Class 8

Locality Sensitive Hashing
Announcements

• HW3 due tomorrow!
• HW4 out now!
• Please fill out feedback form!
Recap

• Johnson-Lindenstrauss Transforms!
Recap

• Intro to Nearest Neighbor Search

<table>
<thead>
<tr>
<th>Method</th>
<th>Space</th>
<th>Query Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear scan</td>
<td>$O(nd)$</td>
<td>$O(nd)$</td>
</tr>
<tr>
<td>Various ways of generalizing the $d = 1$ solution</td>
<td>$n^{O(d)}$</td>
<td>$d^{O(1)} \log n$</td>
</tr>
<tr>
<td>Other heuristics</td>
<td>$O(nd)$</td>
<td>$\Omega(n)$ in the worst case</td>
</tr>
</tbody>
</table>
$c$-near neighbors

Given $y$, find $x_j$ so that
$$
\| y - x_j \|_2 \leq c \left( \min_i \| y - x_i \|_2 \right)
$$

**Goals:**
- Space = $(d \cdot n)O(1)$
- Query time = $o(n)$

(For today all of our points live on the unit sphere.)
$c$-near-neighbors

Imagine that this slide is the surface of the unit sphere…

$$r = \min_{i} (\| x_i - y \|_2)$$

Okay to return this.
Today: \((r, c)\)-near-neighbors

(For today all of our points live on the unit sphere.)

\[ S = \{ x_1, x_2, \ldots, x_n \} \subseteq S^d \]

**Goals:**
- Space = \((d \cdot n)^{O(1)}\)
- Query time = \(o(n)\)

Before:

Given \(y\), find \(x_j\) so that \( \| y - x_j \|_2 \leq c \left( \min_i \| y - x_i \|_2 \right) \)

Given \(y\) so that \( \min_i \| y - x_i \|_2 \leq r \)

find \(x_j\) so that \( \| y - x_j \|_2 \leq c \cdot r \)
$(r, c)$-near-neighbors

Imagine that this slide is the surface of the unit sphere....

Okay to return this.
$c$-NN vs $(r, c)$-NN

$r = \min (\|x_i - y\|_2)$
Fact

• If you can solve \((r, c)\)-nearest neighbors then you can (basically) solve \(c\)-nearest neighbors.

• (See lecture notes).
Goal for today

• A solution to $(r, c)$-approximate nearest neighbors.

• Tool: **Locality-Sensitive Hashing.**
  • Points that are near to each other have a good probability of colliding.
  • Points that are far from each other are unlikely to collide.

• Strategy:
  • Data structure: hash all the $x_i$
  • To query, hash $y$. Return anything in $y$’s bucket.

Our strategy will actually be slightly more complicated than this, but this is the basic idea...
Our Locality Sensitive Hashing Scheme
Our Locality Sensitive Hashing Scheme

• Let $A \in \mathbb{R}^{k \times d}$ have i.i.d. $N(0,1)$ entries.
  • Here, $k = \frac{\pi \log n}{2r}$ (we’ll see why later).
Our Locality Sensitive Hashing Scheme

• Let $A \in \mathbb{R}^{k \times d}$ have i.i.d. $N(0,1)$ entries.
  • Here, $k = \frac{\pi \log n}{2r}$ (we’ll see why later).
• Define $h(x) = \text{sign}(Ax)$

\[ A \]
Our Locality Sensitive Hashing Scheme

• Let $A \in \mathbb{R}^{k \times d}$ have i.i.d. $N(0,1)$ entries.
  • Here, $k = \frac{\pi \log n}{2r}$ (we’ll see why later).
• Define $h(x) = \text{sign}(Ax)$
Our Locality Sensitive Hashing Scheme

- Let $A \in \mathbb{R}^{k \times d}$ have i.i.d. $N(0,1)$ entries.
  - Here, $k = \frac{\pi \log n}{2r}$ (we’ll see why later).
- Define $h(x) = \text{sign}(Ax)$
Actually choose $s$ independent copies of this

- Choose $s = \sqrt{n}$
- Choose $k = \frac{\pi \log n}{2r}$
- For $i = 1, \ldots, s$:
  - Let $A_i \in \mathbb{R}^{k \times d}$ have i.i.d. $N(0,1)$ entries.
  - Define $h_i(x) = \text{sign}(A_ix)$
Outline of group work

• First (problems 1-5) you will show that:
  • If $x, y$ are close, then probably there’s some $i$ so that $h_i(x) = h_i(y)$
  • If $x, y$ are far, then probably there’s no such $i$.

