**CS6301.010.001: HW4 | submitted by Siva Saket Sripada (2021432772)**

**Dimensionality reduction and Spectral Clustering:**

Q2)

***Spectral clustering core code***

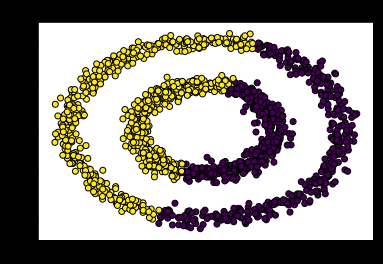
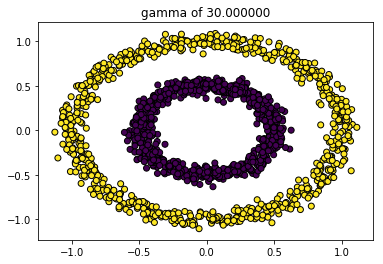
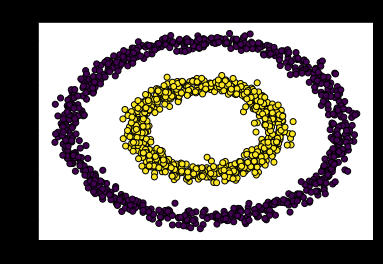
def spectacular\_spectral\_clustering (array,gamma,n\_clusters=2):

A = rbf\_kernel(array=X,gamma=g)

D = [np.diag(np.sum(A[i])) for i in range (np.shape(X)[0])]

eigval, eigvec = scipy.linalg.eigh((D - A),eigvals=[0,1])

return KMeans(n\_clusters=2,random\_state=0).fit\_predict(eigvec)



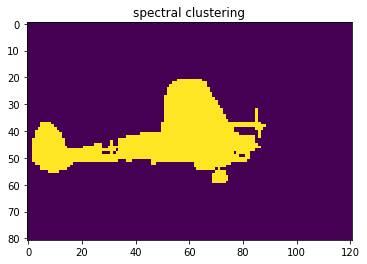
k-means original data spectral\_clustering

**Code for Image-segmentation:**

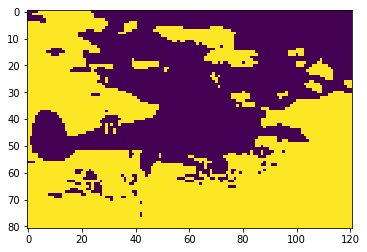
Imgflat = cv2.imread('seg.jpg',0).flatten().reshape(-1,1)

S\_segimg = np.reshape((spectacular\_spectral\_clustering(imgflat,10)),(81,121))

K\_segimg = np.reshape((KMeans(n\_clusters=2).fit\_predict(imgflat)),(81,121))

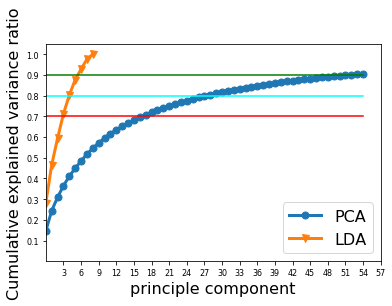
plt.imshow(K\_segimg)

plt.imshow(S\_segimg)



Native k-means clustering

Q1) PCA,LDA and digit recognition



For the USPS data, the k reduced dimensions (k eigenvectors) covering 70, 80 and 90% of total variance:

k70-> 17     k80-> 28     k90-> 52

Val\_Accuracies for different pairs of *α,kf*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Gamma | 0.0001 | 0.001 | 0.01 | 0.1 | 1 |
| K70 | 0.8395 | 0.8595 | 0.8829 | 0.8629 | 0.7659 |
| K80 | 0.8863 | 0.8796 | 0.903 | 0.9064 | 0.806 |
| K90 | 0.9064 | 0.8796 | 0.9197 | 0.9097 | 0.8495 |
| K100 | 0.9264 | 0.8997 | 0.9264 | 0.916 | 0.8361 |

**Without feature selection:**

Based on least validation error, the best pair is  (0.01, 100) with Test\_accuracy of 91.64 %

**With feature selection (post dimensionality reduction)**:

Based on least validation error the best pair is  (0.1, 90) with Test\_accuracy 89.97 %

**Without feature selection, using SVC with default parametrs**:

Based on the same one-vs-all multiclass classification as in SGD, accuracy on usps\_tst is 93.98 %

The eigen-digits are printed in the following page:

Code is below:

from matplotlib.colors import ListedColormap

cmap = ListedColormap(['#b30065', '#178000'])

fig,axes=plt.subplots(4,4,figsize=(15, 20))

for i in range(16):

r, c = np.divmod(i, 4)

axes[r,c].imshow(np.reshape(pca.components\_[i],(16,16)))

axes[r,c].set\_title('eigen-digit='+str(1+i))

