COMS 4771 Machine Learning (2022 Fall)
Problem Set #4

Xinhao Li - xl2778@columbia.edu

2022/12/04

Problem 1

(i)

(a)

Based on the Randomized Response (RR) procedure given in the question and the probability of having the results of heads by flipping an unbiased coin, we have:

$$P[RR(x) = \text{yes}] = k \cdot P[\text{sensitive\_question(x)} = \text{yes}] + (1-k) \cdot \frac{1}{2} \tag{1}$$

This the same case when the answer to sensitive question is no.

$$P[RR(x) = \text{no}] = k \cdot P[\text{sensitive\_question(x)} = \text{no}] + (1-k) \cdot \frac{1}{2} \tag{2}$$

Then for two databases x and y, we can have that x and y have the same (case 1) or different (case 2) answers to the questions. For the first case, considering the requirement of differential privacy, we have:

$$\frac{P[RR(x) = r]}{P[RR(y) = r]} = \frac{k \cdot P[\text{sensitive\_question(x)} = \text{r} \in \{yes, no\}] + (1-k) \cdot \frac{1}{2}}{k \cdot P[\text{sensitive\_question(y)} = \text{r} \in \{yes, no\}] + (1-k) \cdot \frac{1}{2}} = 1 \tag{3}$$

Since $\exp(-\epsilon) \leq 1$ and $\exp(\epsilon) \geq 1$, we have:

$$\exp(-\epsilon) \leq \frac{P[RR(x) = r]}{P[RR(y) = r]} \leq \exp(\epsilon) \tag{4}$$

Next for the second case, databases x and y have different answers. Let's assume that sensitive_question(x)=yes and sensitive_question(y)=no. So we have:

$$P[\text{sensitive\_question(x)} = \text{yes}] = 1 \tag{5}$$

$$P[\text{sensitive\_question(y)} = \text{yes}] = 0 \tag{6}$$

And now, we have:

$$\frac{P[RR(x)=r]}{P[RR(y)=r]} = \frac{k \cdot P[\text{sensitive\_question(x)} = \text{yes}] + (1-k)\cdot\frac{1}{2}}{k \cdot P[\text{sensitive\_question(y)} = \text{yes}] + (1-k)\cdot\frac{1}{2}} = \frac{k+\frac{1}{2}(1-k)}{\frac{1}{2}(1-k)} = \frac{k+1}{1-k}$$

Considering the requirement of differential privacy, we have:

$$\begin{aligned} \exp(-\epsilon) \leq \frac{k+1}{1-k} &\leq \exp(\epsilon) \\ \Rightarrow \frac{e^{-\epsilon}-1}{e^{-\epsilon}+1} \leq k &\leq \frac{e^{\epsilon}-1}{e^{\epsilon}+1} \\ \Rightarrow 0 \leq k \leq \frac{e^{\epsilon}-1}{e^{\epsilon}+1}, &\text{since } k \in [0,1] \end{aligned} \tag{7}$$

This holds true when sensitive_question(x)=no and sensitive_question(y)=yes since:

$$\frac{P[RR(x)=r]}{P[RR(y)=r]} = \frac{k \cdot P[\text{sensitive\_question(x)} = \text{no}] + (1-k)\cdot\frac{1}{2}}{k \cdot P[\text{sensitive\_question(y)} = \text{no}] + (1-k)\cdot\frac{1}{2}} = \frac{k+\frac{1}{2}(1-k)}{\frac{1}{2}(1-k)} = \frac{1-k}{k+1}$$

$$\begin{aligned} \exp(-\epsilon) \leq \frac{1-k}{k+1} &\leq \exp(\epsilon) \\ \Rightarrow 0 \leq k \leq \frac{e^{\epsilon}-1}{e^{\epsilon}+1} & \end{aligned} \tag{8}$$

In conclusion, if we want to make RR to be $(\epsilon, 0)$ differentially private, we should have $0 \leq k \leq \frac{e^{\epsilon}-1}{e^{\epsilon}+1}$.

(b)

Again, based on the Randomized Response (RR) procedure given in the question and the probability of having the results of heads by flipping an unbiased coin, we have:

$$\begin{aligned} P[\text{sensitive\_question(x)} = RR(x)] &= k \cdot P[\text{sensitive\_question(x)} = RR(x) | m < k] + (1-k)\cdot\frac{1}{2} \\ &= 1 \cdot k + \frac{1}{2}(1-k) \\ &= \frac{1}{2}(k+1) \end{aligned} \tag{9}$$

$P[\text{sensitive\_question(x)} = RR(x) | m < k] = 1$ because when m < k, the answer to RR(x) is just the answer to sensitive_question(x).

(ii)

(a)

Let p_x denote the PDF of $M_L(x, f, \epsilon)$, and let p_y denote the PDF of $M_L(y, f, \epsilon)$. We compare the two at some arbitrary point $z \in \mathbb{R}^k$.

$$\begin{aligned}
\frac{p_x(z)}{p_y(z)} &= \prod_{i=1}^{k} \Big(\frac{\exp(-\frac{\epsilon|f(x)_i - z_i|}{\Delta f})}{\exp(-\frac{\epsilon|f(y)_i - z_i|}{\Delta f})} \Big) \\
&= \prod_{i=1}^{k} \exp\Big(\frac{\epsilon(|f(y)_i - z_i| - |f(x)_i - z_i|}{\Delta f} \Big) \\
&\leq \prod_{i=1}^{k} \exp\Big(\frac{\epsilon(|f(x)_i - |f(y)_i|}{\Delta f} \Big) \\
&= \exp\Big(\frac{\epsilon(||f(x)_i - |f(y)_i||_1}{\Delta f} \Big) \\
&\leq \exp(\epsilon), \text{ since } ||x - y||_1 \leq 1
\end{aligned} \tag{10}$$

By symmetry, $\frac{p_x(z)}{p_y(z)} \geq \exp(-\epsilon)$. In conclusion, this procedure can achieve $(\epsilon, 0)$ differential privacy.

