CS5481: Data Engineering - Assignment 1

Instructions

1. Due on Wednesday, Oct. 8, 2025, 18:00:00 PM;
2. You can submit your answers by **a single PDF with the code package** or **a single jupyter notebook** containing both the answers and the code;
3. For the coding questions, besides the code, you are encouraged to additionally give some descriptions of your code design and its workflow. Detailed analysis of the experimental results are also preferred;
4. Total marks are 100;
5. Plagiarism or unjustified late submission will result in the assignment being invalid or points being deducted correspondingly.
6. If you have any questions, please post your questions on the Canvas-Discussion forum.

Question 1 – Online Reviews Data Collection

(20 marks) User-generated reviews on e-commerce and entertainment platforms provide valuable information for text mining and sentiment analysis. However, obtaining high-quality review data can be challenging, especially when official APIs are limited or restricted. In such cases, web scraping can be used to collect data directly from websites.

Your task is to **collect 30 online user reviews** from the websites listed below and prepare them for analysis. The dataset must satisfy the following requirements:

1. Each entry should include the review text along with at least one associated reply or rating (e.g., helpfulness votes, star rating, or comment).
2. Focus only on textual and numerical information; ignore multimedia elements such as images or videos.
3. Store the data in a structured format (e.g., JSON or CSV) with appropriate fields such as review text, user, date, rating, and reply/comment.

Suggested sources for reviews:

- `https://www.imdb.com` (movie reviews)
- `https://www.goodreads.com` (book reviews)
- `https://www.tripadvisor.com` (travel reviews)
- `https://www.amazon.com` (product reviews)

Submit your code and the structured social media data.

Question 2 - Data Cleaning with Regular Expressions

(30 marks) Regular Expressions (Regex) are powerful tools for searching, validating, and transforming text data. During the preprocessing stage of data analysis, Regex helps automate tasks such as filtering, pattern detection, and input validation.

Below are several exercises on Regex in Python. Write the appropriate regex pattern for each task.

1. Write a pattern to check if a string contains only alphabetic characters (uppercase or lowercase, without digits or symbols).
   - Test cases: Python, DataScience, Hello123
2. Write a pattern to find all words that begin with a consonant.
   - Test cases: cat, elephant, dog, owl
3. Write a pattern to verify whether a given string is a valid domain name (e.g., example.com).
   - Test cases: openai.org, invalid@site, my-site.net
4. Write a pattern to extract all numbers (integers) appearing in a text.
   - Test cases: He scored 45 goals in 2022 and 10 goals in 2023.
5. Write a pattern to identify valid file paths with extensions (e.g., .txt, .csv, .jpg).
   - Test cases: /home/user/file.txt, report.doc, /tmp/image.jpg
6. Write a pattern to validate a Canadian postal code format (They are in the format A1A 1A1, where A is a letter and 1 is a digit, with a space separating the third and fourth characters).
   - Test cases: K1A 0B1, 123 456
7. Write a pattern to find strings where the first and last characters are identical.
   - Test cases: level, stats, world
8. Write a regex to validate a password that must include: at least one uppercase letter, one lowercase letter, one digit, one special character, and have a minimum length of 10.
   - Test cases: Secure123!, weakpass, ValidPass#2023
9. Write a regex pattern that matches and extracts dates written in either of the following formats:
   - **mm/dd/yyyy** (e.g., 07/04/2021)
   - **yyyy-mm-dd** (e.g., 2022-12-31)

   Your regex should be able to handle both formats within a single expression.
   - Valid test cases: 07/04/2021, 2022-12-31, 01/01/2024
   - Invalid test cases: 2022/12/31 (wrong separator), 13-2020 (missing parts), 07-04-21 (wrong year digits)
10. Write a regex that matches valid IPv6 addresses.
    - Test cases: 2001:0db8:85a3:0000:0000:8a2e:0370:7334, 1234:5678:90ab:cdef:ghij:0000:0000:0001

Question 3 - Data Processing

(20 marks) Many machine translation datasets, especially those used in research, are provided in structured formats such as XML, which makes it easier to store parallel sentences and metadata. Before we can train a model using these data, we should convert them from XML format to line-based text. Please solve the following questions:

1. Please convert the data in this file (https://github.com/wmt-conference/wmt-format-tools/tree/main/test/sample-data/sample-hyp.xml) to line-based text with your own Python code. You need to remove all punctuation and convert all text to lowercase. You should submit your runnable code and output file.

2. After you obtain the line-based text file, please create a BPE vocabulary (save each BPE token line by line) with a tool called **subword-nmt** (https://github.com/rsennrich/subword-nmt.git). You should submit your runnable code and output file.

Question 4 - Data Visualization

(30 marks) Data visualization is an essential tool for exploring and understanding datasets. Common visualization techniques include bar charts, histograms, pie charts, scatter plots, and heatmaps.

1. Suppose we have a dataset of 500 students containing the following attributes: **Student ID** (Integer; 1–500), **Major** (Categorical; Computer Science, Mathematics, Physics), **Gender** (Binary; Male/Female), and **GPA** (Continuous; 0.0–4.0). Which visualization methods would be appropriate to explore the distribution of each attribute and relationships between attributes?

2. Write a Python program to randomly generate 500 student records based on the descriptions above. Visualize the generated data using the visualization techniques you selected in part (a).

3. Compute the number of students in each major and display the results using a bar chart.

4. In recommendation systems, a simple similarity score between user and item embeddings can be calculated using the dot product. Given two matrices $U \in R^{5\times 8}$ (user embeddings) and $V \in R^{5\times 8}$ (item embeddings), the similarity score is computed as:

$$\text{Similarity}(U, V) = \text{softmax}\left(\frac{UV^T}{\sqrt{d}}\right),$$

   where d is the embedding dimension (8 in this case). Randomly initialize U and V and visualize the similarity scores using a heatmap.