Homework Report #2: Dimensionality Reduction Techniques

Author: Isaac Han

Email: cogitoergosum01001@gmail.com

Class: Data Analysis: Statistical Modeling and Computation in Ilications

Professors: Prof. Uhler, Prof. Jegelka

Table of Contents

1: Data Preparation

The Single-Cell RNA-seq data:

- each row corresponds to a cell
- each column corresponds to the normalized transcript compatibility count (TCC) of an equivalence class of short RNA sequences, rescaled to units of counts per million. You can think of the TCC entry at location of the data matrix **as the level of expression of the -th gene in the -th cell**.

Data Properties

1. `p1`, which is a small, labeled subset of the data. It contains the count matrix along with “ground truth" clustering labels , which were obtained by scientists using domain knowledge and statistical testing. This is for use in Problem 1.
2. `p2_unsupervised`, which contains only a count matrix. This is for use in Problem 2.
3. `p2_evaluation`, which contains a labeled training and test set. This is for use in Problem 2 to evaluate feature selection.

The `p2_unsupervised_reduced` and `p2_evaluation_reduced` folders contain datasets with a reduced number of genes, in case you are unable to run some of the procedures on the larger versions. In particular, a full logistic regression could take 1 or 2 GB of memory to run.

In Problem 1 (autograded), you will explore a small subset of the data, using visualization and clustering methods to discover its structure.

In Problem 2 (written report/peer review), you will use the tools you had from Problem 1 to explore a larger subset of the data. Using clustering combined with logistic regression, you will discover informative features which can be used to distinguish cells of different types.

Finally, in Problem 3 (written report/peer review), you will revisit open-ended decisions you made in your analyses, such as T-SNE hyper-parameters or number of clusters chosen, and explore how robust your end results are to these potentially ambiguous decisions.

2: Importing Data

Treating Tar file and npy format

In order to make npy file compatible with MATLAB following treatment in Python was needed:

```
untar(filename) #It is a matlab command



from scipy.io import savemat # it is a python command
import numpy as np
import glob
import os


#%%
# Import the os module
import os

# Print the current working directory
print("Current working directory: {0}".format(os.getcwd()))

# Change the current working directory
os.chdir(r'C:\Users\Isaac_Han\Desktop\CS\MIT_Data_Science\Homework2\data\p1')

# Print the current working directory
print("Current working directory: {0}".format(os.getcwd()))
#%%

npzFiles = glob.glob("*.npy")
for f in npzFiles:
    fm = os.path.splitext(f)[0]+'.mat'
    X = np.load(f)
    savemat(fm, {'X': X})
    print('generated ', fm, 'from', f)
```

---Written Report Starts Here---:

6: Data Preparation

P2_unsupervised

Making np compatible with MATLAB

As I chose to use MATLAB for the course, I needed to preprocess npy file to be compatible with MATLAB. If you are interested in knowing what I have done, please click the hyperlink.

Loading Data into MATLAB

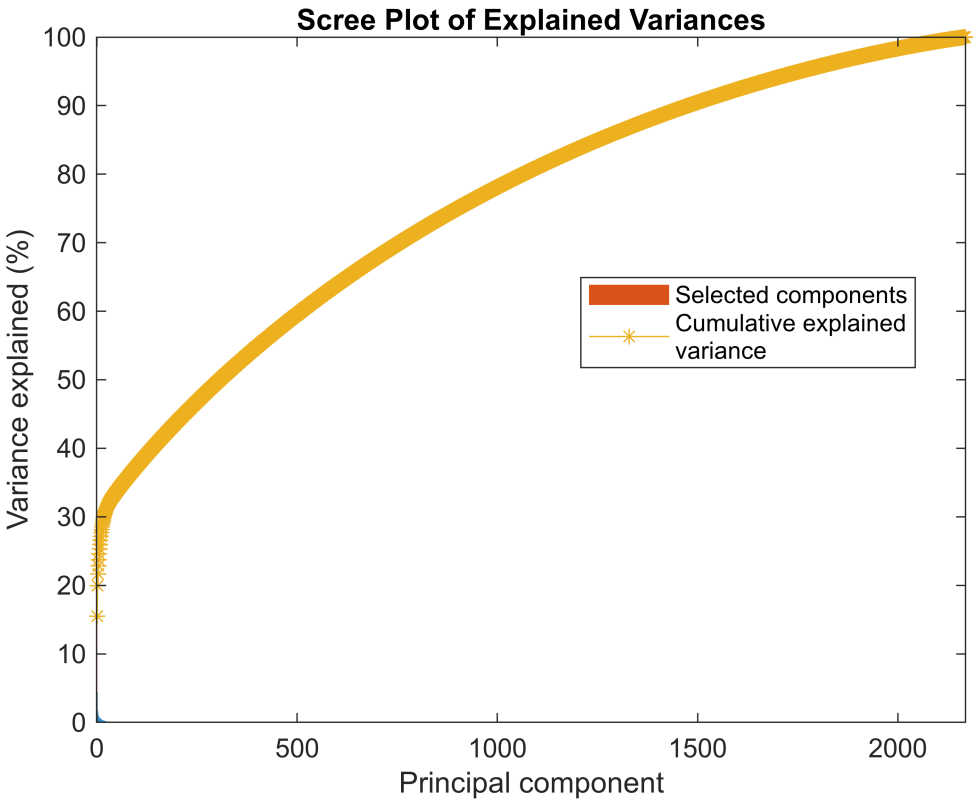
Data Transformation: log2(x+1)

The original high dimensional data is transform by the following mapping;

$$x \rightarrow \log_2(x+1)$$

8: Q2-1 Written Portion

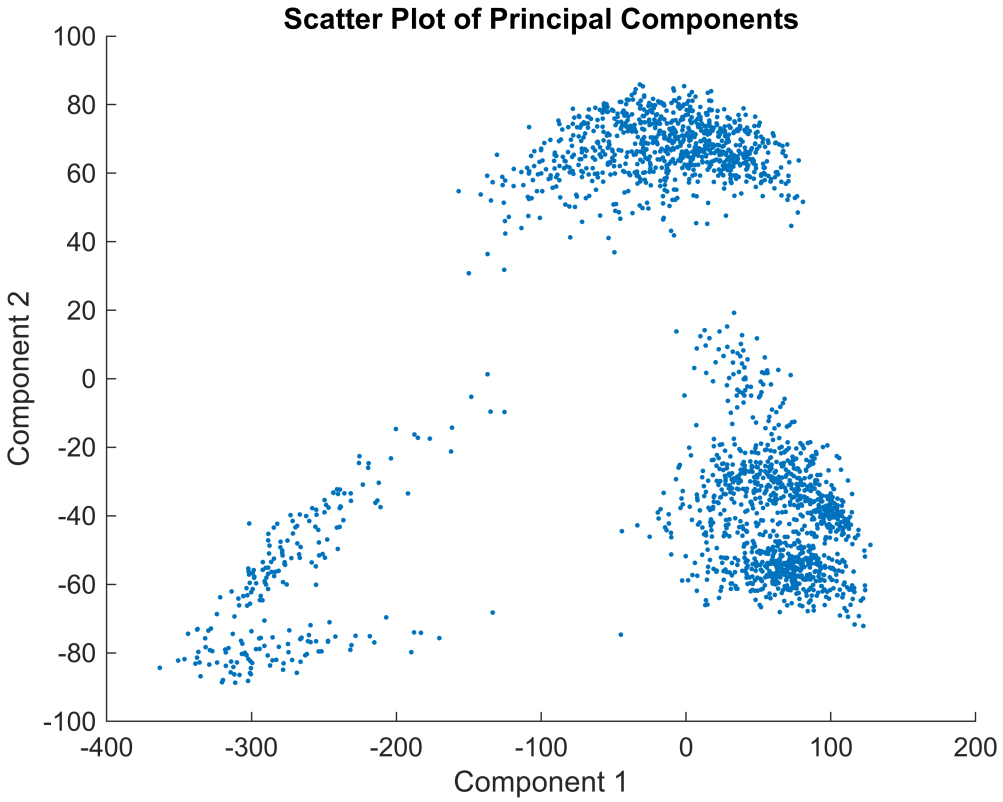
From Scree Plot, it was found that 1253 principal axes are needed to explain at least 85% variance of data projected on principle axes.




```
The number of components needed to explain at least 85% of the variance is 1253
```

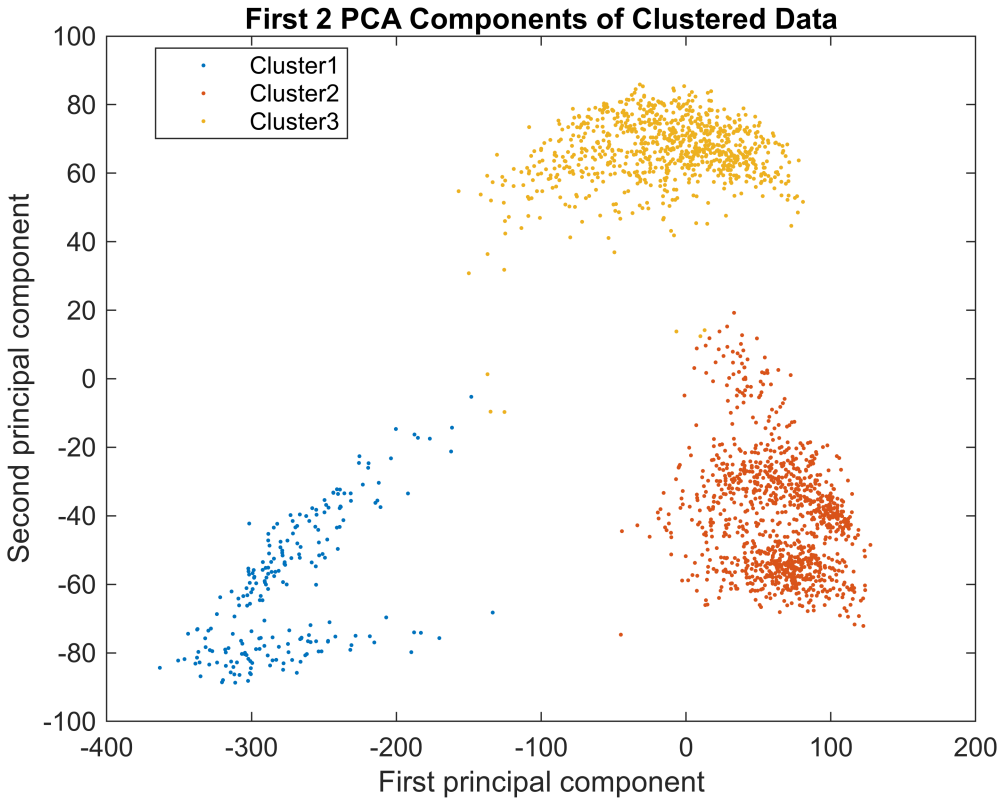
PCA Plot in 2-D

Before deciding what to do with data, one common practice is to scatter plot a high-dimensional dataset into 2 principle axis.