(b)

As the hint given in the question, let's first show that the distribution has a tail bound that guarantees that if Z is drawn from $L(y; b)$ then $P[|Z| \geq bt] = \exp(-t)$. First of all, the equation that is asked us to prove can be reduced as:

$$P\Big[|Y| \geq (\frac{\Delta f}{\epsilon}) \ln(\frac{1}{p})\Big] = p \tag{11}$$

We know that Y is a random variable drawn from $L(y; \frac{\Delta f}{\epsilon})$, where:

$$L(y; b) = \frac{1}{2b} \exp(-\frac{|y|}{b}) \tag{12}$$

Then we will have

$$\begin{aligned}
P[|Y| \geq bt] &= 2P[Y \geq bt] \\
&= \int_{bt}^{\infty} \frac{1}{b} e^{-\frac{y}{b}} dy \\
&= (1 - e^{-\frac{y}{b}})|_{bt}^{\infty} \\
&= e^{-t},
\end{aligned} \tag{13}$$

since L is an even function and $P[|Y| = y] = P[|Y| = -y]$.
Now the left part of equation 11 becomes:

$$P\Big[|Y| \geq (\frac{\Delta f}{\epsilon}) \ln(\frac{1}{p})\Big] = \exp\left(-\ln \frac{1}{p}\right) = p, \forall p \in (0, 1] \tag{14}$$

Problem 2

(i)

The equation that we want to optimize is:

$$\sum_{i=1}^{n} \min_{j \in \{1,....,k\}} ||x_i - c_j||^2 \tag{15}$$

We know the equation 15 should be ≥ 0. Then the best case we can achieve is when equation 15 is equal to 0. However, if we think of minimizing equation 15 over both k and c, we will have $k = n$ and $c_i = x_i$ for i from 1 to n. This will equation 15 reduces to:

$$\sum_{i=1}^{n} \min_{j \in \{1,....,k\}} ||x_i - c_j||^2 = \sum_{i=1}^{n} ||x_i - x_i||^2 = 0 \tag{16}$$

This is a bad idea because every data point will have its own cluster and it will not reveal any useful information or pattern for the data.

(ii)

Let's say that we have 4 data points and they are: $x_i = i$, $\forall i = 1, 2, 3, 4$. Two clusters are supposed to be there, $k = 2$. Suppose that Lloyds method initializes cluster centers at $c_1 = 1$ and $c_2 = 3$. Based on these two centers, the first round valid clusters C_i with respect to center c_i are: $C_1 = \{1\}$ and $C_2 = \{2,\ 3,\ 4\}$. Then, $\sum_{i=1}^{n} \min_{j \in \{1,....,k\}} ||x_i - c_j||^2 = 2$ For the next iteration, the centers are already stable because the average of points in each cluster are exactly at the current cluster center. Hence, the algorithm will stop. However, this is clearly a sub-optimal setting since we can have the new centers at $c_1 = 1.5$ and $c_2 = 3.5$. Consequently, the the clusters should be $C_1 = \{1,\ 2\}$ and $C_2 = \{3,\ 4\}$ since $\sum_{i=1}^{n} \min_{j \in \{1,....,k\}} ||x_i - c_j||^2 = 1$. Therefore, Lloyds method will not be stable to improve the solution.

(iii)

(a)

By implementing the LIoyd's method for k-means algorithm, the results are shown in figure 1, 2, and 3. We can clearly see that running normal LIoyd's method for k-means algorithm does not work for the cases that each pair of clusters is not linearly separable.

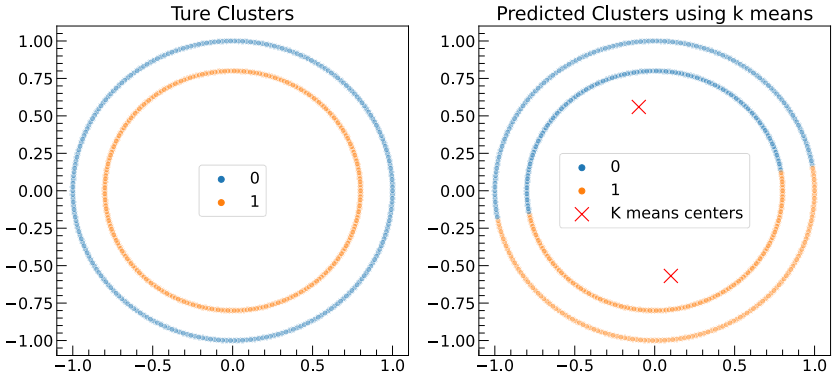

Figure 1: True and predicted clusters using k-means algorithm running on two circles

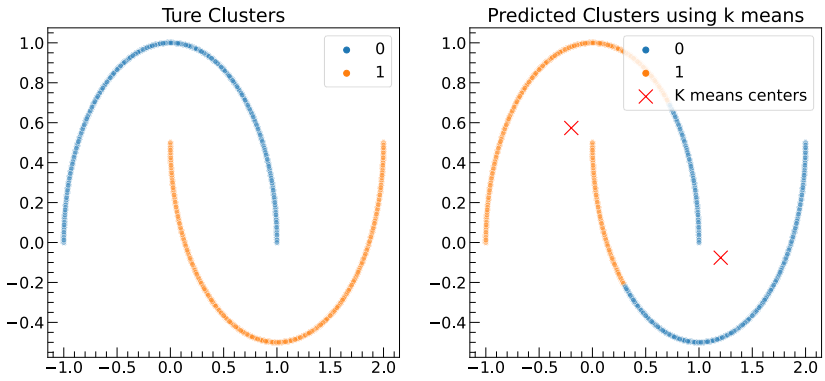

Figure 2: True and predicted clusters using k-means algorithm running on two moons

