Foundations of Data Mining - Homework 3

Idan Shabat, 203511597

January 1, 2018

This homework was done with the collaboration of Mor Geva, 200831618.

1 Question 1

For the first equality, $||A - A_k||_2 = \sigma_{k+1}(A)$, we need to show that for every $x \in \mathbb{R}^d$ with $||x|| = 1$: $||(A - A_k)x|| \leq \sigma_{k+1}(A)$, and that there is such x with $||(A - A_k)x|| = \sigma_{k+1}(A)$. Let x be a unit vector in $\mathbb{R}^d$, and we write:

$$x = \alpha_1 v_1 + \alpha_2 v_2 + ... + \alpha_k v_k + ... + \alpha_r v_r + ... + \alpha_d v_d$$

where $v_1, ..., v_r$ are the right-singular vectors of A, r is the rank of A and $\{v_i\}_{i=1}^d$ is an orthonormal basis. Notice that since x is a unit vector and $\{v_i\}_{i=1}^d$ is an orthonormal basis, we now that $1 = ||x||^2 = \alpha_1^2 + ... + \alpha_d^2$. Also by orthonormality of $\{v_i\}_{i=1}^d$, for every $i = 1, ..., k$:

$$Av_i = U\Sigma V^T v_i = U\Sigma e_i = \sigma_i u_i$$
$$A_k v_i = U_k \Sigma_k V_k^T v_i = U_k \Sigma_k e_i = \sigma_i u_i$$

For every $i = k+1, ..., r$:

$$Av_i = U\Sigma V^T v_i = U\Sigma e_i = \sigma_i u_i$$
$$A_k v_i = U_k \Sigma_k V_k^T v_i = U_k \Sigma_k 0 = 0$$

And for every $i = r+1, ..., d$:

$$Av_i = U\Sigma V^T v_i = U\Sigma 0 = 0$$
$$A_k v_i = U_k \Sigma_k V_k^T v_i = U_k \Sigma_k 0 = 0$$

Therefore, we get that:

$$||(A - A_k)x||^2 = ||\alpha_1(A - A_k)v_1 + ... + \alpha_k(A - A_k)v_k + ... + \alpha_r(A - A_k)v_r + ... + \alpha_d(A - A_k)v_d||^2 =$$
$$= ||\alpha_1(\sigma_1 u_1 - \sigma_1 u_1) + ... + \alpha_{k+1}(\sigma_{k+1}u_{k+1} - 0) + ... + \alpha_r(\sigma_r u_r - 0) + \alpha_{r+1}(0 - 0) + \alpha_d(0 - 0)||^2 =$$
$$= ||\alpha_{k+1}\sigma_{k+1}u_{k+1} + ... + \alpha_r \sigma_r u_r||^2 = \alpha_{k+1}^2\sigma_{k+1}^2 + ... + \alpha_r^2\sigma_r^2 \leq \sigma_{k+1}^2(\alpha_{k+1}^2 + ... + \alpha_r^2) \leq \sigma_{k+1}^2(\alpha_{k+1}^2 + ... + \alpha_d^2) = \sigma_{k+1}^2$$

The last line is true since $\{u_i\}_{i=1}^r$ is an orthonormal set and $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r$ (we saw these two results in class). So finally we got:

$$||(A - A_k)x|| \leq \sigma_{k+1}$$

On the other hand, $(A - A_k)v_{k+1} = \sigma_{k+1}u_{k+1}$, therefore: $||(A - A_k)v_{k+1}|| = ||\sigma_{k+1}u_{k+1}|| = \sigma_{k+1}$.

For proving $\sigma_{k+1} = \min_B ||A - B||_2$, we need to show that for every B with rank k: $\sigma_{k+1} \leq ||A - B||_2$, i.e. for every such B there is a vector $x \in \mathbb{R}^d$ with $||x|| = 1$ and $\sigma_{k+1} \leq ||(A - B)x||$.

Notice that $dim(Span\{v_1, ..., v_{k+1}\}) = k + 1$, and $dim(Ker(B)) = d - rank(B) = d - k$. Therefore, there must be a non-zero vector in the intersection:

$$x \in Span\{v_1, ..., v_{k+1}\} \cap Ker(B)$$

We can assume that $||x|| = 1$ (otherwise we can use $x' = \frac{x}{||x||}$). Since $x \in Span\{v_1, ..., v_{k+1}\}$, we can write:

$$x = \beta_1 v_1 + ... + \beta_{k+1} v_{k+1}$$

and now:

$$||(A-B)x||^2 = ||Ax - Bx||^2 = ||Ax||^2 = ||\beta_1 A v_1 + ...\beta_{k+1} A v_{k+1}||^2 =$$
$$= ||\beta_1 \sigma_1 u_1 + ...\beta_{k+1}\sigma_{k+1} u_{k+1}||^2 = \beta_1^2\sigma_1^2 + ... + \beta_{k+1}^2\sigma_{k+1}^2 \geq \sigma_{k+1}^2(\beta_1^2 + ... + \beta_{k+1}^2) =$$
$$= \sigma_{k+1}^2 ||x||^2 = \sigma_{k+1}^2$$

So we got that $||(A-B)x|| \geq \sigma_{k+1}$.

