Big Data

Solution to HW 2

Question 1

Part a

Denote the length of the stream by n.

Denote: $m = \log M$

For each point x in the steam, we calculate, for all $j = 1, \dots, m$, the distance $d(x, c)$ for all $c \in S_j$.

We keep $|S_j| \le k$ for all j.

Therefore, assuming that distance calculation is $O(1)$, we have that the running time of the algorithm is $O(nkm) = \boldsymbol{O(nk \log M)}$.

Part b

Denote $r_j = 2^{j-1}$ for all $j = 1, \dots, m$.

Denote the optimal k center radius by OPT.

Notice that the described algorithm actually runs m copies of the algorithm we saw in class in parallel. The $j'th$ copy uses $r_j = 2^{j-1}$ as the guess of OPT. So the guesses are $2^0, 2^1, 2^2, \dots, 2^{\log M - 1}$ i.e. $1,2,4, \dots, \frac{M}{2}$. When using a certain guess r_j, the algorithm compares the distances to $2 \cdot r_j = 2 \cdot 2^{j-1} = 2^j$.

Notice that necessarily $1 \le OPT \le M$, because we have that $1 \le d(x, y) \le M$ for all $x \ne y$.

Denote: $D = \max_{x \in X} \min_{c \in S} d(x, c)$

I will prove that: $D \le 4 \cdot OPT$

First, note that necessarily $D \le M$, because it is sufficient to have at least one point in S in order to achieve this, and indeed we have at least one point in S (the first point).

Therefore, in the case that $OPT \ge \frac{M}{2}$ we have:

$$D \le M \le 2 \cdot OPT < 4 \cdot OPT$$

As required.

Let's handle now the case that $OPT < \frac{M}{2}$.

In this case, there exist index $i \in \{1, \dots, m\}$ such that $\frac{r_i}{2} \le OPT$ and $r_i > OPT$.

Let's look at how the algorithm operated on S_i.

When a point x in the stream is not added to S_i, it is due to one of the following reasons:

(1) $\exists c \in S_i \;\; d(x,c) < 2r_i$
(2) $|S_i| = k$

When (1) happens, we can think about x as added to the cluster of c.

We can see that the radius of each cluster would be at most $2r_i$.

Could it happen that (1) was false but x was not added to S_i due to (2)?

Let's imagine for a moment that the algorithm didn't check (2), i.e. didn't care about $|S_i|$ exceeding k.

The distance between any $c, c' \in S_i$ is at least $2r_i > 2 \cdot OPT$.

Since optimal k center radius is OPT, every point in S_i must be in unique cluster of the optimal solution. There are only k such clusters, which means that we would finish with $|S_i| \leq k$ even without checking (2).

It means that for each point x in the stream, if (1) was false, necessarily (2) was false too and x was added to S_i.

Therefore we have that:

$$\forall x \in X \;\; \exists c \in S_i\!: \; d(x,c) \leq 2r_i$$

Since we have $\frac{r_i}{2} \leq OPT$, we get that:

$$\forall x \in X \;\; \exists c \in S_i\!: \; d(x,c) \leq 4 \cdot OPT$$

Therefore we get that:

$$D = \max_{x \in X} \min_{c \in S} d(x,c) \leq 4 \cdot OPT$$

As required.

Part c

The algorithm given in class is better in the following:

- It gives up to k centers (the number of centers we look for) versus $k \log M$, which is an approximation of the desired number of centers.
- Storage required is $O(k)$ versus $O(k \log M)$.
- It doesn't assume the bound M over the pairwise distances.

The algorithm given in class is worse in the following:

- It gives radius of $8 \cdot OPT$ versus $4 \cdot OPT$.

Question 2

Part a

For each unit vector $u \in R^d$, define a hash function $h_u: R^d \to \{0,1\}$ by:

$$h_u(p) = \begin{cases} 1, & p \cdot u \geq 0 \\ 0, & p \cdot u < 0 \end{cases}$$

Define the following hash family H:

$$H = \{h_u : \ u \in R^d, \|u\| = 1\}$$

I will prove that H is a $(\theta_1,\ (1+\varepsilon)\theta_1, 1-\frac{\theta_1}{\pi}, 1-\frac{(1+\varepsilon)\theta_1}{\pi})$ locally sensitive hash family.

