Math 123 **Problem Set 4** **Due: Monday 4/5 at 11:59 p.m. EST**

Instruction: Read the homework policy. For problems 2, 3 and 4, include copies of your code with your final homework submission. You should submit a PDF copy of the homework and any associated codes on Canvas. Your PDF must be a single file, not multiple images.

1. Consider N data points $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N$ lying in $\mathcal{R}^d$. We apply the nonlinear transform $\phi : \mathcal{R}^d \to \mathcal{R}^M$ and obtain data points in feature space $\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), ..., \phi(\mathbf{x}_N)$. Define the kernel matrix $\boldsymbol{K} \in \mathcal{R}^{N\times N}$ as $\boldsymbol{K}_{i,j} = \kappa_{i,j} = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$.

(a) Prove that $\boldsymbol{K}$ is a symmetric and positive semidefinite matrix.

(b) The projected dataset may not be centered. Let's center each projected data point as $\hat{\phi}(\mathbf{x}_i) = \phi(\mathbf{x}_i) - \frac{1}{N}\sum_{k=1}^{N}\phi(\mathbf{x}_k)$. Define a new kernel matrix $\hat{\boldsymbol{K}} \in \mathcal{R}^{N\times N}$ as $\hat{\boldsymbol{K}}_{i,j} = \hat{\kappa}_{i,j} = \hat{\phi}(\mathbf{x}_i)^T\hat{\phi}(\mathbf{x}_j)$. Prove that $\hat{\boldsymbol{K}} = \boldsymbol{K} - \mathbf{1}\boldsymbol{K} - \boldsymbol{K}\mathbf{1} + \mathbf{1}\boldsymbol{K}\mathbf{1}$ where $\mathbf{1} = \frac{1}{N}\begin{pmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \vdots & \vdots & & \vdots \\ 1 & 1 & \dots & 1 \end{pmatrix}$.

(c) Discuss briefly the implication of the result in (b) in carrying out kernel PCA of the data.

2. In this problem, we use random projections to solve the compressive sensing problem. Load the vector $\mathbf{x} \in \mathcal{R}^{1000}$ and the basis $\mathbf{U} \in \mathcal{R}^{1000\times 1000}$ available in the HW4 folder. The columns of $\mathbf{U}$ are an orthonormal basis for $\mathcal{R}^{1000}$.

(a) Plot the representation (coordinate) of the vector $\mathbf{e}_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$ with respect to the basis $\mathbf{U}$. Is $\mathbf{e}_1$ sparse in the standard basis? Is $\mathbf{e}_1$ sparse with respect to the basis $\mathbf{U}$?

(b) Plot the representation (coordinate) of the vector $\mathbf{x}$ with respect to the basis $\mathbf{U}$. Denote your result by $\mathbf{c}$. What can you say about $\mathbf{c}$?

(c) As you see in (b), $\mathbf{x} \in \mathcal{R}^{1000}$ is 3-sparse with respect to the basis $\mathbf{U}$. In particular, in the representation $\mathbf{x} = \sum_{i=1}^{n} c_i\mathbf{U}_i$ where $\mathbf{U}_i$ is the i-th column of $\mathbf{U}$, all except three entries of the coefficient vector $\mathbf{c}$ are nonzero. To find the sparse coefficient vector $\mathbf{c}$, we use the idea of compressive sensing and consider the following minimization program

$$\min_{\mathbf{c}} \quad ||\mathbf{c}||_0 \quad \text{such that} \quad \mathbf{A}\mathbf{c} = \mathbf{b},$$

where $\mathbf{A} \in \mathcal{R}^{m\times n}$ is a random projection matrix. Set $m = 60$ and $n = 1000$. Construct a random projection matrix as follows: $\mathbf{A} = \frac{1}{\sqrt{m}}\mathbf{Z}$ where $\mathbf{Z}$ is a random matrix (e.g. `rand` or `randn` in MATLAB or Python). Generate $\mathbf{b}$ by applying $\mathbf{A}$ on $\mathbf{c}$ i.e. $\mathbf{b} = \mathbf{A}\mathbf{c}$.

(d) In this part, our goal is to see if we can recover $\mathbf{c}$ given the inputs $\mathbf{A}$ and $\mathbf{b}$ obtained from (c). To solve this, we utilize a compressive sensing algorithm, the *Iterative Thresholding* algorithm, proposed in [1]. Use the MATLAB code `IH.mat` to find an approximate $\hat{\mathbf{c}}$. Plot

[1] Blumensath, Thomas, and Mike E. Davies. "Iterative thresholding for sparse approximations." Journal of Fourier analysis and Applications 14.5-6 (2008): 629-654.