2 Question 2

(a) Assume that the SVD of the matrix A is $A = U\Sigma V^T$, and let x be some vector in $\mathbb{R}^d$. We complete the set $\{v_i\}_{i=1}^r$ (of the right-singular vectors of A) into an orthonormal basis of $\mathbb{R}^d$: $\{v_i\}_{i=1}^d$. Now, express x as a linear combination:

$$x = \langle v_1, x\rangle v_1 + ... + \langle v_d, x\rangle v_d$$

We also complete the set $\{u_i\}_{i=1}^r$ (of the left-singular vectors of A) into an orthonormal basis of $\mathbb{R}^n$: $\{u_i\}_{i=1}^n$. Expressing b as a linear combination of $\{u_i\}_{i=1}^n$:

$$b = \langle u_1, b\rangle u_1 + ... + \langle u_n, b\rangle u_n$$

By noticing that $Av_i = \sigma_i u_i$ for every $i = 1, ..., r$ (by definition), and that $Av_i = 0$ for every $i = r+1, ..., d$ (since $r = rank(A)$), we get:

$$||Ax-b||^2 = ||\sum_{i=1}^{d}\langle v_i, x\rangle Av_i - b||^2 = ||\sum_{i=1}^{r}\langle v_i, x\rangle \sigma_i u_i - b||^2 = ||\sum_{i=1}^{r}\langle v_i, x\rangle \sigma_i u_i - \sum_{i=1}^{n}\langle u_i, b\rangle u_i||^2 =$$

$$= ||\sum_{i=1}^{r}(\langle v_i, x\rangle\sigma_i - \langle u_i, b\rangle)u_i - \sum_{i=r+1}^{n}\langle u_i, b\rangle u_i||^2 = \sum_{i=1}^{r}(\langle v_i, x\rangle\sigma_i - \langle u_i, b\rangle)^2 + \sum_{i=r+1}^{n}\langle u_i, b\rangle^2$$

The minimum of the last expression is obtained when for all $i = 1, ..., r$: $\langle v_i, x\rangle\sigma_i - \langle u_i, b\rangle = 0$, i.e. $\langle v_i, x\rangle = \frac{1}{\sigma_i}\langle u_i, b\rangle$. So, the set of the vectors $x \in \mathbb{R}^d$ that minimize $||Ax - b||$ is:

$$X = \{\sum_{i=1}^{r}\frac{1}{\sigma_i}\langle u_i, b\rangle v_i + \sum_{i=r+1}^{n}\alpha_i v_i \mid \alpha_i \in \mathbb{R}\} = V\Sigma^{-1}U^T b + Span\{v_1, ..., v_r\}^{\perp}$$

(note that by definition $\sigma_i = ||Av_i|| \geq 0$, and since $r = rank(A)$, $\sigma_1, ..., \sigma_r$ are all non-zero). For example, the vector $x_0 = V\Sigma^{-1}U^T b \in \mathbb{R}^d$ minimizes $||Ax - b||$.

(b) As we saw in (a), the minimum value of $||Ax - b||^2$ is achieved by x_0, and its value is:

$$||Ax_0 - b||^2 = ||U\Sigma V^T V\Sigma^{-1}U^T b - b||^2 = ||U\Sigma\Sigma^{-1}U^T b - b||^2 = ||UU^T b - b||^2 =$$

$$||U\begin{pmatrix}\langle u_1, b\rangle \\ \vdots \\ \langle u_r, b\rangle\end{pmatrix} - b||^2 = ||(\langle u_1, b\rangle u_1 + ... + \langle u_r, b\rangle u_r) - (\langle u_1, b\rangle u_1 + ... + \langle u_n, b\rangle u_n)||^2 =$$

$$= ||\langle u_{r+1}, b\rangle u_{r+1} + ... + \langle u_n, b\rangle u_n||^2 = \sum_{i=r+1}^{n}\langle u_i, b\rangle^2$$

$$||Ax_0 - b|| = \sqrt{\sum_{i=r+1}^{n}\langle u_i, b\rangle^2} = ||Proj(b, \mathcal{U}^{\perp})||$$

where $\mathcal{U} = Span\{u_1, ..., u_r\}$, and generally $Proj(w, H)$ is the projection of the vector w on the space H.

3 Question 3

Notations:

Let L, L' be two lists of names, such that L' is different from L by one name: Say, L' was obtained from L by replacing one appearance of the name k with an appearance of the name l. Let $[M] = \{1, ..., M\}$ denote the set of all names that appear either in L or in L'.

We denote by $n, n' \in \mathbb{N}^M$ the frequency vectors of L, L' respectively, i.e.

$$n_i = \#\text{ of appearances of } i \text{ in } L$$
$$n'_i = \#\text{ of appearances of } i \text{ in } L'$$

The algorithm from the question will be denoted as A:

$$A(n) = \text{argmax}_i(n_i + N_i)$$
$$A(n') = \text{argmax}_i(n'_i + N'_i)$$

where $\{N_i\}_{i=1}^M, \{N'_i\}_{i=1}^M$ are independent variables with the same distribution $Lap(\frac{1}{\varepsilon})$.

