Machine Learning in Data Science Project Week 8

Eva Aßmann, Paul Vogler

Dezember 2019

1 Data Generation

Six different data sets were generated using empirically determined parameters: Using the scikit-learn make_classification function, a non-cluster data set was generated using random state 2. The data set contained 1000 samples with two informative and non-redundant features. Two classes were predetermined which were separated by the factor 0.3, distributed equally over all data points and each held two clusters. There was no label exchange allowed between the two classes (Fig. 1).
A non-spherical data set was generated using the make_moons function from scikit-learn. Out of 1000 equally distributed samples two interleaving half circles of data points were scattered. Gaussian noise with standard deviation 0.1 was added to the data (Fig. 1).
A data set with many clusters close to each other was generated using the make_blobs function from scikit-learn. At random state 37, out of 1000 equally distributed samples isotropic Gaussian blobs were created around six centroids. Each sample was assigned 2 features and the standard deviation of the clusters was set to 1 (Fig. 1).

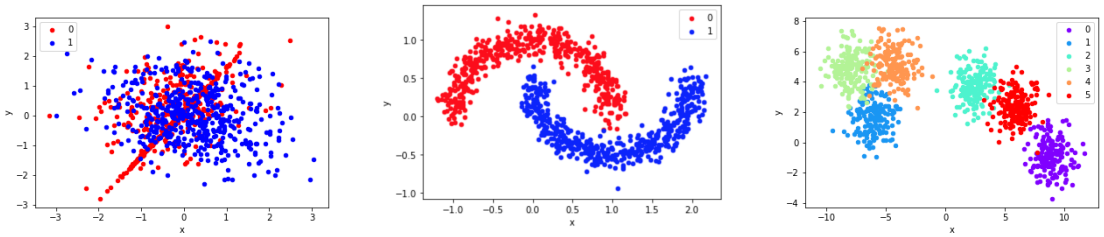

Figure 1: Generated data sets from left to right: non-cluster, non-spherical and many clusters.

Using the scikit-learn make_blobs function, a data set with clusters of different sizes was generated. Three blobs holding 750, 200 and 50 samples with two features each were created at random states 5, 12 and 43, respectively. Each cluster got a standard deviation of 0.8 (Fig. 2).
A data set with clusters of different densities was generated using the scikit-learn make_blobs function. Three blobs with 333, 333 and 334 samples and two features each were created at random states 16, 12 and 34, respectively. For the clusters standard deviations of 0.8, 1.2 and 0.3 were set (Fig. 2).
The sixth data set was generated using the scikit-learn make_blobs function and three predefined cluster centers [[1, 1], [-1, -1], [1, -1]]. The clusters were created defining a standard deviation of 0.4 from 1000 samples with 2 features. After generating the data set, the coordinate range was rescaled along the second dimension from (-2,2) such that a k-means clustering with k=3 would fail to cluster the data. A value range of -200 to 1000 was chosen empirically. K-means with k=3 clusters detected only one of the three clusters correctly, while the other two cluster centers were falsely positioned in the middle of the two remaining clusters (Fig. 3). Data generation yielded six different data structures that density-based clustering and self-organizing maps could be performed on and compared. For each data set, the information about the true labels was added, respectively.
Three additional data sets were generated for the application of the self organizing maps (SOM) only. A 3D

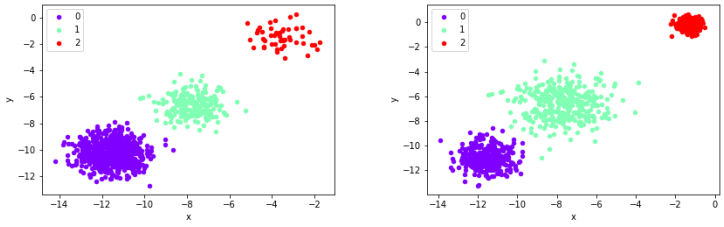

Figure 2: Generated data sets from top to bottom, from left to right: clusters of different sizes and clusters of different densities.

data set was generated using the sklearn function make_blobs for three clusters around the points (5,5,0), (-5,-5,5), (5,-5,-5). Also a 4D set using the same function around (5,5,0,0), (-5,-5,5,5), (5,-5,-5,-5) as center points, and a 5D data set using (5,5,0,0,0), (-5,-5,5,5,5), (5,-5,-5,-5,-5) as center points were created. All of them consisted of 750 samples and used a standard deviation of 0.4.

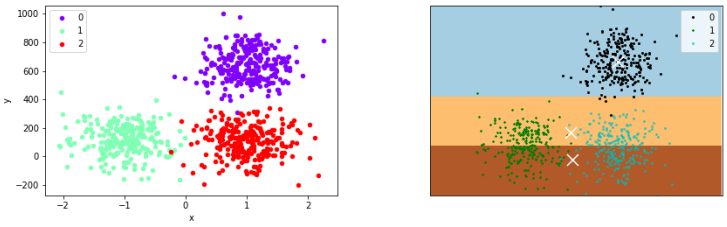

Figure 3: Generated data sets from left to right: 3 clusters with rescaled second dimension, k-means with k=3 applied to the rescaled data set.

