IITM-CS5691 : Pattern Recognition and Machine Learning — Release Date: Apr 5, 2021
Assignment 2 — Due Date: Apr 18, 2021 (11.59pm)

Roll No: CS18B045 — Name: Rishika Varma K
Collaborators (if any): M Harini Saraswathy, Roshini Karedla, Sumanth Nethi
References (if any): https://www.geeksforgeeks.org/elbow-method-for-optimal-value-of-k-in-kmeans/

- Use LaTeX to write-up your solutions (in the solution blocks of the source LaTeX file of this assignment), and submit the resulting single pdf file at GradeScope by the due date. (Note: **No late submissions** will be allowed, other than one-day late submission with 10% penalty or four-day late submission with 30% penalty! Within GradeScope, indicate the page number where your solution to each question starts, else we won't be able to grade it! You can join GradeScope using course entry code **5VDNKV**).
- For the programming question, please submit your code (rollno.ipynb file and rollno.py file in rollno.zip) directly in moodle, but provide your results/answers in the pdf file you upload to GradeScope.
- Collaboration is encouraged, but all write-ups must be done individually and independently, and mention your collaborator(s) if any. Same rules apply for codes written for any programming assignments (i.e., write your own code; we will run plagiarism checks on codes).
- If you have referred a book or any other online material for obtaining a solution, please cite the source. Again don't copy the source *as is* - you may use the source to understand the solution, but write-up the solution in your own words.
- Points will be awarded based on how clear, concise and rigorous your solutions are, and how correct your code is. Overall points for this assignment would be **min**(your score including bonus points scored, 50).

1. (10 points) [SINGULARLY PCA!] Consider a dataset of N points with each datapoint being a D-dimensional vector in $\mathbb{R}^D$. Let's assume that:
   - we are in a high-dimensional setting where $D >> N$ (e.g., D in millions, N in hundreds).
   - the $N \times D$ matrix X corresponding to this dataset is already mean-centered (so that each column's mean is zero, and the covariance matrix seen in class becomes $S = \frac{1}{N}X^TX$).
   - the rows (datapoints) of X are linearly independent.

   Under the above assumptions, please attempt the following questions.

   (a) (3 points) Whereas X is rectangular in general, XX^T and X^TX are square. Show that these two square matrices have the same set of non-zero eigenvalues. Further, argue briefy why these equal eigenvalues are all positive and N in number, and derive the multiplicity of the zero eigenvalue for both these matrices.
   (Note: The square root of these equal positive eigenvalues $\{\lambda_i := \sigma_i^2\}_{i=1,\ldots,N}$ are called the singular values $\{\sigma_i\}_{i=1,\ldots,N}$ of X.)

Solution: Let XX^T has non-zero eigenvalue λ with eigen vector u, this means that $XX^Tu = \lambda u$. Pre-multiplying X^T on both sides, we get

$$\begin{aligned} X^TXX^Tu &= X^T\lambda u \\ X^TX(X^Tu) &= \lambda(X^Tu) \end{aligned} \tag{1}$$

Therefore it is clear that X^TX and XX^T have same non zero eigenvalues.
The elements in the diagonal places in either of matrices XX^T and X^TX are always positive since they are sum of squares of row or column elements appropriately. Thus the eigen values which have same as sign as that of diagonal elements are also positive. The dimensions of XX^T are $N \times N$ and that of X^TX are $D \times D$. Maximum number of non zero eigen values is $N-1$ for XX^T. X^TX has more eigen values than XX^T as $D >> N$. Since both the matrices have same set of non-zero eigenvalues, the remaining eigenvalues of X^TX must be zeroes which is $D-(N-1)$. This implies that the number of 0 eigen values in XX^T is 1 whereas that in X^TX is $D-N+1$.

(b) (2 points) We can choose the set of eigenvectors $\{u_i\}_{i-=1,\dots,N}$ of XX^T to be an orthonormal set and similarly we can choose an orthonormal set of eigenvectors $\{v_j\}_{j=1,\dots,D}$ for X^TX. Briefly argue why this orthonormal choice of eigenvectors is possible. Can you choose $\{v_i\}$ such that each v_i can be computed easily from u_i and X alone (i.e., without having to do an eigenvalue decomposition of the large matrix X^TX; assume $i = 1, \dots, N$ so that $\lambda_i > 0$ and $\sigma_i > 0$)?
(Note: $\{u_i\}, \{v_i\}$ are respectively called the left,right singular vectors of X, and computing them along with the corresponding singular values is called the Singular Value Decomposition or SVD of X.)

Solution: We have seen that if u is an eigenvector of XX^T then X^Tu is the corresponding eigenvector of X^TX for the same non zero eigenvalue.
To find the normalised eigenvectors of X^TX, we multiply the eigen vectors with a constant. The new normalised eigen vector of X^TX would become $\frac{1}{\sqrt{N\lambda}}X^Tu$ for corresponding values of u. Therefore, we choose v_i such that each v_i can be computed from u_i and the transpose of matrix X alone.

(c) (2 points) Applying PCA on the matrix X would be computationally difficult as it would involve finding the eigenvectors of $S = \frac{1}{N}X^TX$, which would take $O(D^3)$ time. Using answer to the last question above, can you reduce this time complexity to $O(N^3)$? Please provide the exact steps involved, including the exact formula for computing the normalized (unit-length) eigenvectors of S.

