

**TALLINNA TEHNIKAÜLIKOOL**

INFOTEHNOLOOGIA TEADUSKOND

Home Assignment 1



Student: Mihhail Daniljuk

Student code: 211579IAPM

Tallinn 2022

Exercise 1. Representative based clustering

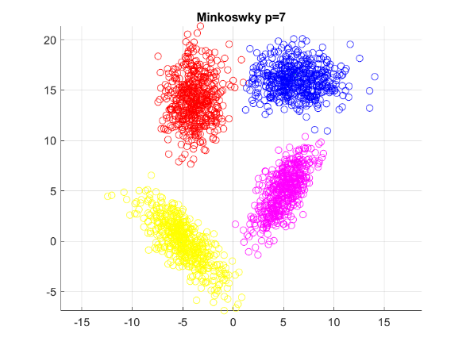
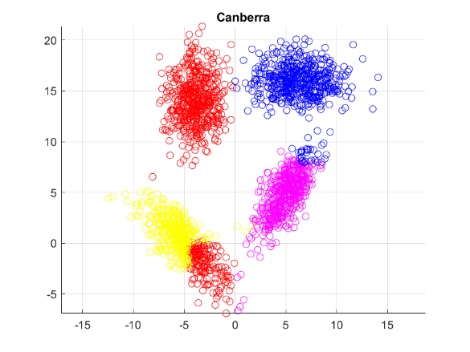
Comparing my K-medoids clustering algorithm using different distance functions I would say that overall clulstering on my particular dataset had given good results. For comparing clustering results I used only visual representation and almost every distance function contributed to similar results, as shown on Fig.1. Canberra Distance function was the only one that provided strange clustering results (Fig 2.), I suppose that, this is because of taking modulus of different features separately and then computing distance with them.

Figure 2.K-Medoids Canberra

Figure 1.K-medoids Minkowsky p=7

Exercise 2. Density based clustering.

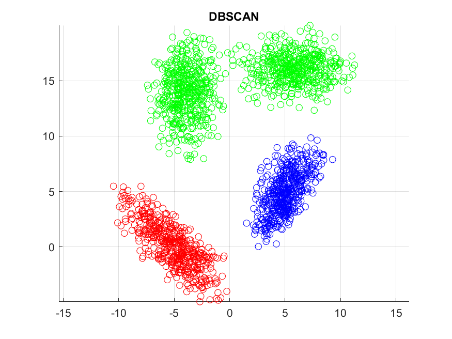
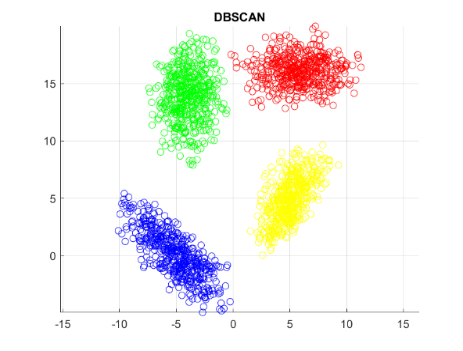
I decided to implement DBSCAN as it seemed to be easier. As my test dataset contains 2 close clusters I had to tune my *epsilon* and *theta (minimum number of neighbors)* parameters. Clustering results with diferent *epsilon* values can be seen on Fig. 3 and 4.

Figure 4. DBSCAN clustering, eps = 1

Figure 3.DBSCAN clustering, eps = 0.85

Exercise 3. Dataset generation

As I’ve had to tune my DBSCAN parameters I discovered its behaviour and with the theory provided about this clustering method I understood that DBSCAN will unite 2 different clusters if they will pass the threshold level. So I created dataset that contains 3 clusters near each other. DBSCAN fails and K-medoids is doing good. Figures can be seen in matlab, but shortly, result is near to Fig. 4 and Fig. 1.

Exercise 4. Feature selection

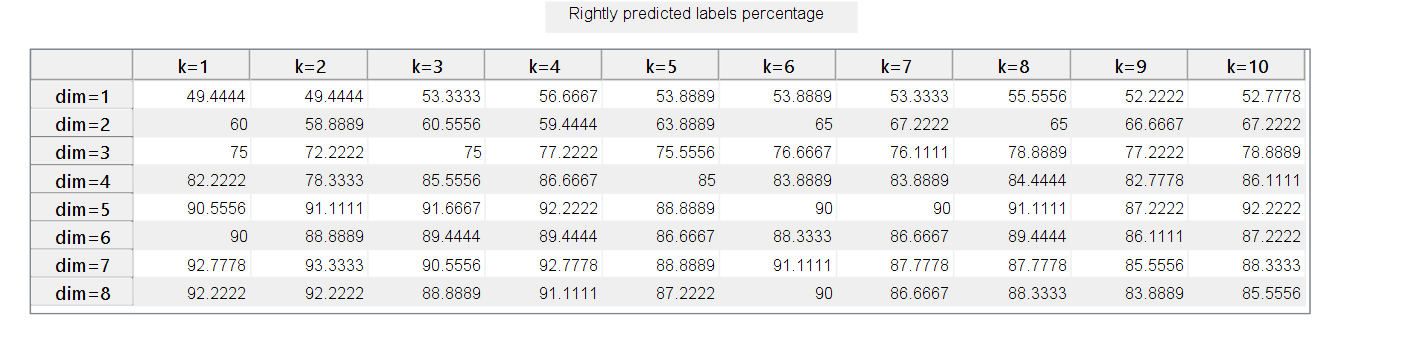
For this excercise I created dataset with easy distinguishable clusters in one dimension and much harder in second dimension, computing Fisher’s score for both features should give much more greater score for first feature than to the second. Graphs provided in matlab

Exercise 5. Decision tree

My implementation of decision tree classifies rightly about 97% of my dataset, which is easy to classify overall. I tried it only with 2-feature dataset with binary labels, so classifying lines are parallel to the axis and it affects model if the dataset has some spiral or ohter complex form datasets. Implementing crossvalidation helped to really understand if my model works correctly, because firstly my code had bugs inside and I got successful results occasionally. Crossvalidation gave insight how model works on some „random“ samples from initial dataset and helped to find the concrete bug.

I think, that in some places I „hardcoded“ decision tree to work on TWO-dimensional dataset. Figures provided in matlab.

Exercise 6. Research.

It looks like increasing the number of dimensions positively affects Knn classification model. *K* parameeter although behaves in two-ways, before some threshold in number of dimensions, increasing *k* parameeter affects results positively, but after the threshold - negatively. Table with results.

Tabel 1. KNN classifier rightly predicted labels percentage depending on dimensionality and k parameter