Claim: For proving that A satisfies differential privacy, it suffices to show that for every name $j \in [M]$:

$$P(A(n) = j) \leq e^{\varepsilon} P(A(n') = j)$$

Proof: For every $S \subseteq [M]$:

$$P(A(n) \in S) = \sum_{j \in S} P(A(n) = j) \leq \sum_{j \in S} e^{\varepsilon} P(A(n') = j) = e^{\varepsilon} \sum_{j \in S} P(A(n') = j) = e^{\varepsilon} P(A(n') \in S)$$

We now prove the following lemma about Laplace distribution:

Lemma: For every two independent variables $N, N' \sim Lap(\frac{1}{\varepsilon})$ and for every $t \in \mathbb{R}$:

$$\frac{P(N \leq t)}{P(N' \leq t+1)} \leq e^{\varepsilon}\ , \quad \frac{P(N \leq t+1)}{P(N' \leq t)} \leq e^{\varepsilon}$$

Proof: Define the functions:

$$\varphi(t) = e^{\varepsilon} P(N' \leq t+1) - P(N \leq t)$$
$$\psi(t) = e^{\varepsilon} P(N' \leq t) - P(N \leq t+1)$$

and it's enough to show that $\varphi(t), \psi(t) \geq 0$ for every t.

We know from the Laplace distribution properties that:

$$\frac{d}{dx} P(N \leq x) = f_N(x) = \frac{\varepsilon}{2} e^{-\varepsilon|x|}$$

(the same holds for N').

In the rest of this question we will use the triangle inequality: $|a+b| \leq |a| + |b|$. We got:

$$\varphi'(t) = e^{\varepsilon} \frac{\varepsilon}{2} e^{-\varepsilon|t+1|} - \frac{\varepsilon}{2} e^{-\varepsilon|t|} = \frac{\varepsilon}{2}(e^{-\varepsilon(|t+1|-1)} - e^{-\varepsilon|t|}) \geq \frac{\varepsilon}{2}(e^{-\varepsilon(|t|+1-1)} - e^{-\varepsilon|t|}) =$$
$$= \frac{\varepsilon}{2}(e^{-\varepsilon|t|} - e^{-\varepsilon|t|}) = 0$$

So φ is monotonically increasing, and since

$$\lim_{t \to -\infty} \varphi(t) = e^{\varepsilon} \lim_{t \to -\infty} P(N' \leq t+1) - \lim_{t \to -\infty} P(N \leq t) = e^{\varepsilon} 0 - 0 = 0$$

we get that $\varphi(t) \geq 0$ for every t.

Similarly:

$$\psi'(t) = e^{\varepsilon} \frac{\varepsilon}{2} e^{-\varepsilon|t|} - \frac{\varepsilon}{2} e^{-\varepsilon|t+1|} = \frac{\varepsilon}{2}(e^{-\varepsilon(|t|-1)} - e^{-\varepsilon|t+1|}) = \frac{\varepsilon}{2}(e^{-\varepsilon(|t+1-1|-1)} - e^{-\varepsilon|t+1|}) \geq$$

$$\geq \frac{\varepsilon}{2}(e^{-\varepsilon(|t+1|)} - e^{-\varepsilon|t+1|}) = 0$$

And:

$$\lim_{t\to-\infty}\psi(t) = e^{\varepsilon}\lim_{t\to-\infty}P(N' \leq t) - \lim_{t\to-\infty}P(N \leq t+1) = e^{\varepsilon}0 - 0 = 0$$

Therefore also $\psi(t) \geq 0$ for every t.

Now, recall the notations L, L', n, n', k, l. In our case, $n_k = n'_k + 1$ and $n_l = n'_l - 1$. Take some name $j \in [M] \setminus \{k, l\}$. For every $x \in \mathbb{R}$:

$$P(A(n) = j|N_j = x) = P(\forall_{i\neq j} n_i + N_i \leq n_j + x) = P(\forall_{i\neq j} N_i \leq n_j + x - n_i) =$$

$$= \prod_{i\neq j} P(N_i \leq n_j + x - n_i) =$$

$$= P(N_k \leq n_j + x - n_k)P(N_l \leq n_j + x - n_l)\prod_{i\neq j,k,l} P(N_i \leq n_j + x - n_i) =$$

$$= P(N_k \leq n'_j + x - n'_k - 1)P(N_i \leq n'_j + x - n'_l + 1)\prod_{i\neq j,k,l} P(N_i \leq n'_j + x - n'_i) =$$

$$= P(N'_k \leq n'_j + x - n'_k - 1)P(N'_i \leq n'_j + x - n'_l + 1)\prod_{i\neq j,k,l} P(N'_i \leq n'_j + x - n'_i) \leq$$

$$\leq e^{\varepsilon}P(N'_k \leq n'_j + x - n'_k)\cdot e^{\varepsilon}P(N'_i \leq n'_j + x - n'_l)\prod_{i\neq j,k,l} P(N'_i \leq n'_j + x - n'_i) =$$

