Technische Universität München
Fakultät für Informatik
Prof. Nassir Navab

WS 15/16
Exercise Sheet 4
November 4, 2015

Exercises in Machine learning in Medical Imaging

Exercise 1 K-Means

In this exercise you will familiarise with K-Means, one of the clustering techniques that was introduced during the lecture. You will implement the class myKMeans, whose structure is detailed in Figure 1.

Implement the constructor of the class myKMeans takes three arguments, the data D which is organised column-wise (each datapoint is one column), the number of clusters K to be discovered and the number or random initialisations that you want to perform in order to ensure convergence. For each of the runs perform the following actions:

a) choose the means that should be used at the beginning. In order to make a sensible choice of means we suggest to use random elements of the dataset using the matlab function randi().

b) create a while loop whose body updates the K-Means as explained during the lecture and that terminates when the K-means update performed in the last step is smaller than a constant (for example 0.0001).

When more than one run is performed, it is necessary to analyse the results of all the runs in order to find the true K-means. We should remember that the k-means are not sorted! For example the first mean of the first run might be the same as the fifth mean of the second run and the third mean of the third run. In this case the algorithm delivered the same result for one of the k-means but we would be unable to discover this if we don't put the k-means obtained in each run into correspondence. We must sort the means such that they correspond across the runs. We will use the k-means of the first run as reference and we will match the k-means of the other runs to those of the first run.

Once they are put into correspondence, we find the agreement of the runs by stacking the k-means of each run in the third dimension of a matrix, and we take their mode using the command mode(matrix,3) (see help mode).

You can test your K-Means implementation on 2D data by using the matrix D contained into the file dummyData.mat, which contains random points clustered into three distinct regions. Since the data is 2D, you will be able to see the means being updated in real time as the algorithm progresses.

Exercise 2 K-Means Segmentation

As you have seen in the lecture, K-Means can be used for segmentation of simple images. In this exercise we want to use the newly implemented class myKMeans and the class Filter-Feature that you implemented during the second session, in order to segment a MRI image. You are going to implement the class SegmentImageKMeans according to the diagram of Figure 2.

The data for this exercise, namely the MRI, grayscale, image is stored in the file mri.mat. For this exercise you will also make use of the content of the file params.mat, which contains

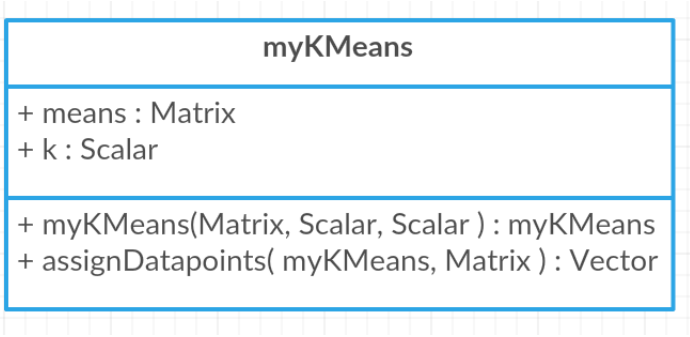

Figure 1: Class diagram for myKMeans.

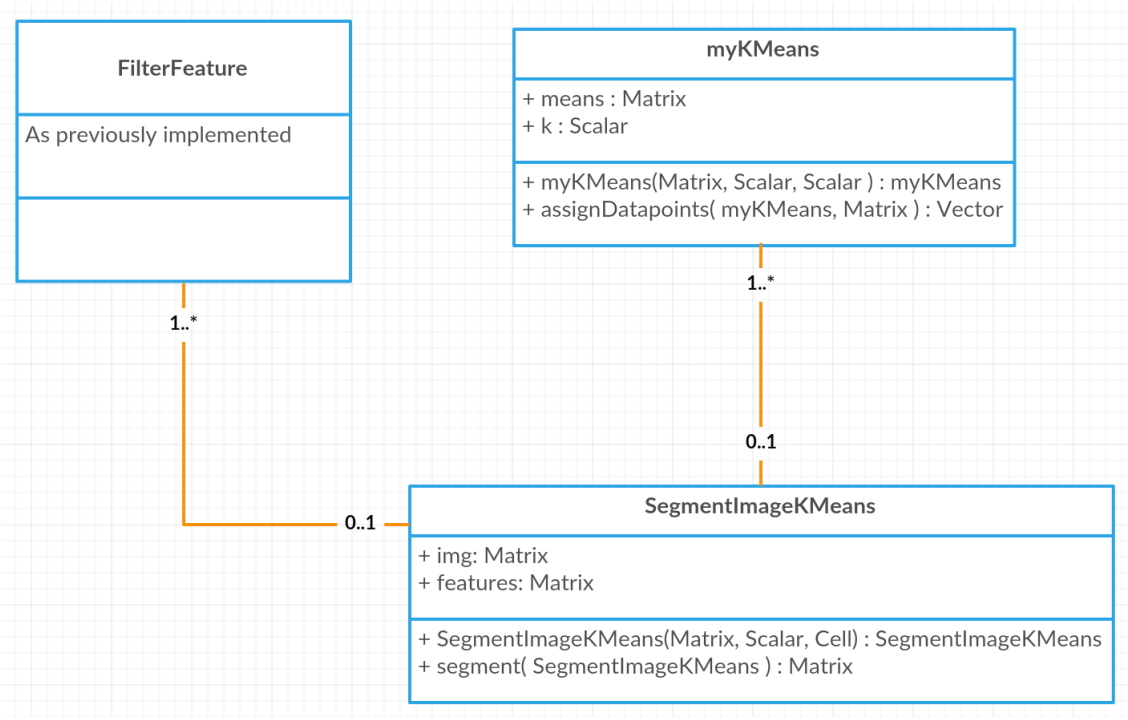

Figure 2: Class diagram for SegmentImageKMeans.

a cell of structures of parameters that are employed to instantiate a cell of objects of class FilterFeature in the constructor of SegmentImageKMeans.

Please complete the implementation of:

- the constructor of SegmentImageKMeans, which is partially written and where you need to instantiate the object of type myKMeans passing the appropriate arguments (please refer to the comment in the code) and store it in obj.kMeansObj.
- the method "segment" where you need to use the features extracted from the data in the constructor and stored in obj.features to segment the image using K-Means. In particular you should make use of the functions obj.kMeansObj.assignDatapoints() first and reshape() later.