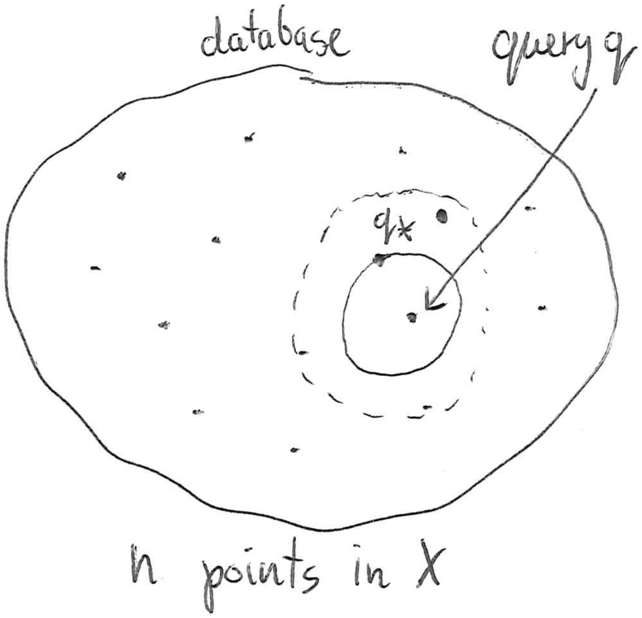
DS-563 / CS-543 Lecture 12 / 2023-03-02

Near neighbor search via Locality Sensitive Hashing (LSH)

Scenario: new point arrives, find something similar in your database



Ideal: return the closest point q_*

With techniques today: return something approximately closest, at distance up to

$$c \cdot \text{dist}(q, q_*)$$

c ↑ some fixed constant

"Approximate Nearest Neighbor Search"

Simplifying the task: Near~~est~~ Neighbor Search

Task: if there is a point at distance r,
return something up to distance $r' = c \cdot r$

Often: To solve "nearest" it suffices to consider small number of instances of "near" with different values of r

Naive solution:

- Assumption: all n points live in d-dimensional space X
- compare all points to the query point: $O(nd)$ time
- Space: $O(nd)$ as well

Want something faster!

Locality sensitive hash function family H

H is an (r, r', p_1, p_2) - locality sensitive hash function family if

for each $u, v \in X$:

$$\text{dist}(u,v) \leq r \Rightarrow \Pr[h(u) = h(v)] \geq p_1$$

and

$$\text{dist}(u,v) > r' \Rightarrow \Pr[h(u) = h(v)] \leq p_2$$

$\boxed{12-2}$

$$(0 < r < r') \quad (0 < p_2 < p_1 < 1)$$

Example: $X = \{0,1\}^d$ $\quad dist(u,v) = \|u-v\|_1$

Hamming distance

$$h_i(u_1 u_2 \dots u_d) = u_i$$

$i \in [d]$

$H = \{h_i : i \in [d]\}$ is a $(0.2d, 0.4d, 0.8, 0.6)$-locality sensitive hash function family

---

Intuition:

- close points likely to be mapped to the same bucket
- far points likely to be mapped to different buckets

How to use this?

Lots of far from q points can still be mapped to the same bucket!

Step 1: Avoiding too many unwanted collisions

New hash function g concatenates results of many random $h \in H$

$$g : X \to Y^k$$

$$g(x) = (h_1^*(x), h_2^*(x), \dots, h_k^*(x))$$

drawn independently from H

Question: What is $\Pr[g(u)=g(v)]$ if $\text{dist}(u,v) \geq r'$?

Answer: $\leq p_2^k$

Want this to be $\sim \frac{1}{n}$ so expected collisions with far points at most constant

Set $k = \left\lceil \frac{\log n}{\log(1/p_2)} \right\rceil$

Probability a close point in the same bucket?

$$\geq p_1^k \geq p_1^{\left(\frac{\log n}{\log(1/p_2)}+1\right)} = p_1 \cdot n^{-\frac{\log(1/p_1)}{\log(1/p_2)}}$$

$$= p_1 \cdot n^{-\rho} \qquad \rho \in (0,1) \qquad \rho = \frac{\log(1/p_1)}{\log(1/p_2)}$$

[This could be small!]

Step 2: Repeat hashing for many independent g

Probability of q in the same bucket as a given close point $\geq p_1 \cdot n^{-\rho}$ $\Rightarrow$ Homework 1

Repeat $\Omega(1/(p_1 \cdot n^{-\rho})) = \Omega\left(\frac{1}{p_1} \cdot n^{\rho}\right)$ times to $\underbrace{\text{“find” the close point}}_{\text{|||}}$ with constant probability

end up in the same bucket
as q

Full approximate near neighbor data structure

Preprocessing: - select $k' = \Theta\left(\frac{1}{p_1} \cdot n^{\rho}\right)$ hash functions g_i, each a "concatenation of results" of $k = \left\lceil \frac{\log n}{\log(1/p_2)} \right\rceil$ independently selected hash functions from H

- create k' hash ~~tt~~ tables with all n points in our database

$$\text{Time: } O(k' \cdot k \cdot d \cdot n) \quad \boxed{= O(n^{1+\rho}\, d \log n)}$$

$$\text{Space: } O(nd + k'n) \quad \boxed{= O(nd + n^{1+\rho})}$$

for typical d dimensional data

for constant p_1 & p_2

Query q

- go over all k' hash tables
- compute the distance of all points hashed to the same buckets as q
- stop when you find a point at distance at most r_2 from q

Expected time:

$$O(\underbrace{k' \cdot k \cdot d}_{\text{computing hash functions}} + \underbrace{k' \cdot O(1) \cdot d}_{\text{expected distance computation for points at distance } > r' \text{ from } q} + d)$$

d: distance computation for the point close to q

$$\boxed{= O(n^{\rho} d \log n)} \quad \text{vs. naive } O(nd)$$

for constant p_1 & p_2