Solution: From (b), the eigen vectors of the matrix $S = \frac{1}{N}X^TX$ can be found using the formula $v_i = \frac{1}{\sqrt{N\lambda}}X^Tu_i$

From this it is clear that we need u_i values which are the eigen vectors of XX^T and therefore finding this would only take $O(n^3)$ time. Thus we can compute PCA using the eigen vectors obtained through this formula.

(d) (3 points) Exercise 12.2 from Bishop's book helps prove why minimum value of the PCA squared error J, subject to the orthonormality constraints of the set of principal axes/directions $\{u_i\}$ that we seek, is obtained when the $\{u_i\}$ are eigenvectors of the data covariance matrix S. That exercise introduces a modified squared error $\widetilde{J}$, which involves a matrix H of Langrange multipliers, one for each constraint, as follows:

$$\widetilde{J} = \mathrm{Tr}\left\{\widehat{U}^T S \widehat{U}\right\} + \mathrm{Tr}\left\{H(I - \widehat{U}^T\widehat{U})\right\}$$

where $\widehat{U}$ is a matrix of dimension $D \times (D - M)$ whose columns are given by u_i. Now, any solution to minimizing $\widetilde{J}$ should satisfy $S\widehat{U} = \widehat{U}H$, and one specific solution is that the columns of $\widehat{U}$ are the eigenvectors of S, in which case H is a diagonal matrix. Show that any general solution to $S\widehat{U} = \widehat{U}H$ also gives the same value for $\widetilde{J}$ as the above specific solution.
(Hint: Show that H can be assumed to be a symmetric matrix, and then use the eigenvector expansion i.e., diagonalization of H.)

Solution: If H can be assumed to be symmetric, we can say that $H^T = H$. Substituting given equation and this condition in the expression for $\widetilde{J}$ we get,

$$\begin{aligned}
\widetilde{J} &= \mathrm{Tr}\left\{\widehat{U}^T S\widehat{U}\right\} + \mathrm{Tr}\left\{H(I - \widehat{U}^T\widehat{U})\right\} \\
&= \mathrm{Tr}\left\{\widehat{U}^T (S\widehat{U})\right\} + \mathrm{Tr}\left\{H(I - \widehat{U}^T\widehat{U})\right\} \\
&= \mathrm{Tr}\left\{\widehat{U}^T (\widehat{U}H)\right\} + \mathrm{Tr}\left\{H(I - \widehat{U}^T\widehat{U})\right\} \\
&= \mathrm{Tr}\left\{\widehat{U}^T (\widehat{U}H)\right\} + \mathrm{Tr}\left\{H - H\widehat{U}^T\widehat{U}\right\} \\
&= \mathrm{Tr}\left\{\widehat{U}^T\widehat{U}H\right\} + \mathrm{Tr}\left\{H - H\widehat{U}^T\widehat{U}\right\} \\
&= \mathrm{Tr}\left\{\widehat{U}^T\widehat{U}H\right\} + \mathrm{Tr}\{H\} - \mathrm{Tr}\left\{H\widehat{U}^T\widehat{U}\right\} \\
&= \mathrm{Tr}\left\{\widehat{U}^T\widehat{U}H\right\} + \mathrm{Tr}\{H\} - \mathrm{Tr}\left\{H^T\widehat{U}^T\widehat{U}\right\} \\
&= \mathrm{Tr}\left\{\widehat{U}^T\widehat{U}H\right\} + \mathrm{Tr}\{H\} - \mathrm{Tr}\left\{(\widehat{U}^T\widehat{U}H)^T\right\} \\
&= \mathrm{Tr}\left\{\widehat{U}^T\widehat{U}H\right\} + \mathrm{Tr}\{H\} - \mathrm{Tr}\left\{\widehat{U}^T\widehat{U}H\right\} \\
&= \mathrm{Tr}\{H\}
\end{aligned} \tag{2}$$

Therefore since the value of $\widetilde{J}$ is always trace of H, it is same for all solutions of the equation.

2. (10 points) [TO SQUARE OR NOT SQUARE WITH K-MEANS]

   (a) (3 points) If instead of squared Euclidean distance, you use ℓ_1 norm in the objective function of (hard) K-means, then what are the new assignment and update equations? If your data contains outliers, would you prefer this over the regular K-means? Justify.

Solution: When d is changed to l_1 norm, the update and assignment steps will change as follows:

Update:

$$\begin{aligned} m^{(k)} &= \mathrm{argmin}_{k'} \Sigma_{n:r_k^{(n)}=1} d(x^{(n)}, k') \\ &= \mathrm{argmin}_{k'} \Sigma_{n:r_k^{(n)}=1} (\Sigma_{i=1}^{I} |x_i{}^{(n)} - k'_i|) \end{aligned} \tag{3}$$

The assignment step essentially remains the same wherein the new k median is the median of the points for which $r_k^{(n)} = 1$. This will minimise the cost function.
When there are outliers, it implies that these contribute significantly more to the cost than the other points. In squared euclidean distances, these already large distances are squared resulting in throwing the algo off much more than when l_1 norm is used where we only add the absolute distance. Thus, I would prefer using l_1 norm.