Claim 1

Let $p, q \in R^d$.

Let $\theta \in [0, \pi]$ be the angle between p and q.

Given a vector $v \in R^d$, let's say that "v separates p and q" if $sign(p \cdot v) \neq sign(q \cdot v)$.

Pick a unit vector $u \in R^d$ randomly uniformly.

Then the probability that u separates p and q is: $\frac{\theta}{\pi}$

Proof

First, let's consider the case $d = 2$.

If $\theta = 0$, it is obvious that the probability that u separates p and q is: $\frac{\theta}{\pi} = 0$.

Assume now $\theta \neq 0$.

Denote by $L \subset R^2$ the line $L = \{x \in R^2 : \ x \cdot u = 0\}$ (u is a normal of L).

Then: u separates p and q iff L separates p and q (i.e. p and q are in different sides of L).

For a vector $v \in R^2$, denote by $a(v)$ the angle between v and the vector $(1,0)^T$.

Denote by $a(L)$ the angle between L and the vector $(1,0)^T$.

W.l.o.g. assume $a(p) < a(q)$.

Then: L separates p and q iff $a(L) \in [a(p), a(q)]$.

By picking u uniformly, $a(L)$ is picked uniformly from $[0, \pi]$. Therefore, the probability that L separates p and q is:

$$\frac{|\,[a(p), a(q)]\,|}{|\,[0, \pi]\,|} = \frac{a(q) - a(p)}{\pi} = \frac{\theta}{\pi}$$

As required.

Now, let's go on to the case $d > 2$.

Denote $W = span\{p, q\}$. Since $\theta \neq 0$, $\dim(W) = 2$.

Denote $W^{\perp} = \{x \in R^d\colon\ x \perp W\}$.

Represent u as $u = w + w^{\perp}$ where $w \in W,\ w^{\perp} \in W^{\perp}$.

So we have that $p \cdot u = p \cdot w$ and $q \cdot u = q \cdot w$.

Therefore: $sign(p \cdot u) \neq sign(q \cdot u)$ iff $sign(p \cdot w) \neq sign(q \cdot w)$.

Let $B = \{b_1, b_2\}$ be an orthonormal basis for W.

Let $p', q', w' \in R^2$ be the representations of p, q, w in the basis B.

Then:

$$sign(p \cdot w) = sign(p' \cdot w') = sign\left(p' \cdot \frac{w'}{\|w'\|}\right)$$

$$sign(q \cdot w) = sign(q' \cdot w') = sign\left(q' \cdot \frac{w'}{\|w'\|}\right)$$

Notice that by picking u randomly uniformly from the unit sphere in R^d, $\frac{w'}{\|w'\|}$ is picked randomly uniformly from the unit circle in R^2 (actually w' might be 0 in case $u \in W^{\perp}$, but this case can be neglected as it happens with probability 0, since $\dim(W^{\perp}) < d$).

By the proof for the case $d = 2$ we have that the probability that $\frac{w'}{\|w'\|}$ separates p and q is: $\frac{\theta}{\pi}$.

Therefore the probability that u separates p and q is: $\frac{\theta}{\pi}$.

■

Now let's prove that H is a $(\theta_1,\ (1+\varepsilon)\theta_1, 1 - \frac{\theta_1}{\pi}, 1 - \frac{(1+\varepsilon)\theta_1}{\pi})$ locally sensitive hash family:

Let $p, q \in R^d$.

Let $\theta \in [0, \pi]$ be the angle between p and q.

Pick $h_u \in H$ randomly uniformly.

Then actually u is picked randomly uniformly from the unit sphere.

Now:

$$\Pr\big(h_u(p) = h_u(q)\big) = \Pr(u \text{ doesn't separate } p \textit{ and } q) = [\textit{by claim } 1] = 1 - \frac{\theta}{\pi}$$

If $\theta \le \theta_1$ then:

$$\Pr\big(h_u(p) = h_u(q)\big) = 1 - \frac{\theta}{\pi} \ge 1 - \frac{\theta_1}{\pi}$$

If $\theta \ge (1+\varepsilon)\theta_1$ then:

$$\Pr\big(h_u(p) = h_u(q)\big) = 1 - \frac{\theta}{\pi} \le 1 - \frac{(1+\varepsilon)\theta_1}{\pi}$$

As required.

