

# COM S 573: Machine Learning

## Homework #4

1. Please put required code files and report into a compressed file “HW#\_FirstName.LastName.zip”
2. Unlimited number of submissions are allowed on Canvas and the latest one will be graded.
3. **No later submission is accepted.**
4. Please read and follow submission instructions. No exception will be made to accommodate incorrectly submitted files/reports.
5. All students are required to typeset their reports using latex. Overleaf (<https://www.overleaf.com/learn/latex/Tutorials>) can be a good start.

### 1. (30 points) Hierarchical clustering

Use the similarity matrix in Table ?? to perform (1) single (MIN) and (2) complete (MAX) link hierarchical clustering. Show each step with dendrogram and the corresponding similarity matrix update. The dendrogram should clearly show the order in which the points are merged. Suppose we choose to use 3 clusters, Show the cut in each final dendrogram.

Table 1: Similarity matrix.

	p1	p2	p3	p4	p5
p1	1.00	0.10	0.41	0.55	0.35
p2	0.10	1.00	0.64	0.47	0.98
p3	0.41	0.64	1.00	0.44	0.85
p4	0.55	0.47	0.44	1.00	0.76
p5	0.35	0.98	0.85	0.76	1.00

**Solution:**

	p1	p2 $\cup$ p5	p3	p4
p1	1.00	0.35	0.41	0.55
p2 $\cup$ p5	0.35	1.00	0.85	0.76
p3	0.41	0.85	1.00	0.44
p4	0.55	0.76	0.44	1.00

	p1	p2 $\cup$ p3 $\cup$ p5	p4
p1	1.00	0.41	0.55
p2 $\cup$ p3 $\cup$ p5	0.41	1.00	0.76
p4	0.55	0.76	1.00

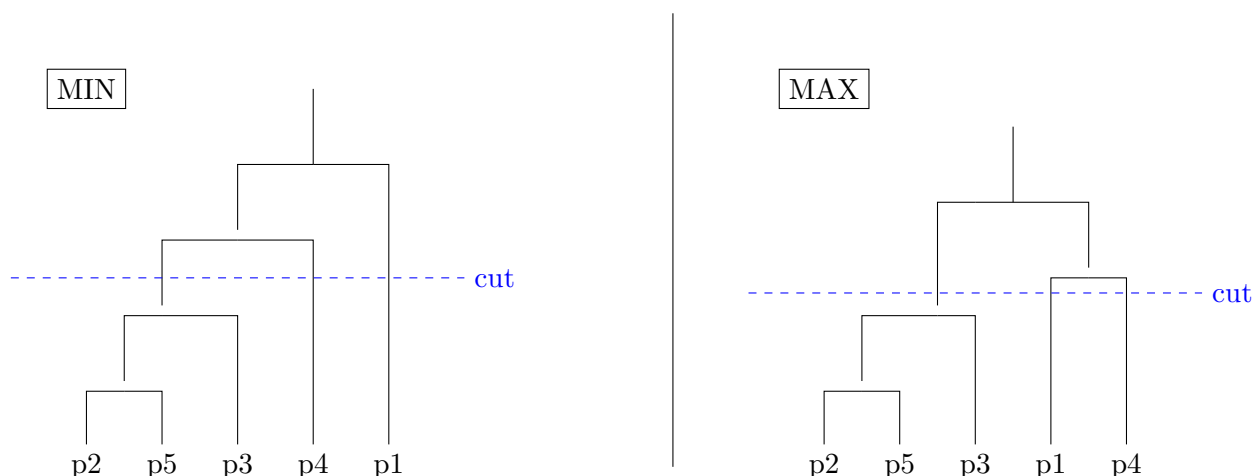
	p1	p2 $\cup$ p3 $\cup$ p4 $\cup$ p5
p1	1.00	0.55
p2 $\cup$ p3 $\cup$ p4 $\cup$ p5	0.55	1.00

Table 2: Hierarchical clustering using MIN. (left to right)

	p1	p2 $\cup$ p5	p3	p4		p1	p2 $\cup$ p5 $\cup$ p3	p4
p1	1.00	0.1	0.41	0.55	p1	1.00	0.1	0.55
p2 $\cup$ p5	0.1	1.00	0.64	0.47	p2 $\cup$ p5 $\cup$ p3	0.1	1.00	0.44
p3	0.41	0.64	1.00	0.44	p4	0.55	0.44	1.00
p4	0.55	0.47	0.44	1.00				

	p1 $\cup$ p4	p2 $\cup$ p5 $\cup$ p3
p1 $\cup$ p4	1.00	0.1
p2 $\cup$ p5 $\cup$ p3	0.1	1.00

Table 3: Hierarchical clustering using MAX. (left to right).



## 2. (20 points) K-Medians Clustering

The K-means algorithm can be summarized as below:

- Select K points as the initial centroids.
- repeat**
- Form K clusters by assigning all points to the closest centroid.
- Recompute the centroid of each cluster.
- until** The centroids don't change.

K-medians clustering is a variation of k-means clustering where it calculates the median for each cluster to determine its center instead of using the mean. Also, K-medians makes use of the Manhattan distance for points assignment.