$$= e^{2\varepsilon}P(N'_k \leq n'_j + x - n'_k)P(N'_l \leq n'_j + x - n'_l)\prod_{i\neq j,k,l} P(N'_i \leq n'_j + x - n'_i) =$$

$$= e^{2\varepsilon}\prod_{i\neq j} P(N'_i \leq n'_j + x - n'_i) = e^{2\varepsilon}P(\forall_{i\neq j} N'_i \leq n'_j + x - n'_i) =$$

$$= e^{2\varepsilon}P(\forall_{i\neq j} n'_i + N'_i \leq n'_j + x) = e^{2\varepsilon}P(A(n') = j|N'_j = x)$$

We used the lemma and the fact that $\{N_i\}_{i=1}^M, \{N'_i\}_{i=1}^M$ are i.i.d.
Consider the following general formula (for an event D and a random variable X):

$$P(D) = \int_{-\infty}^{\infty} P(D|X = x)f_X(x)dx$$

Take $D = (A(n) = j)$ and $X = N_j$ for some name $j \neq k, l$:

$$P(A(n) = j) = \int_{-\infty}^{\infty} P(A(n) = j|N_j = x)f_{N_j}(x)dx \leq \int_{-\infty}^{\infty} e^{2\varepsilon}P(A(n') = j|N'_j = x)f_{N_j}(x)dx =$$

$$= e^{2\varepsilon}\int_{-\infty}^{\infty} P(A(n') = j|N'_j = x)f_{N'_j}(x)dx = e^{2\varepsilon}P(A(n') = j)$$

For $j = k$:

$$P(A(n) = k|N_k = x - 1) = \prod_{i\neq k} P(N_i \leq n_k + x - 1 - n_i) =$$

$$= P(N_l \leq n_k + x - 1 - n_l)\prod_{i\neq k,l} P(N_i \leq n_k + x - 1 - n_i) = P(N'_l \leq n'_k + 1 + x - 1 - n'_l - 1)\prod_{i\neq k,l} P(N'_i \leq n'_k + 1 + x - 1 - n'_i) =$$

$$= P(N'_l \leq n'_k + x - n'_l - 1)\prod_{i\neq k,l} P(N'_i \leq n'_k + x - n'_i) \leq e^{\varepsilon}P(N'_l \leq n'_k + x - n'_l)\prod_{i\neq k,l} P(N'_i \leq n'_k + x - n'_i) =$$

$$= e^{\varepsilon}\prod_{i\neq k} P(N'_i \leq n'_k + x - n'_i) = e^{\varepsilon}P(A(n') = k|N'_k = x)$$

So:

$$P(A(n)=k)=\int_{-\infty}^{\infty}P(A(n)=k|N_k=x)f_{N_k}(x)dx=\int_{-\infty}^{\infty}P(A(n)=k|N_k=x-1)f_{N_k}(x-1)dx=$$

$$=\int_{-\infty}^{\infty}P(A(n)=k|N_k=x-1)\frac{\varepsilon}{2}e^{-\varepsilon|x-1|}dx=\int_{-\infty}^{\infty}P(A(n)=k|N_k=x-1)\frac{\varepsilon}{2}e^{-\varepsilon(|x-1|+1-1)}dx\leq$$

$$\leq\int_{-\infty}^{\infty}P(A(n)=k|N_k=x-1)\frac{\varepsilon}{2}e^{-\varepsilon(|x|-1)}dx=e^{\varepsilon}\int_{-\infty}^{\infty}P(A(n)=k|N_k=x-1)f_{N'_k}(x)\leq$$

$$\leq e^{\varepsilon}\int_{-\infty}^{\infty}e^{\varepsilon}P(A(n')=k|N'_k=x)f_{N'_k}(x)=e^{2\varepsilon}P(A(n')=k)$$

For $j=l$, with a symmetric proof, we get that: $P(A(n)=l)\leq e^{2\varepsilon}P(A(n')=l)$

Concluding all the results, we saw that for every two lists of names L, L' that differ in one name, with frequency vectors $n, n'\in\mathbb{N}^M$:

$$\forall_{j\in[M]}\ P(A(n)=j)\leq e^{2\varepsilon}P(A(n')=j)$$

I.e. A is $(2\varepsilon,0)$-differentially private.

4 Question 4

(a) We prove that the sensitivity of this utility is 1.

For the proof, we use some facts about symmetric difference:

- For every two (finite) sets A, B:

$$|A \triangle B| = |(A \cup B) \setminus (A \cap B)| \leq |A \cup B| \leq |A| + |B|$$

- Associativity: For every three sets:

$$(A \triangle B) \triangle C = A \triangle (B \triangle C)$$

Let S, S' be two databases with distance 1, i.e. $S' = (S \setminus \{s\})$, where $s \in \{1, ..., T\}$ and $s \in S \setminus S'$. Notice that actually: $S' = S \triangle \{s\}$ and $S = S' \triangle \{s\}$.

