Data Algorithm Project

Report on finding similarity between book reviews with Jaccard similarity

Dao Ton Son

Abstract—The report summarize the process and results in finding the similarity between reviews from different books using Jaccard similarity. The report shows the dataset, using Locality Sensitive Hashing(LSH) and Minhash to optimize calculation process to check whether pairs of reviews are alike based Jaccard similarity with discussion of the results

1. Dataset

The data set is imported from Kaggle using the Amazon book review database. More specifically, the project uses the review/text of audience for many book titles to find the similarity between the reviews. The dataset is comprised of 133375 reviews from users containing the name of the book their reviews and other information.

For preprocessing, texts are turned into lowercase and eliminated punctuation and multiple spaces. Any review with less than 5 words are removed since it would produce an empty set when shingle size is 5. Only 78 reviews out of 133375 are not qualified and, it is acceptable to not use it. Reviews are considered on a word-level similarity because it does better on a semantic level compared to character-level, as well as checking for plagiarism in reviews in cases where word order matters, preventing spammer to get good reviews for a book. In addition, it can also be used to recommend books to customers with similar reviews.

One of the problems that arises is that it can be computationally expensive when using our method, which will be discussed later on.

2. Methodologies and Implementation

The approach is adapted from the book "Mining of Massive dataset" section 3.4.3. After preprocessing, each review is constructed into a set of k-shingles, in which k = 5 since it was shown to be an effective shingle for texts and hash to a bucket number to optimize storage space. k = 3 is tested, but the computation is much longer, and the scope of the project does not need to be in such details. The review-shingle pairs are sorted by shingle, which would optimize the process such as helping with the construction of inverted index in Locality Sensitive Hashing(LSH) or with memory efficiency. In addition, all the shingle is hashed using Secure Hash Algorithm 1(SHA1) for a better reproducibity thanks to its deterministic characteristics especially when the reviews are lowercased.

Next step, the Minhash signature is calculated for the review-shingle pair. When selecting the number of permutation functions n, the trade off between accuracy and computational cost must be considered to optimize the process. If n is high the accuracy would be high but also increasing computational cost and vice-versa. After consideration, n = 110, banding m = 22, row = 5 and threshold for Jaccard similarity t = 0.53 are chosen. Due to the sheer number of pairs so if we just calculate the Jaccard similarity between the reviews, it will be very costly. Hence, I pick LSH to construct the candidate pairs with banding m = 22 and row = 3 which is automatically calculated using the datasketch.MinHashLSH when t = 0.53.

Finally, to see if they are similar based on Jaccard similarity, I check each pair of candidates's signatures if they agree in at least a fraction t of components.

3. Discussion

With 133297, there are 499500 pairs of similar reviews. Manually, I handcheck some pairs and see that, repetitively, most of them are similar in the words to an extend which can be a sign of spamming with slight changes in the reviews. There are some exceptions for example where the pairs are totally difference where one is in English and one is in Spanish.

With these findings, we can use it for further investigation for example in plagiarism or spammer since they only tend to change a couple of words, or we can use books with similar reviews as evidence to cross-marketing books if the books are in the same genre and the user like the book. Another application that we can use it as information to train prediction model to rate books based on the reviews we received.

When the data is scaled up from 2Gb to maybe 10 or 100GB, the scaling solution as I used in the method can still hold up but some better solution can be parallel computation or LSH with disk-backed storage for a bigger scale.

4. Declaration

"I/We declare that this material, which I/We now submit for assessment, is entirely my/our own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my/our work, and including any code produced using generative AI systems. I/We understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me/us or any other person for assessment on this or any other course of study."