Data Mining Practical Assignment 5

Henrik Peters (6945965)

Task 1 – k-means:

- **What is the initialization procedure?**
  For initialization first of all a set of k cluster-centres has to be set. The value of k itself usually is set by the user since to find the optimal k is NP-hard. In the given code-snippet the chosen k is 3 with a sample size of 1500 points.
- **How does the algorithm find clusters?**
  Repeat for each data point in the sample:
  1. Assign the point to the cluster with the nearest center.
  2. Calculate the new mean of the cluster with the newly added point using a learning rate, so that the newly added point doesn't have a full impact on the center.
- **When does the algorithm stop?**
  The algorithm stops when the assignments of the cluster's centers don't change anymore. So basically, until the centers have converged into their best position.

Task 2 – best k:

The best number of clusters for 'make_blobs(n_samples=5000, cluster_std=2.5, random_state=42)' is 3. This is because the sum of squared distances is still quite high, while the silhouette coefficient is the highest at k = 3.

Task 3 – clustering comparisons:

- Results:
  - K-Means has problems to detect breaks between clusters. It just takes the distance as an input. Therefor the two circles and the two moons are not separated correctly. Also, the swiss roll is not clustered correctly.
  - Agglomerative clustering also sometimes has trouble to find the correct clusters. For all 3 data inputs 'single' worked best as a parameter.
  - DBSCAN gave good results for all data inputs right away.
- Differences:
  - Agglomerative clustering: the algorithm starts with treating every data point as a single cluster. These clusters are then merged with each other. 2 clusters can merge if they are the 2 closest to each other as well as the distance is not too high. The distance between 2 clusters can be calculated in different ways as the parameters of the function already tell (single, average, complete).
    The biggest difference to k-means is that the agglomerative clustering is not dependent on one distance measurement (mean of cluster) but can use several distance measurement methods. Also, this algorithm has no problem with finding the correct number of k's. If a data point is too far away to merge it with cluster 1, it is the start of a new cluster.
  - DBSCAN: The Density-Based-Spatial-Clustering-of-Applications-with-Noise algorithm is based on the idea of expanding the boundary of a cluster. It starts at a random unvisited data point and expands this 'cluster' into its neighboring data points, if the density is high enough. Density is defined via a radius around each point. If a point can reach a previously defined minimum of points it is considered a dense point in the cluster. If data point can be reached by a dense

point, but itself is not dense, it is considered a reachable point. If a data point is too far away from a cluster border it is classified as noise and will not be part of the cluster.

The difference of this algorithm to k-means is that is doesn’t just use the distance between a point and a cluster, but also takes into consideration the density of a cluster.

Task 4 – self-organizing maps:

- **What is the initialization procedure?**
  First of all, we need to define all input-parameters like the dimensions of the grid, the sigma value. We also need to initialize the grid itself as well as its weights. This is usually done with random values.
- **How does the algorithm determine the best matching unit? Give also an intuitive idea of this concept.**
  A self-organizing map is similar to a neural network. Just that the neurons of a layer are also connected with each other, more precisely with a neurons neighbor. If the network is now fed with an input representing several data points a “Winner”-neuron is found. This winner neuron effects not only deeper layers, but all its neighbors as well and partly activated them. This way the SOM slowly learns how cluster differ from each other. Each cluster activates a different neuron.
  Basically, a Neuron represents a prototype of a cluster and therefor gets activated when a similar cluster is fed into the network.
- **What is the role of the neighborhood function?**
  The neighborhood function determines the degree of weight change between a neuron and its neighbors. Gaussian or Mexican Hat function are mostly used for this. The weight also decreases during training which means the SOM converges towards k-means if it reaches zero.
- **What is the role of the learning rate?**
  The learning rate determines how fast the neurons converge to each other. So basically, it determines the influence a single new training data has.
- **When does the algorithm stop?**
  The algorithm terminates after a fixed number of repetitions. It has no fixed termination criterion. But for sure the is a way to set a termination when the change in weights is very low. Similar to the “early-stopping” method for CNNs.