Home Assignment 1

Data Management Algorithm for Decision Making

November 12, 2025

General Instructions

Please submit a zip file named "Id1_Id2.zip" via 'webcourse' course site with the following files:

- **Written Questions:** PDF file "HW1_Dry.pdf" (typed and in English).
- **Your Code '.py' files - there are 6**
- **Your Code notebook:** "`EX1.ipynb`" (Jupyter Notebook).
- **Execution Results (for control only, all the results should be reported in the PDF file):** "`EX1_Result.html`" HTML file with code output.

Instructions for Saving Jupyter Notebook as HTML

Follow these steps:

1. Download the notebook file ('.ipynb').
2. Open a new Jupyter Notebook and upload the downloaded notebook file to your workspace, and also upload all the code files to the same location.
3. Run the following code in the new notebook:

```python
import nbformat
from nbconvert import HTMLExporter
from nbconvert.preprocessors import ExecutePreprocessor

with open("EX1.ipynb") as f:
    notebook_content = nbformat.read(f, as_version=4)

ep = ExecutePreprocessor(timeout=600, kernel_name='python3')
ep.preprocess(notebook_content)

html_exporter = HTMLExporter()
html_data, _ = html_exporter.from_notebook_node(notebook_content)

with open("EX1_Result.html", "w") as f:
    f.write(html_data)
```

Note: You can adjust the 'timeout' parameter in the code above if needed. There is no requirement for a specific runtime in this exercise.
Make sure that all the results match those you report in your PDF

- Explain your solutions in your own words. Minimize using external *LLMs* tools (e.g., ChatGPT). If you do use LLMs, explain exactly where and how you use them.
- If you use information not covered in the course slides, provide a proper reference.

- If you consulted with classmates, mention who you consulted with. Collaboration is fine as long as you can explain the solution independently.
- Be honest and fair in your work. Ensure you fully understand everything you submit.

Questions

Problem 1: Bloom Filter (30 points). Consider a Bloom filter with m bits and k hash functions. Assume that each hash function uniformly hashes its input to one of the m bits and that we are inserting n elements into the filter.

1. (10 points) Suppose we are designing a Bloom filter with the following specifications:
   • Number of elements to be inserted (n): 1000
   • Number of hash functions (k): 5
   • Desired false positive rate: 0.1

   What is the minimum number of bits (m) required to achieve this false positive rate? Explain your answer.

2. (10 points) When the Bloom filter has too few bits (m) relative to the number of inserted elements (n), the bit array becomes *saturated*, and collisions increase. In this part, you will quantitatively explore how this saturation affects the false positive rate and the choice of k.

   Task (use the given notebook and complete the code):

   - Fix $n = 100$.
   - Vary the Bloom filter size $m \in \{50, 100, 200, 300, 400, 500\}$
   - For each m, simulate the empirical false positive rate for $k \in \{1, 3, 5, 7, 9, 11, 13, 15\}$.
   - Plot the results: complete the code.
     X-axis: number of hash functions k, Y-axis: false positive rate.
     Separate line (and colors) for each m.
   - Based on your results, would you recommend increasing or decreasing k when the memory (m) is limited?

3. (10 points) Assume you have two Bloom filters B_1 and B_2 (both having the same number of bits m and the same hash functions) representing the two sets A_1 and A_2. Let B be the Bloom filter obtained by computing the bitwise Boolean and of B_1 and B_2.

   (a) Is B necessarily the same Bloom filter as the one that would be created by adding the elements of the set $A_1 \cup A_2$ one at a time? If not, provide an example; otherwise, explain why.

   (b) Does B accurately represent the set $A_1 \setminus A_2$ in the sense that it gives a positive answer for a membership query for an element in this set? Explain your answer.

Problem 2: Counting unique items (30 points). In this part, you will explore different methods to **estimate the number of unique items** in a given list. You will implement simulations to evaluate the accuracy of the estimations.

1. Assume a Bloom filter with parameters $m = 100$ bits and $k = 5$ hash functions. Use the provided `BloomFilter` implementation and complete the code in the given notebook.

   (a) (5 points) **Estimation Based on Zero Bits**. Complete the function `estimate_unique_items(items, m, k)` that estimates the number of unique items by counting the fraction of **zero bits** remaining in the Bloom filter. Next, run a simulation for 100 trials. Use the provided code.

   (b) (10 points) **Estimation Based on One Bits**. In this part, you will modify your estimation method to use the number of **1 bits** (bits set to 1) instead of zeros.

      i. Think: If p_0 is the fraction of zero bits, then $p_1 = 1 - p_0$ is the fraction of one bits. Derive a new estimator for n based on p_1.

ii. Implement the function `estimate_unique_items_using_ones(items, m, k)` that uses your new formula.

iii. Compare the results from the two estimation methods (zeros vs. ones). Are they equivalent? Which is more accurate and why?

2. (15 points) In this part, you will estimate the number of unique items using a single hash function and the number of **trailing zeros** in the binary representation of the hash.

   (a) Implement the function `estimate_unique_items_trailing_zeros(items)` based on the method we saw in class. Next, run a simulation for 100 trials, as was done above. For each trial, use the code provided to generate 100 random items with a varying number of duplicates. Compute and print the average absolute estimation error over all trials.

   (b) Modify the previous estimator that used trailing zeros so that it counts **trailing ones** instead. Implement the function `estimate_unique_items_trailing_ones(items)` that: Hashes each item to an integer. It then counts the number of consecutive **1**'s from the least significant bit in the binary representation. Lastly, estimate the number of unique items based on that. Think how and if the estimator should change.

   (c) Compare the results of this estimator with the trailing-zero version. Which one gives a more stable and accurate estimate? why?

Problem 3: Jaccard Similarity (40 points).

1. (5 points) Suppose two sets A and B such that $A \subseteq B$. Prove or disprove the following statement, justifying your answer:

   "If $A \subseteq B$, then their MinHash signatures are guaranteed to match perfectly."

2. (5 points) Prove or disprove the following statement, justifying your answer:

   "As the number of hash functions k increases, the probability that the MinHash similarity approximates the Jaccard similarity for two sets also increases."

3. (5 points) Prove or disprove the following statement, justifying your answer:

   "If two sets A and B have identical MinHash signatures given k hash functions, then $A = B$."

4. (25 points) **Estimating Jaccard Similarity Using Bloom Filters**
   In this part, you will explore an alternative way to estimate the Jaccard similarity between two sets using Bloom filters instead of MinHash.

   (a) **Conceptual (10 points):** Suppose you represent two sets A and B with Bloom filters of the same size m and number of hash functions k. Let $BF(A)$ and $BF(B)$ denote the bit arrays of these Bloom filters. Propose an estimator for the Jaccard similarity based on the fraction of bits that are set to 1 in each filter. Explain intuitively why this could approximate the true Jaccard similarity.

   (b) **Implementation and Simulation (5 points):** Implement this estimator. Complete the code in the given notebook.

   (c) **Explanation (10 points)**: Explain why estimating Jaccard similarity using the fraction of bits set in Bloom filters is generally less accurate than using MinHash. Suggest possible ways to improve the accuracy of your Bloom filter–based estimator.

Good Luck!