Introduction to Scientific Computing

Indian Institute of Technology, Madras

Assignment 2

Maximum Marks: 100 Assigned: March 12, 2024
Deadline: **March 27**, 2024

General Instructions

- You are expected to use the VM for this assignment. Create a directory in your home directory called `assignment_2`. Use this directory to work on the assignment.
- For each question (for question i), create a bash file called `question_i.sh` in the `assignment_2` directory. This bash file should contain the necessary code or commands to solve the respective question.
- We will be using an evaluation script to assess and evaluate your submission. Therefore, kindly ensure that the naming convention (as mentioned in usage section of each question) is strictly adhered to, and that the output which you get from running a script, matches the structure of the sample output.
- For submission, upload the MD5 checksum of the `assignment_2` directory on Moodle. You can use the following command. Make sure that you are in `assignment_2` directory for this command to work as intended.

```
find ./assignment_2/* -exec md5sum {} \; | cut -f 1 -d " " | md5sum
```

- After submitting the MD5 checksum on Moodle, **do not** update any file(s). Doing so will change your checksum, and your submission will not be evaluated.
- You are free to read through various resources. However, please ensure that you cite your sources to avoid plagiarism. Any detected instances of plagiarism will result in penalties.
- Please contact your assigned TA for any doubts or queries regarding this assignment.
- The **soft deadline** for this assignment is **11:59 PM** on **March 27**, 2024. Submissions after this deadline will face a linearly increasing penalty of 10 marks per late day.
- The **hard deadline** for this assignment is **11:59 PM** on **March 31**, 2024. Submissions after this deadline will not be evaluated.

[20 marks] 1. Web scraping is the process of extracting data from a website or any online source. In this era of Large Language Models, web scraping has become commonplace for gathering large quantities of data. Often, the data that is gathered is unusable and requires pre-processing. In this task, you are required to fetch data from an online source and perform some basic manipulations to prepare the data.

NASA maintains an archive of photographs captured by various enthusiasts, along with a brief explanation written by a professional astronomer. You are tasked to create a list of titles of these images that were uploaded on special dates (DD/MM/YYYY) like

[10 marks] (a) dates whose YYYY is divisible by DD,

[10 marks] (b) dates whose YYYY is divisible by MM.

Usage:

```
./question_1.sh
```

Output: Two `.csv` files – `answer_1a.csv` and `answer_1b.csv` for the corresponding parts.

[20 marks] 2. Publicly available datasets are often riddled with errors. In most cases, data visualization reveals such inconsistencies. In this task, you are provided with a dataset of an EV manufacturer that contains multiple parameters. The parameters are mentioned in the header of the dataset.

- Upon inspection, it turns out that all the alphabets in the data (for columns other than `Vehicle Number`) have been mistakenly replaced with their complement (where the complement of the i^{th} letter of the alphabet is $27 - i^{\text{th}}$ letter with the case retained).
- Also, on keen observation, the `SoH` and `SoC` columns are interchanged for `Vehicle Number` AG.
- (Misreported entries) In addition to these errors, there are also obvious entries where the reported `mileage` is non-zero despite `SoC` = 0.
- There are also rows in the dataset where certain parameters are missing. Since that those rows are useless, you may remove them.

You are tasked with correcting these errors to produce a clean dataset and also `Flag` misreported entries as "fake". The dataset is located at `/var/home/Jan24/assignments/assignment_2`.

Usage:

```
./question_2.sh final_dataset.txt > out.csv
```

Input:

```
Vehicle Number, SoC, Mileage(in m), Charging Time(in min), SoH, Driver Name
RB-34-XE  86  11180  12 21 VHMSKKUC
                     40
AG-22-QA  2    1170  81 9  MUAYENW
LT-20-TV  0    5961  90 52 NBBLNBG
```

and so on...

Output:

```
out.csv
Vehicle Number, SoC, Mileage(in m), Charging Time(in min), SoH, Driver Name, Flag
RB-34-XE  86  11180 12 21 ESNHPHFX
AG-22-QA   9  1170  81  2 NFAZBVM
LT-20-TV   0  5961  90 52 MYYOYMT Fake
```

and so on...

[30 marks] 3. Term Frequency Inverse Document Frequency (TF-IDF, for short) is a measure of the importance of a term (t) to a document (d) in a collection of documents ($\mathcal{D}$). For this task, we define

$$\operatorname{tf}(t,\, d) := \frac{f_{t,\,d}}{\sum\limits_{t' \in d} f_{t',\,d}}$$

where $f_{t,\,d}$ is the number of times term t occurs in document d,

$$\operatorname{idf}(t,\, \mathcal{D}) := \log_2\left(\frac{|\mathcal{D}| + 1}{|\{d \in \mathcal{D} : t \in d\}| + 1}\right)$$

where $|\cdot|$ is the cardinality of a set and $|\{d \in \mathcal{D} : t \in d\}|$ is the number of documents where the term t appears in.