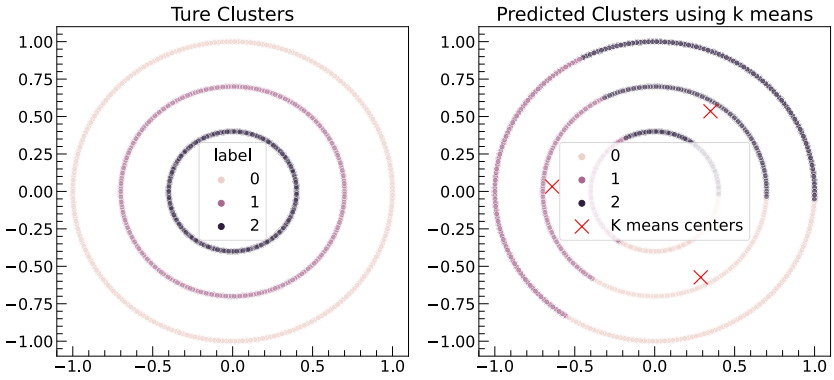

Figure 3: True and predicted clusters using k-means algorithm running on three circles

Figure 4: Demo

(b)

For $k = 2$, the points lying on the cluster boundary should have the same distance to c_1 and c_2. Then we will have:

$$\begin{aligned}
\sum_{i=1}^{n} ||x_i - c_1||^2 &= \sum_{i=1}^{n} ||x_i - c_2||^2 \\
||x - c_1||^2 &= ||x - c_2||^2 \\
||x||^2 + ||c_1||^2 - 2 < x, c_1 > &= ||x||^2 + ||c_2||^2 - 2 < x, c_2 > \\
2 < x, c_2 > -2 < x, c_1 > &= ||c_2||^2 - ||c_1||^2 \\
< x, c_2 - c_1 > &= \frac{||c_2||^2 - ||c_1||^2}{2}
\end{aligned} \tag{17}$$

The linear equation is given by:

$$x^T w = b \tag{18}$$

In our case,

$$\begin{aligned}
w &= c_2 - c_1 \\
b &= \frac{||c_2||^2 - ||c_1||^2}{2}
\end{aligned} \tag{19}$$

Therefore, the cluster boundary induced by minimizing the k-means objective is necessarily linear.

(c)

We know the $n \times n$ matrix is defined as: $L = D - W$. Then we can express matrix L as:

$$\begin{bmatrix}
\sum_j w_{1j} - w_{11} & -w_{12} & -w_{13} & ... & -w_{1n} \\
-w_{21} & \sum_j w_{2j} - w_{22} & -w_{23} & ... & -w_{2n} \\
. & . & . & . & \\
. & . & . & . & \\
. & . & . & . & \\
-w_{n1} & -w_{n2} & -w_{n3} & ... & \sum_j w_{nj} - w_{nn}
\end{bmatrix} \tag{20}$$

Then $f^T(D-W)f$ can be expressed by:

$$
\begin{aligned}
f^T(D-W)f &= \begin{bmatrix} f_1(\sum_j w_{1j} - w_{11}) - f_2 w_{21} - \ldots - f_n w_{n1}, \\ -f_1 w_{12} + f_2(\sum_j w_{2j} - w_{22}) - \ldots - f_n w_{n2}, \\ \ldots, \\ -f_1 w_{1n} - f_2 w_{2n} - \ldots + f_n(\sum_j w_{nj} - w_{nn}) \end{bmatrix}^T f \\
&= \begin{bmatrix} f_1 \sum_j w_{1j} - f_1 w_{11} - f_2 w_{21} - \ldots - f_n w_{n1}, \\ -f_1 w_{12} + f_2 \sum_j w_{2j} - f_2 w_{22} - \ldots - f_n w_{n2}, \\ \ldots, \\ -f_1 w_{1n} - f_2 w_{2n} - \ldots + f_n \sum_j w_{nj} - f_n w_{nn} \end{bmatrix}^T f \\
&= \begin{bmatrix} f_1 \sum_j w_{1j} - \sum_i f_i w_{i1}, \\ f_2 \sum_j w_{2j} - \sum_i f_i w_{i2}, \\ \ldots, \\ f_n \sum_j w_{nj} - \sum_i f_i w_{in}, \end{bmatrix}^T f \\
&= \begin{bmatrix} f_1 \sum_j w_{1j} - \sum_i f_i w_{i1}, \\ f_2 \sum_j w_{2j} - \sum_i f_i w_{i2}, \\ \ldots, \\ f_n \sum_j w_{nj} - \sum_i f_i w_{in}, \end{bmatrix}^T \begin{bmatrix} f_1, f_2, \ldots, f_n \end{bmatrix}^T \\
&= \begin{bmatrix} f_1^2 \sum_j w_{1j} - f_1 \sum_i f_i w_{i1} + \ldots + f_n^2 \sum_j w_{nj} - f_n \sum_i f_i w_{in} \end{bmatrix} \\
&= \sum_i f_i^2 \sum_j w_{1j} - \sum_j f_j \sum_i f_i w_{i1} \\
&= \sum_{ij} f_i^2 w_{ij} - \sum_{ij} f_i f_j w_{ij} \\
&= \frac{1}{2}\Big(\sum_{ij} f_i^2 w_{ij} + \sum_{ij} f_j^2 w_{ij} - 2\sum_{ij} f_i f_j w_{ij}\Big) \\
&= \frac{1}{2}\sum_{ij} w_{ij}\Big(f_i^2 + f_j^2 - 2 f_{ij}\Big) \\
&= \frac{1}{2}\sum_{ij} w_{ij}\Big(f_i - f_j\Big)^2
\end{aligned} \tag{21}
$$

(d)

Based on the definition of w_{ij}, $w_{ij} \in \{0, 1\}$. $(f_i - f_j)^2$ should always be equal or greater than 0. Therefore, we have for any vector $f \in \mathbb{R}^n$:

$$
f^T L f = \frac{1}{2}\sum_{ij} w_{ij}\Big(f_i - f_j\Big)^2 \geq 0 \tag{22}
$$

So, L is positive semi-definite.
To prove that L is symmetric, for all $i \neq j$, based on the expression of L (equation 20) and the definition of w_{ij}, we have:

$$
-w_{ij} = -w_{ji} \rightarrow L_{ij} = L_{ji} \quad \forall i \neq j \tag{23}
$$

Thus, L is also symmetric. Overall, L is a symmetric positive semi-definite matrix.