<u>Part b</u>

Given k unit vectors $u_1, \dots, u_k \in R^d$, define a hash function $h_{u_1,\dots,u_k}: R^d \to \{0,1\}^k$ by:

$$h_{u_1,\dots,u_k}(p) = \left(h_{u_1}(p), \dots, h_{u_k}(p)\right)$$

Define the following hash family H_k:

$$H_k = \{h_{u_1,\dots,u_k} : \ \forall j \ \ u_j \in R^d, \|u_j\| = 1\}$$

Let's prove that H_k is a $\left(\theta_1,\ (1+\varepsilon)\theta_1, \left(1-\frac{\theta_1}{\pi}\right)^k, \left(1-\frac{(1+\varepsilon)\theta_1}{\pi}\right)^k\right)$ locally sensitive hash family:

Let $p, q \in R^d$.

Let $\theta \in [0, \pi]$ be the angle between p and q.

Pick $h_{u_1,\dots,u_k} \in H_k$ randomly uniformly.

Then:

$$\Pr\left(h_{u_1,\dots,u_k}(p) = h_{u_1,\dots,u_k}(q)\right) = \Pr\left(h_{u_j}(p) = h_{u_j}(q) \ \ \forall j = 1, \dots, k\right) = \left(1 - \frac{\theta}{\pi}\right)^k$$

If $\theta \le \theta_1$ then:

$$\Pr\left(h_{u_1,\dots,u_k}(p) = h_{u_1,\dots,u_k}(q)\right) = \left(1 - \frac{\theta}{\pi}\right)^k \ge \left(1 - \frac{\theta_1}{\pi}\right)^k$$

If $\theta \ge (1+\varepsilon)\theta_1$ then:

$$\Pr\left(h_{u_1,\dots,u_k}(p) = h_{u_1,\dots,u_k}(q)\right) = \left(1 - \frac{\theta}{\pi}\right)^k \le \left(1 - \frac{(1+\varepsilon)\theta_1}{\pi}\right)^k$$

As required.

Part c

Denote:

- $p_1 = 1 - \frac{\theta_1}{\pi}$
- $p_2 = 1 - \frac{(1+\varepsilon)\theta_1}{\pi}$
- $\rho = \frac{\ln\left(\frac{1}{p_1}\right)}{\ln\left(\frac{1}{p_2}\right)}$

For all $x, y \in R^d$ denote the angle between them by:

$$dist(x, y) = \cos^{-1}\left(\frac{x \cdot y}{\|x\|\|y\|}\right)$$

Let's define the following algorithm:

Let $k = \log_{1/p2}(n)$

Let $c_1 = 6$

Let $L = \frac{c_1}{p_1^k}$

Notice that:

$$L = \frac{c_1}{p_1^k} = c_1\left(\frac{1}{p_1}\right)^{\log_{1/p2}(n)} = c_1\left(\frac{1}{p_1}\right)^{\frac{\log_{\frac{1}{p_1}}(n)}{\log_{\frac{1}{p_1}}\left(\frac{1}{p_2}\right)}} = c_1 n^{\frac{1}{\log_{\frac{1}{p_1}}\left(\frac{1}{p_2}\right)}} = c_1 n^{\frac{\ln\left(\frac{1}{p_1}\right)}{\ln\left(\frac{1}{p_2}\right)}} = c_1 n^{\rho}$$

We will use L hash tables $T_1, \dots, T_L$. Each hash table T_i will use hash function h_i from the family H_k (which I defined in part b), which is a $\left(\theta_1,\ (1+\varepsilon)\theta_1,\ p_1^k,\ p_2^k\right)$ locally sensitive hash family.

Preprocessing:

1. For $i = 1, \dots, L$
   a. For $j = 1, \dots, n$
      i. Insert x_j to T_i, to the bucket specified by $h_i(x_j)$.

Query:

Given a query $q \in R^d$, we will do the following:

1. For $i = 1, \dots, L$

a. Look at the bucket specified by $h_i(q)$. Scan all vectors (unless we reach the limit described below) in this bucket. If we find there a vector x such that $dist(x,q) \le (1+\varepsilon)\theta_1$, we stop and return x.