Let r be some real number between 1 and T. By definition, $u(S, r) = -|S \triangle R|$ where $R \subseteq \{1, ..., T\}$ s.t. $|S \triangle R|$ is minimal. Similarly, $u(S', r) = -|S' \triangle R'|$ where $R' \subseteq \{1, ..., T\}$ s.t. $|S' \triangle R'|$ is minimal. Particularly, it's implied that:

$$|S \triangle R| \leq |S \triangle R'|, \;\; |S' \triangle R'| \leq |S' \triangle R|$$

Now:

$$u(S, r) - u(S', r) = -|S \triangle R| + |S' \triangle R'| \leq -|S \triangle R| + |S' \triangle R| = -|S \triangle R| + |(S \triangle \{s\}) \triangle R| =$$

$$= -|S \triangle R| + |S \triangle R \triangle \{s\}| \leq -|S \triangle R| + |S \triangle R| + 1 = 1$$

$$u(S, r) - u(S', r) = -|S \triangle R| + |S' \triangle R'| \geq -|S \triangle R'| + |S' \triangle R'| = -|(S' \triangle \{s\}) \triangle R'| + |S' \triangle R'| =$$

$$= -|S' \triangle R' \triangle \{s\}| + |S' \triangle R'| \geq -|S' \triangle R'| - 1 + |S' \triangle R'| = -1$$

So we have: $|u(S, r) - u(S', r)| \leq 1$

On the other hand, if we take $S = \{1, 2\}$, $S' = \{1\}, r = 1$, so the only $R \subseteq \{1, ..., T\}$ that exists with median equals to 1 is $R = \{1\} = S'$. Then:

$$u(S, r) = |S \triangle R| = |\{1, 2\} \triangle \{1\}| = |\{2\}| = 1$$
$$u(S', r) = |S' \triangle R| = |\{1\} \triangle \{1\}| = |\emptyset| = 0$$

So: $|u(S, r) - u(S', r)| = |1 - 0| = 1$

Concluding the results, we got:

$$\max_{r \in (1,T)} \max_{||S - S'|| = 1} |u(S, r) - u(S', r)| = 1$$

(b) Consider the following example:

Take $S = \{1\}$. The median of S is 1, so $u(S, 1) = -|S \triangle S| = 0$. For every $r \in \{2, ..., T\}$, if $R \subseteq \{1, ..., T\}$ has median r s.t. $|S \triangle R|$ minimzed, then $1 \leq |S \triangle R| \leq 2$ (since $|S \triangle R| = 0$ iff $R = S$ and for $R = \{r\}$ we get exactly $|S \triangle R| = 2$).

Therefore, the **un-normalized** probability that the exponential mechanism M returns 1 is $e^{\frac{\varepsilon}{2} u(S,1)} = 1$ and for $r \neq 1$:

$$e^{\frac{\varepsilon}{2} u(S,r)} \geq e^{\frac{\varepsilon}{2}(-2)} = e^{-\varepsilon}$$

and

$$e^{\frac{\varepsilon}{2}u(S,r)} \leq e^{\frac{\varepsilon}{2}(-1)} = e^{-\frac{\varepsilon}{2}}$$

The normalization constant is:

$$\sum_{r=1}^{T} e^{\frac{\varepsilon}{2}u(S,r)} = 1 + \sum_{r\neq 1} e^{\frac{\varepsilon}{2}u(S,r)} \leq 1 + (T-1)e^{-\frac{\varepsilon}{2}}$$

So we finally got:

$$E[|M(S)-1|] = \sum_{r=1}^{T} P(M(S)=r)|r-1| = \sum_{r=2}^{T} P(M(S)=r)(r-1) \geq$$

$$\geq \sum_{r=2}^{T} \frac{e^{-\varepsilon}}{1+(T-1)e^{-\frac{\varepsilon}{2}}}(r-1) = \frac{e^{-\varepsilon}}{1+(T-1)e^{-\frac{\varepsilon}{2}}} \cdot \frac{T(T-1)}{2} \geq \frac{1}{e^{\varepsilon}} \cdot \frac{1}{1+(T-1)} \cdot \frac{T(T-1)}{2} = \frac{1}{e^{\varepsilon}} \cdot \frac{T-1}{2}$$

We see here that if $e^{\varepsilon} = o(T)$, then when T is large, the expectation of the distance $|M(S)-1|$ is large. Since we want a small e^{ε}, maybe even a constant, it is very likely that the expected distance will be large.

(c) We will show that no $\varepsilon > 0$ can make this mechanism $(\varepsilon, 0)-$differential private. Assuming that there is such an ε, we know that for every S, S' with distance 1, and for every $r \in \{1, ..., T\}$:

$$P(M(S)=r) \leq e^{\varepsilon} P(M(S')=r)$$

(denoting the mechanism by M). We take for example $S = \{1,2\}$, $S' = \{2\}$ and $r = 1$. By the definition of M, no matter what is the result of $\lfloor \frac{1}{2} + Lap(\frac{1}{\varepsilon}) \rfloor$, $M(S') = 2$ anyway (because of the rounding). Therefore $P(M(S')=1) = 0$. For S:

$$P(M(S)=1) = P(\lfloor 1 + Lap(\frac{1}{\varepsilon}) \rfloor \leq 1) = P(1 + Lap(\frac{1}{\varepsilon}) < 1) = P(Lap(\frac{1}{\varepsilon}) < 0) = \frac{1}{2}$$

. But then we get:

$$\frac{1}{2} = P(M(S)=1) \leq e^{\varepsilon} P(M(S')=1) = e^{\varepsilon} \cdot 0 = 0$$

In contradiction.