• Then (problems 6,7), you will show how to use this to get a $(c, r)$-near-neighbors scheme.
1. Consider a hash function $h_i : \mathbb{S}^d \to \{\pm 1\}^k$ as defined above. Explain why “$h_i(x) = h_i(y)$” has the following geometric meaning:

Imagine choosing $k$ uniformly random hyperplanes in $\mathbb{R}^d$, and using them to slice up the sphere $\mathbb{S}^d$ like this:

Then $h_i(x) = h_i(y)$ if and only if $x$ and $y$ are in the same “cell” of this slicing. For example, in the picture below $h_i(x) = h_i(y) \neq h_i(z)$.

2. Explain why, for $x, y \in \mathbb{S}^d$, and for any $i = 1, \ldots, s$,

$$\Pr[h_i(x) = h_i(y)] = \left(1 - \frac{\text{angle}(x, y)}{\pi}\right)^k,$$

where $\text{angle}(x, y) = \arccos(x^T y)$ is the arc-cosine of the dot product of $x$ and $y$, aka, the angle between $x$ and $y$.

**Hint:** Think about the geometric intuition in the plane spanned by $x$ and $y$.

3. Suppose that $x, y \in \mathbb{S}^d$. Fill in the blank, using the previous part:

$$\Pr[\forall i \in \{1, \ldots, s\}, h_i(x) \neq h_i(y)] = \ldots$$

(Don’t worry about simplifying, you’ll do that in the next part).

4. Let $x, y \in \mathbb{S}^d$ and suppose that the angle between $x$ and $y$ is pretty small. Using our choices of $s$ and $k$ above, along with extremely liberal use of the approximation that $1 - x \approx e^{-x}$ for small $x$, convince yourself that

$$\Pr[\forall i \in \{1, \ldots, s\}, h_i(x) \neq h_i(y)] \approx \exp\left(-n^{1/2-\text{angle}(x,y)/(2\pi)}\right).$$

5. Fill in the blanks (assuming that your approximation from the previous step is valid):

   (a) If $\text{angle}(x, y) \leq r$, then

   $$\Pr[\exists i \in \{1, \ldots, s\} \text{ so that } h_i(x) = h_i(y)] \geq \ldots$$

   (b) If $\text{angle}(x, y) \geq 5r$, then

   $$\Pr[\exists i \in \{1, \ldots, s\} \text{ so that } h_i(x) = h_i(y)] \leq \ldots$$
Question 1

Interpreting “h(x) = h(y)”
Question 1

• For each row $a^T$ of $A$, we have a hyperplane $\{x \in \mathbb{R}^d : a^T x = 0\}$. 

\[
\begin{bmatrix}
0.2 \\
-1.3 \\
0.7 \\
3.2 \\
-50 \\
0.01
\end{bmatrix}
\begin{bmatrix}
0.2 \\
-1.3 \\
0.7 \\
3.2 \\
-50 \\
0.01
\end{bmatrix}
= 
\begin{bmatrix}
+1 \\
-1 \\
+1 \\
+1 \\
-1 \\
+1
\end{bmatrix}

Interpreting “$h(x) = h(y)$”
Question 1

• For each row $a^T$ of $A$, we have a hyperplane $\{x \in \mathbb{R}^d : a^T x = 0\}$.

• If the corresponding coordinate of $Ax$ is negative, then $x$ lies on one side of the hyperplane, else $x$ lies on the other.

Interpreting “$h(x) = h(y)$”
Question 1

• For each row $a^T$ of $A$, we have a hyperplane $\{x \in \mathbb{R}^d : a^T x = 0\}$.