We will limit the algorithm to check up to total of $c_2 n^\rho$ vectors, where $c_2 = 1200$. If we reach that limit, we stop and return that no appropriate vector was found.

Part d

Denote $S = \{x_1, \dots, x_n\}$.

Let $q \in R^d$ be a query, and assume that $\exists x \in S \;\; dist(x,q) \le \theta_1$.

We need to prove that with probability at least 0.99, the algorithm will return some $y \in S$ such that $dist(y,q) \le (1+\varepsilon)\theta_1$.

First, notice that:

$$p_2^k = p_2^{\log_{1/p_2}(n)} = \left(\frac{1}{p_2}\right)^{-\log_{1/p_2}(n)} = \frac{1}{n}$$

Denote by E_1 the event that x will be mapped to the same bucket as q in at least one hash table, i.e. the event that $\exists i \; h_i(x) = h_i(q)$.

For each i, $\Pr(\, h_i(x) \ne h_i(q)) \le 1 - p_1^k$.

Therefore: $\Pr(\neg E_1) \le \left(1-p_1^k\right)^L = \left(1-p_1^k\right)^{\frac{c_1}{p_1^k}} \le \left(\frac{1}{e}\right)^{c_1} = \left(\frac{1}{e}\right)^6 < \frac{1}{200}$.

Now, denote by F the number of false positives, i.e. the number of times that a vector $z \in S$ such that $dist(z,q) > (1+\varepsilon)\theta_1$ was mapped to the same bucket as q.

Let's bound $E(F)$.

For a vector $z \in S$ which is "far" from q, i.e. $dist(z,q) > (1+\varepsilon)\theta_1$, the expected number of times that z and q are mapped to the same bucket is at most:

$L \cdot p_2^k = c_1 n^\rho \cdot \frac{1}{n} = c_1 n^{\rho-1}$.

In the worst case all n vectors are "far" from q, therefore we can bound $E(F)$ by:

$$E(F) \le n \cdot c_1 n^{\rho-1} = c_1 n^\rho$$

Denote by E_2 the event that $F \le c_2 n^\rho$.

Then:

$$\Pr(\neg E_2) = \Pr(F > c_2 n^\rho) \le [\text{Markov}] \le \frac{E(F)}{c_2 n^\rho} \le \frac{c_1 n^\rho}{c_2 n^\rho} = \frac{6}{1200} = \frac{1}{200}$$

Notice that if both E_1 and E_2 occur, then the algorithm succeeds: E_1 ensures that at least one "good" vector is in some bucket. And E_2 ensures that the number of false positives is not large enough to prevent the algorithm from finding a "good" vector.

What is the probability that both E_1 and E_2 occur?

$$\Pr(\neg E_1 \vee \neg E_2) \leq \Pr(\neg E_1) + \Pr(\neg E_2) \leq \frac{1}{200} + \frac{1}{200} = 0.01$$

Therefore:

$$\Pr(E_1 \wedge E_2) \geq 1 - 0.01 = 0.99$$

Therefore, the probability that the algorithm succeeds to return a "good" vector is at least 0.99, as required.

Let's now analyze the query time complexity.

We have to compute $h_1(q), \dots, h_L(q)$. Each h_i is composed of k "atomic" hash functions. Calculating each "atomic" hash $h_u(q)$ is $O(d)$ (need to compute $u \cdot q$). Therefore the total cost is $O(L \cdot k \cdot d) = O\left(dn^\rho \log_{1/p_2}(n)\right)$ operations.

In addition, we go over at most $c_2 n^\rho$ vectors and compute the angle between q and each of them. Computing the angle between 2 vectors is $O(d)$, therefore the total cost of angles calculations is: $O(dn^\rho)$.

So the total query time is:

$$O\left(dn^\rho \log_{1/p_2}(n) + dn^\rho\right) = O\left(dn^\rho \log_{1/p_2}(n)\right)$$

Regarding space complexity:

We have L hash tables, each containing n vectors. Actually we can store in the hash tables only the indices of the vectors rather than the vectors themselves, so the space needed for this is $O(nL) = O(n \cdot n^\rho) = O(n^{1+\rho})$.

In addition, we have $L \cdot k$ "atomic" hash functions, each is represented by a vector in R^d. The space needed for this is $O(L \cdot k \cdot d) = O\left(dn^\rho \log_{1/p_2}(n)\right)$.