Clustering into 3 groups

Even without using standard methods, it is clear that there are 3 groups. Further, it was asuumed by the question that there are three types of neurons, namely excitory neurons, inhibitory neurons, and non-neuronal cells,



9: Q2: part1 Visualization #1

#1: Providing a visualization which demonstrates the existence of three main brain cell types.

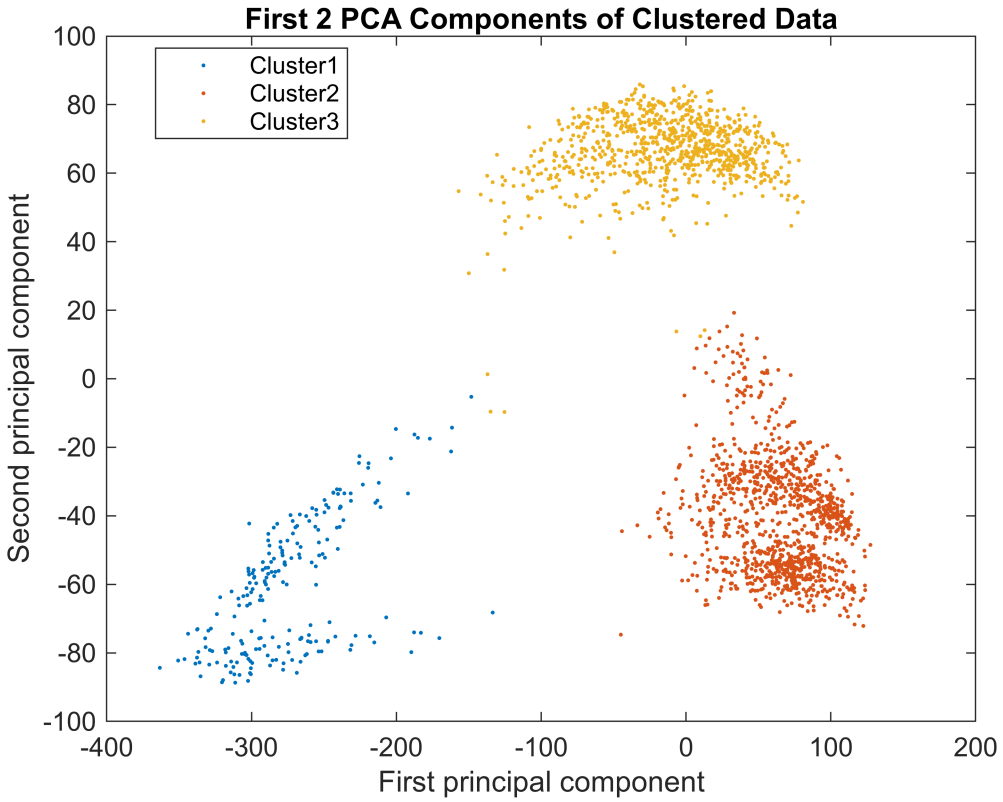
We do not know the exact criterion that classifies those cell types, but we do know, or guess, that there should be enough variance between datas to to cluster them. To extract notable features from a higher dimensional space and represents in lower dimensional space, one particularly popular method is PCA, which structs an orthogonal vectors(unit vector) that preserves maximum amounts of variance.

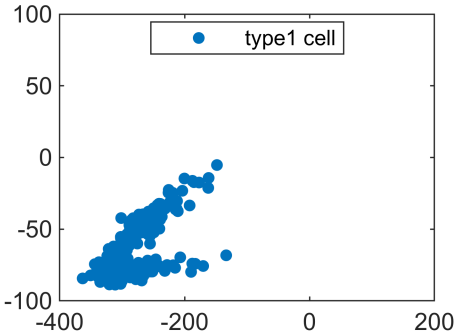
By naked eyes, we can easily see that there are the three groups of data sets. Furthurmore, k-mean method confirms our hypothesis. Without further confirming with an elbow plot or shiloutte methods, above visual plots sufficiently confirms that there indeed three groups of cells.

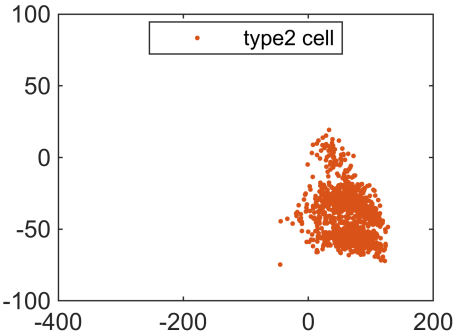
10: Q2: part1 Visualization #2

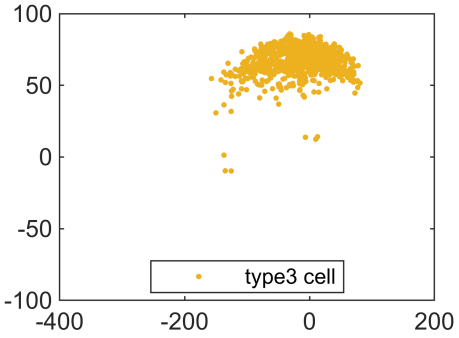
Cell Type Segmentation

It is stated that there can be further features that may distinguish each cell groups into sub-classes. In order to examine the validity the statement, one may apply PCA on each groups again. Before doing so, one may want to separate three clusters that we already distinguished.



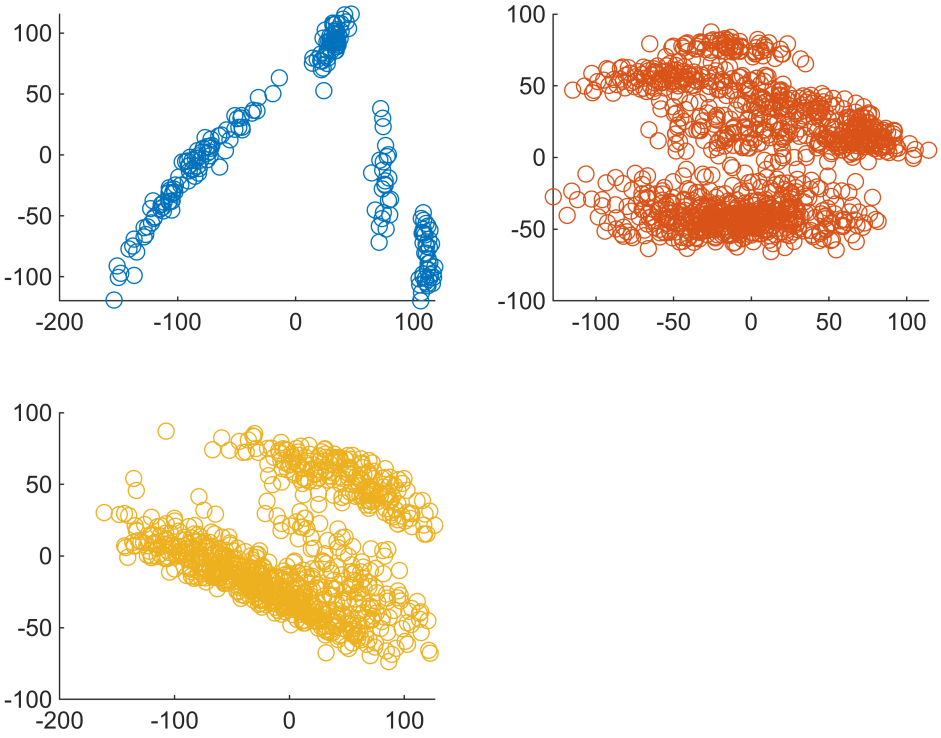






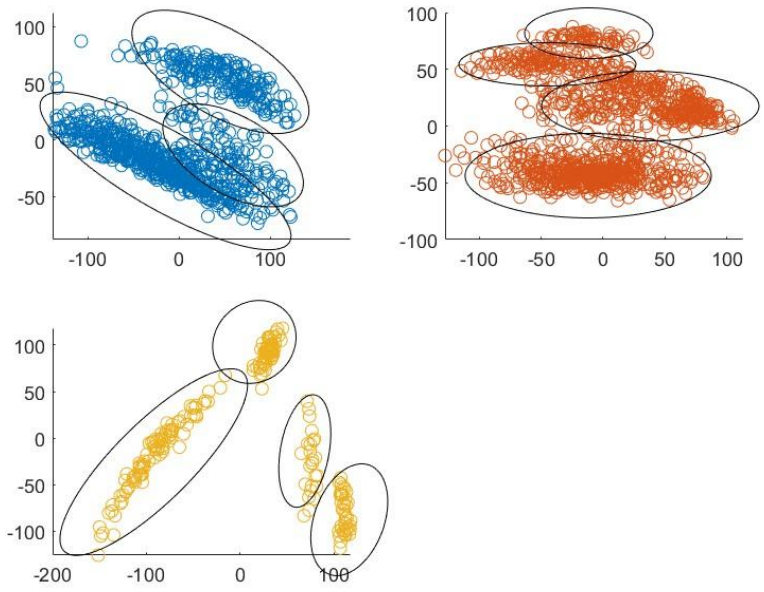
11: Sub-Type of cells: PCA

After segmenting clusters, PCA was applied and plotted against its first two principle axes.



