UNIVERSITY OF COPENHAGEN

FACULTY OF SCIENCE

INTRODUCTION TO DATA SCIENCE

Assignment 4

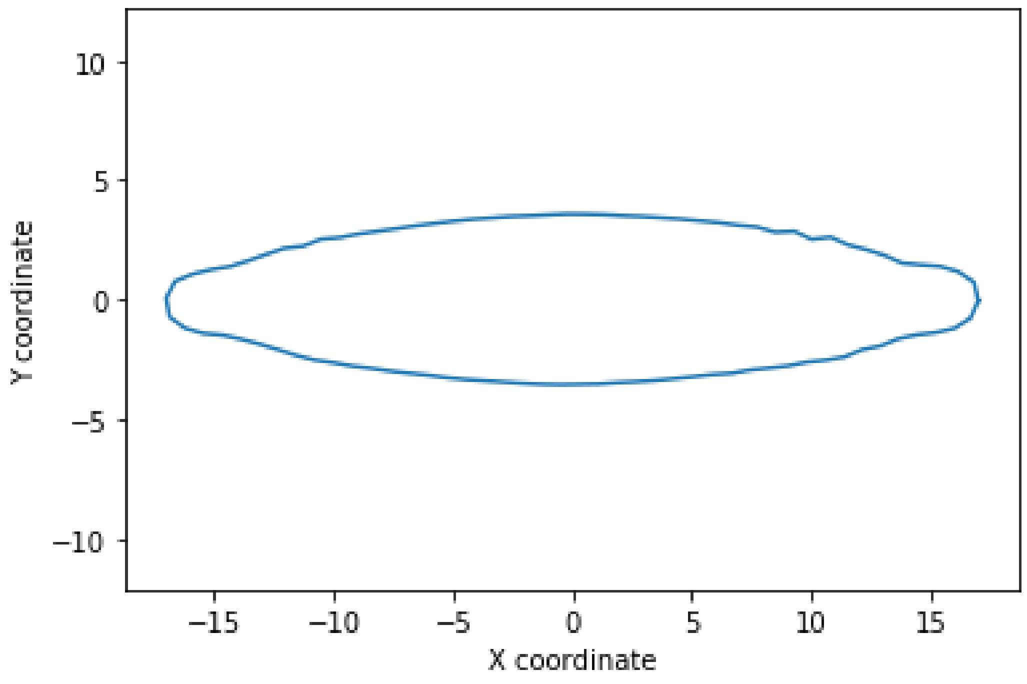
Dimitrios Galinos (`bst265`)