So the total space is:

$$O\left(n^{1+\rho} + dn^\rho \log_{1/p_2}(n)\right) \leq O(dn^{1+\rho})$$

What can we say about ρ?

$$\rho = \frac{\ln\left(\frac{1}{p_1}\right)}{\ln\left(\frac{1}{p_2}\right)} = \frac{\ln(p_1)}{\ln(p_2)} = \frac{\ln\left(1 - \frac{\theta_1}{\pi}\right)}{\ln\left(1 - \frac{(1+\varepsilon)\theta_1}{\pi}\right)} \approx \frac{1}{1+\varepsilon}$$

Therefore, we get that space complexity is:

$$O\left(dn^{1+\frac{1}{1+\varepsilon}}\right)$$

And time complexity is:

$$O\left(dn^{\frac{1}{1+\varepsilon}} \log_{1/p_2}(n)\right)$$

In class we neglected the logarithmic factor. If we do the same here, we get:

$$O\left(dn^{\frac{1}{1+\varepsilon}}\right)$$

Question 3

Let $x \in R^d$.

Let $\varepsilon, \delta > 0$.

Let $= \frac{2}{\varepsilon^2 \delta}$.

We saw in class that if we pick k hash functions $h_1, \dots, h_k: \{1, \dots, d\} \to \{-1,1\}$ from a 4-wise independent hash family, and define a matrix $M \in R^{k \times d}$ by $M_{ij} = h_i(j)$, then we have:

$$\Pr\left(\left|\frac{1}{k}\|Mx\|^2 - \|x\|^2\right| \geq \varepsilon\|x\|^2\right) \leq \delta$$

In our case $d = n^2$ and we have n vectors $x_1, \dots, x_n \in R^d$.

Let's choose $= \frac{1}{n}$, then we get $k = \frac{2n}{\varepsilon^2}$.

Define $f: R^d \to R^k$ by: $f(x) = \frac{1}{\sqrt{k}} Mx$.

Denote $y_i = f(x_i)$.

Let $i \in \{1, \dots, n\}$. Then we have:

$$\Pr(|\|y_i\|^2 - \|x_i\|^2| \geq \varepsilon\|x_i\|^2) \leq \frac{1}{n}$$

Therefore:

$$\Pr(|\|y_i\|^2 - \|x_i\|^2| < \varepsilon\|x_i\|^2) \geq 1 - \frac{1}{n}$$

If we want to "succeed" with all n vectors, we get:

$$\Pr(\forall i \;\; |\|y_i\|^2 - \|x_i\|^2| < \varepsilon\|x_i\|^2) \geq \left(1 - \frac{1}{n}\right)^n$$

The value $\left(1 - \frac{1}{n}\right)^n$ approaches $\frac{1}{e}$ as n grows, but it is less than $\frac{1}{e}$ (which is the required bound). After consulting Prof. Kaplan, he approved that it is okay to use the bound $\left(1 - \frac{1}{n}\right)^n$.

So we got that:

$$\Pr(\forall i \;\; (1 - \varepsilon)\|x_i\|^2 \leq \|y_i\|^2 \leq (1 + \varepsilon)\|x_i\|^2) \geq \left(1 - \frac{1}{n}\right)^n \approx \frac{1}{e}$$

As required.

How much space do we need in order to represent the matrix M?

We actually need to represent $k = \frac{2n}{\varepsilon^2}$ hash functions.

As we saw in class, each hash function is of the form

$$2\big((a_3x^3 + a_2x^2 + a_1x + a_0)\ mod\ T\ mod\ 2\big) - 1$$

Where T is a prime number between d and $2d$ and $a_3, \dots, a_0 \in \{0,1,\dots,T-1\}$.

Since each hash function can be represented by 4 numbers, we can represent the matrix M by $O(k) = O\left(\frac{2n}{\varepsilon^2}\right)$ numbers.

Notice that each of these numbers (and also T) is smaller than $2d = 2n^2$, hence each number can be represented by $O(\log n)$ bits. I assume we are supposed to neglect this logarithmic factor.