Part1: Visualization #2

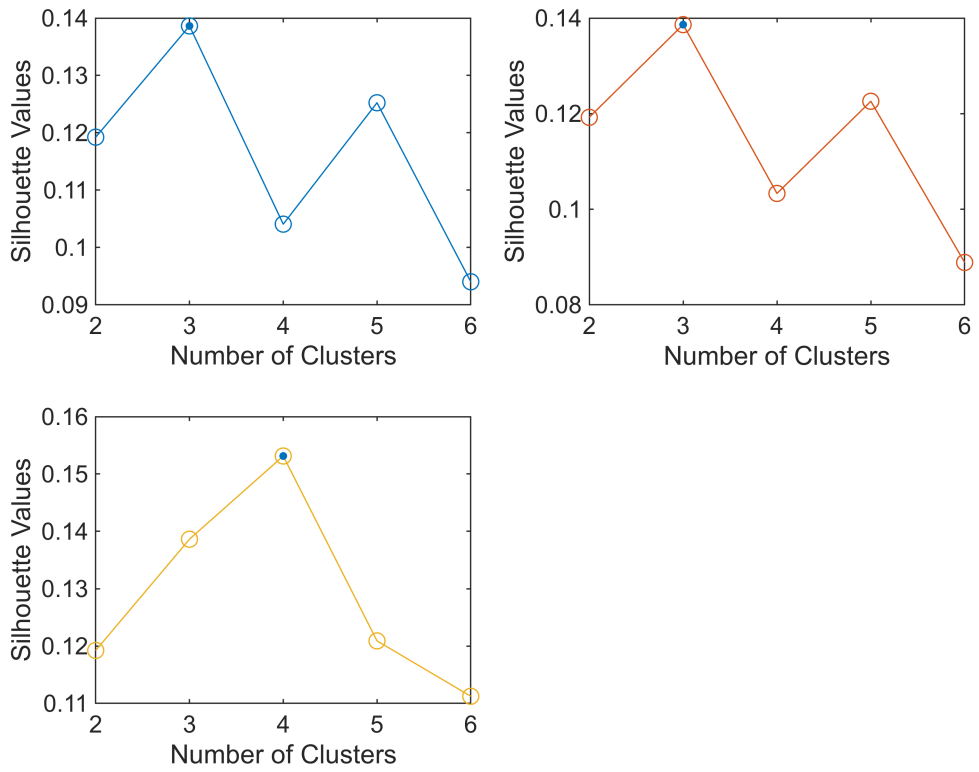
As three classes are segmented and PCA was applied to each of them, each graphs represents one of three major cell types. Annotations are added by hands to indicates that by naked eyes, followings sub classes can be hypothesized.It can be easily seen that there are numerous sub-classes for each cell types.



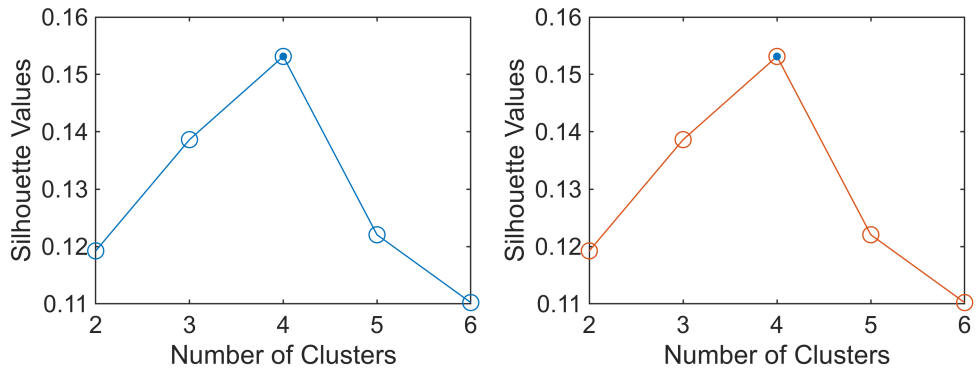
12-Q2: part2 Unsupervised Feature Selection #1

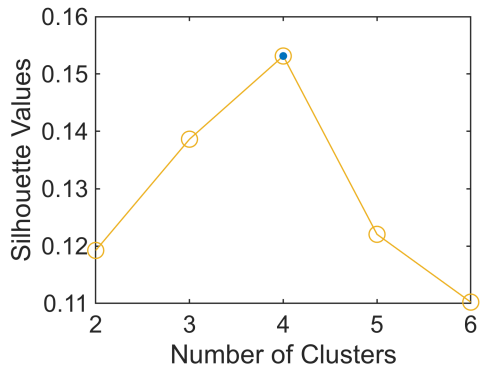
Part2: Subclassification - Silhouette and visual methods

Having classified sub-types by hands, whose approximate upper bounds are given by ellipses, let's confirm our hypothesis with numerical methods - shlhouette plot.



Following is the silhouette score when hierarchical clustering algorithm is used.