(e)

Let's assume that v_1, v_2, ..., v_k vertices are in the connected components C_1, C_2, ..., C_k, respectively. Since they are connected components, this means that all vertices in the connected components are linked to each other by paths. We then have the indicator vector $\mathbf{1}_{C_t} = [w_{t1}, w_{t2}, ..., w_{tn}]^T$ for $1 \leq t \leq k$.

$$\mathbf{1}_{C_t}^T L \mathbf{1}_{C_t} = \frac{1}{2} \sum_{ij} w_{ij} \Big(w_{ti} - w_{tj} \Big)^2 \tag{24}$$

Based on equation 24 and the definition of w_{ij}, we can see that w_{ij} can either be 0 or 1 When $w_{ij} = 0$, $\mathbf{1}_{C_t}^T L \mathbf{1}_{C_t} = 0$. When $w_{ij} = 1$, this means v_i and v_j are connected and in the same set. If v_t is also in the set, then w_{ti} and w_{tj} should be both 1 and if v_t is not in the set, then w_{ti} and w_{tj} should be both 0. This makes $\mathbf{1}_{C_t}^T L \mathbf{1}_{C_t} = 0$ again. So overall, we have:

$$\mathbf{1}_{C_t}^T L \mathbf{1}_{C_t} = 0 \tag{25}$$

Finally, to prove $\mathbf{1}_{C_1}$, $\mathbf{1}_{C_2}$, ..., $\mathbf{1}_{C_k}$ are eigenvectors of L with eigenvalue 0, $L\mathbf{1}_{C_t} = 0$. L is symmetric semi-definite, so it has n orthogonal eigenvectors $v_i \in \mathbb{R}^n \forall i = 1, 2, ..., n$, which forms a baisi of $\mathbb{R}^n$. Thus, we can have $x = \sum_i b_i v_i$

$$\begin{aligned} < x, Lx > &= \Big\langle \sum_i b_i v_i, \sum_i b_i A v_i \Big\rangle \\ &= \Big\langle \sum_i b_i v_i, \sum_i \lambda_i b_i v_i \Big\rangle \\ &= \sum_i \sum_j \Big\langle b_i v_i, \lambda_j b_j v_j \Big\rangle \end{aligned} \tag{26}$$

Because when $i \neq j$, $< v_i, v_j >= 0$(orthogonal)

$$\begin{aligned} < x, Lx > &= \sum_i \sum_j \Big\langle b_i v_i, \lambda_j b_j v_j \Big\rangle \\ &= \sum_i \lambda_i b_i^2 ||v_i||^2 \end{aligned} \tag{27}$$

Since $\lambda_i \geq 0$ and $||v_i||^2 > 0$, for all i, if $< x, Lx >= 0$, $\lambda_i b_i = 0$. Therefore, for all x, $Lx = \sum_i d_i L v_i = \sum_i d_i \lambda_i v_i = 0$. In conclusion we have $L\mathbf{1}_{C_t} = 0$, thus $\mathbf{1}_{C_1}$, $\mathbf{1}_{C_2}$, ..., $\mathbf{1}_{C_k}$ are eigenvectors of L with eigenvalue 0,

(f)

By running this flexible version of k-means with 10 nearest neighbors are selected, we can see from figure 8, 9, and 10 that all data points can be clustered accurately. So by applying k-means in the transformed space, quality of the clustering is indeed improved.

Then, I have also did some tests for the effect of number of nearest neighbors selected on the results. When can see from figure 5, 6, and 7, when only 2 nearest neighbors are selected, the results are pretty poor. When the number of nearest neighbors increased from 2 to 20 or even 100 (since it has already been shown that when $r = 10$ gives perfect results), figure 11, 12, 13, 14, 15, and 16 show that as r increases to a very large value, the results are kind of similar as the standard k-means results.

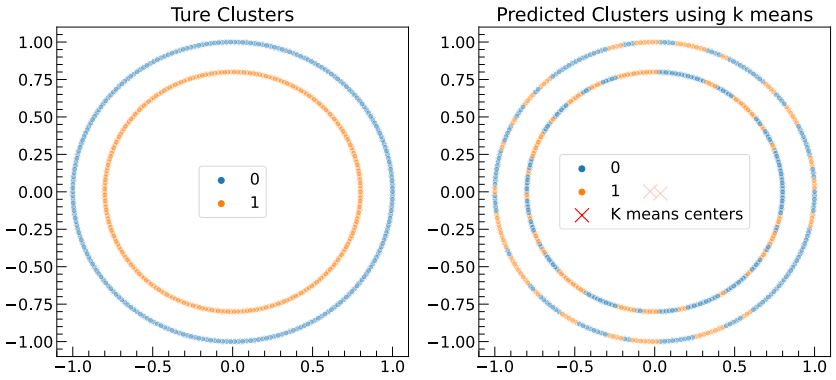

Figure 5: True and predicted clusters using flexible k-means algorithm (r = 2) running on two circles

