SEEM5020 2026 Spring: Homework 5

This assignment is due at 1:30 pm, February 13, 2026. Please bring your solution and submit it at the beginning of the next class. Total weight of Homework 5: 4%.

- **Q1. [10 Point] λ-Snapshot vs. Exponential-Histogram (DGIM)**

  The λ-Snapshot algorithm (Slide 34, Lec 3) and the Exponential-Histogram (DGIM) algorithm are both designed for sliding window counting. Compare the two algorithms in terms of space complexity, query flexibility (e.g., handling arbitrary window sizes $k \leq N$), generalizability (e.g., handling non-binary streams), and implementation simplicity.

- **Q2. [10 Point] Space-Efficient Euclidean LSH**

  In Locality-Sensitive Hashing for Euclidean distance, we often use a composite hash function $g(\mathbf{x}) = (h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_k(\mathbf{x}))$ where each $h_i(\mathbf{x}) = \lceil \frac{\mathbf{u}_i \cdot \mathbf{x} + b_i}{r} \rceil$. Storing the k-tuples into L hash tables directly is space-consuming, costing $O(L \cdot k \cdot n)$ space if we store the full k-tuples.

  Design a strategy to map a k-tuple to a single integer $H(\mathbf{x})$. Your design should hash tuples into a table of size P, where P is a prime number with $P \geq n^3$. Derive an upper bound on the probability that there exist two vectors $\mathbf{x}_1, \mathbf{x}_2$ from the input set of n vectors such that $g(\mathbf{x}_1) \neq g(\mathbf{x}_2)$ but they hash to the same slot. If we directly create an array of size n^3, the space cost will be too high. How can we reduce the space cost to $O(n)$ with $O(k)$ lookup cost?

- **Q3. [10 Point] Success Probability of LSH with Candidate Pruning**

  To bound the query complexity, the LSH search algorithm stops searching when a certain number of candidates have been examined. Suppose we use the standard parameter setting $k = \log_{1/p_2} n$ (assume that k is an integer). We set the number of hash tables $L = 2n^\rho$ and set the stopping threshold to $5n^\rho$ (i.e., the search terminates after examining $5n^\rho$ points that are "far" from the query).

  Analyze the success probability that the LSH query algorithm correctly returns a c-approximate r-near neighbor under this setting.

- **Q4. [10 Point] Using LSH to answer c-approximate Nearest Neighbor query**

  Assume that we want to use the LSH to answer the c-approximate nearest neighbor search query (Ref. to Definition 3 in Slides 4). Suppose that we know $\log_2(d_{max}/d_{min}) = \log_2 n$ where n is the number of data points. Please show how to set L and the stopping threshold (in the above example it stops when $5n^\rho$ points are examined) of the LSH search algorithm so that we return a c-approximate nearest neighbor search with probability at least 4/5.

  Remark: You may need to build multiple LSH structures for different distance thresholds r.