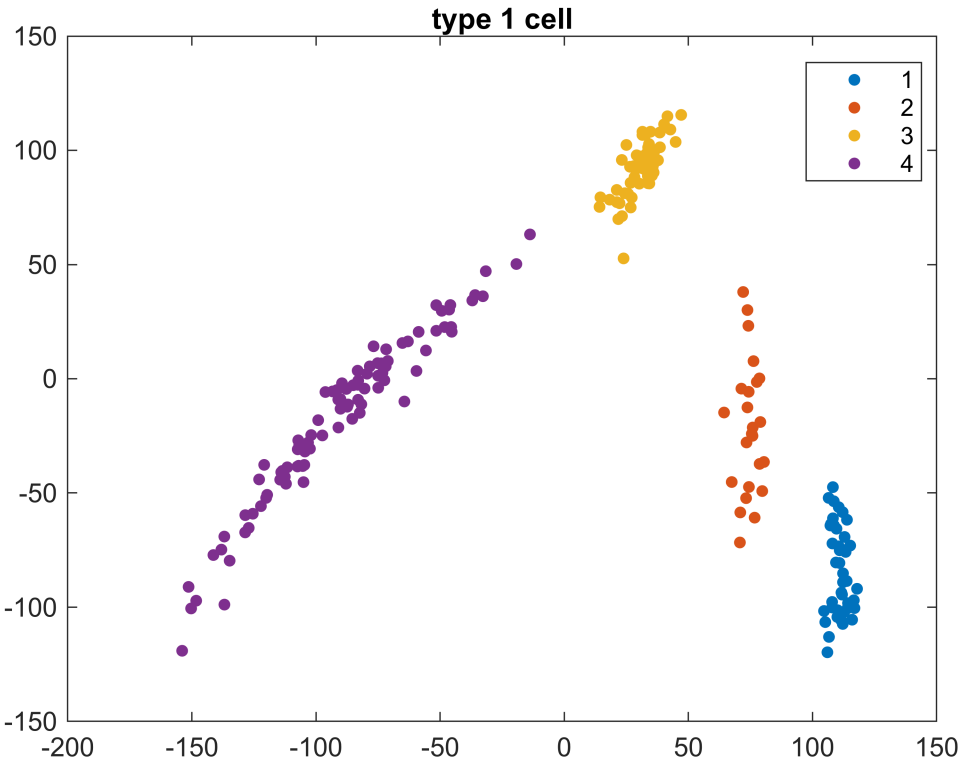


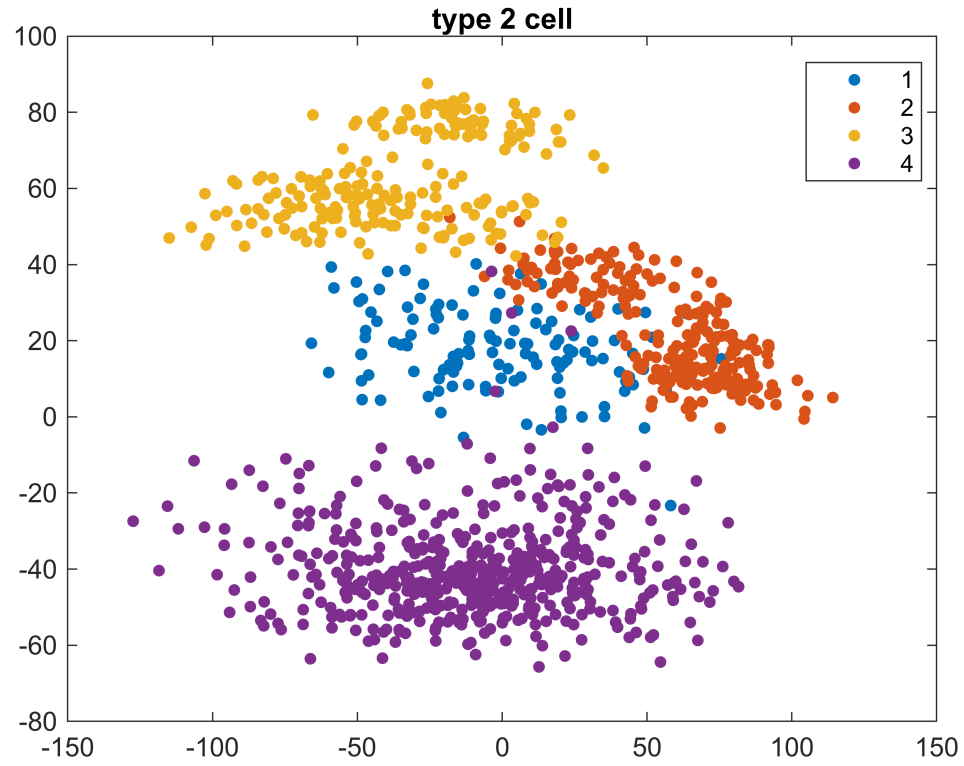


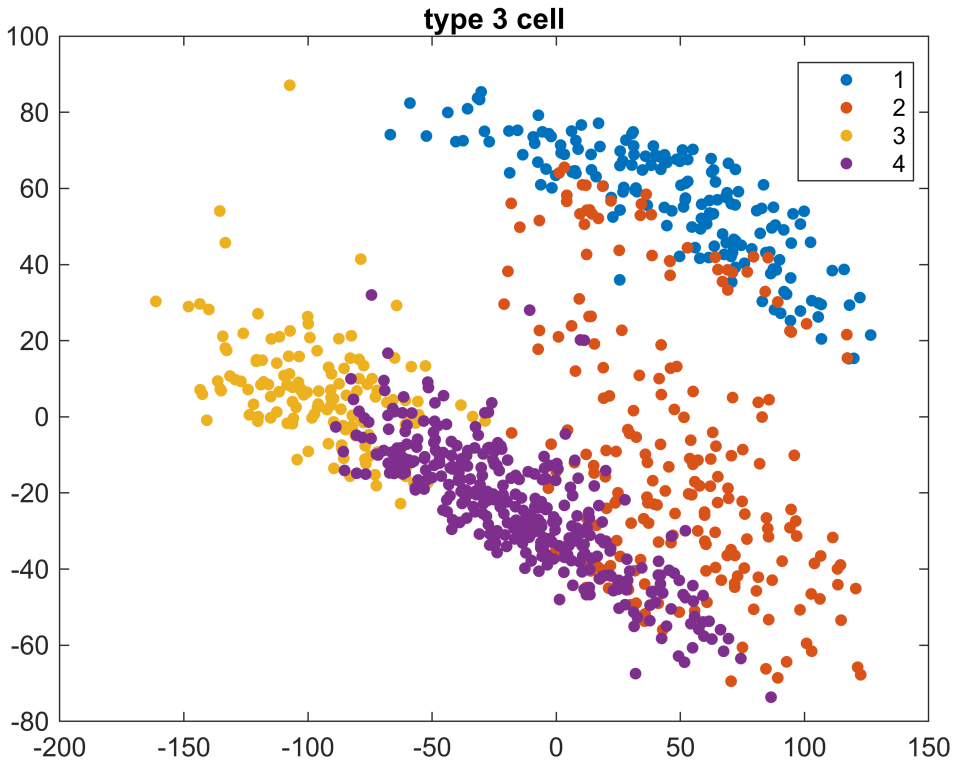
I would give more weight into hierarchical clustering algorithm(called 'linkage' in MATLAB and is a forward propagation algorithm) and choose optimal parameters as (4,4,4), in the tuple notation.

As silhouette scores trturned by hierarchical and kmean algorithm matches, it is concluded that 4 clusters would be optimal.

Plotting with 4 clusters each.


type 1 cell
150
100
50
0
-50
-100
-150
-200
-150
-100
-50
0
50
100
150
1
2
3
4


type 2 cell
100
80
60
40
20
0
-20
-40
-60
-80
-150
-100
-50
0
50
100
150
1
2
3
4



Plotting Alternatives

13 Part2- Multiclass Logistic regression.

Finding functionality satisfies given criterion

According to introduction to *Introduction to Mathine Learning* by Ethem Alpaydin, the result of multiple classes logistic regression, aka softmax function, is given as following,

$$y_i = \hat{p}(C_i|x) = \frac{e^{W_i^T x + W_{i0}}}{\sum_{j=1}^{k} e^{W_j^T x + W_{j0}}} \text{ where } \text{Log}\left(\frac{p(x|C_i)}{p(x|C_k)}\right) = W_1^T x + W_{i0}^0 \text{ when probability distributions are assumed to be normal.}$$

Used Function: Fitcecoc(Fit multiclass models for support vector machines or other classifiers)

Criterions:

Coding: 'onevsall' -For each binary learner, one class is positive and the rest are negative.

This design exhausts all combinations of positive class assignments.

Learner: ' linear' - Linear Classification Model

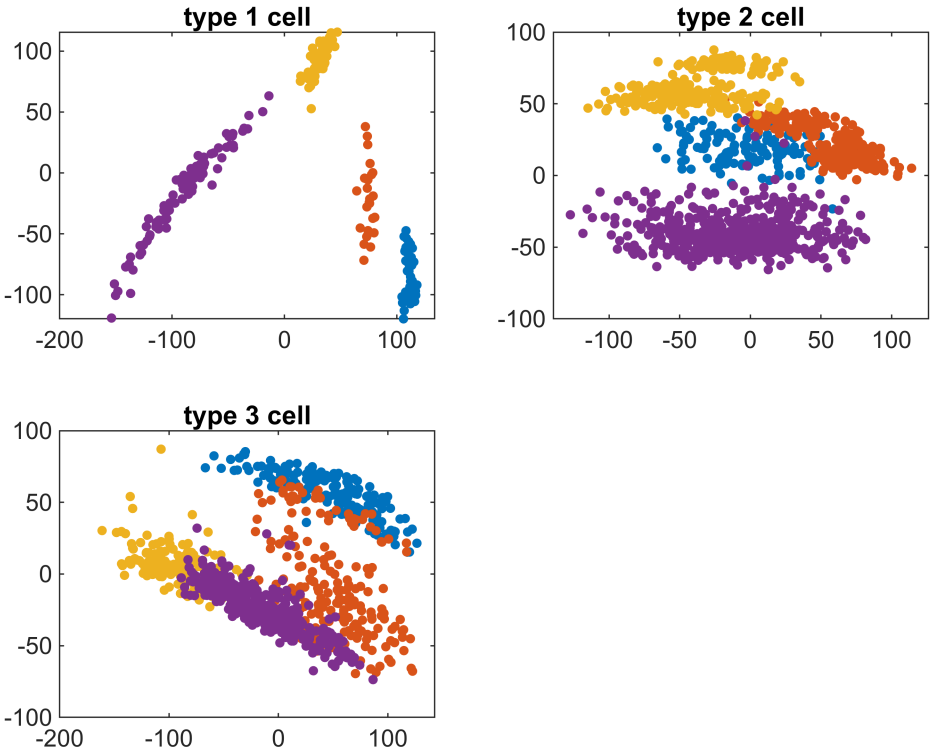
ScoreTransform: 'logit' - $\frac{1}{1+e^{-x}}$

Lambda: regularization strength is manually exlored for the interest of time.

All other options are 'default' option used by Fitcecoc

Applying (multiple logistic regression)

previous choice: (4,4,4)



14-Q2: part2 Unsupervised Feature Selection #2

Partition Dataset p2

Created randomly selected 5 k-fold sets to train and test.

```
c =
K-fold cross validation partition
   NumObservations: 2169
       NumTestSets: 5
         TrainSize: 1736  1735  1735  1735  1735
          TestSize: 433  434  434  434  434
```

Apply Multinomial Linear Regression

to summarize the result, 5-fold cross validation yielded accuracy of

(0.9238, 0.9401, 0.9355, 0.9470, 0.9470) where 1 represents 100 percent.

```
the accuracy of 5-fold crossvalidation was calculated to be (sum(correct)/total length)

    0.9423    0.9055    0.9217    0.9147    0.9700
```

Apply multiclass linear regression

Extract 100 best features

By 'lasso (L1)' regularization, for it example, it returned 12278 non-empty parameters for classifier #1.

```
ans = 1×2
       11990           1
```

To select 100 best features, I added parameters for 12 classifiers.

And by the order of significance, ind100 reports the index that gives the best features whose magnitude is value100 $\sum_{\beta}|\beta|$ where beta is $45768 * 1$ vector,

```
value100 = 100×1
    0.5282
    0.4255
    0.3985
    0.3871
    0.3591
    0.3406
    0.3355
    0.3144
    0.3090
    0.3044
     ⋮
ind100 = 100×1
        4357
        4689
       36986
        8587
       34845
        8590
       31422
       40742
        6043
       36987
      ⋮

ind100_sorted = 100×1
         876
        2550
        2604
        3497
        3841
        3937
        4275
```

```
        4357
        4442
        4689
      ⋮
```

15-Q2: part2 Unsupervised Feature Selection #3

Data Preparation

Logistic Regression Classifier

100 genes selected by random

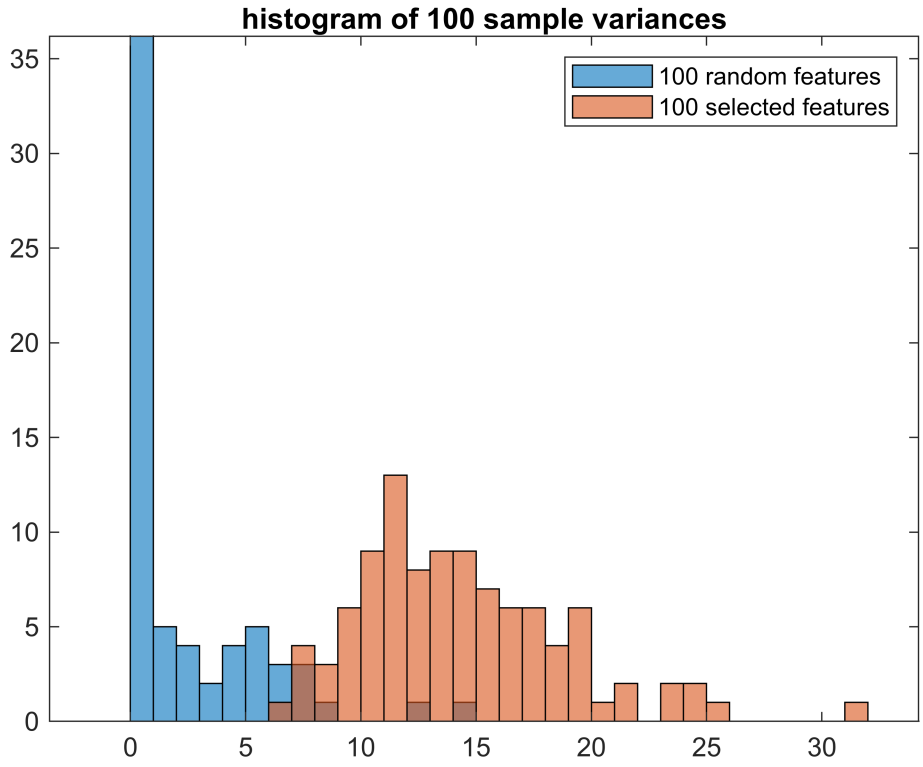
To summarize the result, the random genes gave 40percent accuracy whereas top 100 gens gave accuracy of 92 percent.

```
rand_idx = 100×1
         447
         692
        1014
        1121
        1656
        1856
        3033
        3579
        4872
        5524
      ⋮

the accuracy is calculated to be 0.43592
```