- (8 points) Please show the algorithm of K-medians in the above format. The K-medians algorithm can be summarized as below:
  - Select K points as the initial centroids.
  - repeat**
  - Form K clusters by assigning all points to the closest centroid. To find out the closest point use Manhattan distance.
  - Recompute the centroid of each cluster by taking the median in each dimension of the vector points assigned to that cluster

v. **until** Sum of the Manhattan distances from the points to cluster center is less than some threshold.

(b) (6 points) Please explain how you will compute the median for each cluster.

**Sol:** We find out the median in each dimension of the vector points. To do this, first, we have to sort the  $i$  th entry elements of the data points assigned to that cluster and find the mid value this we call as the median. We repeat this process for all  $i$ .

(c) (6 points) Does K-medians help to avoid the outlier problem? Justify your answer.

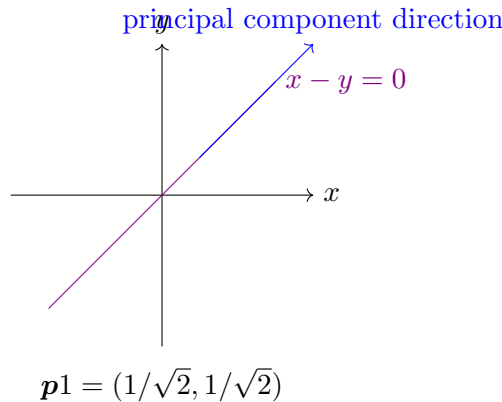
**Sol:** Yes, because K-median sees the outlier as one more extra point and does not consider it as an outlier. Therefore, K-median is robust for outliers.

### 3. (20 points) **Principal Components Analysis**

Given three data points:  $(-1, -1), (0, 0), (1, 1)$ .

(a) (10 points) Show the first Principal Component (actual vector) without using Eigendecomposition. Justify your answer.

**Sol:** Given data points  $(-1, -1), (0, 0), (1, 1)$ . The line passes through the points  $(-1, -1), (0, 0)$ , and  $(1, 1)$  preserving the maximum variance of the data points. Since the slope of points  $(0, 0)$ , and  $(1, 1)$  is 1 and the intercept is 0, the line equation is  $x - y = 0$ . If we consider  $x = t \implies y = t$ . The direction in which the line passes is  $r(t) = (0, 0) + t * (1, 1)$ . Upon normalizing the point  $(1, 1)$  the direction of the principal component is  $(1/\sqrt{2}, 1/\sqrt{2})$ .



(b) (10 points) If use the 1<sup>st</sup> principle component to transform the data into 1-d space. What are the new data?

**Sol:** The new data points are projecting the points on the principal component direction  $(1/\sqrt{2}, 1/\sqrt{2})$  are

$$(-1, -1)^T (1/\sqrt{2}, 1/\sqrt{2}) = -\sqrt{2},$$

$$(0, 0)^T (1/\sqrt{2}, 1/\sqrt{2}) = 0, \text{ and}$$

$$(1, 1)^T (1/\sqrt{2}, 1/\sqrt{2}) = \sqrt{2}.$$

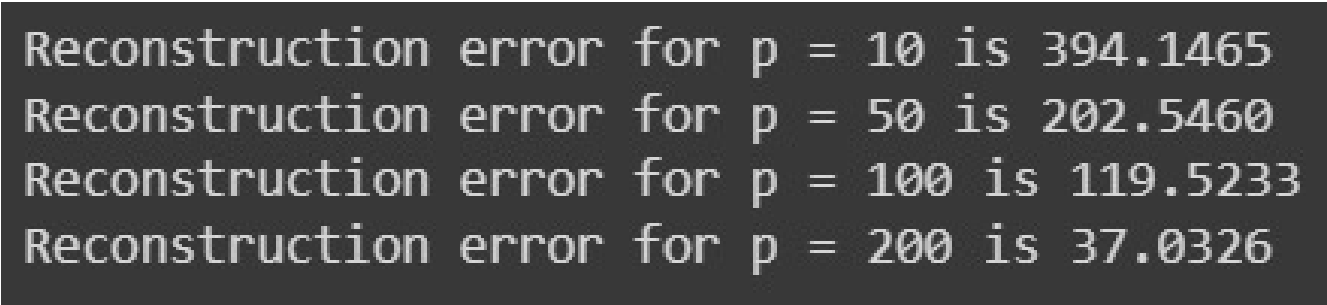
### 4. (30 points) **Principal Component Analysis:**

In this homework, you will apply the principal component analysis to a collection of handwritten digit images from the USPS dataset. The USPS dataset is in the “data” folder: USPS.mat. The starting code is in the “code” folder. The whole data has already been loaded into the matrix  $A$ . The matrix  $A$  has shape  $3000 \times 256$  and contains all the images. Each row in  $A$  corresponds to a handwritten digit image (between 0 and 9) with size  $16 \times 16$ . You are expected to implement your

solution based on the given codes. The only file you need to modify is the “solution.py” file. You can test your solution by running the “main.py” file.