$\hat{\mathbf{c}}$ and $\mathbf{c}$ on the same figure. What can you conclude? Discuss in brief terms.
[**Remark**: If you want to do this in different language other than Python and have issues translating the MATLAB code, feel free to reach out the instructor or TA].

(e) Using the estimated sparse coefficient $\hat{\mathbf{c}}$ in (d), reconstruct the signal using the basis $\mathbf{U}$. Let $\hat{\mathbf{x}}$ be the reconstructed signal. Plot $\hat{\mathbf{x}}$ and $\mathbf{x}$ on the same figure. What can you conclude? Discuss in brief terms.

(f) Briefly discuss the implication of the steps (a)-(e). In particular, comment on the number of measurements required to reconstruct the vector $\mathbf{x}$ that is 3-sparse in the basis $\mathbf{U}$.

3. In this problem, we implement the K-means algorithm and test it on two dimensional datasets.

(a) Implement the K-means algorithm. Your code should take as input a data and number of clusters and should output labels for each data point.

(b) Load the data `gaussians.mat(gaussians.csv)` available in the HW4 folder.

i. Run K-means with K = 2 cluster and 10 replicates. Plot your optimal clustering result i.e. show a scatter plot of the points in different colors corresponding to their K-means label.

ii. Plot the error of the K-means optimization objective as a function of the number of iterations. Is there convergence?

iii. Is the clustering what you expect?

(c) Load the data `ellipses.mat(ellipses.csv)` available in the HW4 folder.

i. Run K-means with K = 2 clusters and 10 replicates. Plot your optimal clustering result i.e. show a scatter plot of the points in different colors corresponding to their K-means label.

ii. Plot the error of the K-means optimization objective as a function of the number of iterations. Is there convergence?

iii. Is the clustering what you expect?

[**Remark**: Replicates here is the number of times the core K-means algorithm is run each time from a different initialization of the centroids. For (a), it is sufficient to write an algorithm for 2 clusters i.e. $K = 2$. The error of the K-means objective is defined as follows

$$E = \sum_{\mathbf{x}_i \in C_1} (\mathbf{x}_i - \mu_1)^2 + \sum_{\mathbf{x}_i \in C_1} (\mathbf{x}_i - \mu_2)^2,$$

where C_1 and C_2 are the two clusters and μ_1 and μ_2 are their respective centers. In brief terms, the error is the sum of two terms, (a) The first term is the sum of squared distances of the points in clusters 1 from the first center, and (b) The second term is the sum of squared distances of the points in clusters 2 from the second center].

4. In this problem, we apply clustering to the latent features obtained from PCA. The dataset we consider is the MNIST dataset which is a database of handwritten digits.

(a) Load the data `mnist_067` available in the HW4 folder. The included data is a subset of the MNIST data consisting of the digits $0, 6$ and 7. Project the data points onto the first two principal components of the data. Run K-means with $K = 3$ clusters and 100 replicates on the latent representations. Plot your optimal clustering result i.e. show a scatter plot of the data points in different colors corresponding to their K-means label.

(b) Recall the results of problem 7 in HW3. Given those results, what could you conclude about your result in (a) ?

[**Remark**: For this problem, you can use an inbuilt PCA function and K-means function from a solver of your choice.]

1

$x_1, x_2, \ldots, x_n \in \mathbb{R}^d \quad \phi: \mathbb{R}^d \to \mathbb{R}^M$

$\Rightarrow \phi(x_1), \phi(x_2), \ldots, \phi(x_n) \in \mathbb{R}^M$

$K_{ij} = \phi(x_i)^T \phi(x_j) \in \mathbb{R}^{n \times n}$

(a) Prove $\underline{K}$ is a positive, semi-definite matrix.

by definition, $\underline{K} = \underline{\Phi}^T \underline{\Phi}$ where $\underline{\Phi} = [\phi(x_1) \mid \cdots \mid \phi(x_n)]$

hence for any $v \in \mathbb{R}^n$,

$\langle v, \underline{K} v \rangle = \langle v, \underline{\Phi}^T \underline{\Phi} v \rangle = \langle \underline{\Phi} v, \underline{\Phi} v \rangle$

$= \|\underline{\Phi} v\|^2 \geq 0$, so $\underline{K}$ is positive semi-definite.