   (b) (2 points) Consider a Gaussian mixture model with scalar covariance matrices: $\Sigma_r = \sigma^2 I$ where σ is a fixed parameter, r represents the mixture-component/cluster, and I the identity matrix. Show that for this model as σ tends to zero, the EM-based soft K-means algorithm (i.e., its assignment/update equations) become the same as the hard K-means algorithm.

Solution: In EM based soft k means algorithm, the E step is given by:

$$r_k^{(n)} = \frac{\pi_k N(x_n \mid \mu_k, \Sigma_k)}{\Sigma_{k'=1}^{K} \pi_{k'} N(x_n \mid \mu_{k'}, \Sigma_{k'})} \tag{4}$$

Here when Σ_k is given by $\sigma^2 I$, Since it is a diagonal matrix, this implies that the above equation will reduce to the following:

$$r_k^{(n)} = \frac{\pi_k e^{-\frac{\|x_n - \mu_k\|^2}{2\sigma^2}}}{\Sigma_{k'=1}^{K} \pi_{k'} e^{-\frac{\left\|x_n - \mu_{k'}\right\|^2}{2\sigma^2}})} \tag{5}$$

Here, when σ tends to zero, There exists some optimum $k = l$ such that $N(x_n \mid \mu_k, \Sigma_k)$ is smallest. When Σ_k tends to null matrix, when $k = l$, $r_k^{(n)}$ value tends to 1 and in for all other k values. This is similar to values of $r_k^{(n)}$ in case of hard k means.

(c) (5 points) We will see how K-means clustering can be done in polynomial time if the data points are along a line (1-dimensional or 1D).

i. (1 point) Consider four datapoints: $x_1 = 1, x_2 = 3, x_3 = 6$, and $x_4 = 7$; and desired number of clusters $k = 3$. What is the optimal K-means clustering in this case?

Solution: Optimal k-means clustering is given by:

$$(1, 3, (6, 7)) = (x_1, x_2, (x_3, x_4)) \tag{6}$$

ii. (1 point) You might think that the iterative K-means algorithm seen in class converges to global optima with 1D datapoints. Show that it can get stuck in a suboptimal cluster assignment for the problem in part (i).

Solution: If the initial values of medians taken are random, in some cases it may result in improper clustering. For example, if the initial medians are 1, 1000, 2000 then during the first iteration all the points are closest to 1 and so the cluster distribution becomes $((x_1, x_2, x_3, x_4), (), ())$. The new first median becomes 4.25 and the remaining medians do not contribute to the distribution. Thus in this case the algorithm gets stuck in a local optimum.

iii. (3 points) Suppose we sort our data such that $x_1 \leqslant x_2 \leqslant \cdots \leqslant x_n$. Show then that any optimal K-means clustering partitions the points into contiguous intervals, i.e. prove that each cluster in an optimal clustering consists of points $x_a, x_{a+1}, \ldots, x_b$ for some $1 \leqslant a \leqslant b \leqslant n$.

Solution:
In this case $d(x, m) = |x - m|$. Proof by contradiction: Assume that the clustering has resulted in a distribution where there is a cluster in which the points are such that it contains $x_a, ...x_b$ and x_k where $x_k < x_{a-1}$ or $x_k > x_{b+1}$. Let value of mean for this cluster is m.
Case1:$x_k < x_{a-1}$ According to our assumption, we know that x_k is present in cluster with mean m but x_{a-1} is not. This would imply that x_k is closer to m compared to x_{a-1}.

$$\begin{aligned} d(x_k, m) &< d(x_{a_1}, m) \\ |x_k - m| &< |x_{a-1} - m| \\ m - x_k &< m - x_{a-1} \\ x_k &> x_{a-1} \end{aligned} \tag{7}$$

But this is contradictory to our assumption. Hence, our assumption is not true.
Case2:$x_k > x_{b+1}$
According to our assumption, we know that x_k is present in cluster with mean m but x_{b+1} is not. This would imply that x_k is closer to m compared to x_{b+1}.

$$\begin{aligned} d(x_k, m) &< d(x_{b+1}, m) \\ |x_k - m| &< |x_{b+1} - m| \\ x_k - m &< x_{b+1} - m \\ x_k &< x_{b+1} \end{aligned} \tag{8}$$

But this is contradictory to our assumption. Hence, our assumption is not true.
Therefore hence proved that optimal clustering can only consist of clusters in contiguous intervals.

iv. (1 point) [BONUS] Show a $O(kn^2)$ dynamic programming algorithm of k-means when the data is 1-dimensional.

Solution: Initially, the points are sorted. This can be done in O(nlogn) time. Assume a matrix in A such that the value $A[i][l]$ represents the cost of the cluster distribution of the points $x_1...x_i$ into l clusters. So we need value of $A[n][k]$.We can fill this matrix using the following recursion.

$$A[i][l] = min_{1<=j<=i}(A[j-1][l-1] + \Sigma_{n=i}^{j} d(x_n^{(k)}, m^k)) \tag{9}$$

For, the second term of the minimum, normally it would take O(n) time, however we can reduce it to linear time by maintaining the sum at each iteration and only adding the newest term. i.e.

$$\Sigma_{n=i}^{j} d(x_n^{(k)}, m^k)) = (\Sigma_{n=i}^{j-1} d(x_n^{(k)}, m^k))) + d(x_j^{(k)}, m^k) \tag{10}$$

Thus this can be calculated in $O(kn^2)$. From this we get the cost of the most optimal clustering for all sets of data points of the form $x_1...x_i$. To get the cluster distribution, we take the result from (iii) and add points till the cost is less than that we calculated. This will give the optimal clustering.