2 Density-based Clustering

Density-based clustering was applied to the six generated data sets using the DBSCAN algorithm of scitkit-learn with different preprocessings and parameter settings. As input parameters for the DBSCAN algorithm three input values were required: the maximum distance ϵ between two samples for one to be considered as in the neighborhood of the other and the number of samples min_samples in a neighborhood for a point to be considered as a core point, as well as the metric to use when calculating distance between instances in a feature or dimension array. The best values for ϵ with min_samples=4 for each data set was determined based on the recommendations by Ester et al. [1] (Fig. 4). This yielded ϵ values of 0.3 (non-cluster), 0.1 (non-spherical), 0.5 (many-clusters), 0.3 (diff-sizes), 0.3 (diff-densities) and 5 (kmeans-fail).
The clustering results were evaluated by number of detected clusters, number of detected noise points and three more measures to compare the clusterings to the ground truth. The adjusted rand index (ARI) measures how similar the two ways of partitioning the data are and ranges from 0 to 1, if they match perfectly. The silhouette index (SI) measures the tightness within the clusters and the separation from neighboring clusters and ranges from -1 to 1, which is the ideal value. For the calculation of the SI, at least two different labelings are required for the clustered data. IF a clustering detects all points as one cluster without any noise, no SI score can be calculated. A clustering's entropy measures the amount of uncertainty within a cluster when comparing the predicted and actual labeling and equals 0 when these match perfectly. The compactness measures the spatial tightness of points within a cluster and is better the smaller its value gets.

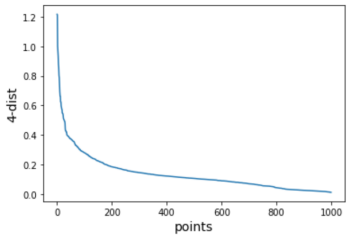

Figure 4: For all points in a set, the min_samples=4 nearest neighbors are calculated. The points are sorted by descending maximum distance and plotted as sorted 4-dist-graph (here for non-cluster). The best ϵ is the threshold point with the maximal distance value for min_samples=4 in the "thinnest" cluster, i.e. the first valley of the sorted 4-dist graph.

A

First, the data sets were clustered with min_samples=4, the derived ϵ values and the euclidean metric (Fig. 5). The correct number of clusters was not detected for any of the sets, but DBSCAN got closest up to one for non-cluster, non-spherical and many-cluster, for which also the littlest noise was produced (Tab. 1, Fig. 5)). For non-cluster and many-clusters sets, the clustering was the most similar to the original partitioning. The tightness within each cluster and separation to neighboring clusters was not really good for any set, but the diff-densities set showed the highest value. DBSCAN showed the smallest average entropy for the clusters in the non-spherical and kmeans-fail sets. Non-spherical and non-cluster showed the best compactness.
Taken all together, the settings within this experiment produced the best overall clusterings for the non-spherical set, while they performed worst for the diff-sizes set.

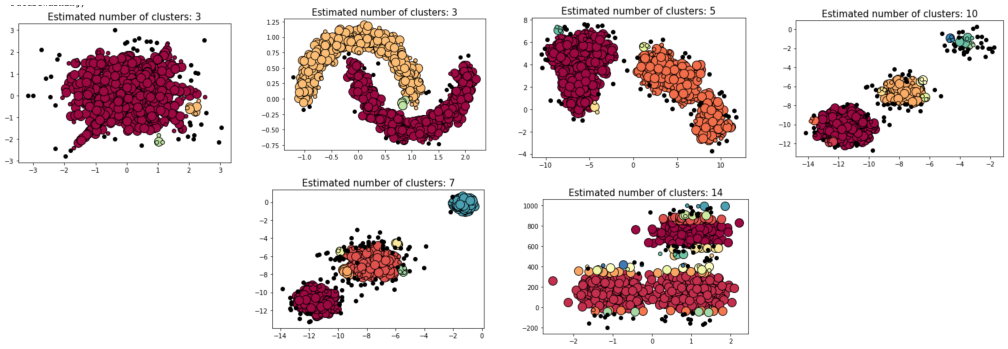

Figure 5: A. DBSCAN clustering results for all six data sets. From left to right, from top to bottom: non-cluster, non-spherical, many-clusters, diff-sizes, diff-densities, kmeans-fail.

Measure	non-cluster	non-spherical	many-clusters	diff-sizes	diff-densities	kmeans-fail
n_cluster	3	3	5	10	7	14
n_noise	43	21	56	78	95	59
ari	0.001	0.949	0.315	0.877	0.84	0.425
silhouette score	0.259	-0.045	0.034	0.126	0.732	0.088
entropy	-2.28	0.11	-1.93	-0.77	-1.34	0.34
compactness	0.59	0.47	7.68	1.96	5.78	2.63

Table 1: A. DBSCAN clustering quality for all six data sets.