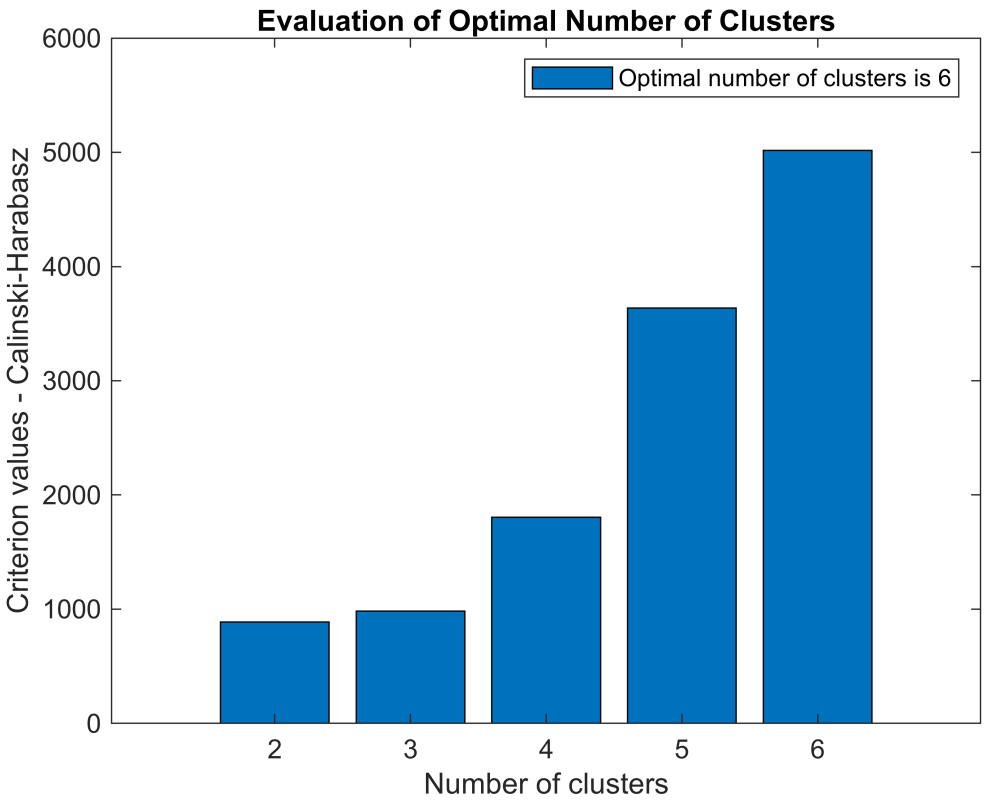
Top 100 genes selected by the previous question

```
the accuracy is calculated to be 0.93592
```