(d) We now define a εN-Locally differential private mechanism M for approximating the median of a set S, where $N = O(\ln(T))$ (will be defined exactly later):

The mechanism M asks each individual $j \in \{1, ..., T\}$, N times, the same question: "Does $j \in S$?". The answer of j to the ith question will be $X_j^i \in \{0,1\}$ ($X_j^i = 1$ means "yes") and will be computed by **randomized response**:

If $x \in \{0,1\}$ is the actual answer to the question, then

$$X_j^i = \begin{cases} x & \text{with probability } \frac{e^{\varepsilon}}{1+e^{\varepsilon}} \\ 1-x & \text{with probability } \frac{1}{1+e^{\varepsilon}} \end{cases}$$

Then, the mechanism computes the set $R_0 = \{j \mid \sum_{i=1}^{N} X_j^i > \frac{N}{2}\}$ (i.e. the set of all j such that most of the questions to the individual j were answered "yes"). The approximated median of S will be the median of R_0.

By definition, this mechanism is εN-Locally differential private, since it accesses each individual N times through a ε-randomizer.

Let Z_j be a random variable that indicating whether $j \in R_0$ or not:

$$Z_j = \begin{cases} 1 & j \in R_0 \\ 0 & j \notin R_0 \end{cases}$$

Then, for every $j \notin S$:

$$E[Z_j] = P\Big(\sum_{i=1}^{N} X_j^i > \frac{N}{2}\Big) = P\Big(\sum_{i=1}^{N} X_j^i - E[\sum_{i=1}^{N} X_j^i] > \frac{N}{2} - E[\sum_{i=1}^{N} X_j^i]\Big) =$$

$$= P\Big(\sum_{i=1}^{N} X_j^i - \frac{1}{1+e^{\varepsilon}}N > \frac{N}{2} - \frac{1}{1+e^{\varepsilon}}N\Big) \leq e^{-\frac{2N^2(\frac{1}{2}-\frac{1}{1+e^{\varepsilon}})^2}{N}} = e^{-\delta N}$$

for $\delta = 2(\frac{1}{2} - \frac{1}{1+e^{\varepsilon}})^2 > 0$.
For every $j \in S$:

$$E[1 - Z_j] = P\Big(\sum_{i=1}^{N} X_j^i \leq \frac{N}{2}\Big) = P\Big(\sum_{i=1}^{N} X_j^i - E[\sum_{i=1}^{N} X_j^i] \leq \frac{N}{2} - E[\sum_{i=1}^{N} X_j^i]\Big) =$$

$$= P\Big(\sum_{i=1}^{N} X_j^i - \frac{e^{\varepsilon}}{1+e^{\varepsilon}}N \leq \frac{N}{2} - \frac{e^{\varepsilon}}{1+e^{\varepsilon}}N\Big) = P\Big(\sum_{i=1}^{N} X_j^i - \frac{e^{\varepsilon}}{1+e^{\varepsilon}}N \leq -(\frac{e^{\varepsilon}}{1+e^{\varepsilon}} - \frac{1}{2})N\Big) \leq$$

$$\leq e^{-\frac{2N^2(\frac{e^{\varepsilon}}{1+e^{\varepsilon}}-\frac{1}{2})^2}{N}} = e^{-\delta N}$$

since $\frac{1}{2} - \frac{1}{1+e^{\varepsilon}} = \frac{e^{\varepsilon}}{1+e^{\varepsilon}} - \frac{1}{2}$. The two inequalities are implied by Hoeffding Bounds.
Now, Notice that:

$$|S \triangle R_0| = \sum_{j \notin S} Z_j + \sum_{j \in S} 1 - Z_j$$

and that if r is the median of R_0, then

$$u(S, r) = -\min_{R \text{ with median } r} |S \triangle R| \geq -|S \triangle R_0|$$

Therefore in expectation:

$$E[u(S, r)] \geq -E[|S \triangle R_0|] = -(\sum_{j \notin S} E[Z_j] + \sum_{j \in S} E[1 - Z_j]) \geq$$

$$\geq -(\sum_{j \notin S} e^{-\delta N} + \sum_{j \in S} e^{-\delta N}) = -Te^{-\delta N}$$

If we take $N = \frac{1}{\delta}(5 + \ln(T)) = O(\ln(T))$, we get:

$$E[u(S, r)] \geq -Te^{-\delta N} = -Te^{-5-\ln(T)} = -T \cdot \frac{1}{e^5} \cdot \frac{1}{T} = -\frac{1}{e^5} \geq -0.007$$

Notice that since the values of u are always non-positive **integers**, if $u(S, r) > -1$, it means that $u(S, r) = 0$, i.e. r is the actual median of S. By Markov inequality, the probability that it **doesn't** happen is:

$$P(u(S, r) < -1) = P(-u(S, r) > 1) \leq E[-u(S, r)] \leq 0.007$$

Concluding everything, we have a εN-Locally differential private mechanism, that returns the right median of a database S with probability ≥ 0.993.