B

The data sets were rescaled using the scikit-learn StandardScaler before beeing clustered with min_samples=4, the derived ϵ values and the euclidean metric. The correct number of clusters was detected for diff-sizes and deviated most for non-spherical (Tab. 2, Fig. 6)). There were no noise points detected for many-clusters, diff-densities and kmeans-fail, the most noise was detected for non-spherical. The clustering for diff-sizes was the most similar to it's original data partitioning and worst for many-clusters and kmeans-fail. The tightness within a cluster and separation to neighbouring clusters was not really good for any of the sets but was highest for diff-sizes and diff-densities and lowest for non-cluster. The smallest amount of average entropy was observed for the non-spherical set and diff-sizes set, which also was most compact. The worst entropy was shown for non-cluster, the worst compactness for many-clusters.
All together, the settings within this experiment produced the best clustering by far for the diff-sizes set, while non-cluster, non-spherical and many-clusters showed the worst cluster quality.

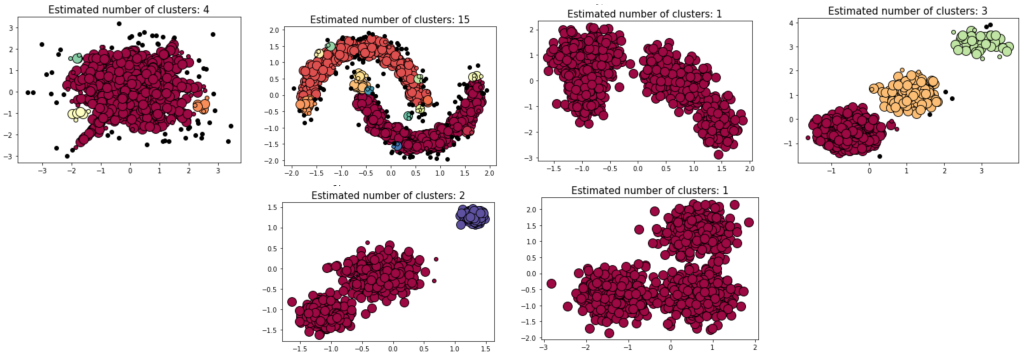

Figure 6: B. DBSCAN clustering results for all six data sets. From left to right, from top to bottom: non-cluster, non-spherical, many-clusters, diff-sizes, diff-densities, kmeans-fail.

Measure	non-cluster	non-spherical	many-clusters	diff-sizes	diff-densities	kmeans-fail
n_cluster	4	15	1	3	2	1
n_noise	55	106	0	6	0	0
ari	0.002	0.653	0.0	0.994	0.572	0.0
silhouette score	0.02	-0.455	N/A	0.720	0.749	N/A
entropy	-2.19	0.01	0.43	0.04	0.33	0.53
compactness	0.65	3.92	9.14	0.03	0.67	1.67

Table 2: B. DBSCAN clustering quality for all six data sets.

C

The data sets were clustered two times using min_samples=4, the derived ϵ values and manhattan and cosine metric, respectively. The experiment with cosine metric clustered all data points into one cluster without any noise points throughout all data sets, which is also why no SI values could be calculated (see explanation above) (Tab. 4). These clustering resulted in very bad ARI scores and undesirable entropy and compactness values. Looking at the clustering quality for the manhattan metric, for none of the data sets the correct number of clusters was detected (Tab. 3, Fig. 7)). The number got closest up to two for non-cluster and non-spherical and deviated the most for diff-densities. For all data sets, there was quite some noise predicted, the smallest amount for non-cluster, the biggest for diff-densities. The clustering for non-spherical got closest to it's original data partitioning and worst for non-cluster. The silhouette score was pretty bad for all of the sets. The entropy was the smallest for non-spherical and the biggest for non-cluster and diff-densities. The

cluster's compactness was best for non-spherical and worst for diff-densities.
The settings of this experiment produced the best clustering for non-spherical and the worst for diff-densities, but overall the clustering qualities were not good.

