Assignment 4: CS 335 & CS 337

Richeek Das : 190260036

30th October 2021

Contents

1 Clustering

1.1 CS 335 KMeans Implementation

(i) Please find the `assignment_4.ipynb` submitted.

(ii) We have implemented the K-MEANS algorithm similar to the one proposed in the lectue slides. We took our cluster center initializations as a uniform random sample of **K** datapoints, where **K** is the number of clusters. We run the algorithm till **convergence**, that is there's no further change in cluster assignments to the points (Note that this is in confidence that K-MEANS is an algorithm that always converges).

DATA 1

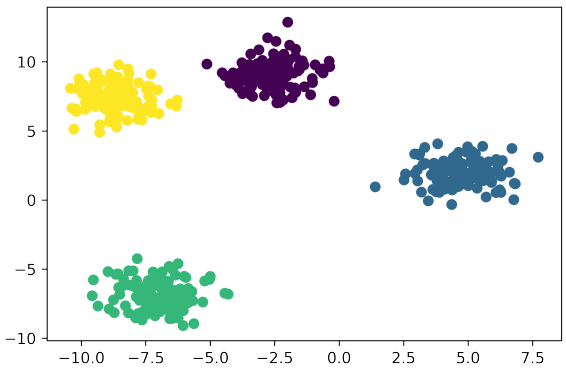
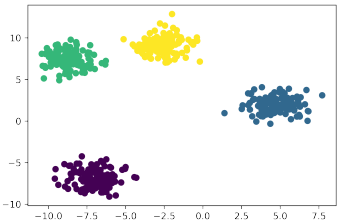
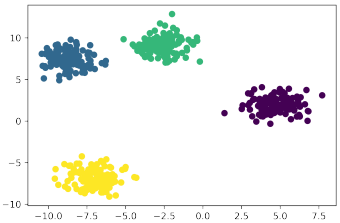


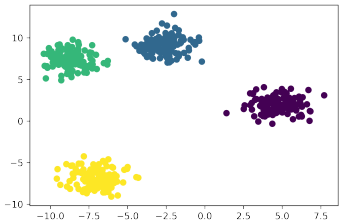
Figure 1: Original Dataset and expected cluster assignments



(a) **SEED=123**



(b) **SEED=53641**



(c) **SEED=87234**

Comments:

- For a linearly separable dataset, like the Dataset 1, we find that under good cluster centre initialisations K-MEANS achieves very good results.
- K-MEANS is highly dependent on the nature of initialisation. If the initialisation is not good, we achieve unintuitive clustering, much because it tries to minimize the plain euclidean distance based on the cluster centre initialisation and gets stuck in a local minima.
- K-MEANS is not suited for the type of clustering expected in Dataset 2 or 3. We can observe that K-MEANS is not suited for non-convex clustering. Dataset 2 and 3 are not linearly separable.
- There are no ideal "cluster centres" for dataset 2 and 3 which will return the type of clusters we want. K-MEANS is not checking the pairwise closeness of cluster points. Its

rather allotting points to the nearest cluster centre. To this end the type of clusters we receive with K-MEANS is totally expected. The clusters we receive do minimize the overall euclidean distance from their allotted cluster centres.

DATA 2

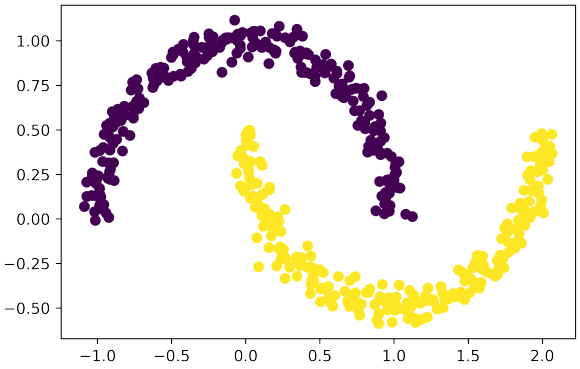
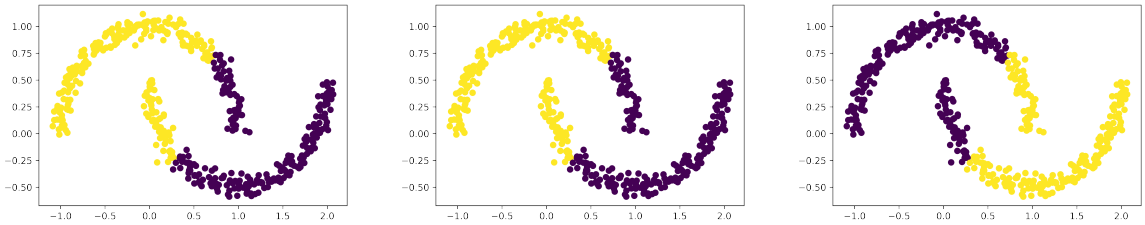