• If the corresponding coordinate of $Ax$ is negative, then $x$ lies on one side of the hyperplane, else $x$ lies on the other.

• Same cell = same side of every hyperplane = same sign in every coordinate.

Interpreting “$h(x) = h(y)$”
Question 2
\[ \Pr[h_i(x) = h_i(x)] = \left(1 - \frac{\text{angle}(x, y)}{\pi}\right)^k \]
Question 2

\[ \Pr[h_i(x) = h_i(x)] = (1 - \frac{\text{angle}(x, y)}{\pi})^k \]
Question 2

\[ \Pr[h_i(x) = h_i(y)] = (1 - \frac{\text{angle}(x, y)}{\pi})^k \]
Question 2
\[ \Pr[h_i(x) = h_i(y)] = (1 - \frac{\text{angle}(x, y)}{\pi})^k \]
Question 3

Q2: \( \Pr[\forall i \in \{1, \ldots, s\}, h_i(x) \neq h_i(y)] = \ldots \)

\( \Pr[h_i(x) = h_i(x)] = (1 - \text{angle}(x, y)/\pi)^k \)
Question 3

\[ \Pr \left\{ \forall i, h_i(x) \neq h_i(y) \right\} = \left( 1 - \left( 1 - \frac{\text{angle}(x_i, y)}{n} \right)^k \right)^s \]

- \( \Pr[h_i(x) = h_i(y)] \) for any one \( i \)
- \( \Pr[h_i(x) \neq h_i(y)] \) for any one \( i \)
- \( \Pr[h_i(x) \neq h_i(y)] \) for all \( s \) values of \( i \)
Question 4
Question 4

With our choice of $s = \sqrt{n}$, $k = \frac{11 \log n}{2r}$, 

$$
\mathbb{P}\left\{ \forall i, h_i(x) = h_i(y) \right\} = \left(1 - \left(1 - \frac{\text{angle}(x,y)}{\pi} \right)^k\right)^s \\
\approx \left(1 - \exp\left(-\log(n) \cdot \frac{\text{angle}(x,y)}{2r}\right)\right)^s \\
\approx \exp\left(-n \cdot n^{\text{angle}(x,y)/2r}\right) \\
= \exp\left(-n^{1/2} \cdot \text{angle}(x,y)/2r\right)
$$
Question 5(a)
Question 5(a)

If \( \angle(x, y) \leq r \),

\[
\mathbb{P}\{ \forall i, \ h_i(x) + h_i(y) \} \leq \exp\left(-n \frac{1}{2} \angle(x, y) r \right) \geq \exp(-1)
\]

\[
\mathbb{P}\{ \exists i, \ h_i(x) = h_i(y) \} \geq 1 - e^{-1}.
\]
Question 5(b)
Question 5(b)

If \( \text{angle}(x, y) \geq 5r \),

\[
\exp\left(-n^{\frac{1}{2} - \text{angle}(x, y)/2r}\right) \leq \exp\left(-n^{\frac{1}{2} - 5/2}\right)
\]

\[= \exp\left(-n^{-2}\right)\]

\[\approx 1 - \frac{1}{n^2}\]

\[P\{\exists i, h_i(x) = h_i(y)\} \leq \frac{1}{n^2}\]
Question 6
Question 6

• Query Algorithm:
  • For $i = 1, 2, \ldots, s$:
    • Compute $h_i(y)$
    • If there’s some $x_j$ so that $h_i(x_j) = h_i(y)$, return it.
Question 6

• Query Algorithm:
  • For $i = 1, 2, \ldots, s$:
    • Compute $h_i(y)$
    • If there’s some $x_j$ so that $h_i(x_j) = h_i(y)$, return it.

• If $\text{angle}(y, x_\ell) \leq r$, then with decent probability there’s some $i$ so that $h_i(x_\ell) = h_i(y)$.
  • In particular, the algorithm will return something.
Question 6

• Query Algorithm:
  • For $i = 1, 2, ..., s$:
    • Compute $h_i(y)$
    • If there’s some $x_j$ so that $h_i(x_j) = h_i(y)$, return it.