March 13, 2020

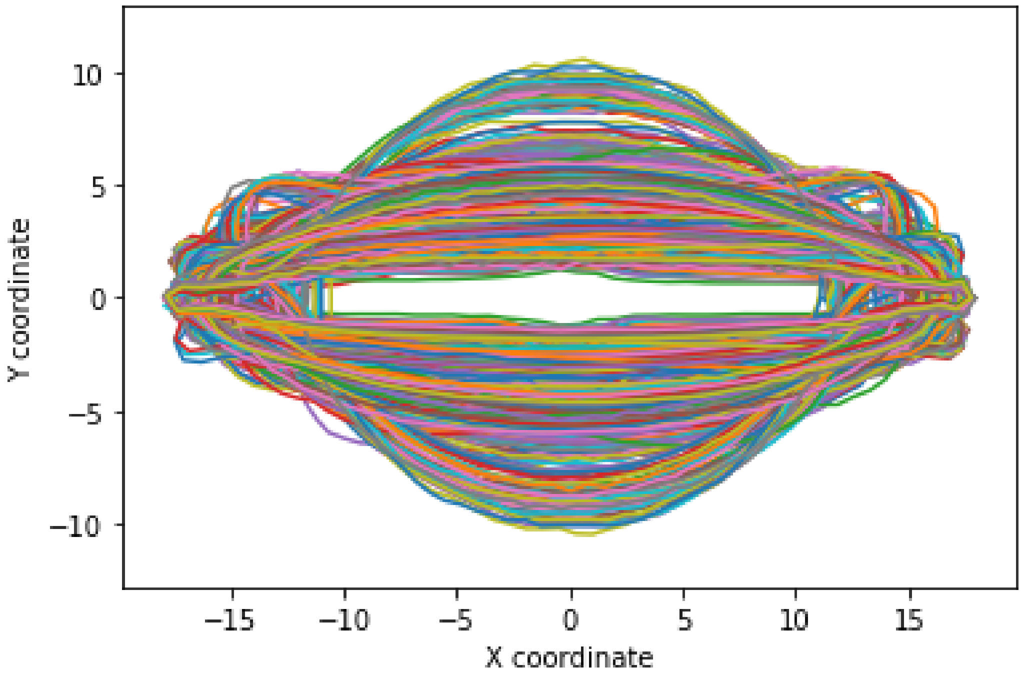
1 Plotting cell shapes

The code is included inside the dimitrios.galinos.zip and is named "assignment4.py" for the main part of the execution while also having the "mds.py" for the multidimensional scaling and the "pca.py" for my own PCA implementation. Note that I have used in all the exercises my own implementation of PCA and have also checked the results with scklearn's PCA in order for them to be the same (or have opposite +/- signs). Please also read the README file.

So, the plot of the first cell can be found bellow, I divided the coordinates to X and Y, connected the first with the last point in order to be able to plot it successfully. I also used plt.axis('equal') as requested.



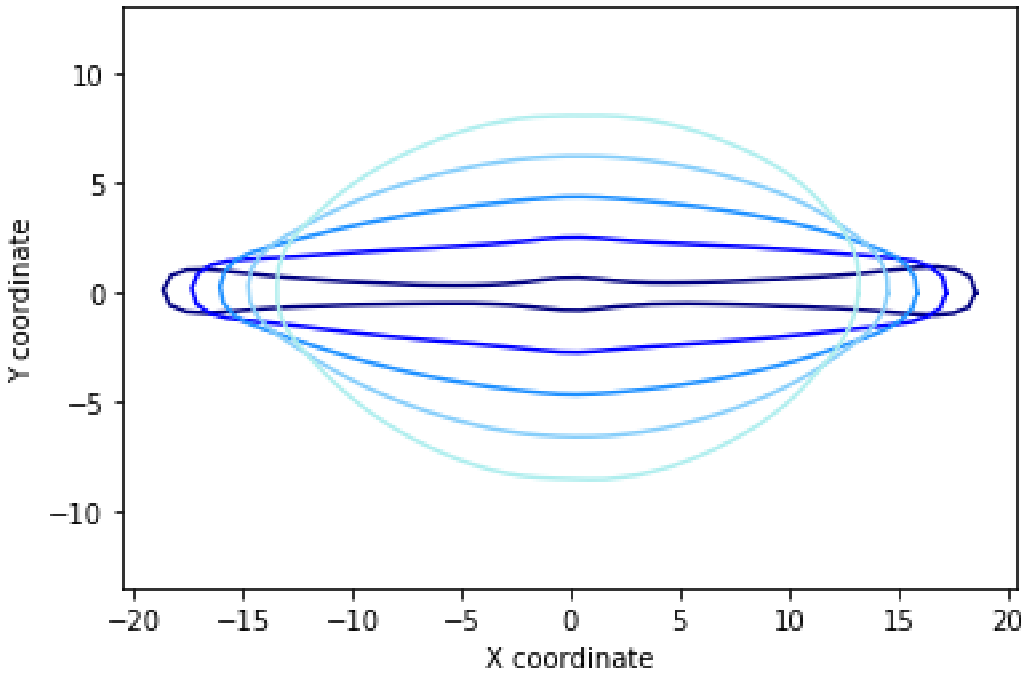
For the next plot I did the same but also included inside the plot every single cell instead of just the first one:



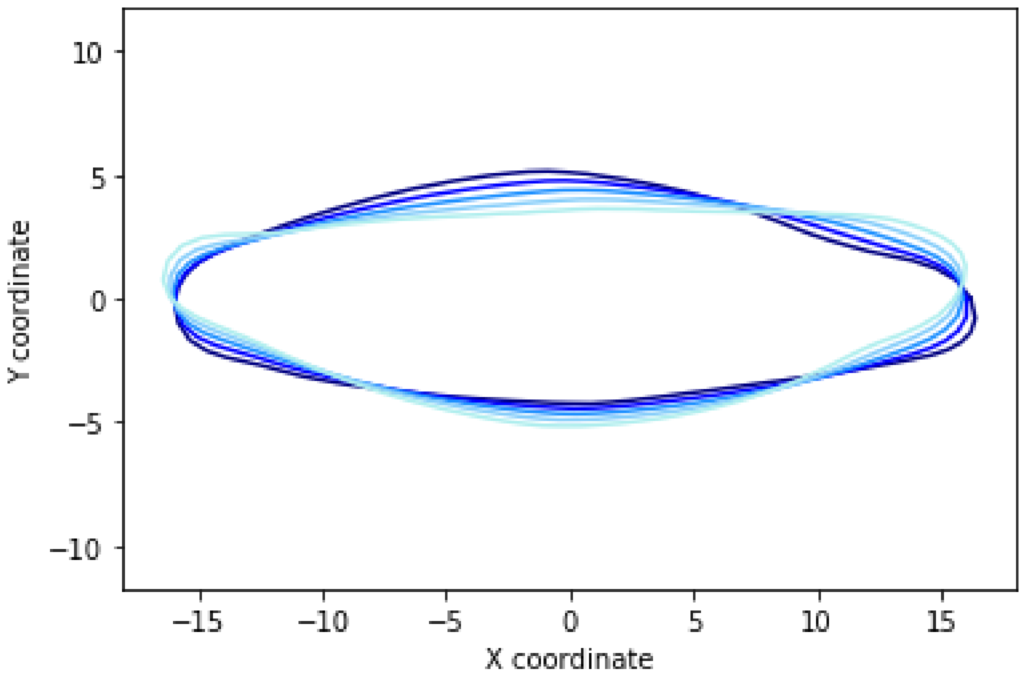
2 Visualizing variance in visual data

I implemented everything as requested and created plots with 5 cells which are some instances of the first three PCs. I also colored those instances starting from dark blue and ending in light-cyan for easier interpretation as requested.

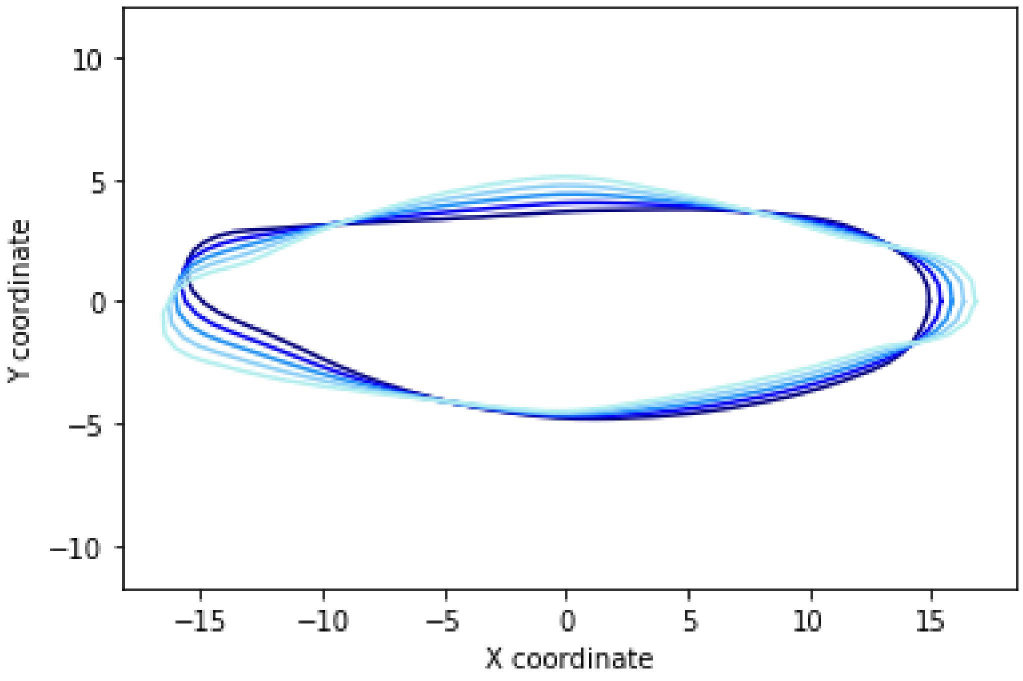
First Principal component instances:



Second Principal component instances:



Third Principal component instances:



From the plots we can deduct that the first principal component actually captures a very big percentage of the variance because its instances cover almost all the instances that can be seen in the plot of the previous exercise where I had to visualize every single cell. The following two principal components capture some small part of the variance which is still important but it is way way smaller than what the first PC actually captures. In general I believe it is safe to say that when using PCA the first principal component captures a very big chunk of the variance and without it we are not gonna be able to perform a meaningful multidimensional scaling.