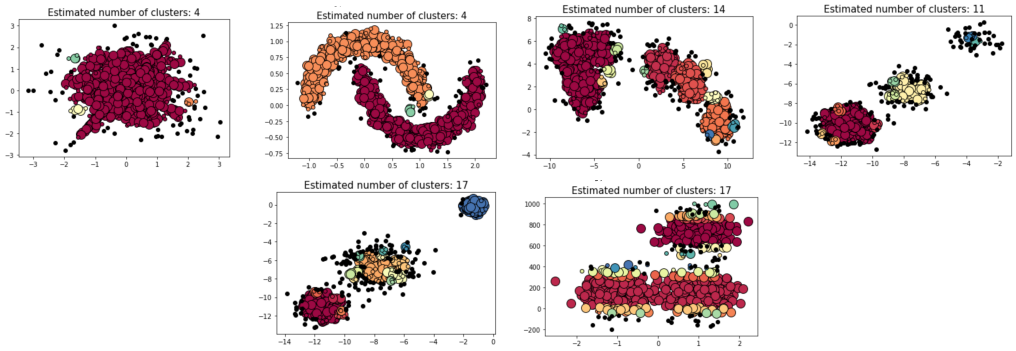

Figure 7: C. DBSCAN manhattan clustering results for all six data sets. From left to right, from top to bottom: non-cluster, non-spherical, many-clusters, diff-sizes, diff-densities, kmeans-fail.

Measure	non-cluster	non-spherical	many-clusters	diff-sizes	diff-densities	kmeans-fail
n_cluster	4	4	14	11	17	17
n_noise	66	50	107	126	133	66
ari	0.003	0.886	0.444	0.755	0.708	0.396
silhouette score	-0.01	-0.1	-0.06	-0.093	0.224	0.073
entropy	-2.59	0.24	-0.76	-0.42	-2.57	0.44
compactness	0.68	0.12	10.38	4.14	74.16	5.48

Table 3: C. DBSCAN manhattan clustering quality for all six data sets.

Measure	non-cluster	non-spherical	many-clusters	diff-sizes	diff-densities	kmeans-fail
n_cluster	1	1	1	1	1	1
n_noise	0	0	0	0	0	0
ari	0.0	0.0	0.0	0.0	0.0	0.0
silhouette score	N/A	N/A	N/A	N/A	N/A	N/A
entropy	0.5	0.5	0.43	0.31	0.53	0.53
compactness	0.5	0.5	9.14	0.4	1.67	1.67

Table 4: C. DBSCAN cosine clustering quality for all six data sets, with and without data scaling.

D

The data sets were again rescaled before beeing clustered two times using min_samples=4, the derived ϵ values and manhattan and cosine metric, respectively. The experiment with cosine metric clustered all data points into one cluster without any noise points throughout all data sets, which is also why no SI values could be calculated (see explanation above) (Tab. 4). These clustering resulted in very bad ARI scores and undesirable entropy and compactness values. Looking at the cluster quality for the manhattan metric, the correct number of clusters was not detected for any of the sets, got closest up to one for diff-sizes and

diff-densities and was worst for non-spherical (Tab. 5, Fig. 8)). No noise was detected for many-clusters and kmeans-fail, the most noise was detected for non-spherical. The clustering for diff-sizes was most similar to it's actual data partitioning and most dissimilar for many-clusters. The average tightness and separation score was 'best' for diff-sizes and worst for non-spherical. The smallest amount of entropy was shown for many-clusters, the biggest for non-clusters. Diff-sizes was most compact, non-spherical the littlest.
All together, the settings of this experiment performed best for diff-sizes and worst for non-spherical.

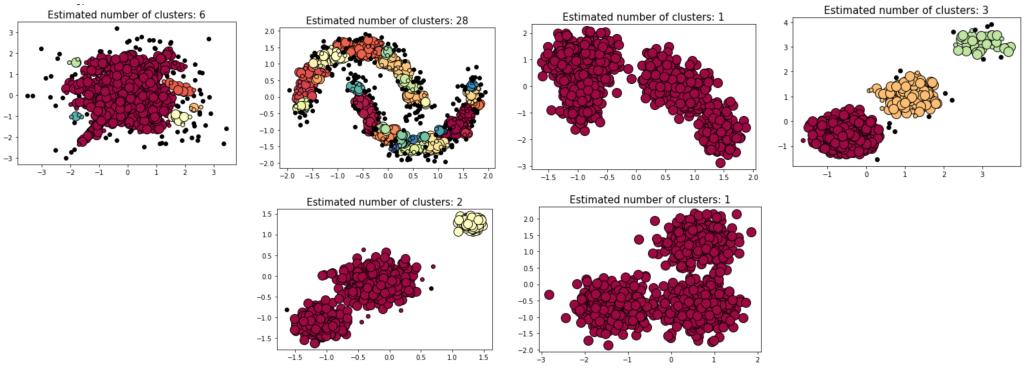

Figure 8: D. DBSCAN clustering results for all six data sets. From left to right, from top to bottom: non-cluster, non-spherical, many-clusters, diff-sizes, diff-densities, kmeans-fail.

Measure	non-cluster	non-spherical	many-clusters	diff-sizes	diff-densities	kmeans-fail
n_cluster	6	28	1	3	2	1
n_noise	80	195	0	15	2	0
ari	0.008	0.085	0.0	0.982	0.571	0.0
silhouette score	-0.067	0.018	N/A	0.709	0.526	N/A
entropy	-1.49	1.14	0.43	0.07	0.35	0.53
compactness	0.84	68.31	9.14	0.08	0.67	1.67

Table 5: D. DBSCAN clustering quality for all six data sets.

