Introduction

This report describes what has been done to solve exercises in Home Assignment 1 and what were the results. To run code, execute 'main.m' file. Code repository: https://gitlab.cs.ttu.ee/totahv/iti8565

Exercise 1. Metric function

Implemented distance functions: Canberra, Mahalanobis, Cosine and Minkowski. Minkowski distance function takes in parameter p, so it can compute also Manhattan, Euclidean and Chebyshev distances. Each distance function takes in two points of arbitrary dimension and can be used in the K-means and DBSCAN algorithms (except Mahalanobis).

Distance function results have been compared against built-in functions and results are in the Table 1.

Table 1 Comparison of distance function implemented in home assignment and compared to Matlab function

Distance	Cosine	Canberra*	Manhattan	Euclidean	Chebyshev	Mahalanobis**
Implemented function	1.94	3.00	13.00	7.88	6.30	2.45
Built-in function	1.94	3.00	13.00	7.88	6.30	2.45

*Matlab does not have built-in function for Canberra distance, used Wolfram Mathematica.

**Mahalanobis distance is more complicated. Unlike other functions, it requires entire dataset to create covariance matrix and then it can be used to test how far are points.

Exercise 2. Representative based clustering

Implemented K-means algorithm. By default, it uses Eucledian distance.

To test results, unit test was written to compare results between built-in K-means algorithm and implemented function. K = 3 value was given up front. Unit test run for N = 10 000 times and result were the same ~90% of the time. If we trust built-in function, then it means that implemented function has flaws.

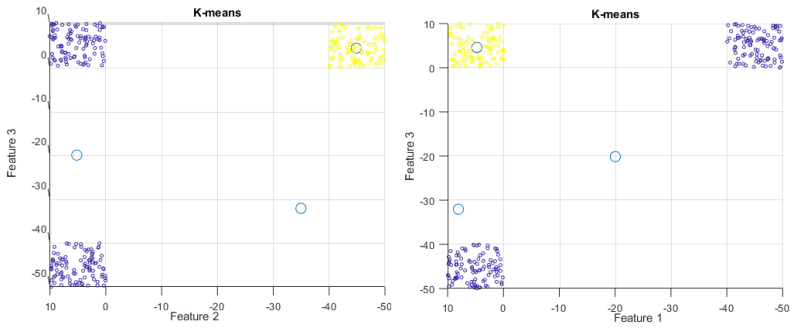

Figure 1 Problem 10% of the time with implemented K-means algorithm

Looking at Figure 1, it seems like implemented K-means algorithm finds first mean value just at the center of two clusters, which makes sense, because if we add two clusters together then mean of two clusters is indeed between two clusters. Second mean finds its center at the yellow cluster. Third mean remains lonely. K-means algorithm clusters correctly 90% of the time.

Exercise 3. Density based clustering

Implemented DBSCAN algorithm. By default, it uses Eucledian distance.

Matlab does not have built-in DBSCAN algorithm. To test results, unit test was written to compare results between DBSCAN algorithm written by Yarpiz (found on the Internet) and home assignment function. Maximum distance = 20 and minimum points = 10 values were given up front. Results were the same, so clustering worked the same for two separately developed algorithms. Figure 2 shows clustering using DBSCAN algorithm.

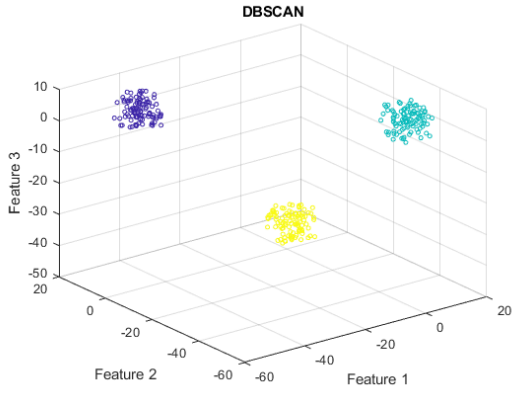

Figure 2 DBSCAN clustering

Exercise 4. Dataset generation

Generated data that has 3 features. In addition to scatterplot, implemented entropy function to describe data. Entropy function is not unit tested because did not find algorithms to compare results with. Figure 3 illustrates features and how values are distributed in selected range, it shows that there are clustering opportunities.

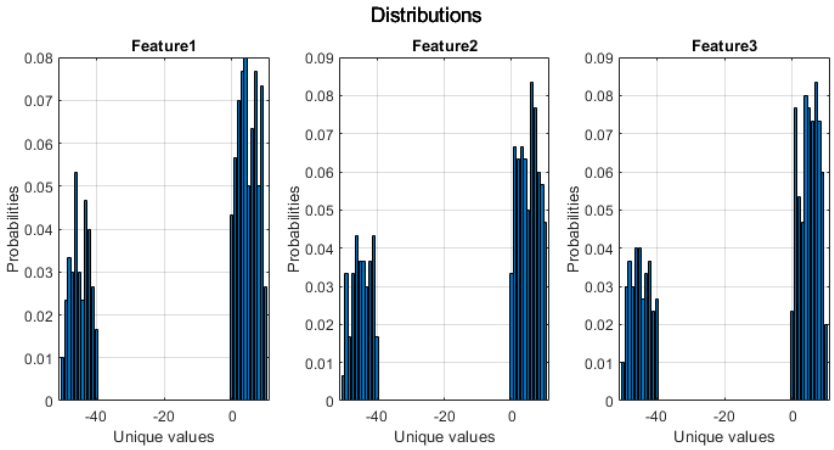

Figure 3 Relation between unique values and probabilities

Conclusion

Implemented 6 distance functions, then K-means and DBSCAN algorithms, generated data, calculated entropy and described distributions using bar plot. K-means algorithm needs improvement, because 10% of the time it clusters incorrectly.