3. (10 points) [THINKING HIERARCHICALLY...] Consider some of the most common metrics of distance between two clusters $A = \{a_1, a_2, \ldots, a_m\}$ and $B = \{b_1, b_2, \ldots, b_n\}$.

- Minimum distance between any pair of points from the two clusters

$$\min_{a\in A, b\in B} ||a - b||$$

- Maximum distance between any pair of points from the two clusters,

$$\max_{a \in A, b \in B} \|a - b\|$$

- Average distance between *all* pairs of points from the two clusters,

$$\frac{1}{m \cdot n} \sum_{i=1}^{m} \sum_{j=1}^{n} \|a_i - b_j\|$$

As discussed in class, we can obtain clusters by *cutting* the hierarchical tree with a line that crosses at required number of points (K).

(a) (2 points) Which of the three distance/dissimilarity metrics described above would most likely result in clusters most similar to those given by K-means? (Consider the hierarchical clustering method as described in class and further *cut* the tree to obtain K clusters. Assume K is power of 2.) Explain briefly.

Solution: The metric most likely to result in clusters similar to k means is the average distance between all pairs of points.

(b) (3 points) Which among the three metrics discussed above (if any) would lead hierarchical clustering to correctly separate the two moons in Figure 1a? How would your answer change (if at all) in case of Figure 1b? Explain briefly.

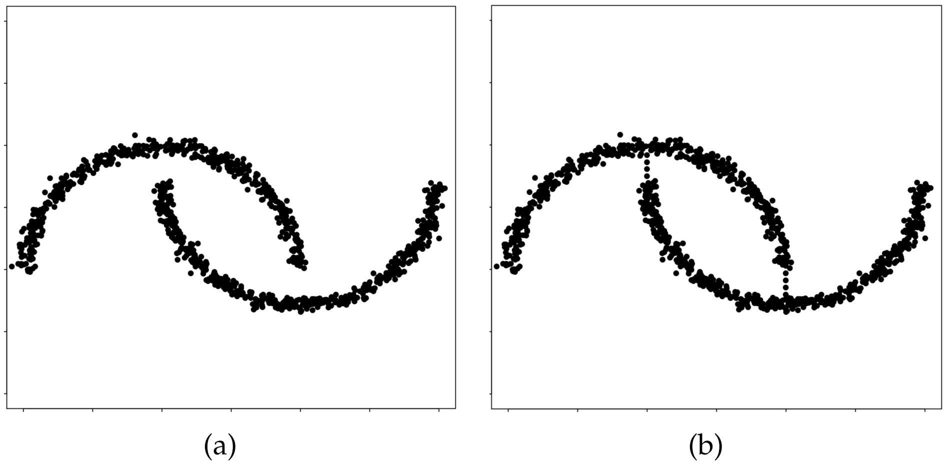


Figure 1: (a) Standard moon crescent distribution. (b) Moon crescent distribution with data points in adjoining area.

Solution: The nature of the distribution in a is such that, although when considering some points the distance to the other moon is closer however for every point there exist particular points which are much closer than the remaining. This is the characteristic of minimum distance between clusters. Thus for a the best metric to differentiate the moons would be minimum distance between any pair of points. In case of b, the distribution is very similar except that there is a connection of closely spaced between the 2 moons. This means that these points might throw the algo off resulting in not differentiating the moons as the minimum distance could be in either direction(in the moon itself or across moons). Thus this metric will also not work for differentiating the moons in b.

(c) (3 points) Consider the distance matrix in Table 1. Show the hierarchy of clusters created by the minimum distance hierarchical clustering algorithm, along with the intermediate steps. Finally, draw the dendrogram with edge lengths indicated.
(Note: You can draw the dendrogram on paper and upload the screenshot.)

Table 1: Distance between nodes

	A	B	C	D	E
A	0	0.73	6.65	4.61	5.24
B	0.73	0	4.95	2.90	3.45
C	6.65	4.95	0	2.24	1.41
D	4.61	2.90	2.24	0	1
E	5.24	3.45	1.41	1	0

Solution: Here, according to the algorithm, choosing the nodes with minimum distance between them and making a cluster. Here the nodes would be A and B with distance between them being 0.73. New table becomes,

	AB	C	D	E
AB	0	4.95	2.90	3.45
C	4.95	0	2.24	1.41
D	2.90	2.24	0	1
E	3.45	1.41	1	0

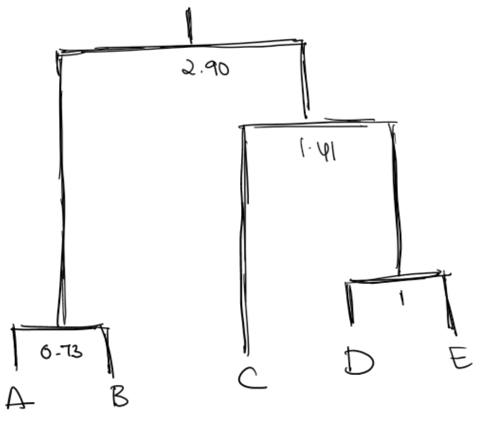
Next cluster would be between E and D with shortest distance 1. Implies the table is

	AB	C	DE
AB	0	4.95	2.90
C	4.95	0	1.41
DE	2.90	1.41	0

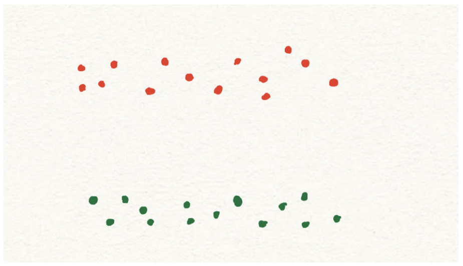
Now shortest distance is 1.41 between DE and C.