E

The data sets were rescaled and as an example for bad parameter setting, they were clustered with euclidean metric, a high number of min_samples=20 and a small degree of ϵ=0.1. In this experiment the cluster quality was pretty bad throughout almost all measures and data sets (Tab. 6, Fig. 9)). The 'better' measurement scores were shown for diff-sizes and diff-densities.

Measure	non-cluster	non-spherical	many-clusters	diff-sizes	diff-densities	kmeans-fail
n_cluster	4	0	1	1	4	0
n_noise	838	1000	965	529	266	1000
ari	0.04	0.0	0.01	0.15	0.74	0.0
silhouette score	-0.34	N/A	-0.08	0.24	0.44	N/A
entropy	-2.61	0.0	-0.38	0.21	-1.31	0.0
compactness	2.51	2.5	14.75	1.53	1.27	4.66

Table 6: E. DBSCAN clustering quality for all six data sets.

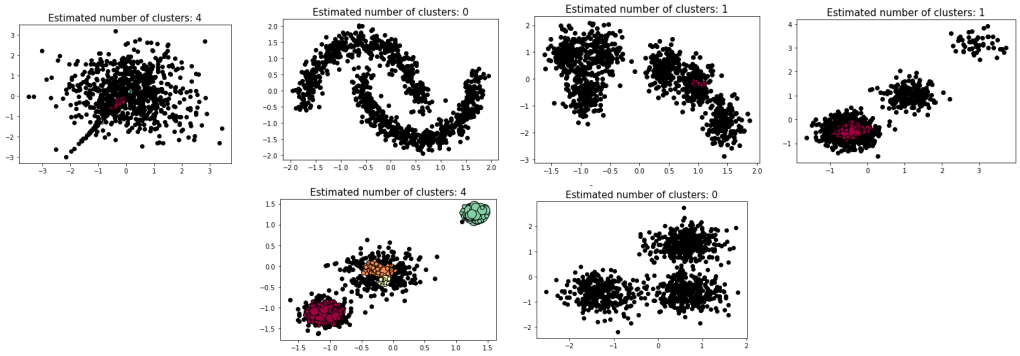

Figure 9: E. DBSCAN clustering results for all six data sets. From left to right, from top to bottom: non-cluster, non-spherical, many-clusters, diff-sizes, diff-densities, kmeans-fail.

F

The iris data set was loaded from scikit-learn. The data set stores feature information on 150 flowers which are categorised into four species. Each flower is described by four feature values which are sepal length and width as well as petal length and width. The best ϵ=0.5 for min_samples=4 was determined as for the data sets above. The iris data was clustered unprocessed and rescaled with min_samples=4, ϵ=0.5 and euclidean metric, respectively. The experiment without data scaling showed the better cluster quality throughout all measures (Tab. 7, Fig. 10)). While the species setosa was detected as one quite qualitative cluster, iris versicolor and virginica are not easy to distinguish and skew the overall clustering quality on the data set. Without data scaling, DBSCAN managed to separate the latter two species at least by three instances.

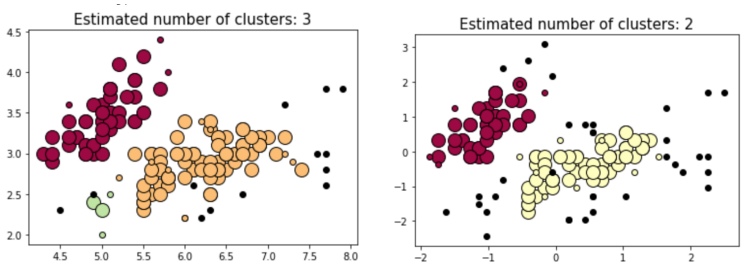

Figure 10: F. DBSCAN clustering results for the iris data set. Left: no data scaling, Right: data scaling

Measure	no data scaling	data scaling
n_cluster	3	2
n_noise	13	33
ari	0.526	0.447
silhouette score	0.381	0.365
entropy	-0.47	0.52
compactness	0.95	1.59

Table 7: F. DBSCAN clustering quality for the iris data set.

In conclusion, density-based clustering with best parameter settings for ϵ and min_samples in case of unscaled data produced qualitative clusterings for non spherical data, while ,in case of data scaling, it worked better for more distinctly grouped data like data clouds of different sizes or densities. Data scaling made

DBSCAN perform very coarsly and detect too little clusters for most of the data sets, except for non-spherical data. While the choice of manhattan metric instead of euclidean metric did not change clustering quality significantly within, cosine metric made the clustering significantly worse for all data sets.

