ASSIGNMENT-2
Map-Reduce and Similar Itemsets Mining

Submission Instructions:

1. **Put all your results in a neatly written PDF file with the plots and outputs. You may take screenshots of your execution to show in the output.**
2. **Put all your code and results in a PDF file in a directory named in the format roll_name. Zip it.**
3. **Submit on GC**
4. **Deadline: 14th February 2025**

Note: I strongly discourage copying code from online resources. Any such activity will lead to a 0 score in this assignment.

TOTAL: 100 POINTS

MIN-HASHING AND LSH

For the following questions, use the D1.txt, D2.txt, D3.txt, and D4.txt files in the minhash directory.

1. **Create k-Grams** [20]

   You will construct several types of k-grams for all documents. All documents only have at most 27 characters: all lowercase letters and space. The space counts as a character in character k-grams.
   - Construct 2-grams based on characters for all documents.
   - Construct 3-grams based on characters for all documents.
   - Construct 2-grams based on words for all documents.

   Remember that you should only store each k-gram once. Duplicates are ignored.

   A: How many distinct k-grams are there for each document with each type of k-gram? You should report 4 × 3 = 12 different numbers.
   B: Compute the Jaccard similarity between all pairs of documents for each type of k-gram. You should report 3 × 6 = 18 different numbers.

2. **Min-Hashing:** [20]

   We will consider a hash family H so that any hash function h ∈ H maps from h: {k-grams} → [m] for m large enough (To be extra cautious, use m ≥ 10,000).

   A: Using 3-grams to build a min-hash signature for documents D1 and D2 using t = {20, 60, 150, 300, 600} hash functions. For each value of t, report the approximate Jaccard similarity between the pair of documents D1 and D2, estimating the Jaccard similarity:

$$\hat{\mathrm{JS}}_t(a,b) = \frac{1}{t}\sum_{i=1}^{t}\begin{cases}1 & \text{if } a_i = b_i \\ 0 & \text{if } a_i \neq b_i.\end{cases}$$

   You should report 5 numbers.

   B: What seems to be a good value for t? You may run more experiments. Justify your answer in terms of both accuracy and time.

3. **LSH:** [20]

Consider computing an LSH using t = 160 hash functions. We want to find all document pairs with Jaccard similarity above $\tau = .7$.

A: Use the formula mentioned in class and the notes to estimate the best values of hash functions b within each of the r bands to provide the S-curve

$$f(s) = 1-(1-s^b)^r$$

with good separation at τ. Report these values.

B: Using your choice of r and b and $f(\cdot)$, what is the probability of each pair of the four documents (using 3-grams) being estimated to have a similarity greater than τ? Report 6 numbers.

4. Min-Hashing on MovieLens dataset [20]

Implement Min-Hashing on the older MovieLens 100k dataset (5MB), which consists of a set of 943 users who have rated 1682 movies. You can download the data from http://www.grouplens.org/node/73. Read the Readme file for details about the data, and process it as you need. For this exercise, we only care about the set of movies that a user has rated and not the ratings. We want to compute the Jaccard similarity between the users.

Compute the exact Jaccard similarity for all pairs of users, and output the pairs of users that have a similarity of at least 0.5. Then, compute the min-hash signatures for the users and compute the approximate Jaccard similarity. Use 50, 100, and 200 hash functions. For each value, output the pairs that have an estimated similarity of at least 0.5 and report the number of false positives and false negatives that you obtain. For the false positives and negatives, report the averages for 5 different runs.

5. LSH on MovieLens dataset [20]

Break up the signature table into b bands with r hash functions per band and implement Locality Sensitive Hashing. The goal is to find candidate pairs with a similarity of at least 0.6. Experiment with r = 5, b = 10 for the table with the 50 hash functions, r = 5, b = 20 for the table with the 100 hash functions, r = 5, b = 40 and r =10, b = 20 for the table with the 200 hash functions. Report the number of false positives and false negatives, taking the average over 5 runs. How do these numbers change if we want a similarity of at least 0.8?