5 Question 5

(a) For every finite set of points in the plane there is a rectangle which contain them. Given the set P, let R_0 be such a rectangle, with left side l_0, right side r_0, upper side u_0 and lower side d_0.

We prove that for every node v in the tree there is a rectangle $r(v)$ s.t. $P \cap r(v)$ is exactly the points in the leaves of the subtree of v.

Proof: We use induction on the distance of v from the root (will be denoted as $\delta(v)$). For the root, clearly R_0 is a good choice, since $R_0 \cap P = P$ and all of P's points are the leaves of the tree.

If v isn't the root, let u be its parent. Consider 4 cases:

- $\delta(u)$ is even, v is u's left child:
  Then $r(v)$ will be the same as $r(u)$, but with $l(u)$ as its lower side. Indeed, $p \in r(v) \cap P$ if and only if $p \in r(u) \cap P$ and p is **above** $l(u)$, and by induction, it happens iff p is a leaf of the subtree of u and it's stored in the **left** subtree of u, i.e. p is a leaf in the subtree of v.
- $\delta(u)$ is even, v is u's right child:
  Then $r(v)$ will be the same as $r(u)$, but with $l(u)$ as its upper side. Indeed, $p \in r(v) \cap P$ if and only if $p \in r(u) \cap P$ and p is **below** $l(u)$, and by induction, it happens iff p is a leaf of the subtree of u and it's stored in the **right** subtree of u, i.e. p is a leaf in the subtree of v.
- $\delta(u)$ is odd, v is u's left child:
  Then $r(v)$ will be the same as $r(u)$, but with $l(u)$ as its right side. Indeed, $p \in r(v) \cap P$ if and only if $p \in r(u) \cap P$ and p is **left** to $l(u)$, and by induction, it happens iff p is a leaf of the subtree of u and it's stored in the **left** subtree of u, i.e. p is a leaf in the subtree of v.
- $\delta(u)$ is odd, v is u's right child:
  Then $r(v)$ will be the same as $r(u)$, but with $l(u)$ as its left side. Indeed, $p \in r(v) \cap P$ if and only if $p \in r(u) \cap P$ and p is **right** to $l(u)$, and by induction, it happens iff p is a leaf of the subtree of u and it's stored in the **right** subtree of u, i.e. p is a leaf in the subtree of v.

Following the inductive proof, we can exactly specify the sides of $r(v)$: Let $v_1, v_2, ..., v_m$ is a path from v ($= v_1$) to the root ($=v_m$) in the tree. Let $down(v)$ be the first v_i s.t. v_{i-1} is its **left** child and $\delta(v_i)$ is **even**; $up(v)$ be the first v_i s.t. v_{i-1} is its **right** child and $\delta(v_i)$ is **even**; $right(v)$ be the first v_i s.t. v_{i-1} is its **left** child and $\delta(v_i)$ is **odd**; and $left(v)$ be the first v_i s.t. v_{i-1} is its **right** child and $\delta(v_i)$ is **odd**.

Then the sides of $r(v)$ are exactly: $l(down(v))$, $l(up(v))$, $l(right(v))$, $l(left(v))$.

(b) Let R be a query rectangle with $|P \cap R| = k$. For every node v such that $r(v) \subseteq R$, we conclude all of the leaves in the subtree of v in the output. Since the size of a subtree is proportional to the number of its leaves, the sum of the sizes of these subtrees is proportional to the size of the output $= k$. This is also the time needed to deal with this kind of nodes.

The search also visit nodes v such that $r(v)$ intersect R but not contained in it. That means that a side of $r(v)$ is intersected with a side of R, and we call the nodes of this kind **grey nodes**.

Let l be a horizontal side of R. For any node v with left child $v.left$ and right child $v.right$, if $\delta(v)$ is even, then $l(v)$ is horizontal, and therefore only one of the ranges $r(v.left)$, $r(v.right)$ can intersect l. If $\delta(v)$ is odd, then $l(v)$ is vertical, and it could be that both $r(v.left)$, $r(v.right)$ intersect l. Anyway, the number of nodes v s.t. $r(v)$ intersects l in some level of the 2-d tree, is twice their number in the previous-previous level. So, we can bound their total number in the tree by:

$$1 + 2 + 2 + 4 + 4 + 8 + ... < 2 \sum_{i=1}^{\log(n)} 2^{\lfloor \frac{i}{2} \rfloor} \leq 2 \sum_{i=1}^{\log(n)} \sqrt{2}^i = 2\frac{\sqrt{2}^{\log(n)} - 1}{\sqrt{2} - 1} = O(\sqrt{n})$$