3 Self-Organizing Maps

The minisom package was used to apply a self organizing map (SOM) to the three data sets generated during data extension, to the non-spherical 2D data set and to the iris data set as a real world example. For each set a 7x7 SOM grid with 2 to 5 dimensions and a learning rate of 0.5 was visualized using three different plots. For the 3D data set additionally a 10x10 grid, a 20x20 grid and a 3x3 grid were applied. The three visualisations used are a grey-scaled U-Matrix, that shows short distances between network nodes in light colors and contains markers for the original classes, a class assignment plot using multiple pie charts to show the class distribution per U-Matrix sector and the last plot that shows a color-scaled U-Matrix. The clustering was evaluated using four metrics: the Silhouette score, that measures cluster distinction based on the squared error in and between the clusters, the Adjusted Rand Index (ARI), that shows how similar the original labels and the predicted ones are, overall entropy, that measures the cluster entropies by using the proportion of correct and incorrect class assignments, and compactness, here measured by the mean of the squared error of each cluster.

For the 2D data set the points were well classified with an ARI of 0.98 but the distance between clusters was not significant, resulting in a low Silhouette score (Tab. 8). The Entropy and Compactness also show that a few clusters contain wrongly assigned classes to a cluster group of a dominant other class in the same region (Fig. 11). A slight resemblance of the moon shapes can also still be seen in the plots with added classes.

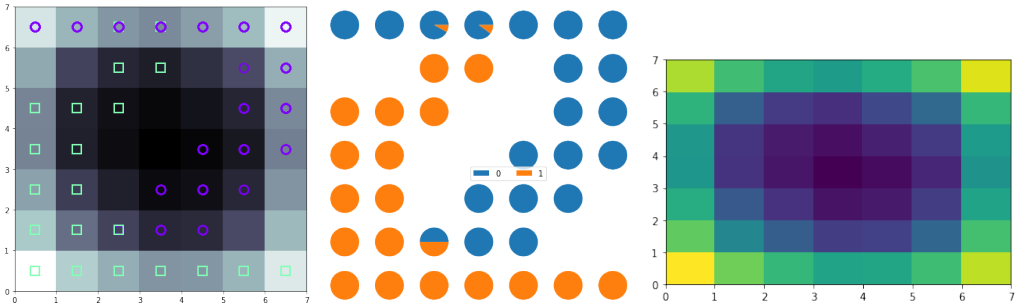

Figure 11: U-Matrix with added classes, class distribution in pie charts per U-Matrix region and the U-Matrix in color for the 7x7 SOM on the 2D non-spherical data set.

For the 3D data set the points were perfectly classified, resulting in an ARI of 1 and 0 Entropy and Compactness for the 7x7, 10x10 and 20x20 grid (Tab. 8). The Silhouette score was also very high and the same between the three grids. But for the 3x3 grid the predicted label are highly different from the original classes with an AR of only 0.57. This is because two clusters fell entirely into one of the few quadrants of the SOM and one cluster is therefore overshadowed by another class, resulting in only 2 clusters (Fig. 15).
For the 4D data set the points were perfectly classified (Fig. 16), resulting in an ARI of 1 and 0 Entropy and Compactness for the 7x7 grid (Tab. 8). The Silhouette score was also high showing that the clusters are very distinct.
Similarly to before, for the 5D data set the points were perfectly classified (Fig. 17), resulting in an ARI of 1 and 0 Entropy and Compactness for the 7x7 grid (Tab. 8). The Silhouette score was again high indicating that the clusters are very distinct.
For the iris data set the points were correctly classified into three clusters, but with multiple map sectors where the class was not clear. The indecisive regions were always between class 1 and 2 (corresponding to the flower petals versicolor and virginica) (visible in the first two plots of Fig. 18). Class 0 (corresponding

Metric/Dataset	2D	3D 7x7	3D 3x3	3D 10x10	3D 20x20	4D	5D	Iris
# clusters	2	3	2	3	3	3	3	3
Silhouette score	0.331	0.925	0.702	0.925	0.925	0.921	0.916	0.516
ARI	0.984	1.0	0.571	1.0	1.0	1.0	1.0	0.745
Overall Entropy	0.037	0.0	0.333	0.0	0.0	0.0	0.0	0.299
Compactness	0.004	0.0	1.333	0.0	0.0	0.0	0.0	0.1

Table 8: Clustering evaluation using different metrics for the datasets with SOM.

to the petal of setosa) was correctly classified. This could be a result of that the two flower petals that show overlaps (versicolor and virginica) have similar petal shapes, while the setosa petal has a distinctly different shape. The ARI score was high at 0.75, but the Silhouette score showed that the clusters aren't very distinct. The few classification errors between versicolor and virginica also show in the Entropy and Compactness score of 0.3 and 0.1 respectively. (Tab. 8).
Overall all maps other than the 3x3 map on the 3D data set were aligned well to the data and estimated the data classes accurately. Some estimations like for the 2D and iris data sets might work better with higher grid resolution. Only for the 3x3 map there were not enough meshes in the grid to align the SOM accurately to the data clusters.