Figure 3: Original Dataset and expected cluster assignments



(a) **SEED=123** (b) **SEED=53641** (c) **SEED=87234**

DATA 3

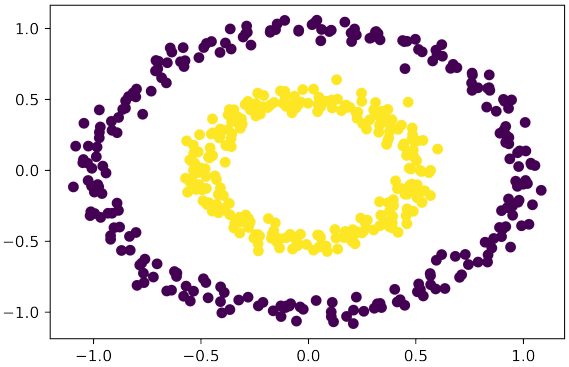
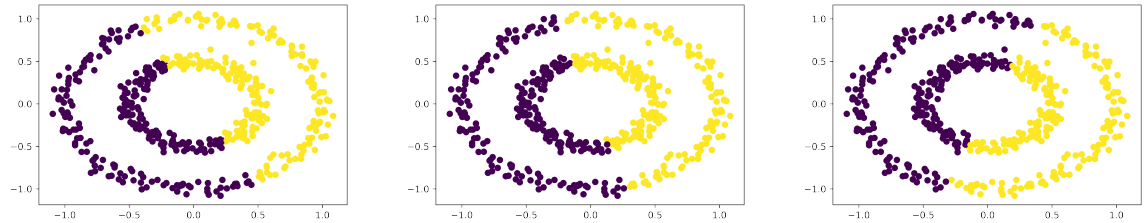


Figure 5: Original Dataset and expected cluster assignments



(a) **SEED=123** (b) **SEED=53641** (c) **SEED=87234**

(iii) Ideally we would want the cluster centre initialisations to be close to the true cluster centres. It should be at least in the same global minima bucket of the convex error function as the true cluster. But definitely we can't ensure this just by looking at the data. So we test out the following initialisation strategies:

(a). **Randomly sample K data points as the K cluster centres:**

- This is the strategy we adopt for this problem.
- It is easy to see the idea behind this. We would expect the true cluster centres to lie close to the mode of the cluster data points. So picking random data points as cluster centres seems plausible.
- If random initialisation leads to two cluster centres getting initialised with data points lying in the same cluster, it is quite possible that this initialisation strategy will perform poorly. So we also explore another algorithm.

(b). **kmeans++ :** [link to publication] [Link to a reference used to understand this topic]. The basic idea here is to choose the initial points far apart from each other (but with some differences with the Farthest Point Method). As they outline:

- We start this initialisation procedure like the previously outlined one (that is by choose random points from the data).
- We choose the following points such that its likely to lie at a large distance from its previously assigned set of points. They perform the sampling of the point from a probability distribution that is proportional to the squared distance of a point from the first centre.
- The remaining points are generated by a probability distribution that is proportional to the squared distance of each point from its closest centre. So, a point having a large distance from its closest centre is more likely to be sampled.

But this method takes up additional computational capacity to build the initial cluster centres. We choose to go for **(a)** as described above for this assignment.

2 Kernel design and Kernelized clustering

2.1 CS 337: Proving kernel validity

Prove that the function $K_\sigma : \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}$ defined as $K_\sigma(\mathbf{x}, \mathbf{y}) = \exp\left(\frac{-\|x-y\|^2}{2\sigma^2}\right)$ is a valid Kernel.

Solution.

We will use **sum, product and positive scaling closure** properties of Kernels to prove this.

Observation: *If $K(x, y)$ is a valid kernel then $\exp(K(x, y))$ is also a valid kernel.*

Proof: We can use Taylor's expansion around 0.

$$\begin{aligned}\exp(K(x,y)) &= 1 + K + \frac{K^2}{2} + \frac{K^3}{6} + \cdots \\ &= \sum_{i=0}^{\infty} \frac{K(x,y)^i}{i!}\end{aligned}$$

By product and positive scaling closure properties every term of the summation is a valid kernel, i.e. $K(x,y)^i$ is a valid kernel because $K(x,y)$ is a valid kernel. Also, $\frac{1}{i!} > 0$, so $\frac{K(x,y)^i}{i!}$ is a valid kernel.