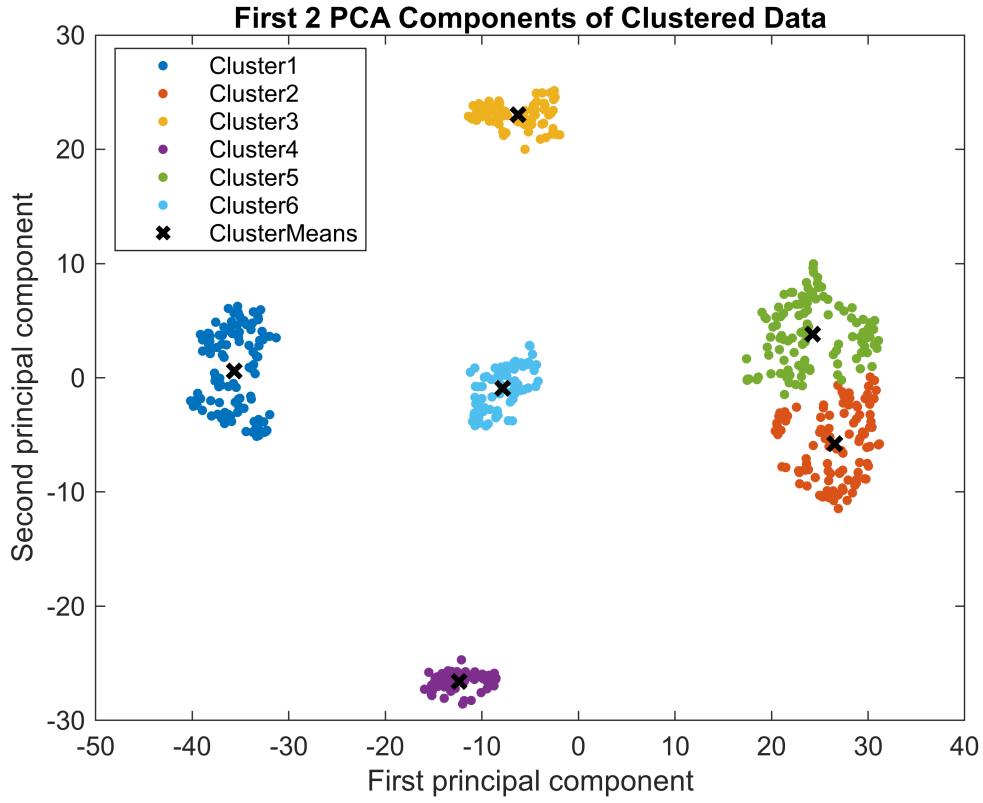


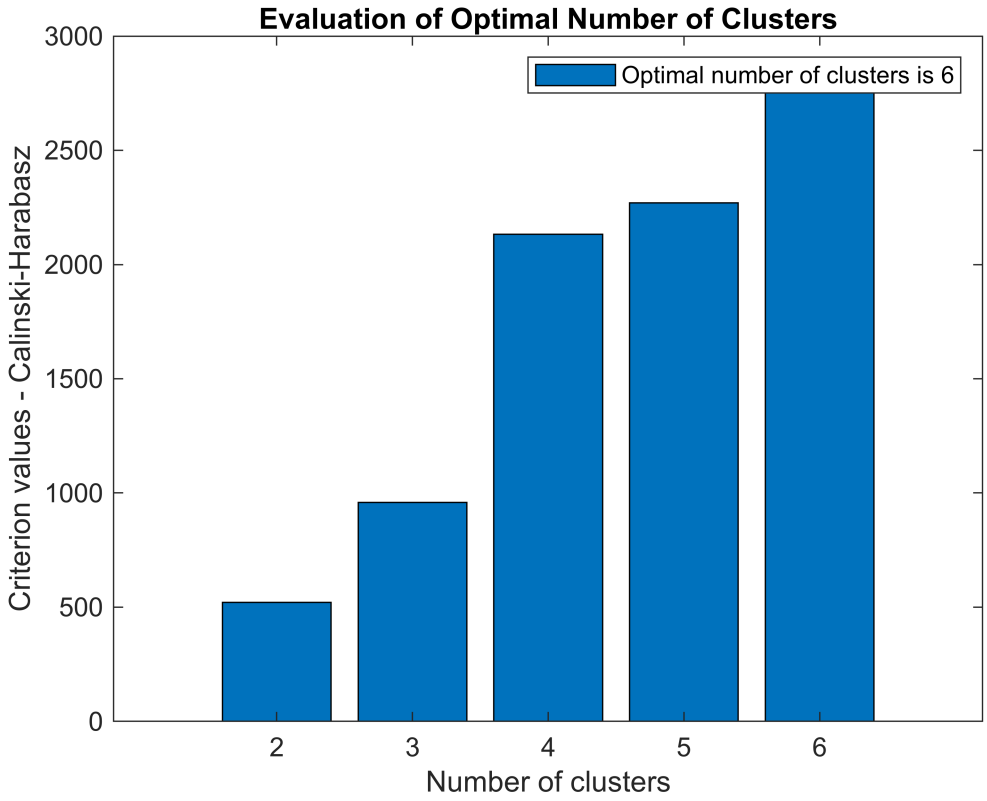
16-Q3- Influence of Hyper-parameters #1

Plotting T-SNE with various # of principal axes

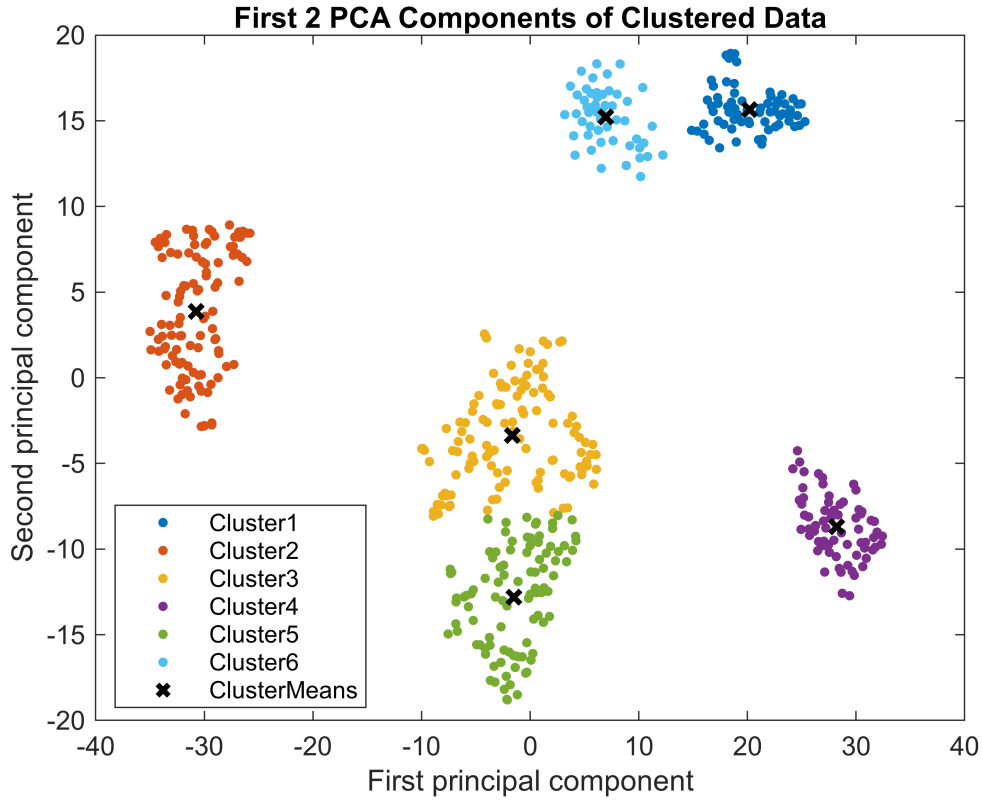


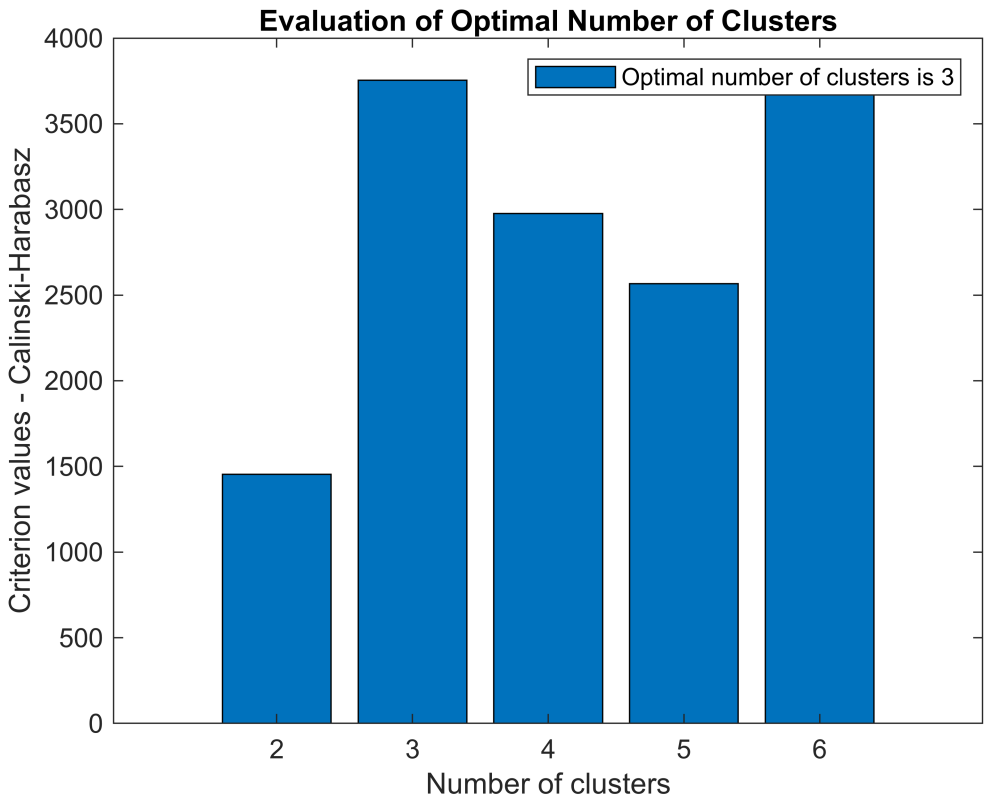

```
Optimal number of clusters is 6
```



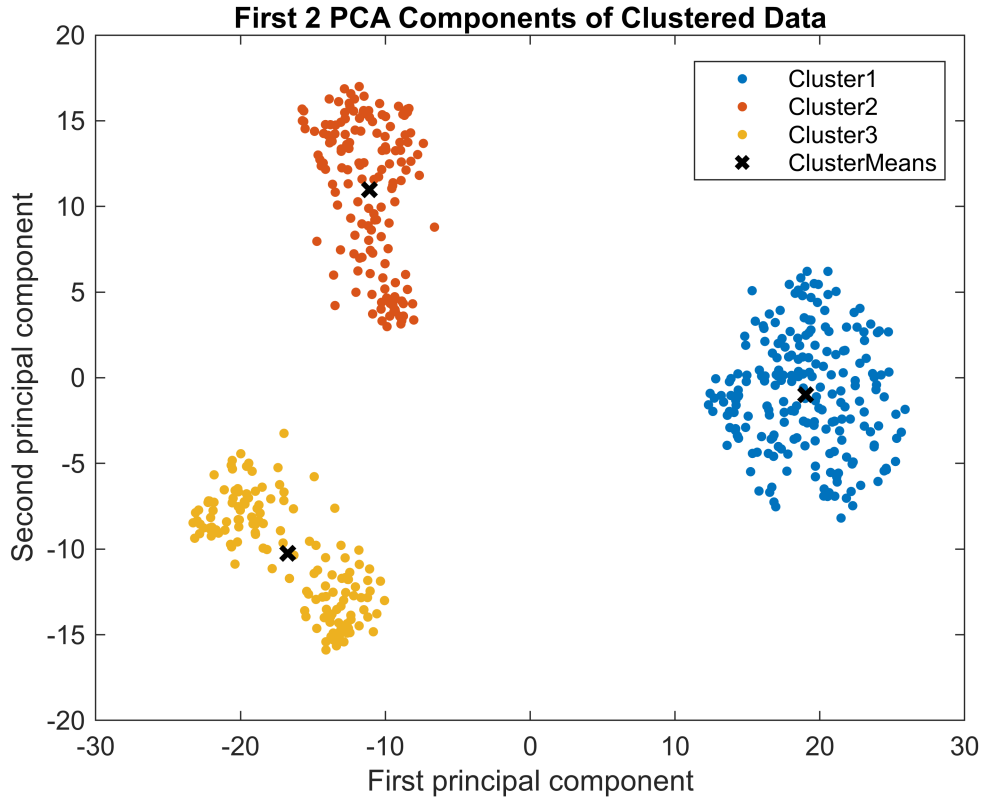


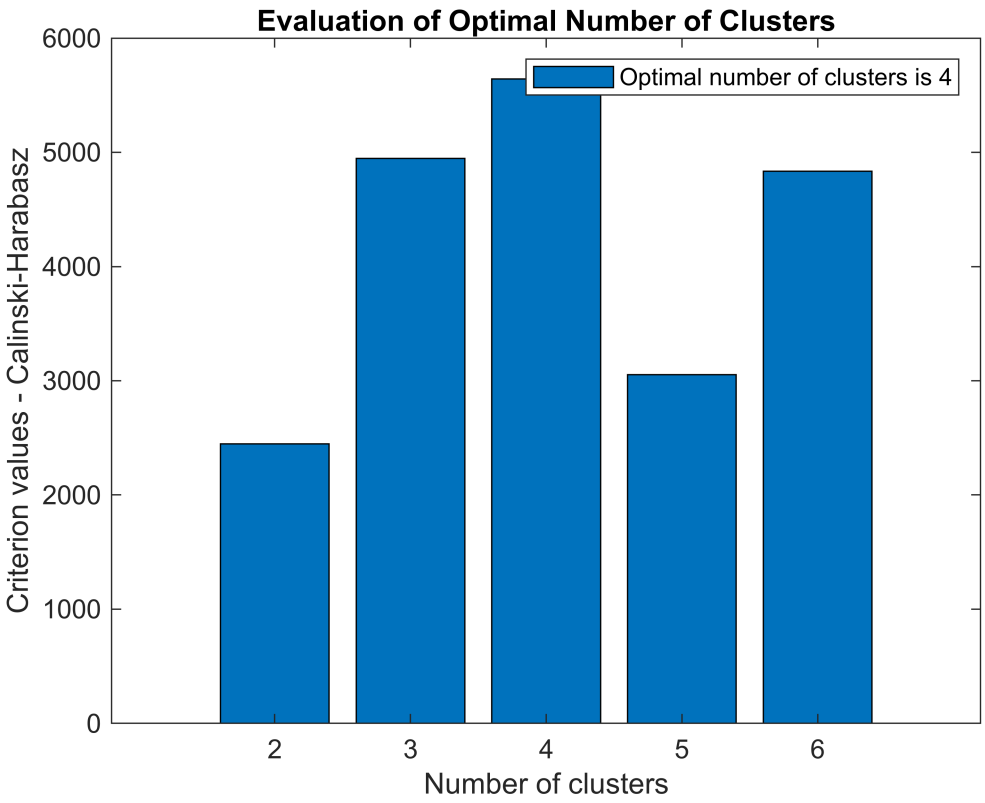

```
Optimal number of clusters is 6
```