4 Discussion

DBSCAN clusters points from multidimensional data in a hierarchical manner using a bottom up approach and uses a distance measure as well as a condition to expand the clusters from randomly chosen points. It worked well for clusters with enough distance between them regardless of shape, but is highly susceptible to parameter and scaling changes. SOMs can also be applied to multidimensional data sets and try to fit a lower dimensional map of neurons on the data, also using a distance measure, but additionally using dimensionality reduction techniques. It does not classify by itself, but the most common class in a sector of the SOM can determine the class of a cluster. In summary SOM aligns possible cluster center points to the data and provides a low-dimensional view, while DBSCAN is an algorithm that does the actual clustering.
The data sets that both DBSCAN and SOMs were applied to are the 2D non-spherical and the iris sets.
The non-spherical set was well clustered using DBSCAN with the euclidean and manhattan norm, but was a lot worse on the rescaled sets for both of them. The class distribution added to the SOM fitted the data good as well, except for the ends of the "moon arcs", where the groups overlapped. The SOM had overall the higher ARI score, was more distinct regarding the silhouette score. It also had a lot lower entropy and compactness.
For the iris data set DBSCAN and SOM found the desired number of 3 clusters on the standard data set, for the rescaled one DBSCAN only found two clusters. Here the SOM hat the higher ARI as well around 0.75 while the ARI for DBSCAN averages around 0.5. the silhouette score is also higher for SOM, showing a higher cluster distinction. SOM also has the lower entropy and compactess measures.
Overall the SOM with applied classes was better at differentiating the clusters.

References

[1] Ester, M., et al. (1996) A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.

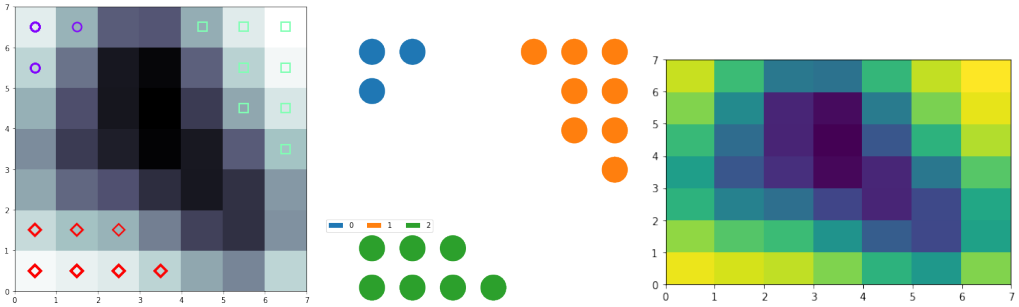

Figure 12: U-Matrix with added classes, class distribution in pie charts per U-Matrix region and the U-Matrix in color on the 3D data set for the 7x7 SOM...

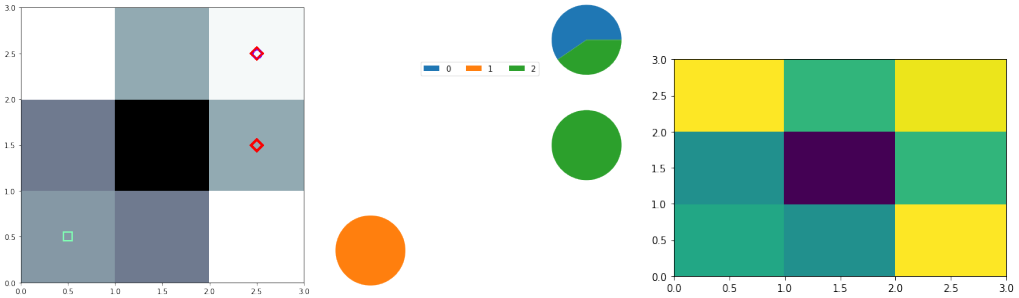

Figure 13: ...and for the 3x3 SOM...

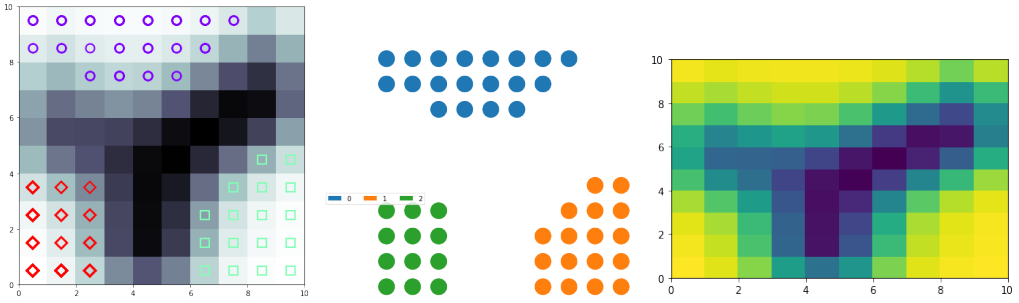

Figure 14: ..and for the 10x10 SOM...