	AB	CDE
AB	0	2.90
CDE	2.90	0

The only remaining 2 clusters are combined to form the root. The dendrogram will be as follows:



(d) (2 points) Which distance metric (minimum, maximum and average) is more likely to generate following results given in Figure 2a for K = 2? Why?



(a)

Figure 2: Result produced for K = 2 clusters. Red points belong to one cluster and green to the other.

Solution: As the width of the cluster is considerably more than the distance between the 2 clusters, if average or maximum distance metrics would make clusters vertically with points on the same side rather than horizontally with points at the 2 ends of the given clusters. Thus using minimum distance metric would be favourable to getting clusters of this form.

4. (10 points) [CUTTING SPECTRAL APART] One of the several ways to express a given dataset is by

using a *graph*. Each of the N datapoints in the dataset can be thought of as a vertex/node in a graph and any two datapoints can be connected in the graph with an edge whose non-negative weight W_{ij} indicates the similarity between the ith and jth datapoints. We will look at methods to partition this graph into two clusters, especially one that gained early prominence in computer vision. These methods can be recursively applied to partition the graph into any required number of clusters.

(a) (1 point) A graph cut is a technique that separates a given graph into two disjoint sets of vertices and the degree of similarity (formally called the *cut cost*) between the two sets is given by the sum of weights of the edges between the sets (i.e., edges whose two endpoint nodes lie in different sides of the partition). The obvious method to separate the data into two is by choosing a partition that has the minimum cut cost. What do you think is/are the drawback(s) of this method? (Hint: Think about the sizes of the two sets in the partition.)

Solution: This method has 2 steps. The first step takes $O(n^2)$ and the second step takes $O(n^3)$ time. Thus overall the complexity of this technique is $O(n^3)$. This is quite high time complexity and so for larger data sets it takes a large amount of time which is not ideal. This is one of the drawbacks. Sometimes, when there are isolated nodes, the minimum cut technique cuts off the isolated nodes because of the smaller values resulting when they are partitioned.

(b) (2 points) Due to the above drawback(s), we use a variation of the min cut method called normalized cut to partition the graph into two. The problem of finding the minimum-cost normalized cut can be reduced to this problem:

$$min_y \frac{y^T(D-W)y}{y^TDy} \text{ subject to } y \in \{1, -c\}^N \text{ and } y^TD\mathbf{1} = 0$$

where y_i takes one of the two discrete values $\{1, -c\}$ to indicate which side of the cut/partition the ith datapoint belongs to, W is the symmetric $N \times N$ similarity (non-negative edge-weights) matrix, D is a diagonal matrix called the degree matrix with $d_{ii} = \Sigma_j W_{ij}$, and $\mathbf{1}$ is a vector whose entries are all ones. This expression (not including the constraints) is called the *Generalized Rayleigh's Quotient* (GRQ). The matrix in the numerator, $D - W$ is the called the Laplacian Matrix, denoted by L. Prove that the Laplacian matrix is a singular matrix.

Solution: The matrix L is such that the sum of values in its rows and columns always add to 0. Thus from this we can conclude that for a vector $X = [1, 1...1]$, $LX = 0$. Thus 0 is an eigen value of the matrix L. As we could show that there exists an eigen value and corresponding eigen vector for L it is clear that L is a singular matrix.

(c) (3 points) As the above minimization problem is NP-hard with the two constraints, we first let go of both constraints. Then, the above GRQ can be minimized over $y \in \mathbb{R}^N$ by solving the generalized eigenvalue system $(D - W)y = \lambda Dy$. Show that this equation can be expressed in the form $(ALA)z = \lambda z$, by expressing A, z in terms of D, W, y. Compute the eigenvector corresponding to the smallest eigenvalue of the matrix $M = ALA$.

$$\begin{aligned} y_1 &= \mathbf{1} \\ D^{-1/2}z_1 &= \mathbf{1} \\ z_1 &= D^{1/2}\mathbf{1} \end{aligned} \tag{13}$$

Solution: Here, we have the equation $(D - W)y = \lambda Dy$. Pre-multiplying by $D^{-1/2}$ on both sides and replacing y with $D^{-1/2}z$ of it we get,

$$\begin{aligned} D^{-1/2}(D - W)y &= D^{-1/2}\lambda Dy \\ &= \lambda D^{-1/2}Dy \\ &= \lambda D^{1/2}y \\ D^{-1/2}(D - W)D^{-1/2}z &= \lambda D^{1/2}D^{-1/2}z \\ D^{-1/2}(D - W)D^{-1/2}z &= \lambda z \end{aligned} \tag{11}$$