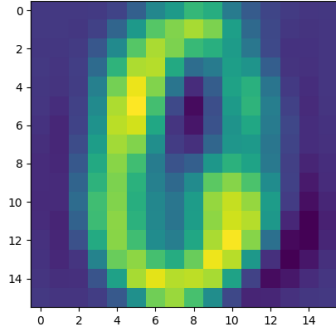
- (a) (15 points) In PCA, we obtain a projection matrix or reduce matrix  $\mathbf{U} \in \mathbb{R}^{d \times p}$ . Based on  $\mathbf{U}$ , we project the original centered data  $\tilde{\mathbf{X}} \in \mathbb{R}^{d \times n}$  into reduced data  $\mathbf{Z} \in \mathbb{R}^{p \times n}$ . Complete the `_do_pca()` method. You only need to center the data instead of applying mean normalization. Your code will be tested on  $p = 10, 50, 100, 200$ , total four different numbers of the principal components.
- (b) (5 points) Based on the projection matrix  $\mathbf{U}$  and reduce data  $\mathbf{Z}$ , we can reconstruct the original data  $\mathbf{X}'$  by  $\mathbf{U}\mathbf{Z}$  and adding back the original means. Here you need to Complete the `reconstruction()` method to reconstruct the reduced data.
- (c) (5 points) Based on the reconstructed data  $\tilde{\mathbf{X}}'$ , we can compute measure the reconstruction error by  $\|\mathbf{X} - \mathbf{X}'\|_F^2$ . Complete the `reconstruct_error()` function to measuring the reconstruction error.
- (d) (5 points) Run “main.py” to see the reconstruction results and summarize your observations from the results into a short report. When you run the “main.py” file, a subset (the first two) of the reconstructed images based on  $p = 10, 50, 100, 200$  principal components will be automatically saved on the “code” folder. Please attach these images into your report also.

**Sol:** From Fig 1 we can see that as the number of principal components increases the reconstruction error decreases.

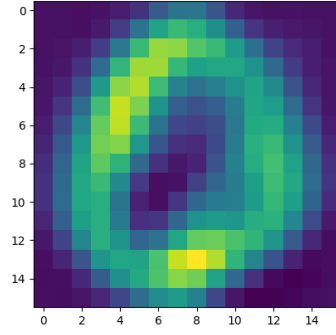


```
Reconstruction error for p = 10 is 394.1465
Reconstruction error for p = 50 is 202.5460
Reconstruction error for p = 100 is 119.5233
Reconstruction error for p = 200 is 37.0326
```

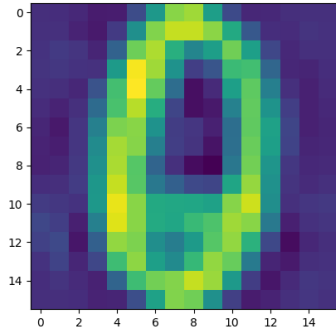
Figure 1: Reconstruction Error



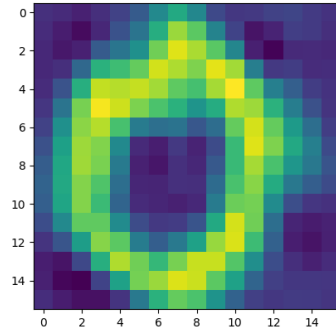
(a) Reconstructed Image 1 with  $d = 10$



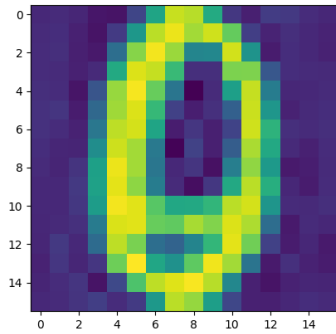
(b) Reconstructed Image 2 with  $d = 10$



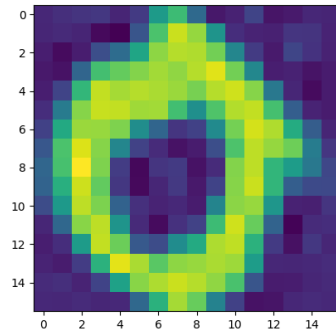
(c) Reconstructed Image 1 with  $d = 50$



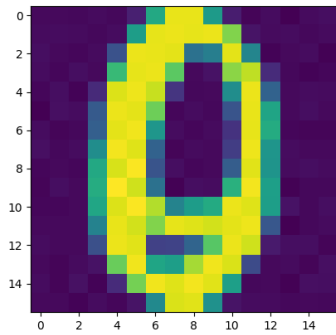
(d) Reconstructed Image 2 with  $d = 50$



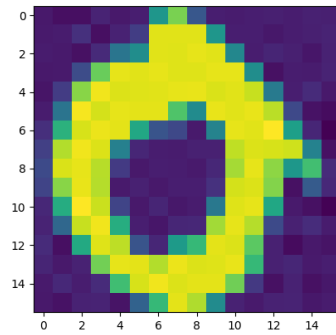
(e) Reconstructed Image 1 with  $d = 100$



(f) Reconstructed Image 2 with  $d = 100$



(g) Reconstructed Image 1 with  $d = 200$



(h) Reconstructed Image 2 with  $d = 200$