National University of Singapore
School of Computing
CS2109S: Introduction to AI and Machine Learning
Semester 2, 2024/2025

Tutorial 7
Unsupervised Learning

Summary of Key Concepts

In this tutorial, we will discuss and explore the following key learning points/lessons.

1. K-means algorithm
   (a) K-medoids algorithm
2. Kernel K-means clustering
   (a) Gaussian radial basis function (RBF) kernel
3. SVD

A K-means algorithm

The K-means algorithm is given as follows:

Algorithm 1: K-means clustering

```
for $k = 1$ to $K$ do
    $\mu_k \leftarrow$ random location
while not converged do
    for $i = 1$ to $m$ do
        $c^{(i)} \leftarrow argmin_k||x^{(i)} - \mu_k||^2$
    for $k = 1$ to $K$ do
        $\mu_k \leftarrow \frac{1}{|\{x^{(i)}|c^{(i)}=k\}|} \sum_{x \in \{x^{(i)}|c^{(i)}=k\}} x$
```

Note that a key fact of the K-means algorithm is that the algorithm never increases the loss function.

1. From the key fact, it is clear that every iteration produces a partition with a lower or equal loss. Prove from this fact that the K-means algorithm always converges. Convergence is when the centroids/medoids do not change after an iteration of the algorithm.
2. Although K-means always converges, it may get stuck at a bad local minimum. As mentioned in the lecture, one method of circumventing this is to run the algorithm multiple times and choose the clusters with the minimum loss. Suggest some other ways to help the algorithm get closer to the global minimum.
3. You are given the following data.

   Cluster the 6 points in table 1 into **two** clusters using the K-means algorithm. The two initial centroids are (0, 1) and (2.5, 2).

i	1	2	3	4	5	6
x	1	1	2	2	3	3
y	0	1	1	2	1	2

Table 1: 6 data points on a 2D-plane

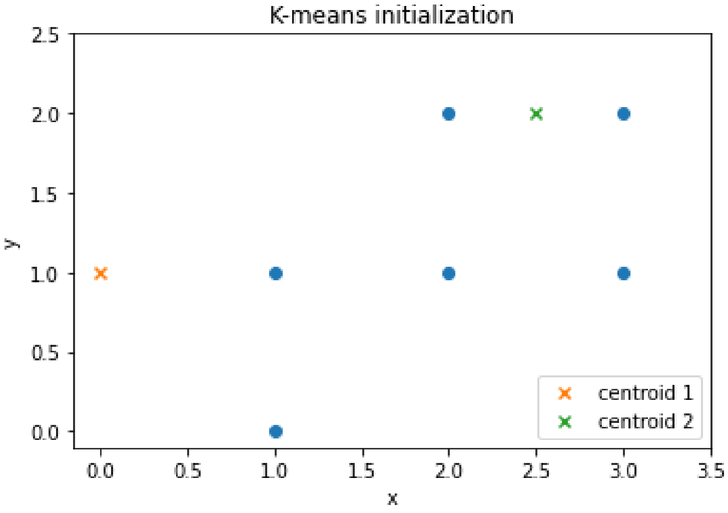

Figure 1: Initial configuration of K-means

4. Cluster the 6 points in table 1 into **two** clusters using the K-medoids algorithm. The initial medoids are point 1 and point 3. The K-medoids algorithm is given in Algorithm 2.

Algorithm 2: K-medoids clustering

```
for $k = 1$ to $K$ do
    $\mu_k \leftarrow$ random data point $x^{(i)}$
while not converged do
    for $i = 1$ to $m$ do
        $c^{(i)} \leftarrow argmin_k||x^{(i)} - \mu_k||^2$
    for $k = 1$ to $K$ do
        $\mu_k \leftarrow \frac{1}{|\{x^{(i)}|c^{(i)}=k\}|} \sum_{x \in \{x^{(i)}|c^{(i)}=k\}} x$
    for $k = 1$ to $K$ do
        $\mu_k \leftarrow argmin_{x^{(i)} \in \{x^{(i)}|c^{(i)}=k\}}||x^{(i)} - \mu_k||^2$
```

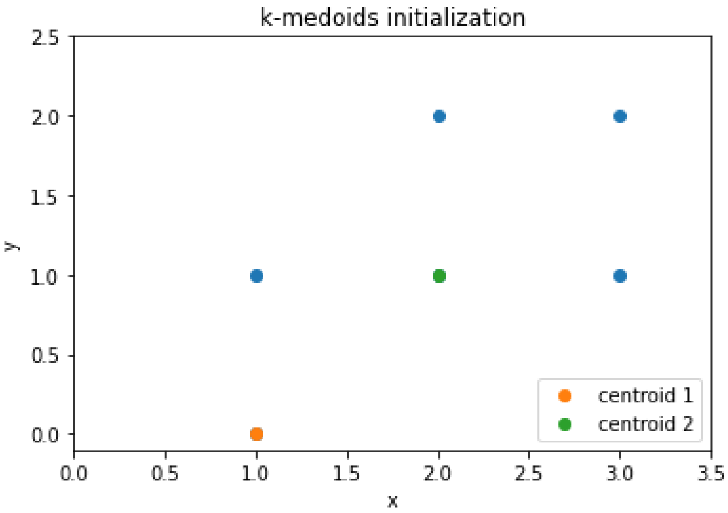

Figure 2: Initial configuration of K-medoids

B Kernel K-means clustering

1. Given the set of data points in the figure below, Clustering 1 should be considered better than Clustering 2. Can the K-means algorithm achieve the better clustering? Why or why not?

