CS5691: Pattern recognition and machine learning

Assignment 1

Name: Chetan Reddy N

Roll Number: ME19B093

Question 1

Q1 (i):

- The PCA code is implemented in the function `get_pc()`.
- The `show_scatter_PC()` function displays the calculated principle components.
- The variance explained by a particular principal component is essentially the associated eigenvalue (when data is centered) as proved in the class. Therefore, the following results were obtained:
    - Variance explained by principal component 1: **17.1319 (54.18%)**
    - Variance explained by principal component 2: **14.4896 (45.82%)**

Q1 (ii):

- Centering is essentially for a proper implementation of PCA.
- The given data already appears to be centered to a precision of 10^{-7}.
- On centering it further, the mean was made to be upto a precision of 10^{-17}. There were barely any noticeable changes. The reconstruction loss when considering only 1 principal component:
    - L2 reconstruction loss (1 principal component) for Centered Data: **2.9030077367**
    - L2 reconstruction loss (1 principal component) for Non Centered Data: **2.9030077417**
    - A very slight increase in error can be observed for Non-Centered Data.

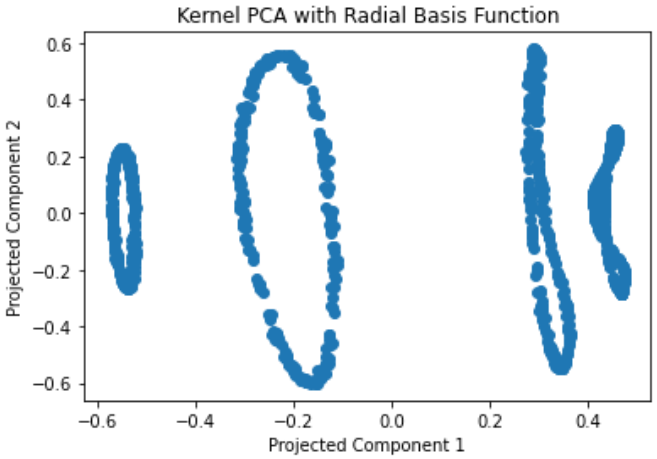
- To highlight the importance of centering, the dataset was deliberately decentred by adding a value. On doing this, the following was observed:
    - L2 reconstruction loss (1 principal component) for Centered Data: **2.9030077367**
    - L2 reconstruction loss (1 principal component) for Non Centered Data: **2.9362356470**
    - The error increases by 0.03 for non-centered data

Q1 (iii):

- Kernel PCA was implemented using the given kernels. The functions `get_kernel_components_poly()` and `get_kernel_components_rbf()` give the top 2 projected components. The results for kernel A (d=2, d=3) and kernel B (σ = 0.1,0.2…1) are visualised

Q1 (iv):

- Kernel PCA with **kernel B i.e a radial basis function (rbf)** seems to be the better choice for this dataset because the projected components are linearly seperable which will be useful for downstream tasks (like clustering as will be seen in Q2). In particular, kernel B with sigma = 3 has 4 linearly seperable regions (probably corresponding to the 4 rings of data).



Question 2

Q2 (i):

- The `run_llyods()` implements the lloyds algorithm. We can optionally pass an initialisation for the means. The results have been visualised for 5 random initialisations and number of clusters = 4. Three plots have been included:
    1. Data with initialised Means
    2. Clustered Data with the final position for the means
    3. Error vs Iterations

Q2 (ii):

- Fixing the initialisation, the number of clusters are varied and the voronoi regions for each have been visualised and depicted in the colab file. Two plots are included:
    1. Data with Initialised Means
    2. Clustered data with the voronoi region boundaries

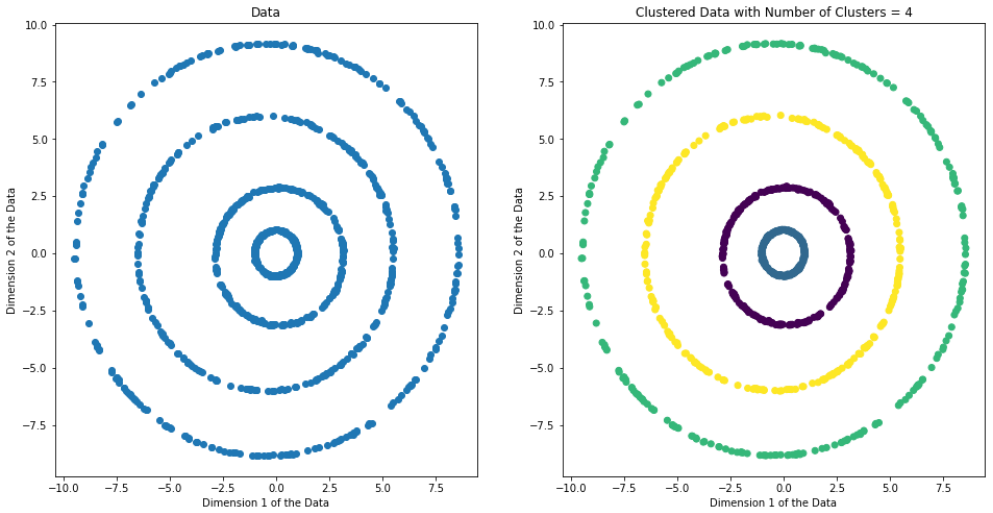
Q2 (iii):

- Spectral clustering was implemented with various choices of kernels, however, the natural clustering of 4 rings was very difficult to achieve. Although H^* had linearly seperable regions, lloyds algorithm was not completely successful on it. The results are shown in the colab notebook. Radial Basis Function Kernel is used because the data is linearly seperable into 4 regions using it.

- Something to notice is that spectral clustering and kernel PCA are very similar. In kernel PCA, the top few components are considered for the downstream tasks. Spectral clustering can be considered as that downstream task where the number of components is the number of clusters (k).

- A slightly modified way of clustering was implemented. Instead of perform lloyds algorithm on the k-dim data in H*, it was observed that the 1D data from kernel PCA (by considering the top 1 component) was linearly seperable and easily clusterable. Therefore, the following was done:
    1. Perform kernel PCA on X
    2. Consider the top 1 component for each datapoint and consider this as the new dataset.
    3. Perform lloyd's algorithm on this 1D data and obtain Z (assignment) vector
    4. Visualise the results

- The modified method was found to be a perfect version for the given dataset and 4 clusters (rings) were obtained using the RBF kernel (sigma = 3)



Q2 (iv):

- The mapping given by $l = \arg\max_{j=1,2,\ldots k} v^j_i$ generates the assignment vector directly from H*. The results were visualised and it was seen that it was not as good as spectral clustering.