The TF-IDF index is thus computed as

$$\text{tf-idf}(t,\, \mathcal{D}) := \frac{1}{|\mathcal{D}|} \sum_{d \in \mathcal{D}} \operatorname{tf}(t,\, d) \times \operatorname{idf}(t,\, \mathcal{D}).$$

Note: The definition of TF-IDF index may vary. For the purpose of this question, please stick to the above definition.

You are given a `.csv` in which each row is considered as a document (d) and the rows constitute the collection of documents ($\mathcal{D}$). Assume that only periods ('.') and commas (',') are only punctuations present in the documents.

[20 marks] (a) Given a term t, return its TF-IDF index (accurate to 4 decimal places).

Input:

```
id, document
1, this is a sample sentence.
2, there are 3 sentences in this sample.
3, this is a placeholder sentence.
```

Usage:

```
./question_3.sh document.csv sentence
```

Output:

```
0.0553
```

[10 marks] (b) If no arguments are passed when calling `question_3.sh`, return the top-5 terms (with values) in decreasing order of TF-IDF index.

Input:

```
id, document
1, this is a sample sentence.
2, there are 3 sentences in this sample.
3, this is a placeholder sentence.
```

Usage:

```
./question_3.sh document.csv
```

Output:

```
placeholder, 0.0667
a, 0.0553
is, 0.0553
sentence, 0.0553
there, 0.0476
```

[30 marks] 4. Structured Query Language (SQL) is extensively used to manage databases and is designed to query data in relational databases. In this exercise, you are tasked to replicate one of SQL's fundamental features `JOIN` using (preferably) `awk` or a combination of `join`, `sort` and `sed` (and other commands as needed).

The `JOIN` clause is used to combine rows from two (or more) tables based on some relation common between them. SQL offers four types of `JOIN`s (Fig. 1), namely

- `INNER JOIN`: Returns records that have matching values in both tables,
- `LEFT JOIN`: Returns all records from the left table, and the matched records from the right table,
- `RIGHT JOIN`: Returns all records from the right table, and the matched records from the left table,
- `FULL (OUTER) JOIN`: Returns all records when there is a match in either left or right table.

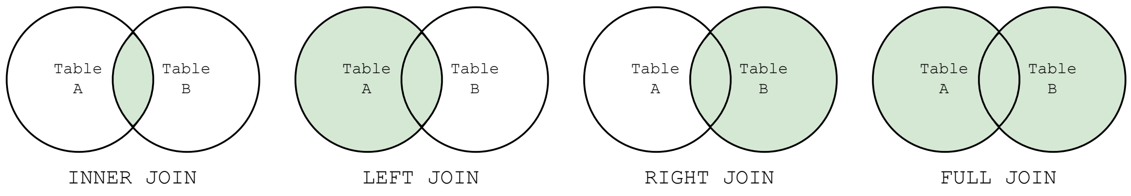

Figure 1: Types of `JOIN`s in SQL

Note: You may refer to SQL documentation or any online source to read about the different types of `JOIN`s.

[30 marks] (a) Write a bash script with flags ('-I' for `INNER JOIN`, '-L' for `LEFT JOIN`, '-R' for `RIGHT JOIN` and '-F' for `FULL JOIN`) to parse two `.csv` files (with fixed columns) and output the joined `.csv` file.

[10 bonus] (b) Extend sub-part (a) to adapt for generic `csv` files (no restriction on number of columns). You may assume that the columns names across the two files will be identical.

Input:

```
file_1.csv
ID, Roll
1, AE23B005
2, AE23B010
4, AE23B013
5, AE23B020

file_2.csv
Roll, Name
AE23B005, BHAVESH
AE23B010, GUHAAN
AE23B011, HEMANT
AE23B013, KISHOREKUMAR
```

Usage:

```
./question_4.sh -F file_1.csv file_2.csv > out.csv
```

Output:

```
out.csv
ID, Roll, Name
1, AE23B005, BHAVESH
2, AE23B010, GUHAAN
NULL, AE23B011, HEMANT
4, AE23B013, KISHOREKUMAR
5, AE23B020, NULL
```

Note: Ensure that columns of `out.csv` are in the order mentioned in the sample output. The rows need not be in any specific order.