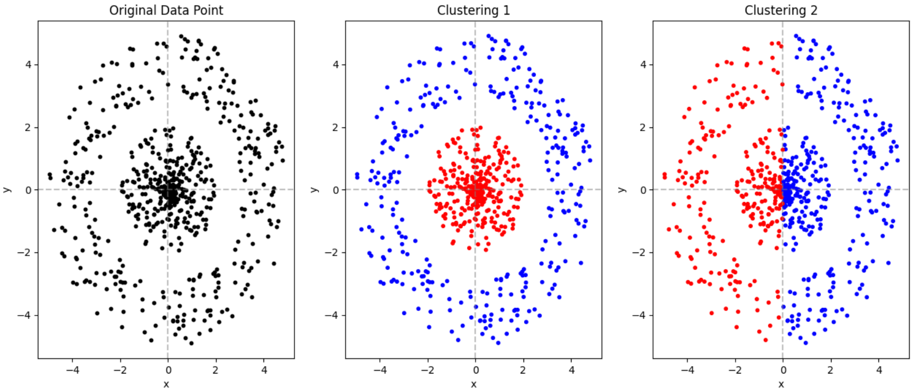

Figure 3: Different clustering approaches on the same dataset

2. We would like to extend the K-means algorithm to non-linearly separable cases. To do that, write the squared Euclidean distance between two points $\mathbf{p}_i$ and $\mathbf{p}_j$ using vector norms (length) and the dot product. How can we apply the kernel trick?

3. For the remainder of the problem, you are given the following data.

   Given five points in Table 2 and two initial centroids, (1,1) and (-4,-4), we perform K-means clustering with two clusters. After convergence, we obtain:

   - The first cluster has centroid (1, 1) and contains points 1, 2, 3, and 5.
   - The second cluster has centroid (-4, -4) and contains point 4.

p	x	y
1	0	0
2	4	4
3	-4	4
4	-4	-4
5	4	-4

Table 2: 5 data points on a 2D-plane

The table below shows that the K-means algorithm has converged.

Point	Squared distance to first centroid	Squared distance to second centroid	Assigned cluster
1	2	32	1
2	18	128	1
3	34	64	1
4	50	0	2
5	34	64	1

- Centroid 1 $\mu_1 = ((-4,4)+(0,0)+(4,-4)+(4,4))/4 = (1,1)$.
- Centroid 2 $\mu_2 = (-4,-4)/1 = (-4,-4)$.

The Gaussian radial basis function (RBF) kernel maps data into a higher-dimensional space, where clusters potentially become linearly separable. For a point $\mathbf{p}_i = (x_i, y_i)$, the formula for the RBF kernel $k_{\text{rbf}}(\mathbf{p}_i, \mathbf{p}_j)$ is given by:

$$k_{\text{rbf}}(\mathbf{p}_i, \mathbf{p}_j) = \exp\left(-\frac{\|\mathbf{p}_i - \mathbf{p}_j\|^2}{2\sigma^2}\right),$$

where σ is a parameter that controls the width of the kernel. Set $\sigma = 4$. Calculate $k_{\text{rbf}}(\mathbf{p}_i, \mathbf{p}_j)$ for $i, j = 1, \ldots, 5$. Explain what the value $k_{\text{rbf}}(\mathbf{p}_i, \mathbf{p}_j)$ represents.
Note: The matrix with the entries $k_{ij} := k_{\text{rbf}}(\mathbf{p}_i, \mathbf{p}_j)$ is called a kernel matrix (for a given kernel function).

4. (Optional) Evaluate the distance to the previous cluster centers μ_1 and μ_2 using the kernel trick and the kernel matrix from the previous question.
Hint: The kernel trick is applied only to dot products between the data points and not *directly* to dot products between a data point and the cluster mean.

C SVD

PCA can be thought of as a form of lossy compression. In this question, we will compress some data, provided in the form of an image for better visualisation. We will be making use of a classic test image `mandrill.png`, which can be downloaded from https://upload.wikimedia.org/wikipedia/commons/a/ab/Mandrill-k-means.png.

You have been provided with a notebook `Tutorial7.ipynb` to aid you in answering the questions below.

1. The current choice of $k = 9$ does not produce a very nice output. What is a good value for k? Justify your answer.
2. For the value of k you select in part 1, what is the space saved by doing this compression?
3. What are the drawbacks of this form of compression?

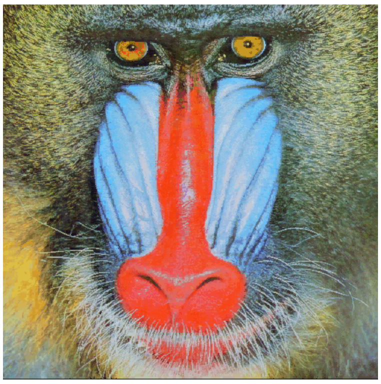

Figure 4: Image of Mandrill

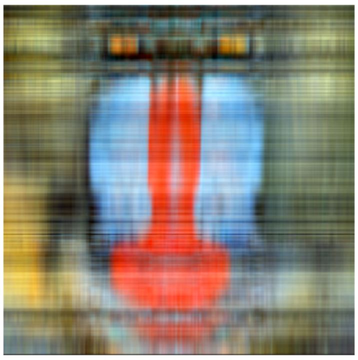

Figure 5: Image of Mandrill after k=9