3 Critical thinking

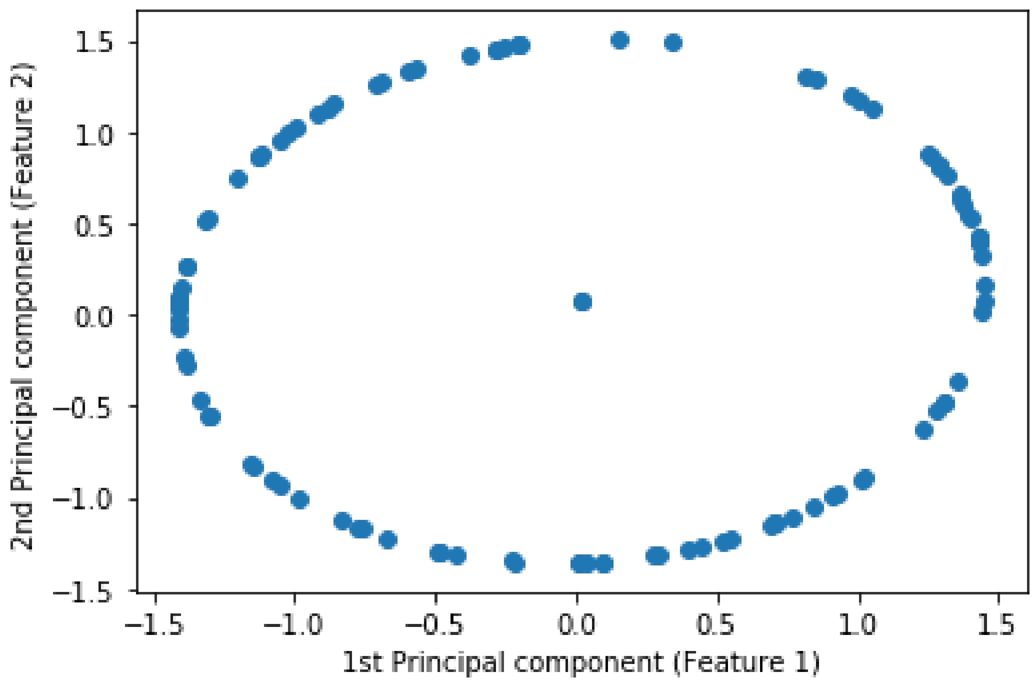
a) First of all, it is important to mention that the way PCA works it is affected by scale. Meaning if we have bigger numbers in some component due to a different scale measure, or due to the fact that there are just more of these, PCA will maximize that component's variance way more, even if that particular component doesn't actually capture the biggest part of the variance of the whole dataset.

- **Centering**: Centering simple centers the data to zero mean. It will help any visualizations look a bit better because everything will be centered around zero but it won't actually help PCA into efficiently maximizing the variances.

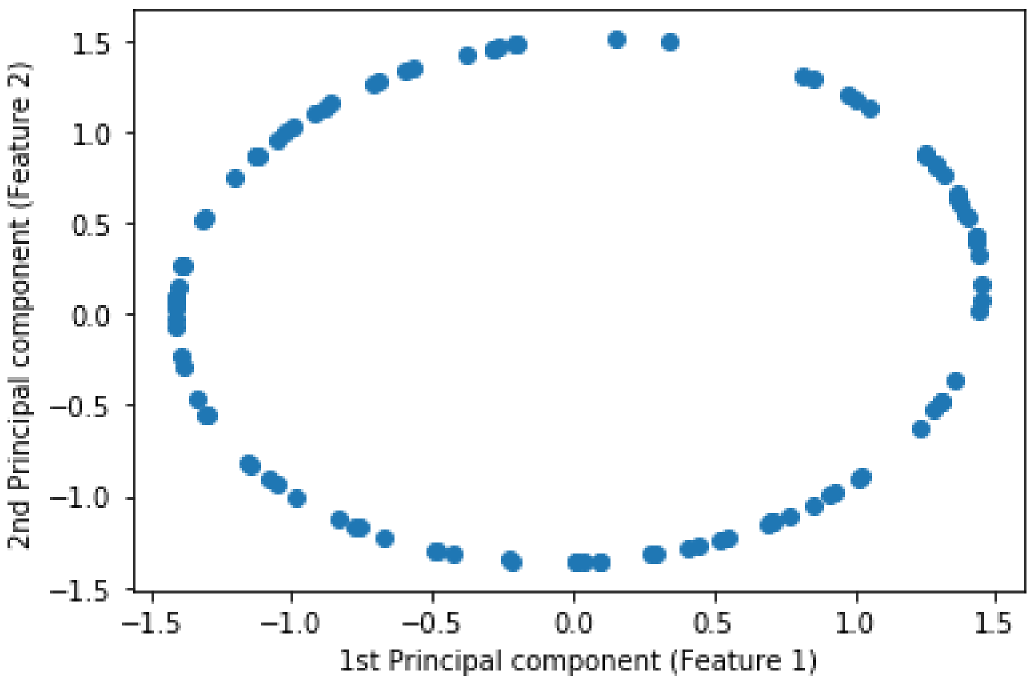
- **Standardization**: Standardizing is the best thing we can do before applying PCA because it normalizes the data to unit variance and zero mean. That means that in a visualization we will have the benefits of centering but moreover the PCA will efficiently maximize the variances of each component because with the transformation of the data into unit variance we actually remove the scaling problem I mentioned above.
- **Whitening**: Gives us unit variance which is very helpful when applying PCA in order to avoid the scaling problem but the lack of centering in this preprocessing technique makes it inferior to Standardization.