Comparing this with $(ALA)z = \lambda z$ we get, $A = D^{-1/2}$ and $L = (D - W)$. From intial substitution $y = D^{-1/2}z$ by pre multiplying with $D^{1/2}$ we get

$$\begin{aligned} y &= D^{-1/2}z \\ D^{1/2}y &= D^{1/2}D^{-1/2}z \\ D^{1/2}y &= z \end{aligned} \tag{12}$$

Thus, $z = D^{1/2}y$. Here, as we know $D - W$ has minimum eigen value as 0. So, we can also say that ALA will also have minimum eigen value as 0 (Since, D is a diagonal matrix which means $D^{-1/2}$ is a symmetric matrix and so transpose of A is A itself. Here L is semi definite and it is known that ALA^T is semi definite which is the same as ALA). We know that the first eigen vector of $D - W$ is $\mathbf{1}$.

So, consequently, considering first eigen vector of $D - W$ as y_1 and that of ALA as z_1,

(d) (4 points) Now, let's bring back the constraint that $y^T D\mathbf{1} = 0$ (the constraint that y takes only two discrete values can remain relaxed as a final real-valued solution $\{y_i\}$ can be clustered using 2-means for instance to identify the desired partition). Prove that the GRQ above (subject to $y^T D\mathbf{1} = 0$) is minimized when y is the eigenvector corresponding to the second smallest eigenvalue of the generalized eigenvalue system $(D - W)y = \lambda Dy$.
(Hint: You can use the following fact. Let A be a real symmetric matrix. Under the constraint that x is orthogonal to the $(j-1)$ eigenvectors corresponding to the $(j-1)$ smallest eigenvalues

of A, the Rayleigh's quotient $\frac{x^T A x}{x^T x}$ is minimized when x is the eigenvector corresponding to the j^{th} smallest eigenvalue.)

Solution: Considering D is a diagonal matrix and W is a symmetric matrix, we can say that $D - W$ is also symmetric. At the same time, it is clear that $D^{1/2}$ and consequently $D^{-1/2}$ are also symmetric. Thus A, L are symmetric implies ALA is symmetric. The given GRQ can be converted to A,L,z terms in the following way,

$$min_y \frac{y^T(D-W)y}{y^T D y} = min_z \frac{z^T(ALA)z}{z^T z} \tag{14}$$

Modifying the constraint as well we get,

$$\begin{aligned} y^T D 1 &= 0 \\ (y^T D^{1/2})(D^{1/2} 1) &= 0 \\ (D^{1/2} y)^T (D^{1/2} 1) &= 0 \\ z^T z_1 &= 0 \end{aligned} \tag{15}$$

This implies that the solutions of the constraint are the eigen vectors of ALA that are perpendicular to its first eigen vector.

From the hint, we can observe that the minimising framework given there is the same as the converted GRQ, with ALA corresponding to the matrix A and x corresponding to z, we also have the condition that z must be such that it is orthonormal to the first eigen vector of ALA when oredred according to their eigen values. Thus value of j-1 in this case is 1 which means value of j is 2. Therefore the GRQ is minimised when x is the eigen vector corresponding to the second smallest eigen value. This is Z_2. The corresponding y value would be the y_2 which is the eigen vector corresponding to the second smallest eigen value of the generalized eigenvalue system $(D - W)y = \lambda D y$.

Therefore hence proved.

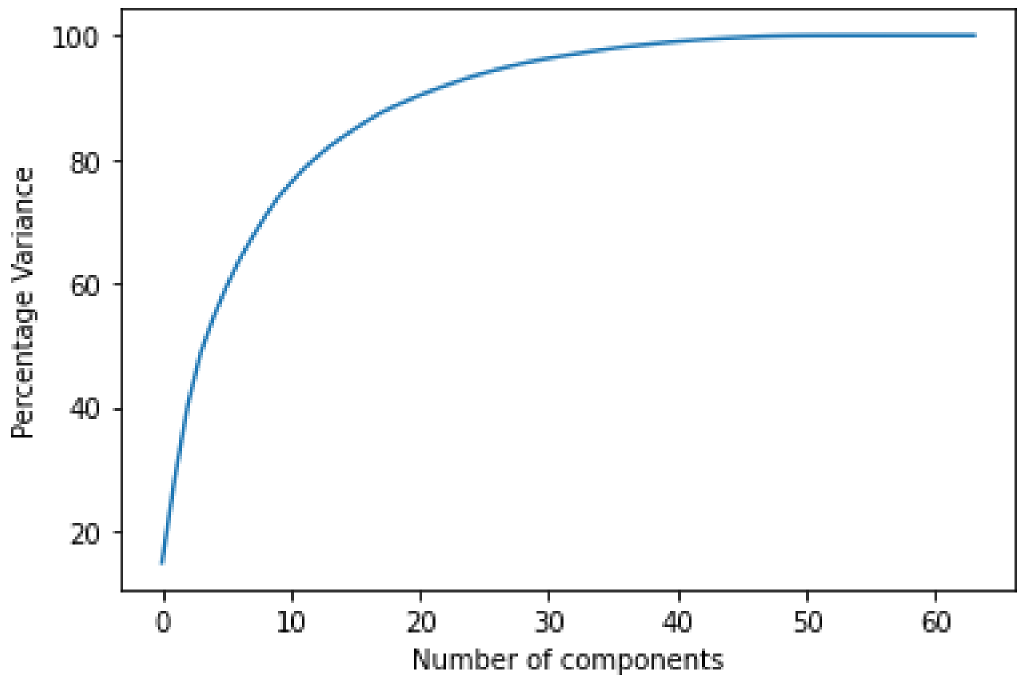
5. (10 points) [LIFE IN LOWER DIMENSIONS...] You are provided with a dataset of 1797 images in a folder here - each image is 8x8 pixels and provided as a feature vector of length 64. You will try your hands at transforming this dataset to a lower-dimensional space, and clustering the images in this reduced space.