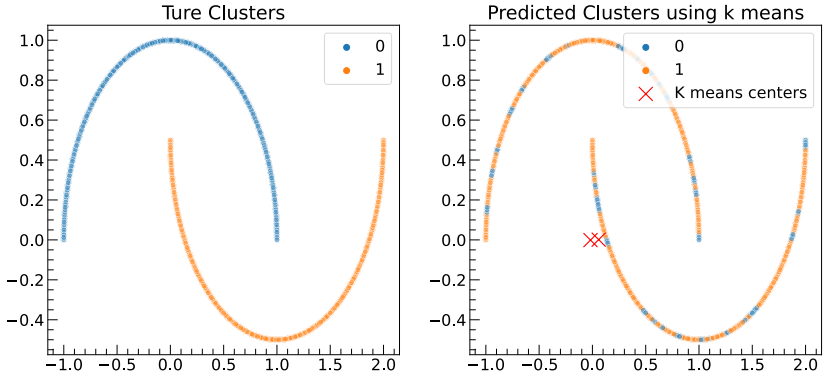

Figure 6: True and predicted clusters using flexible k-means algorithm (r = 2) running on two moons

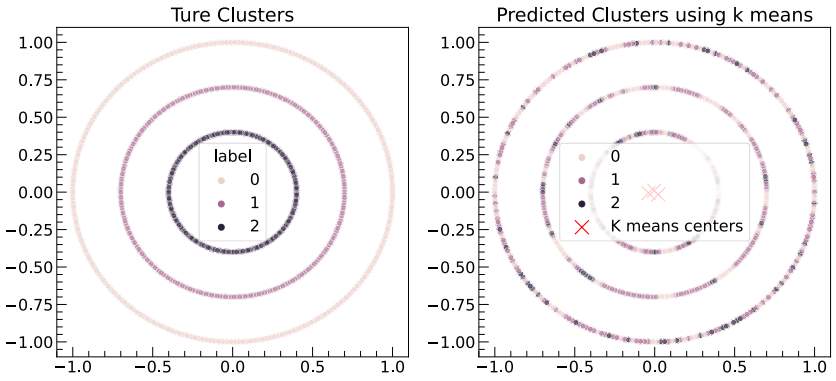

Figure 7: True and predicted clusters using flexible k-means algorithm (r = 2) running on three circles

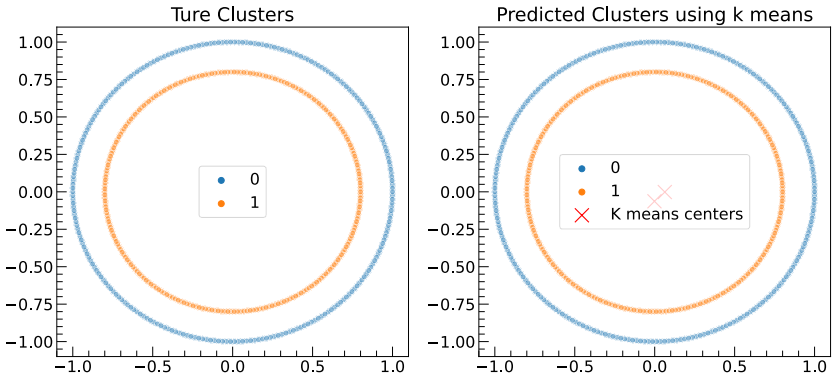

Figure 8: True and predicted clusters using flexible k-means algorithm (r = 10) running on two circles

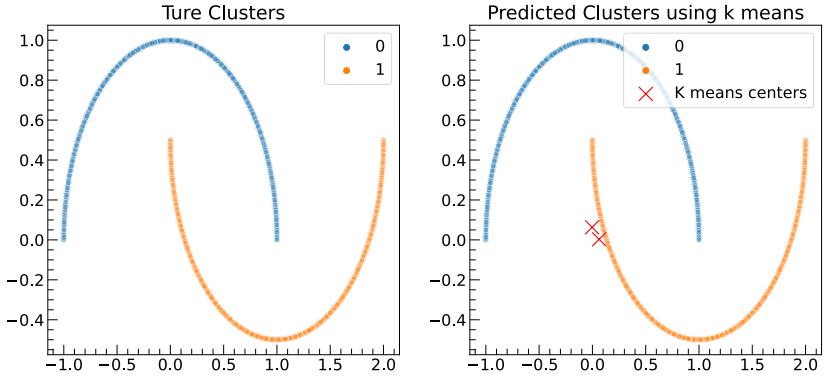

Figure 9: True and predicted clusters using flexible k-means algorithm (r = 10) running on two moons

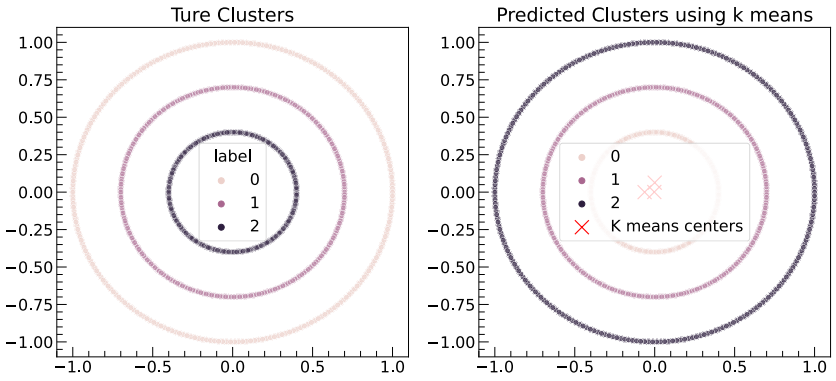

Figure 10: True and predicted clusters using flexible k-means algorithm (r = 10) running on three circles