• If $\text{angle}(y, x_\ell) \leq r$, then with decent probability there’s some $i$ so that $h_i(x_\ell) = h_i(y)$.
  • In particular, the algorithm will return something.

• If the algorithm returns $x_j$ then with high probability $\text{angle}(y, x_j) \leq 5r$.
  • If $\text{angle}(y, x_j) > 5r$, $\Pr[\exists i, h_i(x_j) = h_i(y)] \leq \frac{1}{n^2}$, and we can union bound over all $x_j$ to say that never happens whp.
Question 7
Question 7

Using $\frac{2}{\pi} \angle(x, y) \leq ||x - y||_2 \leq \angle(x, y)$, we can conclude:

- If $\angle(x, y) \leq r$, $\mathbb{P}\{ \exists i, h_i(x) = h_i(y) \} \geq 1 - \frac{1}{n}$.

  $||x - y||_2 \leq \frac{\pi}{2} \cdot r$

- If $\angle(x, y) \geq 5r$, $\mathbb{P}\{ \exists i, h_i(x) = h_i(y) \} \leq \frac{1}{n^2}$

  $||x - y||_2 \geq 5r$

So just fiddle with the value of “c” and the same analysis will still work.
Wrapping up: Time and Space

• Space:

  \[ k = O(\log n) \]
  \[ s = \sqrt{n} \]

• Update time:
Wrapping up: Time and Space

• Space:
  • $s$ different $k \times d$ matrices $A_i$: $O(d \cdot \sqrt{n} \cdot \log n)$
  • $s$ hash tables, each with $2^k$ buckets: $O\left(\sqrt{n} \cdot 2^{O(\log n)}\right) = n^{O(1)}$
  • The elements of $S$ themselves: $O(nd)$

• Update time:
Wrapping up: Time and Space

• Space: $n^{O(1)}$
  • $s$ different $k \times d$ matrices $A_i$: $O(d \cdot \sqrt{n} \cdot \log n)$
  • $s$ hash tables, each with $2^k$ buckets: $O\left(\sqrt{n} \cdot 2^{O(\log n)}\right) = n^{O(1)}$
  • The elements of $S$ themselves: $O(nd)$

• Update time:
Wrapping up: Time and Space

• Space: \( n^O(1) \)
  - \( s \) different \( k \times d \) matrices \( A_i \): \( O(d \cdot \sqrt{n} \cdot \log n) \)
  - \( s \) hash tables, each with \( 2^k \) buckets: \( O(\sqrt{n} \cdot 2^O(\log n)) = n^O(1) \)
  - The elements of \( S \) themselves: \( O(nd) \)

• Update time:
  - \( s \) different \( k \times d \) matrix-vector multiplies: \( O(s \cdot k \cdot d) = O(d \sqrt{n} \cdot \log n) \)
  - Going through all \( s \) hash tables and look in \( h_i(y) \)'s bucket to see if there's anything else: \( O(s) = O(\sqrt{n}) \)

\[ k = O(\log n) \]
\[ s = \sqrt{n} \]
Wrapping up: Time and Space

- **Space:** \( n^{O(1)} \)
  - \( s \) different \( k \times d \) matrices \( A_i \): \( O(d \cdot \sqrt{n} \cdot \log n) \)
  - \( s \) hash tables, each with \( 2^k \) buckets: \( O(\sqrt{n} \cdot 2^{O(\log n)}) = n^{O(1)} \)
  - The elements of \( S \) themselves: \( O(nd) \)

- **Update time:** \( O(d \cdot \sqrt{n} \cdot \log n) = o(n) \) when \( d \) isn’t too big.
  - \( s \) different \( k \times d \) matrix-vector multiplies: \( O(s \cdot k \cdot d) = O(d \sqrt{n} \cdot \log n) \)
  - Going through all \( s \) hash tables and look in \( h_i(y) \)’s bucket to see if there’s anything else: \( O(s) = O(\sqrt{n}) \)
Recap

• We can use dimension reduction (that smells a bit like JL) to make an efficient c-near-neighbors algorithm!