Overall, the best idea is to apply Standardization, the second best is Whitening and the worse is Centering. Standardization gives us the benefits of both the other two methods of preprocessing while the other two lack something.

b) I ran PCA on the toy dataset, after I first standardized it and visualized the projection onto the first 2 PCs getting this plot:



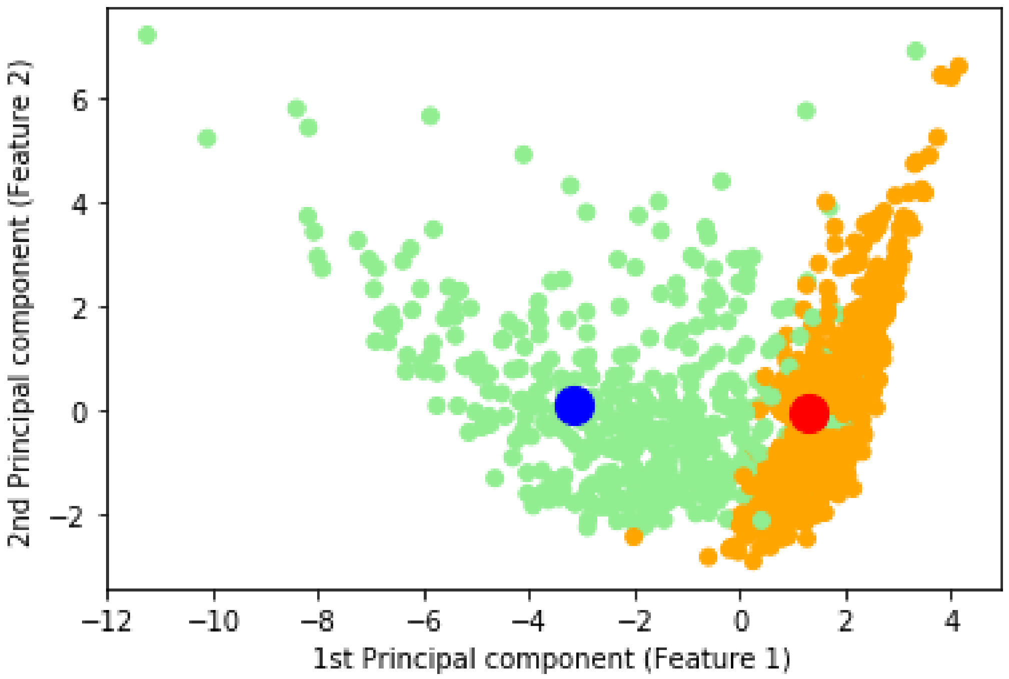
I then removed the last two datapoints and got the following plot:



After some digging around in the original toy dataset I found the last two datapoints which both were [0,0,0,0] and I was expecting that their 2D projection would be somewhere close to [0,0]. So the removal of those two actually removing the dot in the center of the circle was something we could expect.

4 Clustering II

I performed a 2-means clustering exactly as requested. I used sklearn.cluster.KMeans, I initialized the two starting points to be the first two data points of XTrain they way it is shown in the exercise text, run the KMeans algorithm with the following parameters: n_clusters=2 (2 cluster centers), n_init=1 (running only 1 trial), init=start (my starting point of the first two data points of XTrain) and algorithm=full in order to run the classical EM-Style algorithm. Furthermore I visualized the XTrain dataset by projecting it onto its first principal components coloring the class 0 with light green and the class 1 with orange. I then took the two cluster centers I found and projected them as well onto the first two principal components found in the previous step and visualized them together with the data points using blue and red colors.



The clusters appear to be meaningful because the cluster centers look pretty close to where the real cluster centers should be according to the classes of the 2D projection of the dataset.