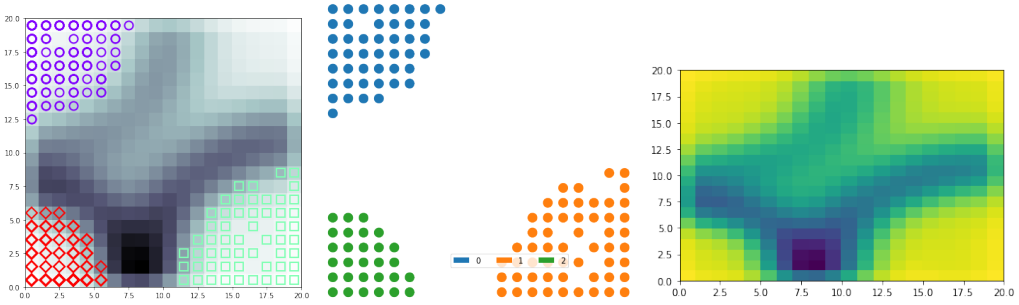

Figure 15: ..and for the 20x20 SOM.

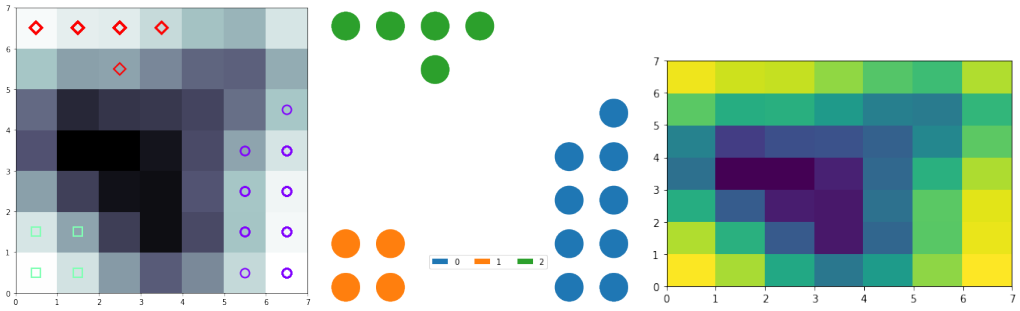

Figure 16: U-Matrix with added classes, class distribution in pie charts per U-Matrix region and the U-Matrix in color for the 7x7 SOM on the 4D data set.

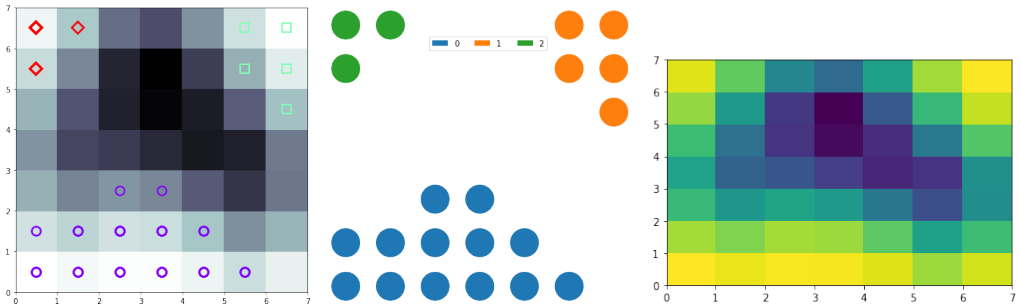

Figure 17: U-Matrix with added classes, class distribution in pie charts per U-Matrix region and the U-Matrix in color for the 7x7 SOM on the 5D data set.

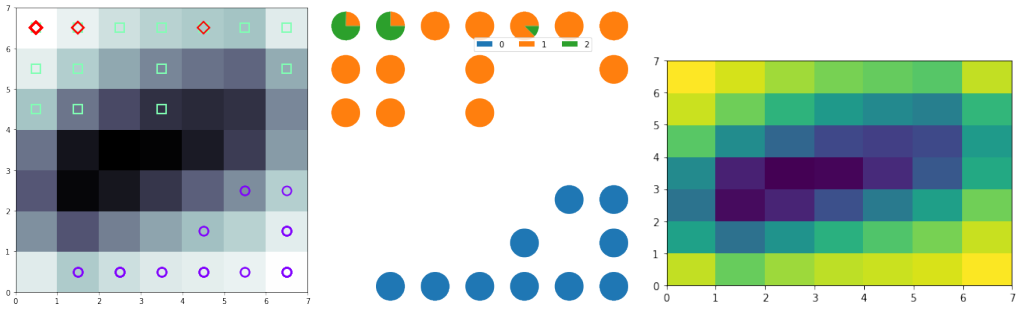

Figure 18: U-Matrix with added classes, class distribution in pie charts per U-Matrix region and the U-Matrix in color for the 7x7 SOM on the iris data set.