Introduction

The objective of this take home is to assess your data engineering skills. As part of this project, you will demonstrate competencies in **Data modeling** - through unifying different types of data (json, csv etc.), **Data engineering and pipelining** - through efficient and clear code, and **Data analysis** - through analyzing this unstructured data.

Part 1 - Data Engineering and Deduplication

Objective

The objective of the first part of this data engineering take home project is to design and implement a data pipeline that can efficiently extract, transform, and load (ETL) large volumes of data from various sources into a single data structure, using appropriate tools and technologies.

The pipeline should be scalable, reliable, and easy to maintain, and the resulting data should be stored in a format that is optimized for efficient querying and analysis. The project should demonstrate proficiency in data modeling, transformation and pipeline orchestration; and should adhere to best practices in data engineering.

Data

Following data has been scraped from indeed.com. Job links file contain the list of scraped job postings and job descriptions file contains job postings themselves.
Note that the job link file contains multiple queries from different zip codes for the same job key.

- Indeed us job link (job links file) file
- Indeed us job description (job descriptions file) file

Tasks

1. Merge job links file and job descriptions file on key(s) that you find appropriate.
2. Write a function to clean html tags and special characters from job descriptions, and make it as clean as possible.
3. Use job description, job title, and company fields to remove duplicate entries from the list. Remove duplicate rows from data but keep a list of jk values (see table below).
   Note: that you are expected to make this comparison with an optimal solution (e.g your solution should not have a O(n^2) time complexity)
4. Create an index for all job postings after deduplication (index_dup).
5. [Bonus] group job titles that are nearly identical based on a similarity score, create a new data structure to store the results

Expected Output

Combine data and provide us with a final csv file (for tasks 1-4). Keep in mind that we only provide a small subset of the data to save time and computational effort, however we expect to see code that could scale for over 100M+ postings.

1. Index dup: new key for all combined and deduplicated data
2. Jk list column: if the row is duplicate, mark list jk's of all other duplicate rows in a list.

Other columns	job_desc	index_dup	jk_list
		1	['19d2c865646d7481', 'ac8786c7ff655f19']
		2	['2e5cb1d5984a5ab6']
		3	['54cdab9620b07bd5', 'e602eaa2ed0edd4b', '1389b8b5741aaf3b']

Note: We expect you to fully explore the data, identify potential problems, and propose solutions

Conceptual Questions (please provide answers)

- Imagine that you receive daily data from 50,000 different job posting / employer sites. The data are in csv, json and other formats with different columns/ fields with approximate 20,000,000 daily job postings in volume.
    - What type of database and platform would you use to process and store the data?
    - What technologies and tools would you use?
    - How do you serve it to an end user (via some sort of dashboard/BI tool? Or would you develop something from scratch?)

Part 2 - Text Preprocessing and Segmentation

Objective

The goal of the second part is to assess your skills in text processing and text segmentation. Text preprocessing is an important step in natural language processing (NLP) because it helps to convert raw text data into a structured format that can be easily analyzed and processed by machine learning algorithms. You can exploit regex or any other function/tool to complete the following tasks. The target is to be able to do it as fast as possible so it can be scalable to large datasets.

The objective of this data engineering take home project is to develop a system that can effectively preprocess, segment unstructured text data, and implement regular expressions that can effectively extract, transform, and validate text data based on specific patterns and rules.

The project should demonstrate proficiency in using regular expressions to perform tasks such as searching and replacing text, parsing data, and validating input. The resulting regex patterns should be efficient, accurate, easily maintainable, and scalable, and should adhere to best data engineering practices.

Data

You are going to use the job description from the earlier data (job descriptions file) containing html code from part 1.

Tasks

Extract the following from job postings:

a. Salary
b. Company domains
c. URLs, websites,
d. Contact Information
e. Relevant dates (mm/dd/yyyy)
f. US address, location specific information, zip code, state

Regex should take care most, but if you are interested in doing this with NLP you can look at NER (may not perform as well as regex)

Expected Output

Expected output is a json file containing all the relevant information, e.g. {'salary': 10000, 'address': ..., etc.}. Keep in mind that some job descriptions may not include all of this information. Therefore, it's normal to encounter job listings where some or all of this information is not provided.

Based on your code, provide an answer to the following questions:

- What is the average number of fields (listed above) found per job listing?
- What percentages does each segment correspond to within the job descriptions?