Total Points: 100 (5%)

Assignment 1
COMPSCI 753: Algorithms for Massive Data

Due: 13 Aug at 11:59 pm 2023

- **General Instruction:** The main objective of this assignment is to build a document retrieval system using the LSH framework. The assignment is divided into three parts. Your task is to develop a Python program that accomplishes the following components:
  1. Construct LSH Hash Tables for All Movie Plots Articles
  2. Perform Nearest Neighbor Search for Each Query
  3. Investigate the Impact of the hash size (k) on the estimation quality in terms of MAE and runtime

- **Datasets:** Please refer to the movie plot dataset (**bitvector_all.csv**), which consists of 5000 movie plot articles. To achieve an efficient nearest neighbor (NN) search for any queries, please utilize the dataset to construct an LSH Hash Table. Each movie plot within these files is formatted as a tab-separated line with three columns:

  `movie_id <TAB> word_features <TAB> movie_genre`

  1. movie_id: This column contains a unique string ID for each movie.
  2. word_features: This column represents a sequence of TAB-separated binary values, indicating the occurrence of specific tokens in binary format.
  3. movie_genre: This column specifies the genre of the movie.

  For the original movie plots corresponding to the dataset, you can access them in the movie_plots all.csv and movie_plots_query.csv files, which are included in the movieplot.zip file available on Canvas.

- **Submission:** Please submit a **single report** (.pdf) and the **source code with detailed comments** (**.py or .ipynb or .html**) on Canvas by 23:59, Sunday 13 August 2023. It is important to include your student ID, UPI, and name in the submission file.

- **Penalty Policy:** Please note that the assignment will not be accepted after the final deadline, unless special circumstances such as sickness with a medical certificate or family/personal emergencies are present. In such cases, exceptions may be made. Penalties for late submissions will be calculated as a percentage of the assignment's mark.
  1. By 23:59, Sunday 13 August 2023 (No penalty)
  2. By 23:59, Monday 14 August 2023 (25% penalty)
  3. By 23:59, Tuesday 15 August 2023 (50% penalty)

1 Construct LSH Hash Tables for All News Articles [45 marks]

(A) Load bitvector_all.csv. Construct a feature vector for each movie plot article in the dataset (D). Please report the number of articles ($|D|$) and the number of features (n) for the loaded data. **[5 marks]**

(B) Construct a family of MinHash functions in the LSH family. You can choose a prime number, denoted as p, and coefficients $0 < a, b < p$. Let the tunable hash size k, and report the family of MinHash functions you have generated with $k \in \{2, 4, 8, 16\}$, respectively. **[10 marks]**

(C) Construct LSH hash tables using your hash functions with bucket size ($m = 600$) with $k = 2$. Report the data dimension of your signature matrix in terms of the number of rows, and the number of articles. [**20 marks**]

(D) Please report the collision distribution of all articles hashed into m buckets using a histogram plot, where the x-axis is $m = 600$ buckets, y-axis refers to the number of colliding articles. Please report the summation of articles across buckets and comment on your findings. [**10 marks**]

2 Nearest Neighbor Search [40 marks]

(A) Query the LSH tables and return the top-5 articles that have the highest estimated Jaccard similarities as the answer. Consider articles with an article ID range from 4996 to 5000 as the query set $Q = \{4996, 4997, 4998, 4999, 5000\}$. To compute the search results for each query document $q \in Q$ by following the steps: **[25 marks]**

- **Step 1:** Find the set of articles D_q that collide with $q \in Q$ in at least one hash table.
- **Step 2:** Compute the estimated Jaccard similarity between $q \in Q$ and each article in D_q, denoted as $\hat{J}(d_i, q)$, where d_i represents each article in the entire set of articles D.
- **Step 3:** Report the top-5 articles with the highest estimated Jaccard similarity for query $q \in Q$. Arrange the list in descending order based on estimated Jaccard similarity. You need to provide five lists in total for the search results, each corresponding to one query $q \in Q$. In each list, the article with the highest estimated Jaccard similarity should be ranked at 1. Each row of the list should follow the following format for each query $q \in Q$:


```
movie_id <TAB> estimated_Jaccard_sim <TAB> movie_genre
```


(B) Consider the same query set Q, compute true Jaccard similarity for query $q \in Q$ and all articles in the dataset D, i.e., $J(d_i, q)$ for each article $d_i \in D$. Please report the list of top-5 articles with the highest true Jaccard similarity in descending order for each article $q \in Q$. Each row of the list should follow the format for each query $q \in Q$: **[15 marks]**

```
movie_id <TAB> Jaccard_sim <TAB> movie_genre
```

3 Estimation Quality and Efficiency [15 marks]

(A) Investigate the impact of the hash size $k \in \{2, 4, 8, 16\}$ on the estimated Jaccard similarity estimation. Consider articles with an article ID range from 4001 to 5000 as the query set Q. For each value of hash size k, compute the mean absolute error (MAE) for the pairs of articles (d_i, q), where $1 \leq i \leq |D|$ and $q \in Q$. Please plot the MAE of MinHash estimator on different values of k. In particular, MAEs with different values of k on x-axis (i.e., 2, 4, 8, 16) and MAE values on y-axis. The MAE is defined as follows. [**10 marks**]

$$\text{MAE} = \frac{\sum_{q \in Q} \sum_{i=1}^{|D|} |J(d_i, q) - \hat{J}(d_i, q)|}{|Q| \times |D|}. \tag{1}$$

(B) Consider articles with an article ID range from 4001 to 5000 as the query set Q with $k = 2$. Compare the query time in Question 2(A) and Question 2(B) on average across the query set Q in milliseconds and comment on their differences (if any). [**5 marks**]