BLG 454E Learning From Data (Spring 2018)

Homework III

1 Question 1

1.1 Part a

One of the most importing step to analyze the data is visualizing. So correct visualization can highlight the problems that we encounter. So we can use some techniques to visualize the data. Also another motivation is to reduce the computation that we will do since by reducing dimensions we get faster results. The accuracy can be bad when we reduced the dimension but it can be in acceptance treshold. Either of them it's crucial thing to do.

1.2 Part b

Lets say we have w values that is in the dimension of n where $d < n$ so our $x = x_1 * v_1 + x_2 * v_2 + ... + x_n * v_n$ When we reduce the dimension to the d we get $y = x_1 * v_1 + x_2 * v_2 + ... + x_d * v_d$ for measure the error we can make $|| x - y ||^2$ So this will give us a basic error measurement.

1.3 Part c

The left figure shows that the datas that uses 2 features and it is easy to see that linearly seperable. But after applying PCA and reduce the dimension to the 1 dimension. We cannot seperate the class points so applying PCA is not a good choice for this data. So performance is bad for 2d to 1d on this data.

1.4 Part d

PCA relies to linear assumptions, since it is focused to find orthogonal projections of the data that contains highest variance. So if data is not linearly correlated PCA wouldn't handle this kind of data.

1.5 Part e

In order to apply PCA i followed these steps. First calculated covarince matrix with numpy.cov then by using this covariance matrix i calculated eigen values and eigen vectors. Since we have 64 features, eigen vector matrix shape is (64,64) and our aim is to choose features that affects higher in dataset. So i choose first 2 eigen vectors for 2 features. After calculation for showing the class of the data, i plotted the random 200 points as you can see below.

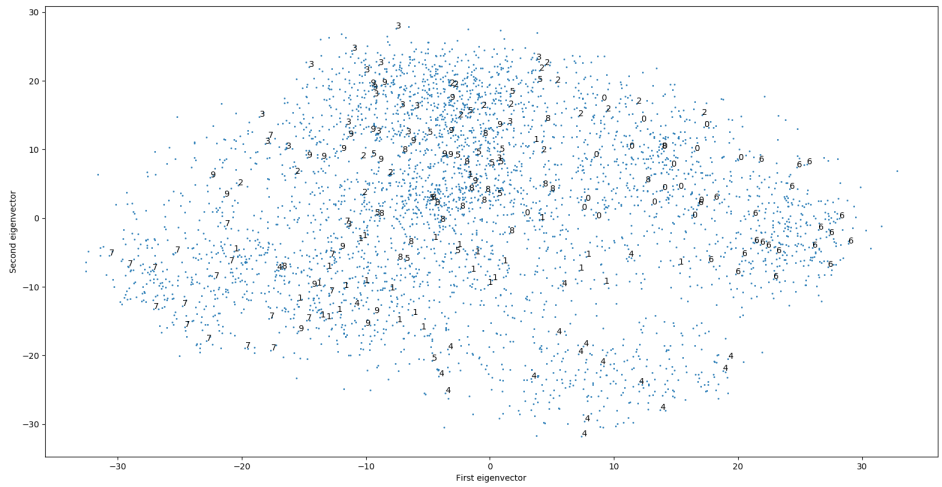
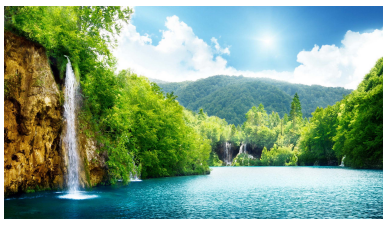


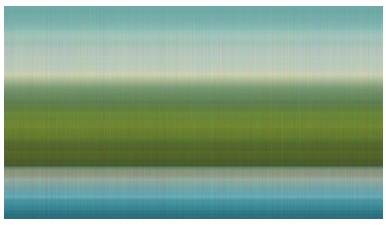
Figure 1: PCA result of the python code

2 Question 2

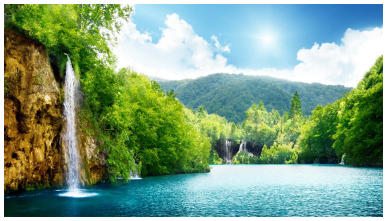
For reading image i used scipy.imread and after reading image i split the image into the red,green and blue channels. Then i calculate $A^T * A$ and then i calculate the eigen vectors and eigen values of this matrix. I used linalg.eig function because calculating manually would be so hard and inefficient. The eigen values that i get are not sorted so i sort the eigen values since we would like to find the values that has impact on the image most. Then the result of the $A * V * S^{-1}$ we get U matrix and so by this we have U,S and V matrices that gives A matrice when calculating $U * S * V^T$. There are different methods for calculating the SVD but the methodology that i used was like this. You can see the images for different ranks and original image.



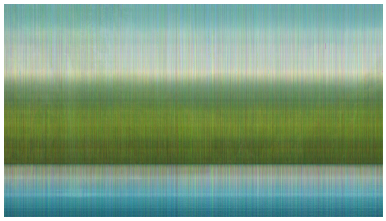
(a) Original Image



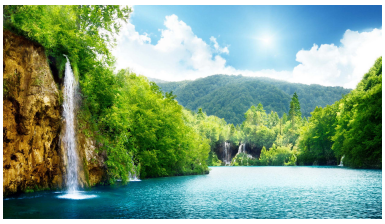
(b) Using 1 term



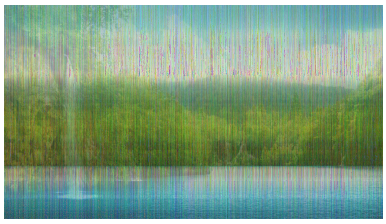
(c) Original Image



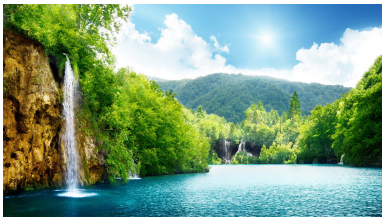
(d) Using 5 term



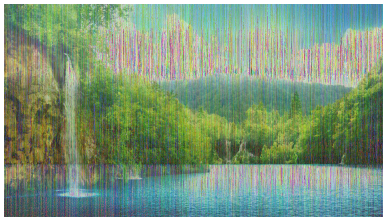
(e) Original Image



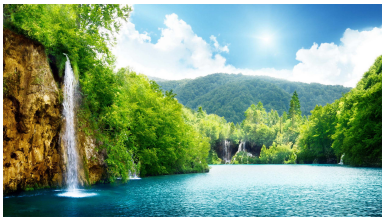
(f) Using 20 term



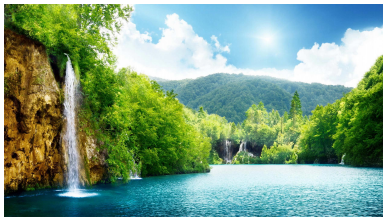
(g) Original Image



(h) Using 50 term



(i) Original Image



(j) Using 100 term

Figure 2: The original image and its the compressed results are displayed.