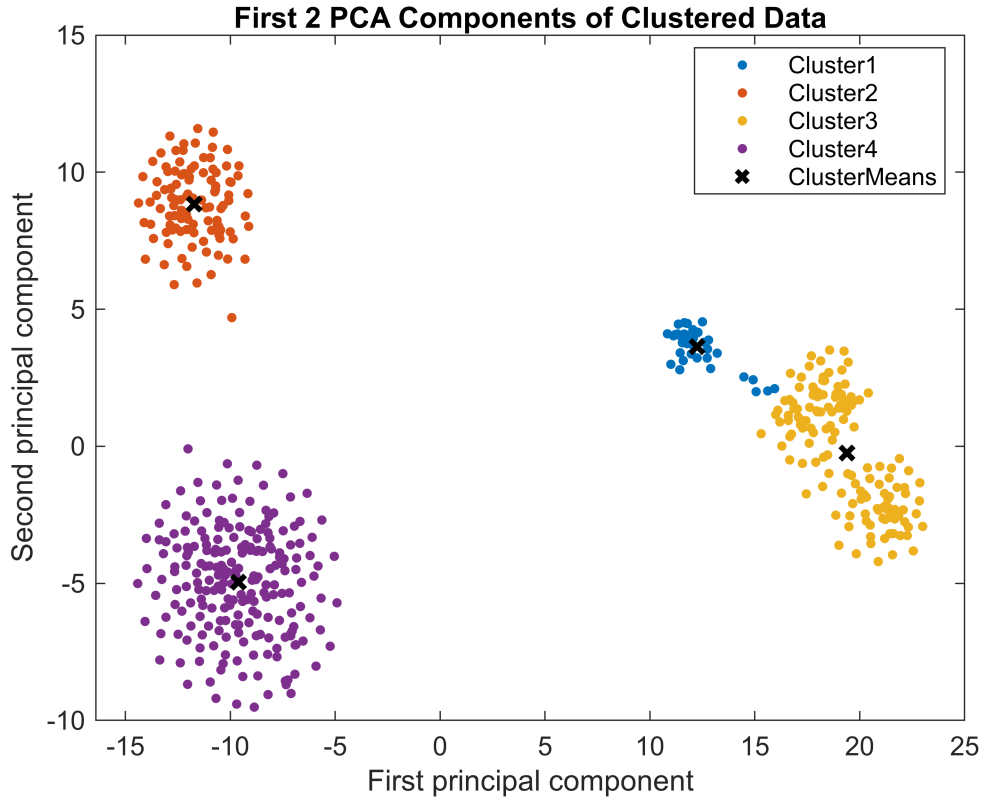


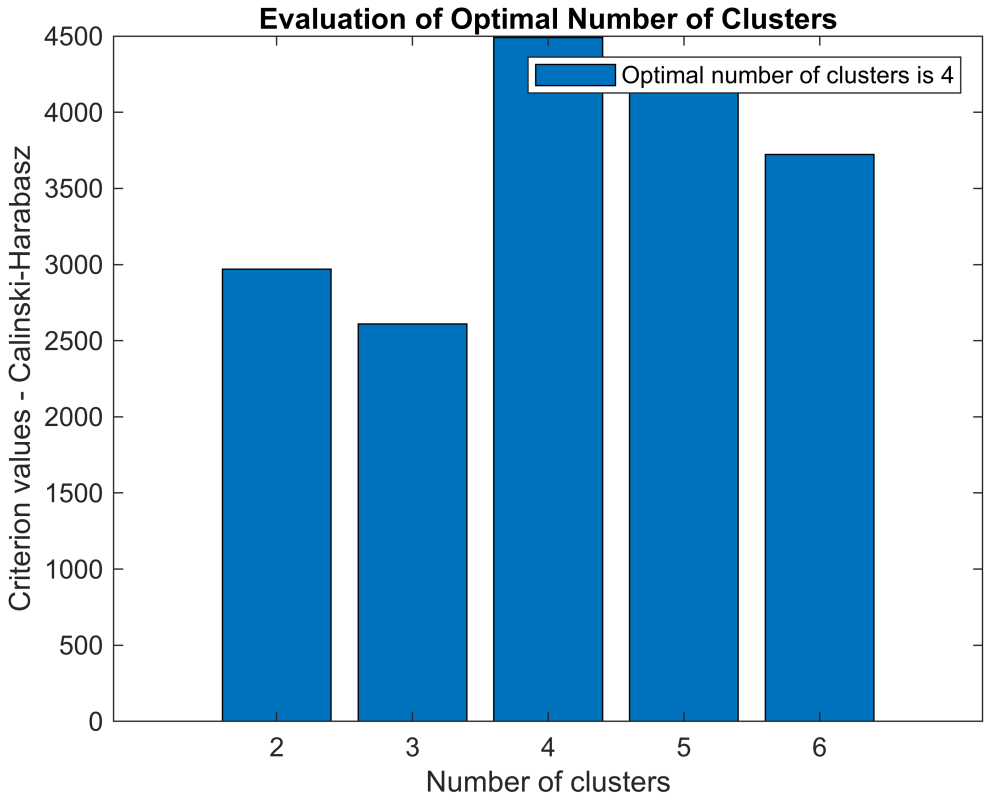

```
Optimal number of clusters is 3
```



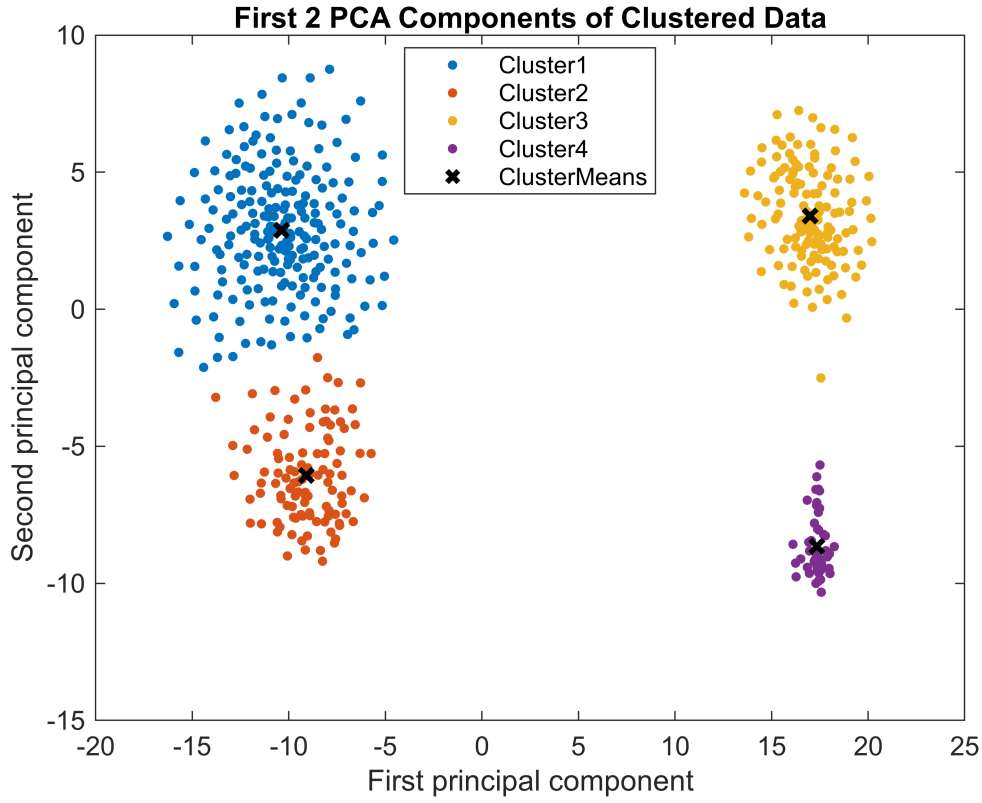



```
Optimal number of clusters is 4
```






```
Optimal number of clusters is 4
```



17-Q3- Influence of Hyper-parameters #2

Category B- Effect of number of PC's chosen on clustering

T-SNE calculates pairwise distances between high-dimensional points and create lower dimensional points that preserves high dimensional distance as possible, which is noted by minimizing Kullback-Leibler divergence. As we lower dimensions from PCA, we observed that explained variance goes down; the reduced complexity of matrix was traded for reduced explained variance.

Since using reduced numbers of axes results in lowered explained variance, it can be observed that several points stick together and the shape of clusters are not spherical. This is simply because reducing principal axes increased the chance that possibly distinct points to stick together due to proejction; one can easily think that projecting 2-d points into 1-d would generally decrease squared euclidean distances between them.

On the same note, as potentially distinct pairs of points projected together, the shape of clusters have a higher chance of being eccentric while having higher dimensional information would decrease the chance for two random points having low squared euclidean distance. Consider the plots with 500 PCs, most of them are evenly spread out because having more coordinates(PCs), it is generally hard to have small squared euclidean distances.

$$\because \text{Squared Euclidean Distance} = \sum_j \sum_i (x_i - x_j)^2$$

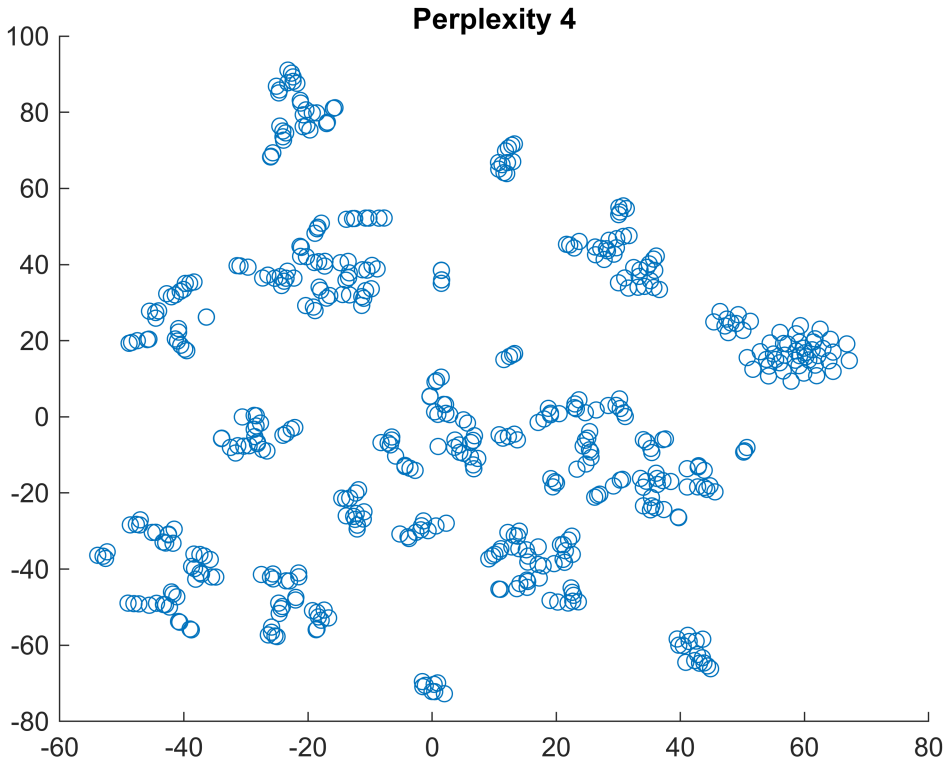
$$\text{and } x_i - x_j = \sum_d \text{coordinate difference in each dimensions}$$