Therefore $\exp(K(x,y))$ is a valid kernel.

Now, since $\exp(K(x,y))$ is a kernel, by **Mercer's Theorem**, $\exists$ a feature map $\phi(\mathbf{x}) : \mathbb{R}^n \to \mathrm{H}$, s.t.

$$\exp(K(x,y)) = \phi(x)^T \phi(y)$$

where, H is a Hilbert space.

Now, the given kernel:

$$\begin{aligned}\exp\left(\frac{-\|x-y\|^2}{2\sigma^2}\right) &= \exp\left(\frac{-\|x\|^2 - \|x\|^2 + 2x^T y}{2\sigma^2}\right) \\ &= \exp\left(\frac{-\|x\|^2}{2\sigma^2}\right) \times \exp\left(\frac{-\|y\|^2}{2\sigma^2}\right) \times \exp\left(\frac{x^T y}{\sigma^2}\right)\end{aligned}$$

We know that $x^T y$ is a valid kernel with an identity feature map. Given $\sigma^2 > 0$, we know $\frac{x^T y}{\sigma^2}$ is also a valid kernel. This means, $\exists\, \phi$ s.t.

$$\exp\left(\frac{x^T y}{\sigma^2}\right) = \phi(x)^T \phi(y)$$

Therefore, rewriting the kernel we get,

$$\begin{aligned}\exp\left(\frac{-\|x-y\|^2}{2\sigma^2}\right) &= \exp\left(\frac{-\|x\|^2}{2\sigma^2}\right) \times \phi(x)^T \phi(y) \times \exp\left(\frac{-\|y\|^2}{2\sigma^2}\right) \\ &= \left(\phi(x) \exp\left(\frac{-\|x\|^2}{2\sigma^2}\right)\right)^T \left(\phi(y) \exp\left(\frac{-\|y\|^2}{2\sigma^2}\right)\right)\end{aligned}$$

Therefore, we can construct the projection space of the given kernel $\phi'(x) = \phi(x) \exp\left(\frac{-\|x\|^2}{2\sigma^2}\right)$. Since a valid feature map exists, the given kernel is a valid kernel!

2.2 CS 337: Simple Kernel Design

1. *Is there a condition on r_1 , r_2 such that the vanilla KMeans (vanilla means we need to run the algorithm as is on the given data without transformations of any kind) algorithm gives us the correct clusters?*

Solution:

No, there does not exist a pair of r_1, r_2 such that vanilla KMeans algorithm gives us the correct clusters. This is because K-Means cannot cluster non-convex datasets.

Proof: We propose a proof by contradiction for the given concentric dataset case. Let 2 distinct cluster centers c_1(for r_1) and c_2 (for r_2) be such that K-means can perfectly cluster the dataset, after we are done with the *fit()* function.

Wlog assume $r_2 < r_1$. Assume $c_i < r_i$ means cluster center c_i lies inside circle with radius r_i. We consider the following **cases**:

(a) $c_1 < r_2$ and $c_2 < r_2$: If we take the line joining c_1 and c_2, it will intersect the inner circle at points p_1 and p_2. As, p_1 and p_2 belong to the cluster corresponding to c_2, their distances from c_2 should be less than their corresponding distances from c_1. Since straight line gives us the shortest distance, it is obvious that the above claim will lead to a contradiction.

(b) $r_2 < c_1 < r_1$ and $c_2 > r_1$: The line joining c_1 and c_2 intersect the outer circle. Let p_1 lies between c_1 and c_2 and p_2 lies on the opposite. Both of these points should lie closer to c_1, but again that leads to a contradiction.

(c) $r_2 < c_2 < r_1$ and $c_1 > r_1$: Follows the same idea.

(d) $c_1 > r_1$ and $c_2 > r_1$: The point on the outer circle that lies on the line joining the origin and c_2. This point is closer to c_2 than c_1, giving a contradiction.

Therefore, using exhaustive cases, we have shown that for any r_1, r_2 and corresponding c_1, c_2 vanilla K-Means won't be able to correctly cluster the given dataset of concentric circles.

2. *For the configurations of r_1 , r_2 that are not clusterable, can you suggest a kernel that will help KMeans identify the correct clusters? Specify both the transformation $\phi(x)$ and the kernel function $k(x, x')$. Further show that the kernel function you propose is a valid kernel.*

Solution:

The basic intuition for formulating a good kernel for clustering concentric circles is to utilise the similar radius of the data points lying in a particular cluster. To use the radius of the clusters for clustering, we can use $\|x\|_2^2$ as the cluster metric. We will **first propose the kernel** and derive its **transformation space $\boldsymbol{\phi(x)}$**.