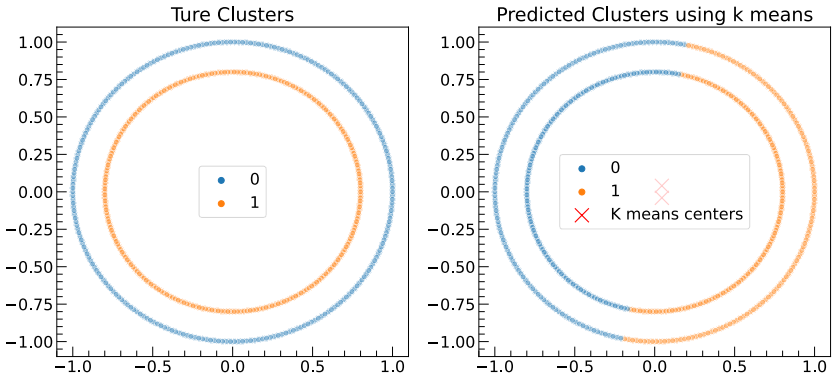

Figure 11: True and predicted clusters using flexible k-means algorithm (r = 20) running on two circles

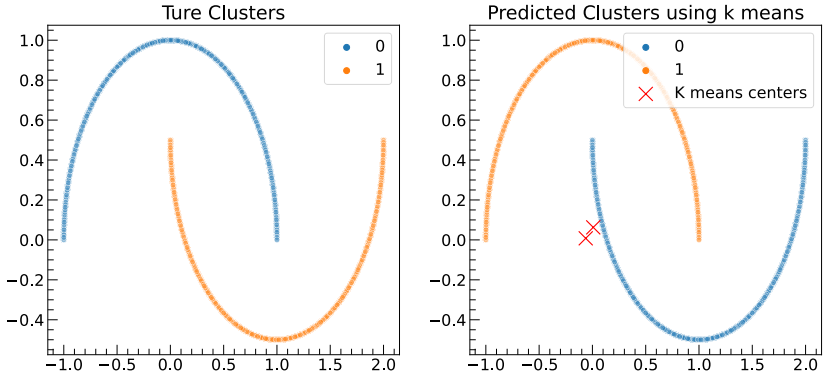

Figure 12: True and predicted clusters using flexible k-means algorithm (r = 20) running on two moons

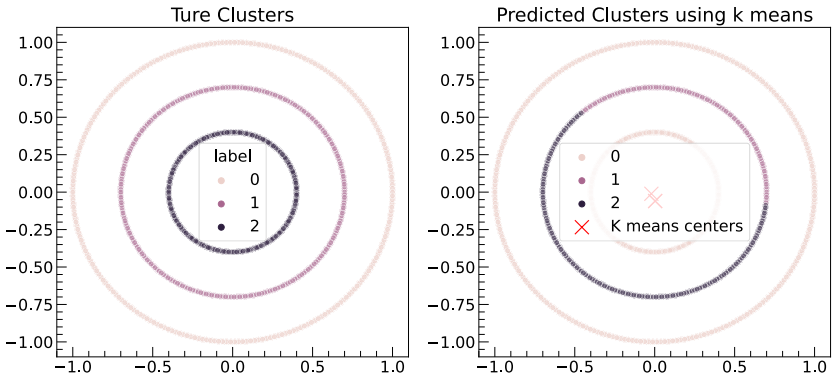

Figure 13: True and predicted clusters using flexible k-means algorithm (r = 20) running on three circles

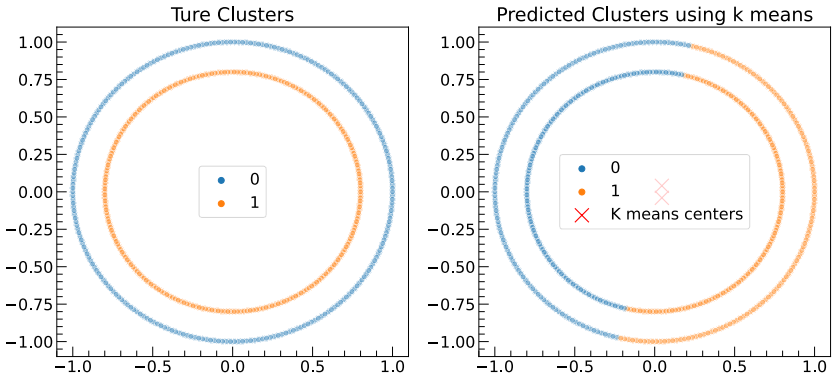

Figure 14: True and predicted clusters using flexible k-means algorithm (r = 100) running on two circles

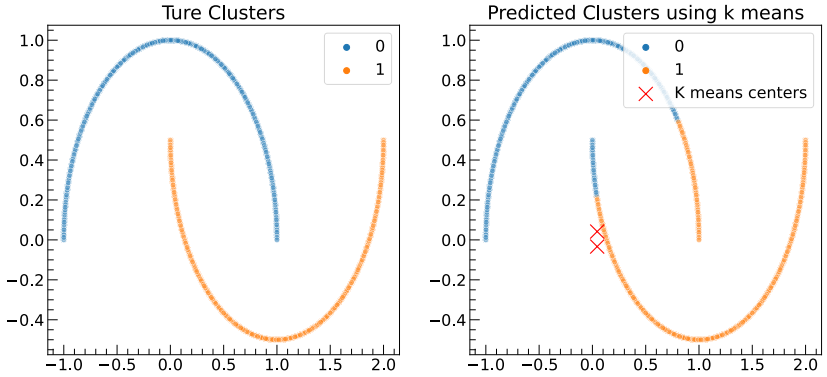

Figure 15: True and predicted clusters using flexible k-means algorithm (r = 100) running on two moons

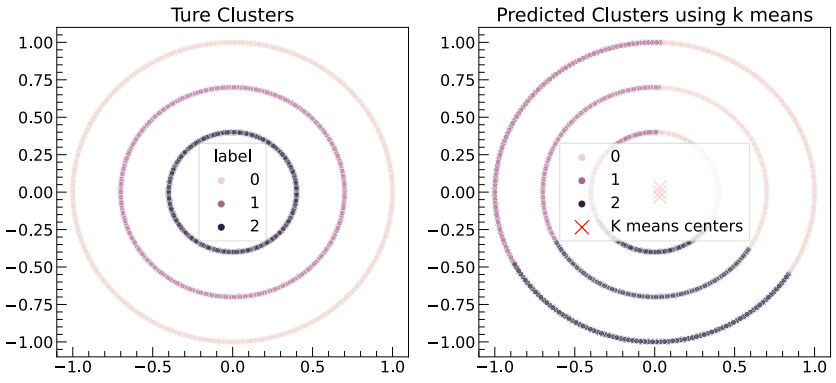

Figure 16: True and predicted clusters using flexible k-means algorithm (r = 100) running on three circles

