Data Science Through a Geometric Lens Fall 2019

Homework 3

Due: Wednesday 11/13/19, 5pm

- **Solving 3 of the following 5 problems** will lead to full credit. You may attempt more than 3 problems, but the grading will be based on the 3 problems with the highest scores.
- Email the solutions to both the instructor and TA (emails listed on the course website).
- 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.

1 Problem 1: Dynamic Data

This problem considers an extension of the approximate near neighbor search (ANNS) data structure that handles insertions and deletions. You may work with Hamming distance, Euclidean distance, or generally in terms of an LSH family.

(a) Insertions and deletions: after the initial dataset X has been processed, there will be additional vectors that must be inserted into X or removed from X. Queries should search for close vectors in the updated input set. Provide an ANNS data structure for this variant.

(b) What is the time complexity of inserting/deleting a vector from your ANNS data structure (both in terms of the k, L parameters, and in terms of n for the k, L settings used in lecture)?

2 Problem 2: 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).

3 Problem 3: 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.

4 Problem 4: Small Hamming Distance

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 $O(d \log n)$. 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 $O(d^2 \log n)$. 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 $O(d \log(nd))$. Space $O(nd^2)$.

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

5 Problem 5: Implementing ANNS

Implement and test the approximate near neighbor data structure using locality sensitive hashing (either for Hamming distance or Euclidean distance or Cosine similarity).

(a) Find a dataset of $n \geq 500$ vectors, either randomly generated or from a public repository (e.g., UCI, ScikitLearn, etc). Split off 10% of the dataset to be the query set, and the rest will be the input dataset. The dimensionality should be fairly large (e.g., ≥ 32).

(b) Implement the LSH family and the ANNS data structure, parameterized by k, L, the distance threshold r, and the approximation factor $c \geq 1$. You may also want an additional parameter to limit the number of comparisons if a query has no close point in the dataset.

(c) Compare the accuracy of the ANNS data structure with a linear scan (which should have 100% recall). Use a value of r that will give interesting results (e.g., there are only a few vectors in the dataset at distance $\leq r$ from most queries).

(d) Provide results (in a table or plot, clearly labeled) for the efficiency and recall of the ANNS data structure as you vary the two parameters k, L and possibly also for different distance thresholds. How does the ANNS recall compare to linear scan? Is there a way to choose k and L so that the approximate algorithm achieves $\geq 90\%$ recall, while performing much fewer comparisons than the linear scan? How do the best k and L compare to the theoretical results?