By a symmetric proof, we can prove that the number of nodes v in the tree s.t. $r(v)$ intersect some vertical side of R is $O(\sqrt{n})$. Finally, the number of grey nodes visited in the algorithm is exactly the number of nodes v s.t. $r(v)$ intersects some side of R, which is at most $4O(\sqrt{n}) = O(\sqrt{n})$. The total sum of the nodes visited in the algorithm, which is proportional to its running time (since in each one we perform a constant number of steps), is:

$$O(\sqrt{n}) + O(k) = O(\sqrt{n} + k)$$

(c) A k-d Tree is a binary tree that stores a set $P \subseteq \mathbb{R}^k$ in its leaves. Each node v with distance $\delta(v)$ from the root corresponds to a hyperplane $h(v)$: If $i - 1 = \delta(v) \mod k$, then $h(v) = Span\{e_i\}^{\perp} + c_v$ for some point $c_v \in \mathbb{R}^k$, such that half of the points p in v's subtree satisfy $\langle p - c_v, e_i \rangle \geq 0$, and the other half satisfy $\langle p - c_v, e_i \rangle < 0$ ($e_i = (0, ..., 0, 1, 0, ..., 0)^T \in \mathbb{R}^k$, where the 1 is in the i'th coordinate). The first half are stored in v's left subtree, and the second half are stored in v's right subtree.

The generalized algorithm for finding the points $P \cap R$ for some query box R is as follows (here we also denote v's left and right child as $v.left$ and $v.right$ respectively):

Start at the root and for each internal node v which is visited perform the following: If one of the ranges $r(v.left)$ or $r(v.right)$ is contained in R, output all of the leaves in the subtree of $v.left$ or $v.right$, respectively. If one of the ranges only intersect R but not contained in it, continue and visit the corresponding node $v.left$ or $v.right$.

(d) We first define an algorithm that finds an initial guess for the nearest neighbor, with a regular search of x in the k-d tree (in general, we search for x in a subtree with root r and distance δ from the original root):

Algorithm 1 Find Initial Guess

```
1: procedure SEARCH(x,r,δ)
2:     if r is a leaf then return r
3:     i ← (δ mod k) + 1
4:     if ⟨x − c_r, e_i⟩ ≥ 0 then SEARCH(x,r.left,δ + 1)
5:     else SEARCH(x,r.right,δ + 1)
```

Now, the k-d tree nearest neighbor algorithm finds an initial guess p with distance d from x. Then it climbs up the tree with a iteration node v. In every step, it checks whether the other subtree has a better guess for the nearest neighbor - which can happen only if the distance between x and the range of this subtree is smaller than d. If so, it replaces p with the better guess (and d with the better distance).

Algorithm 2 Find Nearest Neighbor

```
1: procedure NN(x,r,δ)
2:     if r is a leaf then return r
3:     p ← SEARCH(x,r,δ)
4:     d ← ||p − x||
5:     v ← p
6:     while v.parent is not null do
7:         if dist(x, h(v.parent)) < d then
8:             newSuspect ← NN(x,OTHERCHILD(v.parent,v),δ + 1)
9:             if d > ||newSuspect − x|| then
10:                p ← newSuspect
11:                d ← ||newSuspect − x||
12:        v ← v.parent
       return p
```

Note that $dist(x, h(u))$ is the distance between x and the hyperplane $h(u) = Span\{e_i\}^{\perp} + c_v$ for some $i \in \{1, ..., k\}$. We also used the simple function OTHERCHILD(u,v), which for a node u and its child v returns u's other child:

$$\text{OTHERCHILD}(u, v) = \begin{cases} u.left & u.right = v \\ u.right & u.left = v \end{cases}$$

Correctness: Let n be the actual nearest neighbor. For every point p, the distance $d = ||p - x||$ is at least $||n - x||$, and therefore n is inside the sphere of radius d around x. It means that every range $r(v)$ in the k-d tree which contain n is intersected with this sphere, i.e. for all ancestors u of n: $dist(x, h(u)) < d$. Since the path from the initial guess to the root has to pass through some ancestor u of n, a recursive call to NN will be performed on the subtree of u. By induction (on the level of the tree) we can now prove that this call will find n, and therefore the whole algorithm will find it (since the guess can only be improved each step).

Time Complexity: In the worst case, the time needed for this algorithm in $O(kn)$, as we show in the following example:

Suppose that all of the n points in the original set P are located on the unit sphere in $\mathbb{R}^k$: $\{x \in \mathbb{R}^k \mid ||x|| = 1\}$, and the point x is $0 \in \mathbb{R}^k$. Then, in each step of the algorithm $d = 1$, because there is no point closer to x than 1. Then every hyperpalne (which is parallel to some axis) that passes through any point in P intersects the current sphere, i.e. $dist(x, h(v)) \leq d$ for every node v in the k-d tree. It means that the whole tree will be traversed. Notice that in any internal node we perform a calculation like $dist(x, h(v))$, which takes $O(k)$ time, so concluding all, we have $O(kn)$ time for the whole algorithm.