   Please use the template .ipynb file in the same folder to prepare your solution. Provide your results/answers in the pdf file you upload to GradeScope, and submit your code separately in this moodle link. The code submitted should be a rollno.zip file containing two files: rollno.ipynb file (including your code as well as the exact same results/plots uploaded to Gradescope) and the associated rollno.py file.

Write the code from scratch for both PCA and clustering. The only exception is the computation of eigenvalues and eigenvectors for which you could use the numpy in-bulit function.

(a) (3 points) Run PCA algorithm on the given dataset. Plot the cumulative percentage variance explained by the principal components. Report the number of principal components that contribute to 90% of the variance in the dataset.

Solution:



The number of components that contribute to 90% of the dataset are 20.

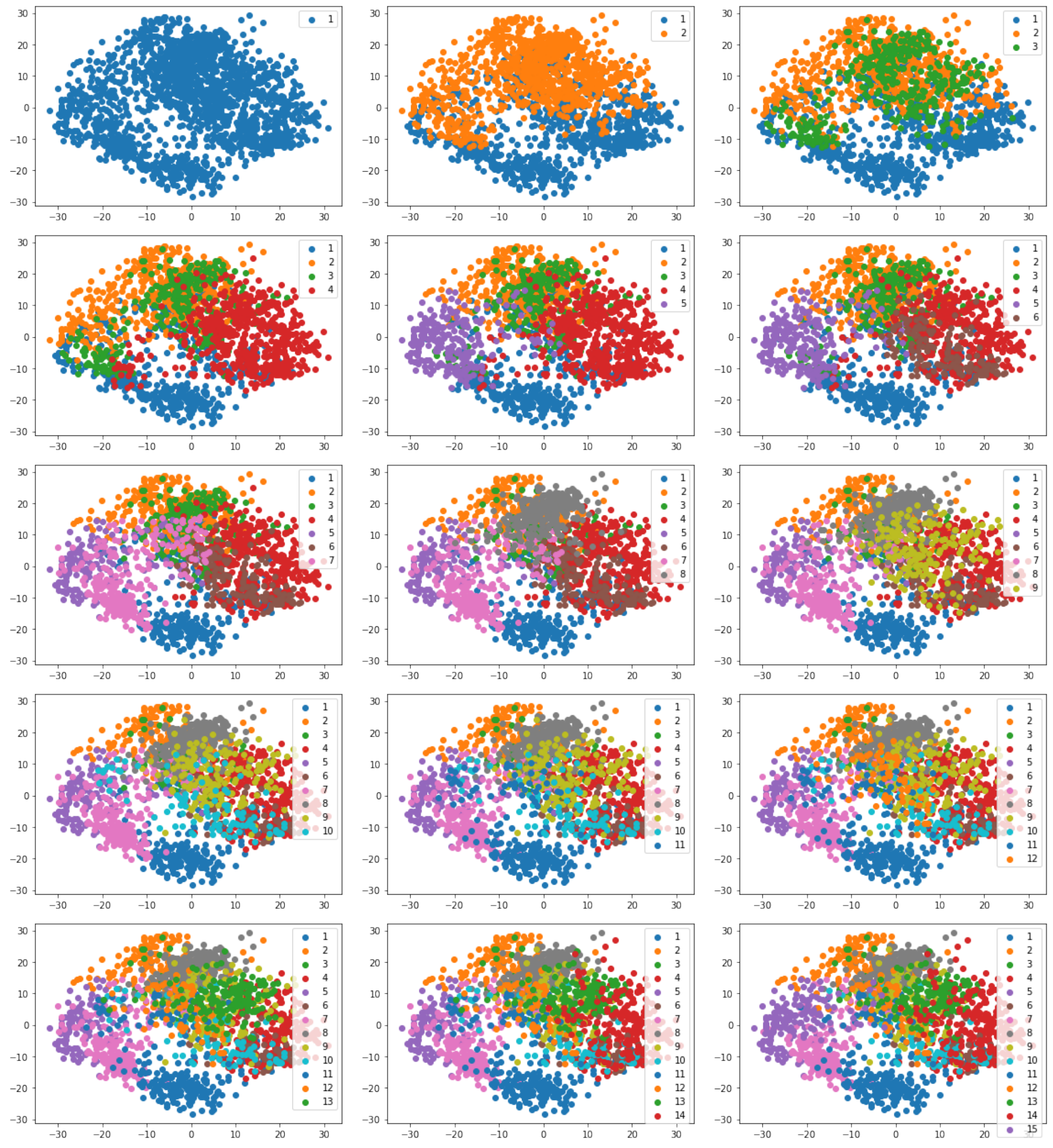
(b) (3 points) Perform reconstruction of data using the dimensionality-reduced data considering the number of dimensions [2,4,8,16]. Report the Mean Square Error (MSE) between the original data and reconstructed data, and interpret the optimal dimension $\widehat{d}$ based on the MSE values.