Problem 3

(i)

For fixed y_i, the derivative of the optimization function is:

$$\begin{aligned}\frac{\partial}{\partial y_i}\sum_{\substack{i,j\\ i\neq j}}(||y_i-y_j||-\pi_{ij})^2 &= 2\sum_{\substack{i,j\\ i\neq j}}(||y_i-y_j||-\pi_{ij})\ \frac{\partial}{\partial y_i}(||y_i-y_j||-\pi_{ij})\\ &= 2\sum_{\substack{i,j\\ i\neq j}}(||y_i-y_j||-\pi_{ij})\ \frac{y_j-y_i}{||y_j-y_i||}\\ &= 2\sum_{\substack{i,j\\ i\neq j}}(1-\frac{\pi_{ij}}{||y_j-y_i||})y_j-y_i \end{aligned} \tag{28}$$

(ii)

The optimization with respect to a fixed y_i is convex.
As been introduced in lecture 4, to prove a function is called convex, iff for any two points x, x' and $\beta\in[0,1]$, it should satisfy:

$$f(\beta x+(1-\beta)x')\leq \beta f(x')+(1-\beta)f(x') \tag{29}$$

Let's define:

$$f(y_j)=\sum_{\substack{i,j\\ i\neq j}}(||y_i-y_j||-\pi_{ij})^2 \tag{30}$$

So for the left hand side of equation 29, we have:

$$\sum_{\substack{i,j\\ i\neq j}}(||y_i-\beta y_j+(1-\beta)y_j'||-\pi_{ij})^2=\sum_{\substack{i,j\\ i\neq j}}(||\beta(y_i-y_j)+(1-\beta)(y_i-y_j')||-\pi_{ij})^2 \tag{31}$$

Because $||x+y||\leq||x||+||y||$, we can have:

$$\begin{aligned}&\sum_{\substack{i,j\\ i\neq j}}(||\beta(y_i-y_j)+(1-\beta)(y_i-y_j')||-\pi_{ij})^2\\ &\leq\sum_{\substack{i,j\\ i\neq j}}(\beta||y_i-y_j||+(1-\beta)||y_i-y_j'||-\pi_{ij})^2\\ &=\sum_{\substack{i,j\\ i\neq j}}(\beta[||y_i-y_j||-\pi_{ij}]+(1-\beta)[||y_i-y_j'||-\pi_{ij}])^2\\ &\leq\beta\sum_{\substack{i,j\\ i\neq j}}([||y_i-y_j||-\pi_{ij}])^2+(1-\beta)\sum_{\substack{i,j\\ i\neq j}}([||y_i-y_j'||-\pi_{ij}])^2 \end{aligned} \tag{32}$$

This proves the equation 29 and means that the optimization problem is a convex problem.

(iii)

The code has been submitted online.

(iv)

First the 3D swill roll and swill roll with hole are shown in figure 17 below. The results using

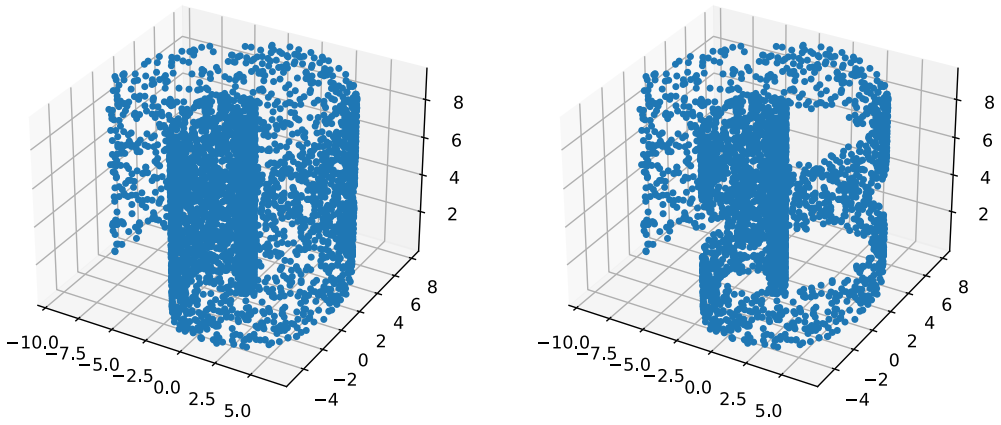

Figure 17: 3D swill roll and swill roll with hole

my own developed non-linear dimensionality reduction are shown in figure 18, 19, 20, and 21 for swiss roll and figure 22, 23, and 24 for swiss roll with hole.
For swiss roll, we can see that when the number of NNs are small, the dimensionality reduction fails. But as the number of NNs increases to 500 and beyond, especially when the number of NNs is chose as 1000, the shape of the swiss roll can be preserved.
For the swiss roll with hole dataset, my non-linear dimensionality reduction algorithm also works. The upper and lower part of the swiss roll with hole have been preserved clearly(figure 24). However, it seems like that the overall geometry of the swill role can be preserved easily, but it struggles a little bit with the hole inside the roll.