We propose the following **kernel**:

$$k(x, x') = \exp\left(\gamma\left(\|x\|_2 + \|y\|_2\right)\right) \tag{1}$$

Let's derive the **transformation space** for this proposed kernel:

$$\begin{aligned} k(x, x') &= \exp\left(\gamma\left(\|x\| + \|y\|\right)\right) \\ &= \exp\left(\gamma\|x\|\right)\exp\left(\gamma\|y\|\right) \end{aligned}$$

Therefore, the **transformation space** is $\boldsymbol{\phi(x) = \exp\left(\gamma\|x\|_2\right)}$, since we assume $\|x\|$ is a scalar under the current problem settings.

$$k(x, x') = \langle\phi(x), \phi(y)\rangle = \phi(x)^T\phi(y) = \exp\left(\gamma\|x\|\right)\exp\left(\gamma\|y\|\right)$$

since the transformation is from $\mathbb{R}^2 \longrightarrow \mathbb{R}$

Proof - Proposed kernel is a Mercer Kernel (hence valid):

By theorem, a kernel $K(x, y)$ is a Mercer kernel if $\int_x \int_y K(x, y)g(x)g(y)dxdy \geq 0$ for all square integrable functions $g(x)$.

Therefore,

$$\begin{aligned} \int_x\int_y K(x, y)g(x)g(y)dxdy &= \int_x\int_y \exp\left(\gamma\left(\|x\|_2 + \|y\|_2\right)\right)g(x)g(y)dxdy \\ &= \int_x\int_y \exp\left(\gamma\|x\|\right)\exp\left(\gamma\|y\|\right)g(x)g(y)dxdy \end{aligned}$$

Using Fubini's Theorem we can write the iterated integral as a product of integrals:

$$\int_x \int_y \exp(\gamma\|x\|) \exp(\gamma\|y\|) g(x)g(y)dxdy = \int_x \exp(\gamma\|x\|) g(x)dx \int_y \exp(\gamma\|y\|) g(y)dy$$
$$= \left(\int_x \exp(\gamma\|x\|) g(x)dx\right)^2 \geq 0$$

Therefore our proposed kernel is a Mercer Kernel and valid.

Results with this kernel:

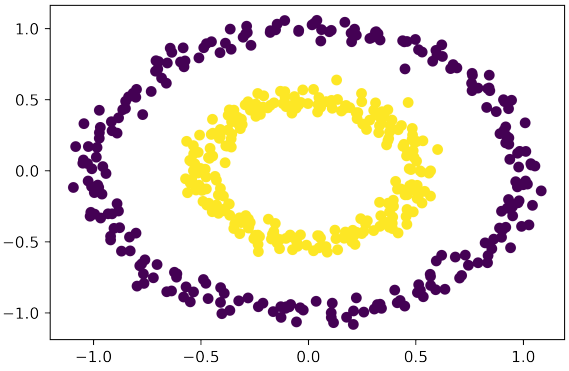
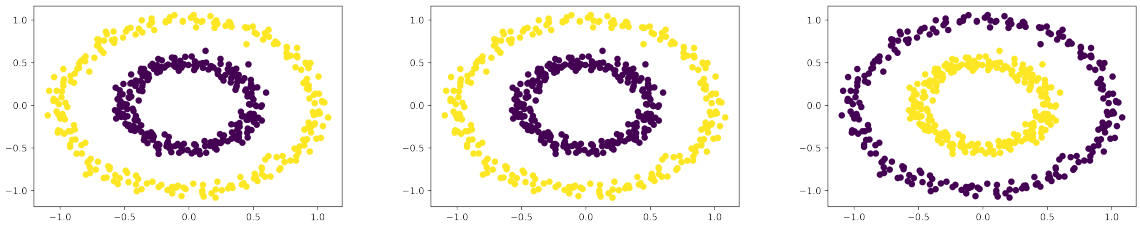


Figure 7: Original Dataset and expected cluster assignments



(a) **SEED=123** (b) **SEED=53641** (c) **SEED=87234**

Figure 8: Kernel K-MEANS Clustered Dataset 3

In the fit function we find and store the radius of the clusters as the *cluster_centers*. In the predict function we calculate the squared distance between the cluster centers and norm of the data points. We allot the data point to the nearest cluster.