Solution:
The Mean square error for 2 dimensions = 858.9447808487326
The Mean square error for 4 dimensions = 616.1911300562695
The Mean square error for 8 dimensions = 391.79473611497656
The Mean square error for 16 dimensions = 180.9397032573786
Optimal dimension is 16

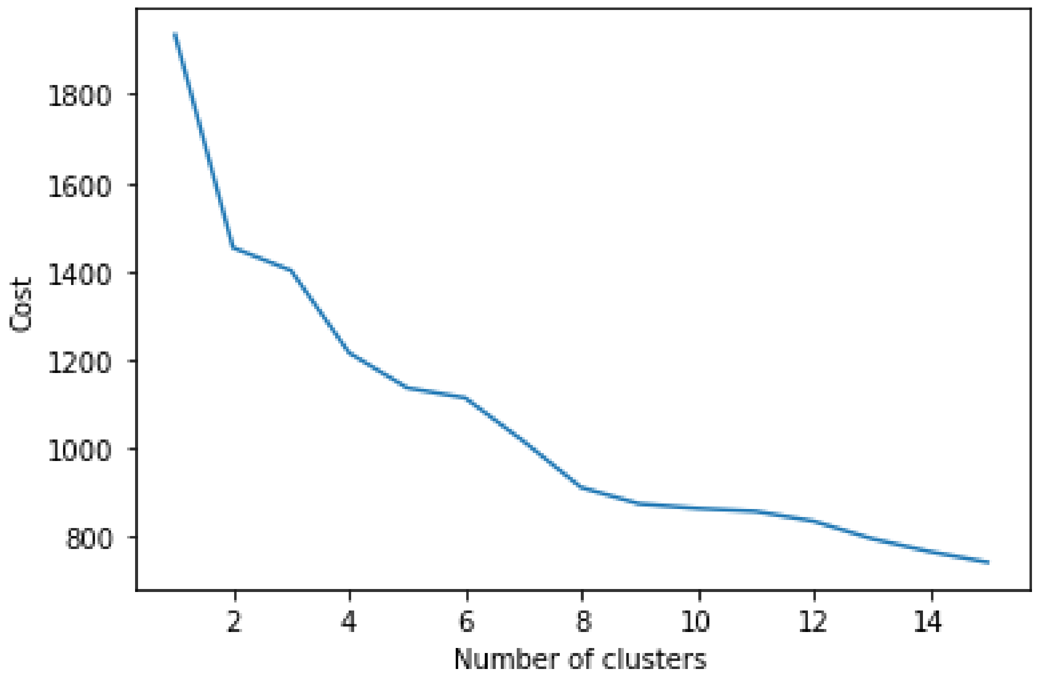
(c) (3 points) Apply K-means clustering on the reduced dataset from last subpart (b) (i.e., the $\mathbb{R}^{64}$ to $\mathbb{R}^{\widehat{d}}$ reduced dataset; pick the initial k points as cluster centers during initialization). Report

the optimal choice of K you have made from the set [1...15]. Which method did you choose to find the optimum number of clusters? And explain briefy why you chose that method.
Also, show the 2D scatter plot (consider only the first two dimensions of optimal $\widehat{d}$) of the datapoints based on the cluster predicted by K-means (use different color for each cluster).

Solution: The 2D scatter plots of the clusters predicted by k means for all the given values of k are as follows:



The method that I used to find the optimal number of clusters is the elbow method. This method involves plotting the total cost of the clusters corresponding to each number of clusters value and then choosing the value at which the decrease in cost starts becoming linear AKA the elbow of the plot. The plot that was obtained for the given data set is as follows:



Here the elbow of the plot appears to be at 9. Therefore the optimal number of clusters for this data set would be 9.

I used the elbow method to get optimal number of clusters because it also takes into consideration the number of clusters along with the cost. If we were to take only the cost which is done in methods like mse, then automatically the most optimal number of clusters would be given as the highest number clusters. But this is not true as this is the result of over fitting rather than the number of clusters that are actually present. The elbow method takes into consideration and reports the number of clusters as the point after which the total cost does not rapidly decrease.

(d) (1 point) Summarise and explain your observations from the above experiments. Is the PCA+K-means clustering consistent with how your brain would cluster the images?

Solution:
From the first part we observe that majority of the 64 components contribute very little

percentage variance and so it makes sense to reduce our data to a lesser dimension for analysis. It is clear that a lot of the dimensions are inherently dependent on the others from this. From the second part we see that greater the number of dimensions that we reduce the data to, lesser is mean squared error of the reconstructed data. This also appears reasonable since with more number of dimensions more of the variation is represented consequently preserving more of the original data. Thus the error would be lower. In the third part we see the data clusters as scatter plots of the first 2 dimensions. On first glance it does not appear as though the clustering was done properly as the clusters overlap and in many cases it does not appear as though it is optimally clustered. However, this can be explained by the fact that we are only plotting 2 of the 16 dimensions and so these plots do not capture the essence of the clusters properly. This is also reinforced because, when we run the algorithm for the reduced data of 2 dimensions, we can see that the clusters look much more appropriate as there all aspects of the data are represented.