COMP47470 Big Data Programming
Assignment 1

February 17, 2023

1 Project Outline

Submission Details

Submission Deadline: 10th of March, 2023, 17:00.

This project is worth 30% of the overall grade for the module. The project is graded out of 100 marks. Please review the School's plagiarism policy before completing this project. It is important that any sources used for code or writing are **clearly referenced** within the code and/or cited appropriately in accompanying documentation.

Submission format - zip file containing the following files:

1. `data-collection.ipynb` (Tasks from Section 2)
2. `conceptual-design.pdf` (Tasks from Section 3.1)
3. `data-management.ipynb` (Tasks from Section 3.2)

Note: Don't include downloaded data. Your script from Section 2 should do this when it is executed anyway.

Project Setup

In this project, we are going to perform the following tasks using big data programming techniques and frameworks:

- Data Collection (Bash)
- Data Managenent (MySQL/MongoDB)

Within the project folder, you should have the following files:

1. `parse_article.py`

2. `parse_article.sh`

3. `article-ids.txt`

It is recommended to move all project and setup files to the same directory to simplify the project workflow.

Scenario

The research and development department of Big Four Inc. is looking to invest in machine learning research. Machine learning, however, is a large and ever-growing field, meaning it can be difficult to know where to start. You have been asked to determine the trends of the field in the last few years and report them back. You and your technical colleagues will be the users accessing the database created and will be using it in order to derive summary information that you will report back to stakeholders. Note that if this database ends up being useful, you may be asked by management to add new data as the field evolves and may need to eventually do so at a high frequency.

2 Data Collection (20 marks)

Finding trends in a particular field requires that we use some sort of data that summarises topics within the field over time. As it happens, there is an open source website that updates a regular summary of nearly everything in existence - Wikipedia. In this section, we will download historical versions of the "Machine Learning" Wikipedia Page from the last few years to gauge the trends in the field. ese can be appended to the following URL to get the historical versions of the page: `https://en.wikipedia.org/w/index.php?title=Machine_learning&oldid=`
For example:
`https://en.wikipedia.org/w/index.php?title=Machine_learning&oldid=1138583125`

2.1 Download Data

To download the history of the page, we look at the history of the page. For the sake of ease, the id of each version of the page from the last 10 years, at an interval of six months, has been included in the `article-ids.txt` file.

Tasks

Complete the following section using Bash. Put any code that you execute into a IPython notebook called `data-collection.ipynb`.

1. Create a folder called `wikipedia-ml-raw`. You should include a test to check if this directory already exists and only create the folder if it does not.
2. Read ids from the text file `article-ids.txt`.
3. For each id, download the historical HTML file, giving each file a name in the format

   ```
   "machine-learning-<articleId>.html"
   ```

4. Place each file into the `wikipedia-ml-raw` folder. Your directory should look like this:

   ```
   wikipedia-ml-raw
   |
   ------> machine-learning-1063409105.html
           ...
   |
   ------> machine-learning-999142720.html
   ```

2.2 Clean data

Once we have downloaded the data, we need to do some additional pre-processing. This is not a text-analytics module, so the parsing script is provided in Python. Take the `parse_articles.sh` and follow the instructions. Note that your directory must be formatted in the manner described in Section 2.1 for this to work seamlessly.

The preprocessing script will:

- Parse HTML.
- Extract article title.
- Extract article text.
- Extract references.
- Remove anything non-linguistic or atypical punctuation.
- Extract date of html file update.
- For each html file, write the article paragraphs to one text file and references to another.

Tasks

Put any code that you execute into a Python notebook called `data-collection.ipynb`.

1. Run `parse_articles.sh`
    - The script will output two files: an `article` and `ref` text file. Each file is identified by the article title and date (there is only one article per date stated).
    - Check that the articles are in the following format:
      For articles

      `article_<article title>_<year>_<month>_<day>.txt`

      For article references:

      `refs_<article title>_<year>_<month>_<day>.txt`
    - Check that the resulting directory of cleaned data looks like this:

```
!ls
 >> article_Machine learning - Wikipedia_2013_1_2.txt
    ...
    article_Machine learning - Wikipedia_2023_1_1.txt
    refs_Machine learning - Wikipedia_2013_1_2.txt
    ...
    refs_Machine learning - Wikipedia_2023_1_1.txt
```

3 Data Management

Now that we have cleaned our raw data, let's make a plan to store our data.

3.1 Conceptual Design (40 marks)

Before we store our data, we need to look at the structure and determine the data model that suits our dataset the most. In order to do this, complete the following tasks.

Tasks

Put the following tasks in a PDF file called `conceptual-design.pdf`. Please follow the word count and remember - sometimes less is more!

1. Using the scenario that was stated in Section 1, list the likely requirements for the database (max 500 words).
2. Based on the structure of the parsed data, do you think that a relational database or a document database is most appropriate? Motivate your choice using the concepts you have learned (max 500 words).
3. Outline the structure of your database. This should include the fields in your data, any relationships that exist etc. You can get a bit creative here, using a simple a diagram (for instance a entity-relationship model if you choose a relational database) or a schematic listing fields if you choose a document based model. Remember to choose fields that will be useful to any downstream CRUD operations you think you'd need.

3.2 Database Implementation (40 marks)

Using the plan you've outlined in Section 3.1, implement your database in either MySQL or MongoDB, depending on the data model you choose.

Tasks

You can use either Bash or IPython for the following tasks. Put any code that you execute into a Python notebook called `data-management.ipynb`.

1. Populate your database using the design from 3.1.
2. Use queries to answer the following questions. **Include the queries with your answers, the answers alone will not get you any marks**:
   (a) List the unique years in which the data was collected.
   (b) List the number of documents for each year.
   (c) Count the number of years that each of these terms appear in the article body:
      - "supervised"
      - "semi-supervised"
      - "unsupervised"
      - "meta-learning"
   (d) What is the earliest year that each of these terms appeared in the article body?
   (e) Assuming that references in the reference field are split by a `\n` character, calculate the number of references for each year.