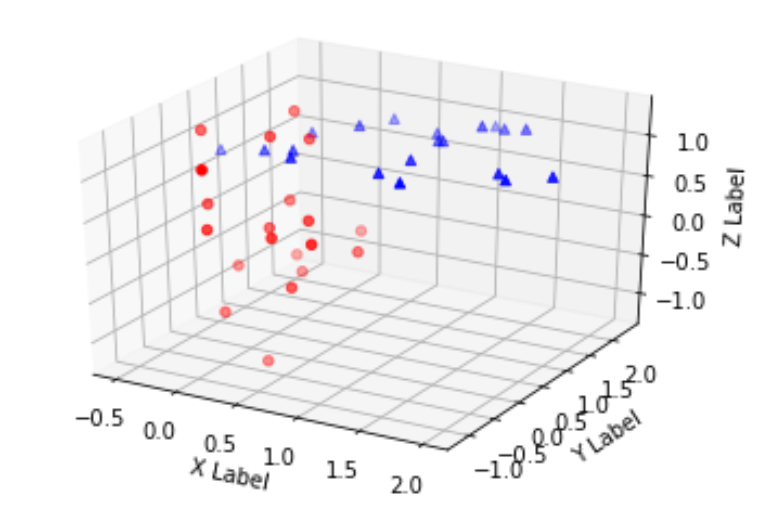
Problem 1

1. See code.

2. The scatter plot does not look like anything, and it is hard to tell which data point is where



3. I used the covariance formula to calculate the value of each entry of the covariance matrix. The result is

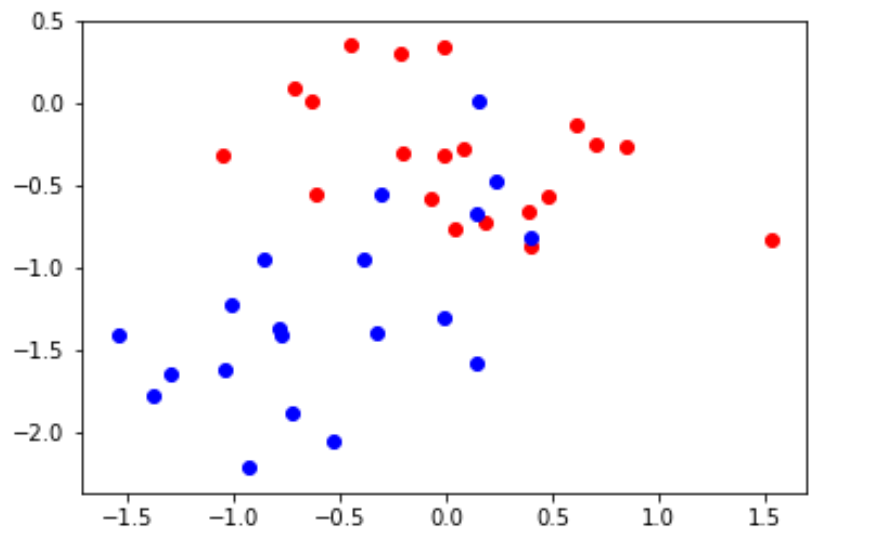
[[ 0.36212565 0.21606788 0.0938059 ]

[ 0.21606788 0.74300616 0.23599682]

[ 0.0938059 0.23599682 0.38017863]]

Which is the same as the result I got from np.cov()

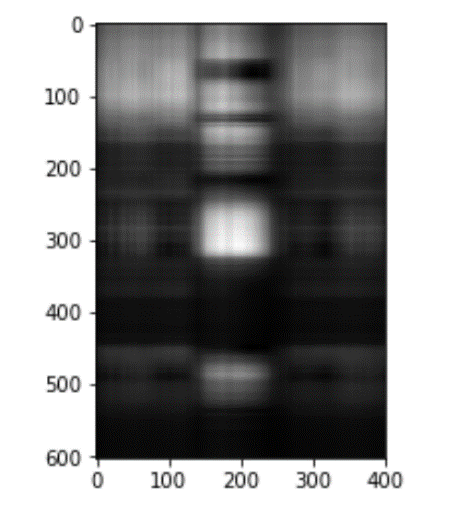
4.



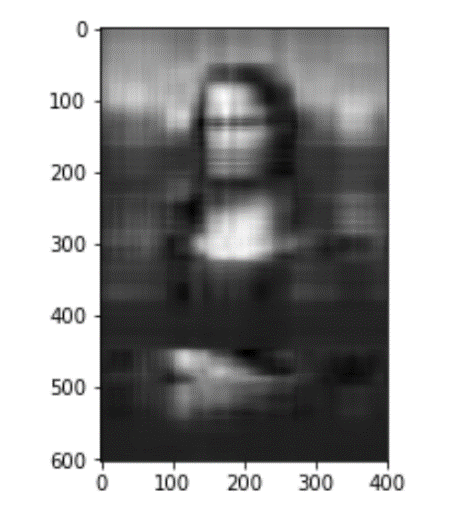
We now have a 2d image (the plane we got from the 2 vectors) The PCA makes it much easier to distinguish between the two sets of data.

Problem 2

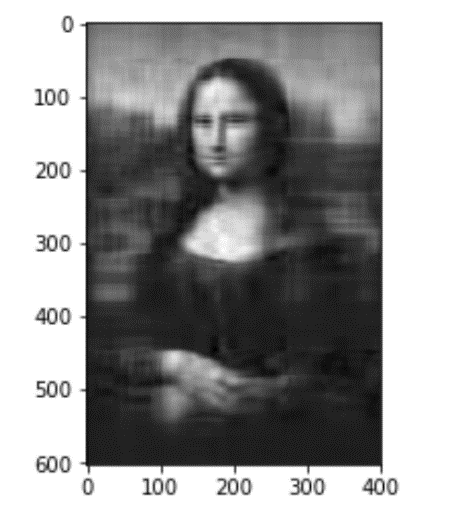
K = 2



K = 5



K = 10



2. With the svd, what we get is three things that can be used to reconstruct the image: u, s and v.

u is a matrix of shape 603 x 400; s is an array of length 400, and s is of shape 400 x 400.

If we zero pad ranks after, say, the top 2 ranks, we will need 603\*2 + 2 + 400 \*2 floating point numbers to reconstruct the image. Each floating point number in python is 64 bits (sometimes 32 bits), so we will need (603\*2 + 2 +400\*2) \* 64 bits to contain all information needed to reconstruct the image.

Same goes for k = 5 and k = 10

Assuming the floating point numbers we are using are 64 bits, then

For k = 2, 128512 bits

For k = 5, 321280 bits

For k = 10, 642560 bits

For the original image, 2 bytes \* 603 \* 400 = 3859200 bits