(b) center each $\phi(x_i) \longmapsto \phi(x_i) - \frac{1}{n}\sum_{j=1}^{n} \phi(x_j) = \hat{\phi}(x_i)$

define $\underline{\hat{K}}_{ij} = \hat{\phi}(x_i)^T \hat{\phi}(x_j)$ $\qquad \underline{1} = \begin{pmatrix} 1/n & \cdots & 1/n \\ \vdots & & \vdots \\ 1/n & \cdots & 1/n \end{pmatrix}$

Prove $\underline{\hat{K}} = \underline{K} - \underline{1}\underline{K} - \underline{K}\underline{1} + \underline{1}\underline{K}\underline{1}$

notice:

$\underline{\hat{\Phi}} = \left[\phi(x_1) - \frac{1}{n}\Sigma\phi(x_j) \mid \cdots \mid \phi(x_n) - \frac{1}{n}\Sigma\phi(x_j)\right]$

$= \underline{\Phi} - \underline{\Phi}\underline{1}$, thus

$\underline{\hat{K}} = \underline{\hat{\Phi}}^T \underline{\hat{\Phi}} = (\underline{\Phi} - \underline{\Phi}\underline{1})^T (\underline{\Phi} - \underline{\Phi}\underline{1}) = \Longrightarrow$

$$= \Phi^T\Phi - \mathbb{1}\Phi^T\Phi - \Phi^T\Phi\mathbb{1} + \mathbb{1}\Phi^T\Phi\mathbb{1}$$

$$= K - \mathbb{1}K - K\mathbb{1} + \mathbb{1}K\mathbb{1}$$

(c) The implication of the result of (b) is that to carry out KPCA, we don't really need to worry about the function ϕ; we only need to start with a good kernel.

this extends even to centering the data, which looks like it depends on ϕ, but actually we can do it entirely using K.

2

$x \in \mathbb{R}^{1000}$ $U \in \mathbb{R}^{1000 \times 1000}$, U an orthonormal basis

(a) plot $e_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$ wrt U

$e_1 = c_1 u_1 + \cdots + c_{1000} u_{1000}$

so $[e_1]_U = \begin{bmatrix} c_1 \\ \vdots \\ c_{1000} \end{bmatrix}$

$U \begin{bmatrix} c_1 \\ \vdots \\ c_{1000} \end{bmatrix} = e_1$

$\begin{bmatrix} c_1 \\ \vdots \\ c_{1000} \end{bmatrix} = U^T e_1$

e_1 is sparse in the standard basis, however it is <u>not</u> sparse in U.

(b) x, on the other hand, <u>is</u> sparse wrt U.

(c) see MatLab

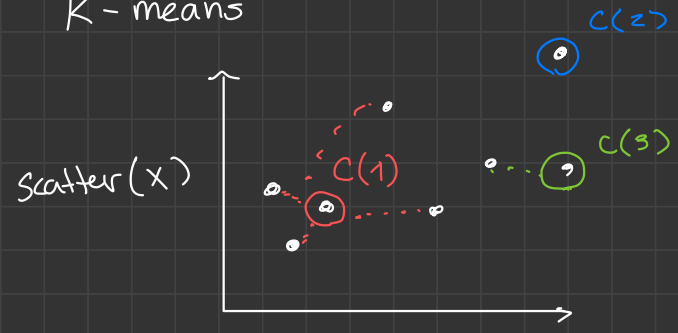
(d) we get a sparse vector in U, but it is not close to c

(e) $\hat{x}$ and x, however, look very similar so it looks as though the reconstruction was pretty close

3 (a) implement K-means

$X \in \mathbb{R}^{n \times 2}$

Clusters = K = 3



R = randperm(n)

C_initial = X(R(1:K), :) $\in \mathbb{R}^{3 \times 2}$ 3 random data points

Dist = zeros(1, K)

Y = zeros(n, 1)

SCRATCH WORK

for i = 1 : n

K
- Dist(1) = norm(X(i) − C(1))
- Dist(2) = norm(X(i) − C(2))
- Dist(3) = norm(X(i) − C(3))

[M, I] = min(Dist)

Y(i) = I

for l = 1 : K

$$C(l) = \frac{1}{\#\text{ of } l\text{s s.t. } Y(i)=l} \sum x_i$$

4 (a) See Matlab implementation

(b) Compared with the plot obtained in the previous assignment, the k-means plot draws very sharp lines between categories. 6's and 0's both have loops, which we could previously see by how some of their respective plots were mixed in with each other. K-means categorizes everything by distance, so there is no mixing allowed.