Nearest neighbor searching

Preprocess points \( P = \{ p_1, \ldots, p_n \} \)
Later, given point \( q \)
quickly find \( \min_{p \in P} \| q - p \| \)

Euclidean distance in \( \mathbb{R}^d \)
Fast alg/small space when \( d \leq 2 \)

Brute Force: \( O(dn) \) time

\( d=1: \) Binary search, \( O(\log n) \) time \( O(n) \) space

\( d=2: \) Voronoi diagram [1906]
Insert: $O(1)$ exp modif.
Query: $O\left(\frac{2}{n}\right)$ exp work for
$n$th pt
\Rightarrow $O(\log n)$ exp time

$V - E + F = 2$
$V \leq 2n - 4$  $E \leq 3n - 6$
Approx. nearest neighbor query:

Return any point \( p' \in P \) s.t.

\[ d(p, q) \leq c \cdot \min \{ d(p, q), d(p', q) \} \]

with prob \( 1 - \delta \).

**Locality-sensitive hashing**

**Approx. near neighbor search**

Given \( P \) to preprocess

Given point \( q \), distances \( r, R \)

If there is a point \( p \in P \) s.t. \( d(p, q) \leq r \)

return a point \( p' \in P \) s.t. \( d(p', q) \leq R \)

with prob \( 1 - \delta \).
A family $H$ of hash functions is \textit{locally sensitive} if

\begin{align*}
\text{If } d(p, q) < r & \implies \Pr[h(p) = h(q)] \geq \frac{1}{2} \\
\text{If } d(p, q) > r & \implies \Pr[h(p) = h(q)] \leq \frac{1}{2}
\end{align*}

"Points" are $d$-bit vectors

\[ \text{dist}(p, q) = \#1's \text{ in } p \oplus q \quad \text{Hamming dist} \]

\[ p = 0000111101 \quad \text{dist} = 3 \]
\[ q = 0010011100 \]

\[ h_i(p) = p_i = \text{ith bit in } p \quad H = \{ h_i | 1 \leq i \leq d \} \]

\[ \Pr[h(p) = h(q)] = \frac{d - \text{dist}(p, q)}{d} = 1 - \frac{d(p, q)}{d} \]

\[ \frac{1}{2} = 1 - \frac{3}{8} \quad \frac{1}{2} = 1 - \frac{3}{8} \]

Pick \( k \cdot l \) functions \( h_{ij} \in H \) independently

\[ 1 \leq i \leq k \quad 1 \leq j \leq l \]

For all $j$: \( h_j(p) = (h_{1j}(p), \ldots, h_{kj}(p)) \in \{0, 1\}^k \)

Define "collide" = For some $j$, $h_j(p) = h_j(q)$
\[ \exists j \ \forall i \ h_{ij}(p) = h_{ij}(q) \]

\[ P_{\text{collision}} = 1 - (1 - \Pr[h_{ij}(p) = h_{ij}(q)]^k)^l \]

\[ = 1 - (1 - \left(1 - \frac{\text{dist}}{d}\right)^k)^l \]

For each \( j \):
- For each \( p \in P \)
  - Store \( p \) at \( T_j[h_{ij}(p)] \)

Query \( q \):
- For each \( j \):
  - \( S \leftarrow T_j[h_{ij}(q)] \)
  - Check every point in \( S \)
  - If close pt found, return it

Stop after checking 2l points.