Algorithms for Big Data Fall 2020

Homework 4

Due: Monday 11/09/20, 5pm PT

- **Solving 3 of the following 4 problems** will lead to full credit. You may attempt all 4 problems, but the grading will be based on the 3 problems with the highest scores.
- You may work in groups of size 1-3. If you do, please hand-in a single assignment with everyone's names on it. It is strongly encouraged to type up the solutions in Latex.
- If the question asks to prove something, you must write out a formal mathematical proof.
- If the question involves analyzing an algorithm, you must formally explain the time and/or space usage, along with the approximation guarantees (when applicable).
- When you are asked to prove a bound, it suffices to prove it up to multiplicative constants, i.e., using $O(\cdot)$, $\Theta(\cdot)$, or $\Omega(\cdot)$ notation. No need to optimize (multiplicative) constants!
- You may use other resources, but you must cite them. If you use any external sources, you still must provide a complete and self-contained proof/result for the homework solution.

Problem 1: Set Similarity

Consider a dataset X that consists of sets of integers in the universe $\{1, 2, \ldots, n\}$, i.e., X is a set of sets. For example, there may be a set $A \in X$ which is $A = \{1, 4, 33\}$, and another set $B = \{2, 4, 33\}$. One way to measure the similarity of two non-empty sets is using Jaccard similarity:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}.$$

In the above example, $J(A, B) = \frac{|\{4,33\}|}{|\{1,2,4,33\}|} = 2/4 = 1/2$.

(a) Prove that the Jaccard distance $d(A, B) = 1 - J(A, B)$ is a valid distance metric for pairs of sets A, B (that is, show that it satisfies the three properties of a metric).

(b) Consider the following LSH family for Jaccard distance. Each hash function will be based on a random permutation π of the universe $\{1, 2, \ldots, n\}$. Then, we let h_π operate on sets as follows:

$$h_\pi(A) = \operatorname*{argmin}_{a \in A} \pi(a),$$

so $h_\pi(A)$ is the "minimum valued" element in A according to π. Prove that

$$\Pr[h_\pi(A) = h_\pi(B)] = J(A, B),$$

where the probability is over a uniformly random permutation $\pi : [n] \to [n]$.

(c) Explain how you might use the hash family h_π as an LSH family for ANNS under Jaccard distance. What values of r, c make sense? What values of p_1 and p_2 can you achieve? You do not need to analyze the formal near neighbor algorithm (a high-level description suffices).

Problem 2: 1D is Easy

Provide an algorithm for exact nearest neighbor search on the real line $\mathbb{R}$. For simplicity, assume you have a dataset of n vectors $X \subseteq \mathbb{R}$ such that each vector can be represented using $O(\log n)$ bits (for example X may consist of integers between $-n^2$ and $+n^2$).

The overall space of the algorithm should be $O(n \log n)$. Given a query $q \in \mathbb{R}$ you want to find the closest vector to q in X, that is, output

$$\operatorname*{argmin}_{x \in X} |x - q|.$$

A nearest neighbor query with your data structure should use $O(\log n)$ comparisons; so, the total query time should be $O(\log^2 n)$.

Hint: Consider a binary search tree over the vectors of X, and break up the real line into intervals. For each interval, store the largest and smallest vectors from X in the interval.

Problem 3: Small Hamming Distance

Update (11/4/20): The previous query times were incorrect and they have been corrected.

Let $X \subseteq \{0,1\}^d$ be a dataset of n vectors consisting of d bits each. This problem will show how to solve exact nearest neighbor for Hamming distance one and two. For each subproblem, design a deterministic data structure, prove that it works as desired, and analyze the time/space.

(a) Given a query $q \in \{0,1\}^d$ output a vector $x \in X$ with $d_{\mathrm{H}}(q,x) = 1$ if there exists such a vector. Query time $O(d^2 \log n)$. Space $O(nd)$.

(b) Given a query $q \in \{0,1\}^d$ output a vector $x \in X$ with $d_{\mathrm{H}}(q,x) \leq 2$ if there exists such a vector. Query time $O(d^3 \log n)$. Space $O(nd)$.

(c) Given a query $q \in \{0,1\}^d$ output a vector $x \in X$ with $d_{\mathrm{H}}(q,x) \leq 2$ if there exists such a vector. Query time $O(d^2 \log(nd))$. Space $O(nd^2)$.

Hint: Consider the nd possible vectors that have Hamming distance one from vectors in X.

Problem 4: Implementing Dimensionality Reduction

Implement and test the Johnson-Lindenstrauss dimensionality reduction method from Lecture 10. The goal here is for you to explore how well the dimensionality reduction works as you change the matrix and the dimensionality of the embedded data. Discuss your findings in addition to providing the experimental results. You do not need to provide code, and you can use whatever programming language you are comfortable with.

(a) Find *two* datasets of $n \geq 200$ points, either randomly generated or from a public repository (e.g., UCI, ScikitLearn, etc). The dimension of the dataset should be at least $d \geq 100$.

(b) Provide results (in a table or plot, clearly labeled) for the distortion of the projected points versus the original points, as you increase the dimensionality of the embedded points (e.g., compare the distortion as you scale from a small number of dimensions to the true dimension of the dataset). Is the behavior the same or different for the two datasets?

(c) Replace the normal distribution with ± 1 random variables. How does the embedding change (better, worse, different, ...)?

(d) **Extra Credit.** Consider the sparse JL transform, where the matrix has entries in $\{-1, 0, 1\}$ where ± 1 occurs with equal probability for the nonzero entries, but some fraction of each row is fixed to be 0 (the 0 entries chosen randomly in each row). How many non-zero entries do you need to achieve comparable distortion to the case where all entries are normal or ± 1?