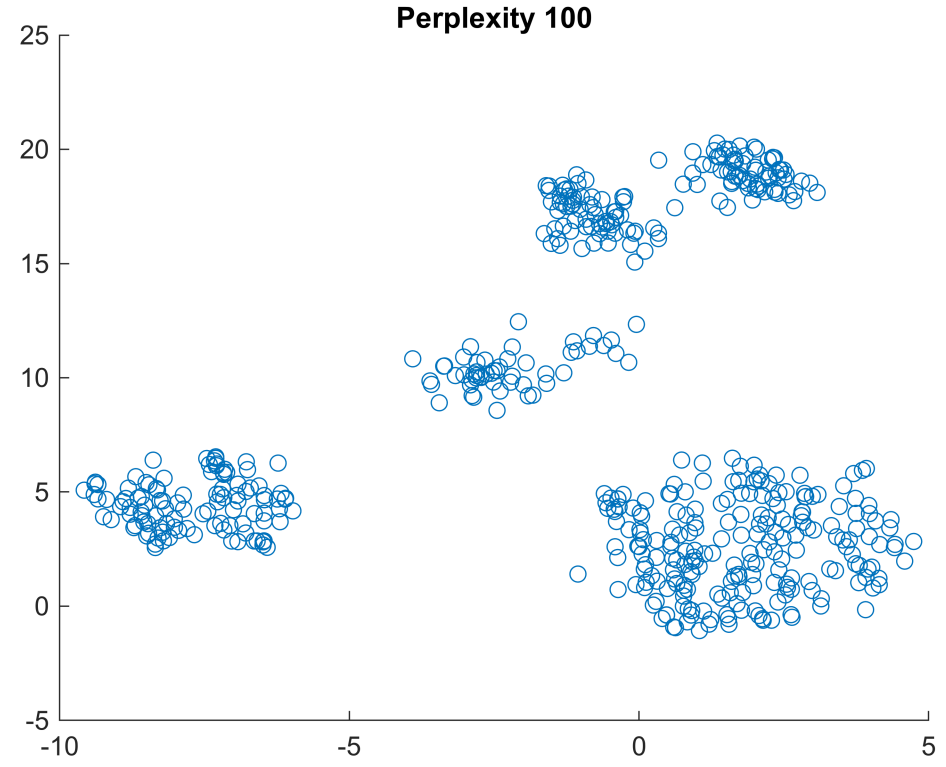
$$\therefore \text{Having more dimensions mean higher expected magnitude for square euclidean distance.}$$

Category A- Effect of perplexity and exaggeration

Effects of perplexity

According to MATLAB documentation, perplexity is a measure of the effective number of neighbors of a given point. And tsne performs a binary search over the deviation to achieve a fixed perplextiy for each points. In other words, having higher perplexity would likely results in two points with higher euclidean distance to be embedded together.


Perplexity 4
100
80
60
40
20
0
-20
-40
-60
-80
-60
-40
-20
0
20
40
60
80


Perplexity 100
25
20
15
10
5
0
-5
-10
-5
0
5

It can be easily seen that many of clusters in perplexity 4 graphs are embedded together and formed a cluster in the perplexity 100 graph.

Effects of Exaggeration

According to MATLAB documentation, during the first gradient descent steps, `tsne` weighs the probabilities *pij* by given exaggeration value, which create more space between clusters in the output Y.

where the equations of conditional probabilities are given as follows;

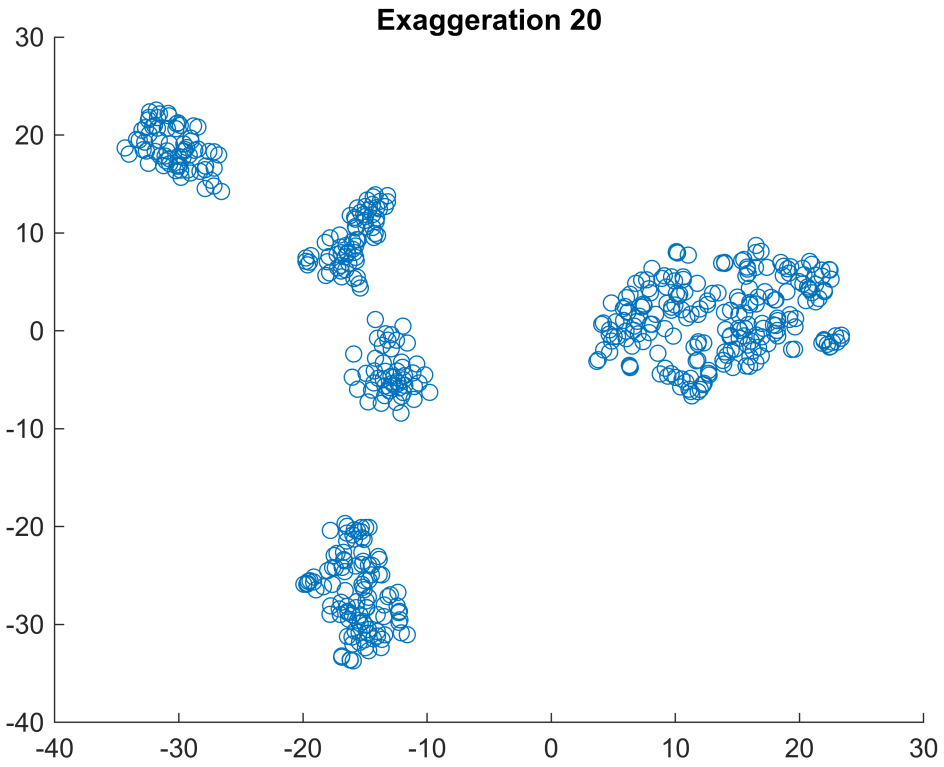
Define the conditional probability of j given i as

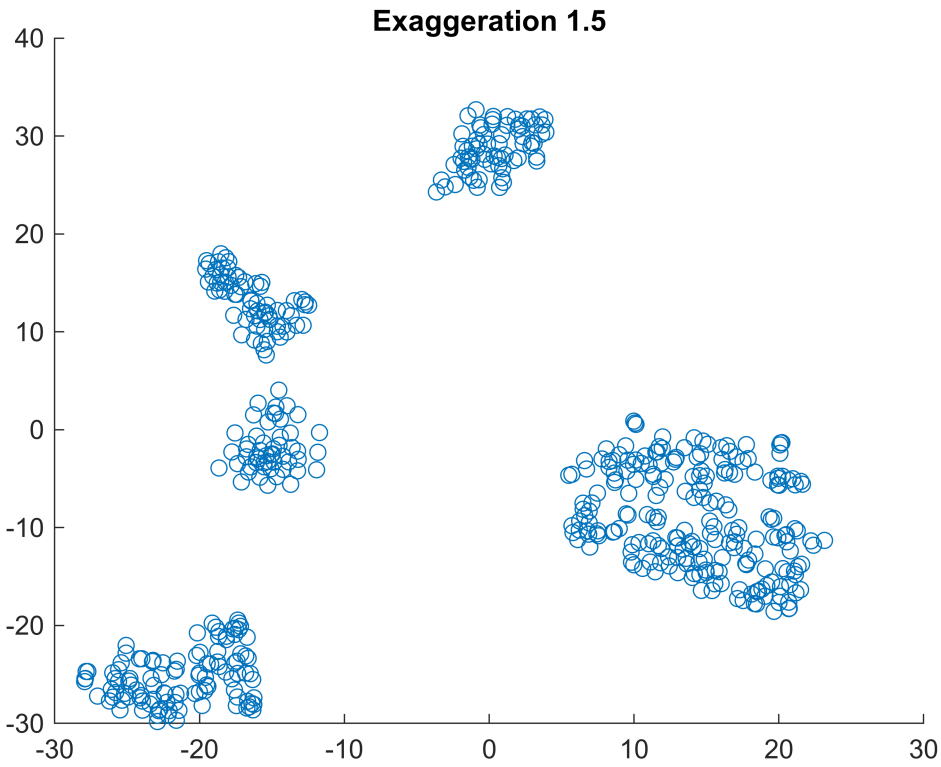
$$p_{j|i} = \frac{\exp\left(-d(x_i, x_j)^2/\left(2\sigma_i^2\right)\right)}{\sum_{k \neq i} \exp\left(-d(x_i, x_k)^2/\left(2\sigma_i^2\right)\right)}$$

$$p_{i|i} = 0.$$

Then define the joint probability p_{ij} by symmetrizing the conditional probabilities:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}, \qquad (1)$$





The figure with an exaggeration of 20 has larger empty spaces between clusters. It can be also observed that a loosely embedded cluster (on the rightmost for each graphs) shows clear density difference; the rightmost cluster in exaggeration 1.5 seems to be 'split' in half while in exaggeration 20, it is connected stronger.

Exaggeration makes points easier to move relative to one another, which resulted in spaces between clusters and splitting between clusters.