Project 4: Clustering Project

Data set

The data source is https://www.aminer.org/citation, version 13, as it is the most detailed one in JSON format. You can also check the schema of the respective data set on the same page under the "Description" link.

Goal

Can clustering similar research articles together simplify the search for related publications? How can the content of the clusters be qualified? And over each cluster how can we recommend the most similar papers leveraging clustering?

You are required to cluster the papers in the dataset, you need to use at least the abstract and the title of the paper. Then, you should build a simple search engine on top that recommends similar papers based on search by title.

Steps

1. Read the dataset using Spark, common from previous projects
2. Do exploratory data analysis to help you extract features,
3. Keep only the English documents,
4. Preprocessing: the goal is to clean and preprocess the text to prepare it to represent it in vectors. It is a mandatory step in NLP projects to preprocess the text. You can have a look in this article to explore some of well-known preprocessing steps and find how they can be done in Spark Mlib: https://towardsdatascience.com/nlp-text-preprocessing-a-practical-guide-and-template-d80874676e79, required pre-processing
    a. Remove stop words,
    b. Remove custom stop words: research papers will often frequently use words that don't actually contribute to the meaning and are not considered everyday stop words and should be removed to enhance the accuracy. Examples of custom stop words are ['doi', 'preprint', 'copyright', 'peer', 'reviewed', 'org', 'https', 'et', 'al', 'author', 'figure','rights', 'reserved', 'permission', 'used', 'using', 'biorxiv', 'medrxiv', 'license', 'fig', 'fig.', 'al.', 'Elsevier', 'PMC', 'CZI', 'www']
    c. Remove Punctuation, use this Regex: '!()-[]{};:'"\,<>./?@#$%^&*_~' to remove it,
    d. Convert text to lower case,
5. Vectorization: convert the data into format that can be handled by ML algorithms. You can have a look on https://spark.apache.org/docs/latest/mllib-feature-extraction.html, some useful techniques are:
    a. TF-IDF: this will convert string-formatted data into a measure of how important each word is to the instance out of the literature as a whole. See, https://www.youtube.com/watch?=hc3DCn8viWs for more details,
    b. Word2vec, https://www.youtube.com/watch?v=3eoX_waysy4
6. Clustering: You can try K-means clustering. To determine K, you can run the elbow method. You can use PCA to reduce the dimensions while still keeping 95% of the variance in the data for better performance and hopefully remove some noise/outliers

7. Search engine: you can implement this via a very basic recommender function that takes as input a paper title and N as the top-most N closest papers. You recommend the top-most N based on the most similar (Cosine similarity) papers to the input paper title in the cluster to which it belongs.