Xin Wen

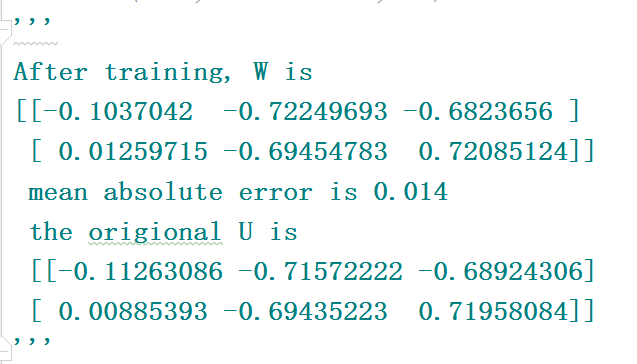
Note:

1. The python code is based on Python 3, Numpy, Sklearn, Tensorflow, matplotlib

2. in the python code, there is a customized function defined as NN(train, label, iteration). This function is used in NN PCA and NN LDA. Because they are in the same class, they can not be called twice, which leads to variable reusing error in Tensorflow. So I saved the output and comment out the running command at line 75 and 112. You can enable any of it to check the NN running result.

1. see code
2. see attached plot Q2
3. [[-0.11263086 -0.71572222 -0.68924306]

[ 0.00885393 -0.69435223 0.71958084]]

1. see attached plot Q4
2. we already know that when projecting from 3D to 2D, we need a matrix of 3 by 2. So I set up a simple 1 layer NN.
3. see code
4. 
5. see attached plot Q8
6. From the plot, the difference is hard to be noticed by human eyes unlike standard PCA, NN PCA is based on learning from training sample, which is generated by standard PCA. Because we already know that the standard PCA is in the shape of (3,2), so we only generated an one-layer NN, and generated good result
7. the NN LDA W is

[[ 0.89537019 0.19827998 0.1902625 ]

[-0.07452867 -0.67853248 0.70091814]]

original W is

[[ 1.29872254 5.22240153 5.02100442]

[-0.04185976 -4.8763595 4.88727284]]

see attached plot Q10

compare to PCA, the performance of NN LDA is not as good as NN PCA, mean error = 0.87 vs 0.014. It’s noticed that LDA is based on the classification label [0,1,2,3] of sample data. However, this is not used in the NN training. So the additional distance between different labels did not benefit NN training.