We have also compare our results obtained from 2D non-linear embedding with 2D PCA reduction as well as the 2D Isomap reduction. The results using PCA and Isomap from sklearn are shown in figure 25, 26, 27, 28, 29, and 30.
Basically, we can see PCA can always give the shape of the swiss roll. However, for the case that the swill roll has the hole, PCA can not preserve the hole shape inside and can only generate the same results as doing dimensionality reduction on pure swill roll data whatever how many number of nearest neighbors (NN) are used.
For Isomap, we can clearly see that when the number of NN is small, it can not preserve the shape of both swiss roll and swiss roll with hole well. However, once we increase the number of NNs to 500, the perfect results are achieved. For swiss roll, not only the geometric shape in is been preserved, some kind of thickness of the swiss roll is also been preserved. For swill roll with hole, we can also see that the hole inside the swiss roll is also been preserved. But as we considered more and more NNs, Isomap generally yields the same results as PCA.

In conclusion, when the number of NNs selected is small, this non-linear embedding fails. This is because the Euclidean distance between two points can be small in swiss roll or

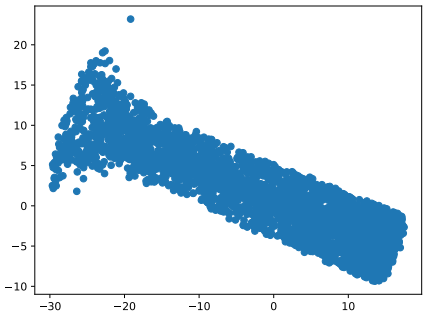

Figure 18: Swiss roll 20 NNs

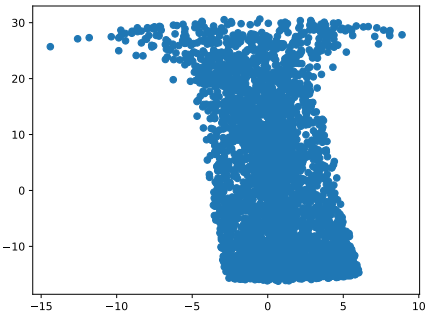

Figure 19: Swiss roll 100 NNs

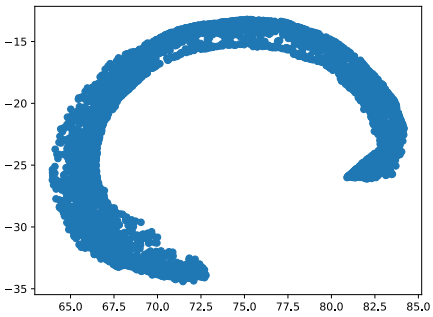

Figure 20: Swiss roll 500 NNs

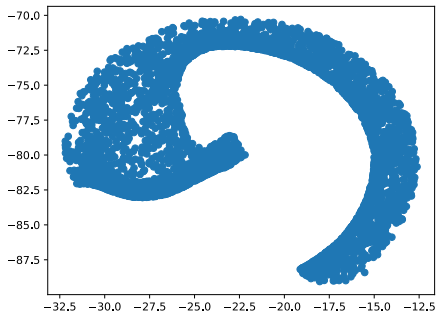

Figure 21: Swiss roll 1000 NNs

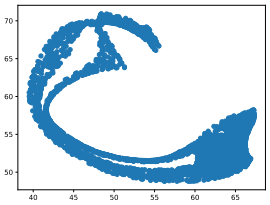

Figure 22: Swiss roll with hole 200 NNs

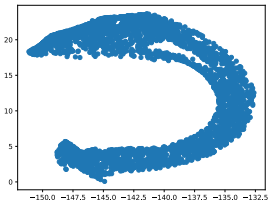

Figure 23: Swiss roll with hole 500 NNs

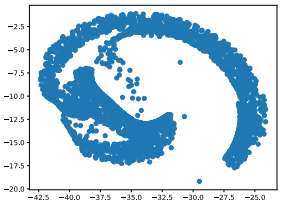

Figure 24: Swiss roll with hole 1000 NNs

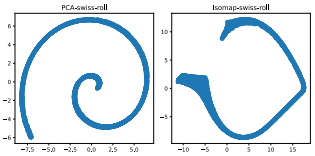

Figure 25: NN=200

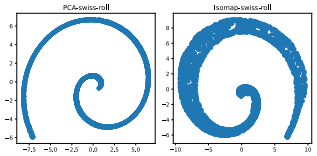

Figure 26: NN=500

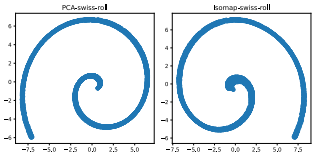

Figure 27: NN=1000

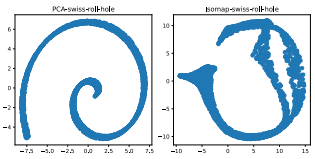

Figure 28: NN=200

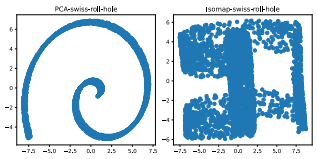

Figure 29: NN=500

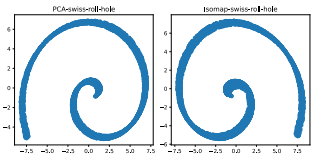

Figure 30: NN=1000

swiss roll with hole, but their distance over the manifold on which the data points reside can be large. This non-linear embedding also works on the swiss roll with hole dataset but the result can definitely been improved because choosing the number of NNs is extremely hard. Because when more points are considered, the hole in the geometry will be neglected, but when less points are considered, the geometry can not be learned. However, when the number of NNs selected is large (more appropriate) for swiss roll dataset, this non-linear embedding succeeds and to some extent can perform better than PCA. But we should also notice that the running time for this algorithm is far more slow than PCA.

Appendix

Code

- k_means.py
- non_linear_dim_reduction.py
- run_k_means.py
- run_non_linear_dim_reduction.py

Useful discussions with

- Yueyue Ma
- Yiwei Wang

References

- Useful info related to Q1 (ii) (a).
- 1-Useful info related to Q2.
- 2-Useful info related to Q2
- 1-Useful info related to Q3.
- 2-Useful info related to Q3.