the average number of reviews for restaurants in the `city`

The collection of businesses is provided in a file that follows the same format as the original file provided by Kaggle (`yelp_business.csv`). The contents of the file we use for testing might vary.

Running the script:

Your script should be run as follows:

```
% python3 dstats.py filename.csv city
```

For example:

```
% python3 dstats.py yelp_business.csv Toronto
```

Hint:

Note that the `filename.csv` and the `city` need to be passed to the program as command line arguments. The `argparse` module makes it easy to write user-friendly command-line interfaces.

B. Distribution Statistics (30%, 10% each)

Write a python program (`dist-stats.py`) that given a collection of businesses in a file `filename.csv` and a name of a city `city` **computes** and **prints out (in the STDOUT)** the following:

- `restaurantCategoryDist`: a frequency distribution of the number of restaurants in each category of restaurants (e.g., Chinese, Japanese, Korean, Greek, etc.) in a descending order of popularity (from the most popular category to the least popular), along with the average number of stars per category. The output should be one line per pair of values as follows:

  ```
  category:#restaurants
  ```

  for example:

  ```
  Korean:120
  Italian:110
  ...
  ```

- `restaurantReviewDist`: a frequency distribution of the number of reviews submitted for each category of restaurants (e.g., Chinese, Japanese, Korean, Greek, etc.) in a descending order (from the most reviewed category to the least reviewed), along with the average number of stars received per category. The output should be one line per triplet as follows:

  ```
  category:#reviews:avg_stars
  ```

  for example:

  ```
  Korean:580:4.5
  Italian:110:3.8
  ...
  ```

- create a bar chart that shows the *top-10* of `restaurantCategoryDist` found earlier, where the *x-axis* represents the restaurant category and the *y-axis* represents its frequency (`#restaurants`).

The collection of businesses is provided in a file that follows the same format as the original file provided by Kaggle (`yelp_business.csv`). The contents of the file we use for testing might vary.

Running the script:

Your script should be run as follows:

```
% python3 dist-stats.py filename.csv city
```

For example:

```
% python3 dist-stats.py yelp_business.csv Toronto
```

Hint: Use the `matplotlib.pyplot` module to create the plot. Follow the example at pythonspot[2] about using `matplotlib` to create a bar chart: https://pythonspot.com/en/matplotlib-bar-chart/

[2] https://pythonspot.com

C. Creating the Yelp Social Network (20%)

Write a python program (`yelp-network.py`) that given a collection of users in a file `filename.csv` creates a file `yelp-network.txt` that represents the social network of Yelp friends. The social network will be represented as a graph `G(V,E)`, where `V` is a set of vertices/nodes representing the Yelp users and `E` is a set of links/edges representing friendships between Yelp users. The graph/network should be represented in a file using the edge list format. An edge list is a list that represents all the edges in a graph. Each edge is represented as a pair of nodes occupying a new line in the file. For example, a small fully connected triangle-like graph between nodes `a1`, `a2`, `a3` would be represented in the `yelp-network.txt` file as:

```
a1 a2
a2 a3
a3 a1
```

Note that the order of the lines does not matter, and edges are bidirectional (so either "`a1 a2`" or "`a2 a1`" should be listed but not both).

The collection of users and their friends is provided in a file that follows the same format as the original file provided by Kaggle (`yelp_user.csv`). The contents of the file we use for testing might vary.

Running the script:

Your scripts should be run as follows:

```
% python3 yelp-network.py  yelp_user.csv
```

D. Computing Network Statistics (20%)

Write a python program (`graph-stats.py`) that given as input a file that represents a network as an edge list, **computes** and **prints out (in the STDOUT)** the following:

- the number of nodes (`|N|`) and edges (`|E|`) of the network. The output should be as follows:
  `#nodes:|N| #edges:|E|`
- `nodeDegreeDist:` a frequency distribution of the node degrees of the network, in a descending order of frequency (from the highest to the lowest frequency). The degree of a node is the number of edges that are incident to the node (i.e., #neighbors). The output should be one line per pair of values as follows:
  `nodeID:nodeDegree`
  for example:
  `JJ-aSuM4pCFPdkfoZ34q0Q:14`
  `pzpbr9mlagHhDRdin8DvPQ:12`
  `...`
- `avgNodeDegree`: The average node degree of the graph. The output should be as follows:
  `avgNodeDegree:X`

The network is provided in a file that represents a network as an edge list. The contents of the file we use for testing might vary.

Running the script:

Your script should be run as follows:

```
% python3 graph-stats.py   yelp-network.txt
```