Question 4

Part a

First, note that if $u, v \in R^d$ are unit vectors, then:

$$u \cdot v = \|u\| \cdot \|v\| \cdot \cos\big(\theta(u,v)\big) = 1 \cdot 1 \cdot \cos\big(\theta(u,v)\big) = \cos\big(\theta(u,v)\big)$$

Therefore:

$$(1) \qquad (u \cdot v)^2 = \cos^2\big(\theta(u,v)\big) \quad \forall u, v \in S^{d-1}$$

Now, let $c \geq 1$.

Let $i, j \in \{1, \dots, n\}$.

Denote $e_1 = (1,0,0,\dots,0)^T \in R^d$.

Let $A \in R^{d\times d}$ be a rotation matrix such that $Ax_i = e_1$.

Denote $w = Ax_j$.

Since A is a rotation matrix, it preserves angles, i.e.:

$$\theta(x_i, x_j) = \theta(e_1, w)$$

$$\Rightarrow \ \cos^2\left(\theta(x_i, x_j)\right) = \cos^2\big(\theta(e_1, w)\big)$$

Since $e_1, w \in S^{d-1}$, we get by (1) that:

$$\cos^2\left(\theta(x_i, x_j)\right) = (e_1 \cdot w)^2$$

On the other hand:

$$e_1 \cdot w = w_1$$

Where w_1 is the first coordinate of w.

Therefore:

$$(2) \quad \cos^2\left(\theta(x_i, x_j)\right) = w_1^2$$

Note that w is actually a random unit vector drawn uniformly from the unit sphere.

Let $t \in (1, d)$. Using the inequality we saw in class, we have that:

$$\Pr\left(w_1^2 > \frac{t}{d}\right) < e^{-\frac{t(d-1)}{2d}}$$

(in class we used $t = 1 + \varepsilon$).

Substituting (2) we get:

$$\Pr\left(\cos^2\left(\theta\left(x_i, x_j\right)\right) > \frac{t}{d}\right) < e^{-\frac{t(d-1)}{2d}} \leq [for\ d \geq 2] \leq e^{-\frac{t \cdot d/2}{2d}} = e^{-\frac{t}{4}}$$

This is true for a single pair i, j.

If we consider all pairs, we get (using the union bound):

$$(3) \quad \Pr\left(\exists \mathrm{i,j} \quad \cos^2\left(\theta\left(x_i, x_j\right)\right) > \frac{t}{d}\right) < \binom{n}{2} e^{-\frac{t}{4}} < \frac{n^2}{2} e^{-\frac{t}{4}}$$

We want that:

$$\begin{aligned}
&\frac{n^2}{2} e^{-\frac{t}{4}} \leq \frac{1}{n^c} \\
&\Rightarrow \quad e^{-\frac{t}{4}} \leq 2n^{-(c+2)} \\
&\Rightarrow \quad -\frac{t}{4} \leq \log 2 - (c+2)\log n \\
&\Rightarrow \quad t \geq -4\log 2 + 4(c+2)\log n
\end{aligned}$$

Choose $t = 4(c+2)\log n$. Then we have:

$$\Pr\left(\exists \mathrm{i,j} \quad \cos^2\left(\theta\left(x_i, x_j\right)\right) > \frac{4(c+2)\log n}{d}\right) \leq \frac{1}{n^c}$$

Denote $c' = 4(c+2)$. Then:

$$\Pr\left(\forall \mathrm{i,j} \quad \cos^2\left(\theta\left(x_i, x_j\right)\right) \leq \frac{c' \log n}{d}\right) \geq 1 - \frac{1}{n^c}$$

As required.

<u>Part b</u>

Let c be a large number.

By part a, we get that with probability at least $1 - \frac{1}{n^c}$ we have that:

$$\forall \mathrm{i,j} \quad \cos^2\left(\theta\left(x_i, x_j\right)\right) \leq \frac{4(c+2)\log n}{d}$$

$\frac{4(c+2)\log n}{d}$ is almost 0 because $d \gg \log n$.

Therefore $\cos^2\left(\theta\left(x_i, x_j\right)\right) = \left(x_i \cdot x_j\right)^2$ is almost 0 for all i, j.

Therefore $\left(x_i \cdot x_j\right)$ is very close to 0 for all i, j.

This means that with high probability $1 - \frac{1}{n^c}$ we have that x_i, x_j are almost orthogonal for all i, j.