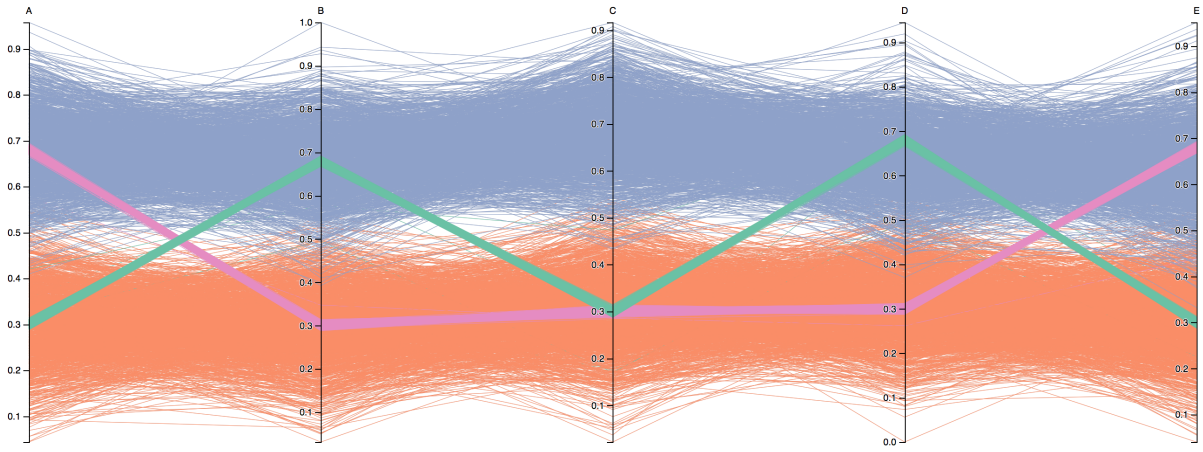
TNM048—Information Visualization

Data Mining

January 31, 2020

Kahin Akram
kahin.akram.hassan@liu.se



1 Introduction

Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar to each other than to those in other groups (clusters). It is a main task of exploratory data mining, and a common technique for statistical data analysis, used in many fields, including machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics. In this lab you will work with a common clustering technique called $K-$ means.

2 The Setup

As the previous lab we will use D3.v4.js JavaScript library. Download the files from the course home-page. **Use this link to D3.js Version 4** and **this link to K-means** to make sure you know what and how to do it. Please read about K-means before starting!!

Choose a preferred editor and make sure to comment the code well, this comes in handy when presenting the code to the assistant. ***Also remember to have the project folder on the W hard-drive or if you are using your own computer you can use i.e Xampp or Python Flask to run a server.*** If you are using a university computer make sure to access the page via ***www.student.itn.liu.se/∼student_id***

3 The Data

The data sets we will use to test the clustering algorithm can be found under the data folder. There are three data sets.

1. Data1-2clusters.csv
2. Data2-2clusters.csv
3. Data3-Xclusters.csv

The first two data sets are small and contain very distinct clusters that you could use to verify that your algorithm works. The third and larger data set contains an unknown number of clusters. The final task of this lab is to find the shape of these clusters. Keep in mind that the third data set is large and can take a longer time to load than the other two. However, if robust code is implemented this should not be an issue.

4 K-means algorithm

K-means is one of the simplest unsupervised learning (unlabeled) algorithms that solves the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters fixed a priori. The main steps of the algorithm are:

1. Randomly place K points into the space represented by the items that are being clustered. These points represent the initial cluster centroids.
2. Assign each item to the cluster that has the closest centroid. There are several ways of calculating distances and in this lab we will use the Euclidean distance:
$$\sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \tag{1}$$
3. When all objects have been assigned, recalculate the positions of the K centroids to be in the centre of the cluster. This is achieved by calculating the average values in all dimensions.
4. Check the quality of the cluster. Use the sum of the squared distances within each cluster as your measure of quality. The objective is to minimize the sum of squared errors within each cluster:
$$\min \sum_{i=1}^{k} \sum_{x_j \in S_i} \parallel x_j - \mu_i \parallel^2 \tag{2}$$
where μ_i is the centroid (mean of the points) of each cluster S_i. Iterate (repeat steps 2-4) as long as the quality is improving, once there is no or minimal (less than a threshold) improvement break your algorithm.

Task 1:
Implement the $K-$means clustering algorithm according to the above description. It should be possible to specify an arbitrary number of clusters. Use data sets (1) and (2) to verify that the algorithm works as expected.
Write your code in the ***kmenas.js*** file. The function has to return an array of assignments that will be used to color code the polylines in parallel coordinates according to which cluster they belong to.

5 Analyze the clustering results

Unfortunately there is no general theoretical solution to find the optimal number of clusters for any given data set. A simple approach is to compare the results of multiple runs with different k classes. In addition, the random initialization of the centroids can influence the result.

Task 2:
Read in data set (3) into the the function ***main.js***. Run the algorithm multiple times with varying number of clusters and study the results. What do you think is a good number of clusters for this specific data set (cluster number in ***pc.js***). What are the shapes of the clusters? Discuss your results with the lab instructor.