4ME310 - Assignment 2
Intelligent techniques to identify patterns from data

Kamil Janeček

28.11.2019

In this assignment, we implemented 2 intelligent methods for clustering. Clustering is a type of unsupervised learning in which we divide objects into some groups (clusters). The similarity of objects in one cluster should be greater than the similarity of objects in multiple clusters.

There are multiple ways of deciding if two objects are similar. In the algorithms we implemented. The similarity is calculated using the distance between vectors of object features. There are multiple types of distances. In particular, we use

- Euclidean distance
- Squared Euclidean distance
- Manhattan distance

1 Clustering algorithms

We implemented these 2 algorithms:

- K-Means
- Partitioning around medoids (PAM)

The implementation was done in Python using numpy library for doing effective vector calculations.

1.1 K-Means

K-Means algorithm divides objects into **k** clusters. This number of clusters must be selected upfront. The algorithm is fast and simple with pretty good results. The steps are:

1. Take random k objects and deem them as current centroids
2. Assign every object to the closest centroid (by calculating the distance)

3. Find new centroids by calculating the new centre of each cluster
4. If no terminating condition is true then go to step 2.

There are multiple possible terminating conditions. In our implementation we implemented these:

- Limiting max iterations
- Inertia (average distance to centroid) change is lower than the tolerance value
- Centroids didn't change between iterations

There are multiple possible optimizations of this algorithm. One of them might be doing some intelligent selection of initializing centroids instead of random selection. As this algorithm is initialized randomly each execution might give different results. It's recommended to run it multiple times and select the best result (using some criteria, e.g. lower inertia value).

1.2 Partitioning around medoids

PAM algorithm is similar to k-means but in k-means, the centroids are averages of every point in the cluster in PAM it must be some object from the set. That means it has to be a real object not only an imaginary one. This makes PAM more robust to outliers than k-means.

We implemented the original PAM algorithm. Steps of the algorithm are:

1. Select k random objects as initializing medoids
2. Assign every object to closest medoid
3. Calculate the total cost of the configuration (sum of in-cluster distances to medoids)
4. While the cost decreases (or until max iterations are reached)
   (a) For each pair of (medoid, non-medoid) try to swap these and recalculate the cost
   (b) If the cost is lower than best-known cost then save this configuration

Just by reading the configuration we can see that this algorithm is much slower than k-means because instead of simply calculating new centroid we have to do $k(n-k)$ steps for selecting new medoids.

There are multiple optimizations of this algorithm but we implemented the original idea. The initial initialization of the algorithm is not really important as it will try all possibilities (if not interrupted). The implementation is generally nearly deterministic as if there are multiple medoid candidates with the same distance we will always select the first one. This algorithm is also called k-medoids.

1.3 Example

We created an artificial dataset to show how these algorithms work. Both of these algorithms are doing clustering around some point and that means they would not work if the shape of clusters is not round. For these kinds of data, we should use different algorithms e.g: DBSCAN.

We can see the importance of good initialization for k-means in figure 1 and also that PAM has the same results after each execution in figure 2. But we must note that PAM is much slower than k-means. In this example (2 features, 1200 observations) k-means finishes under 0.01 second but PAM takes nearly 5 seconds to finish. This might be a problem in our implementation (although it's optimized to use vectorized operations as much as possible) but it is also expected based on the complexities of the algorithms. Having this in mind running k-means multiple times and taking the best result might be better than using PAM but this is an artificial example with nicely formed clusters.

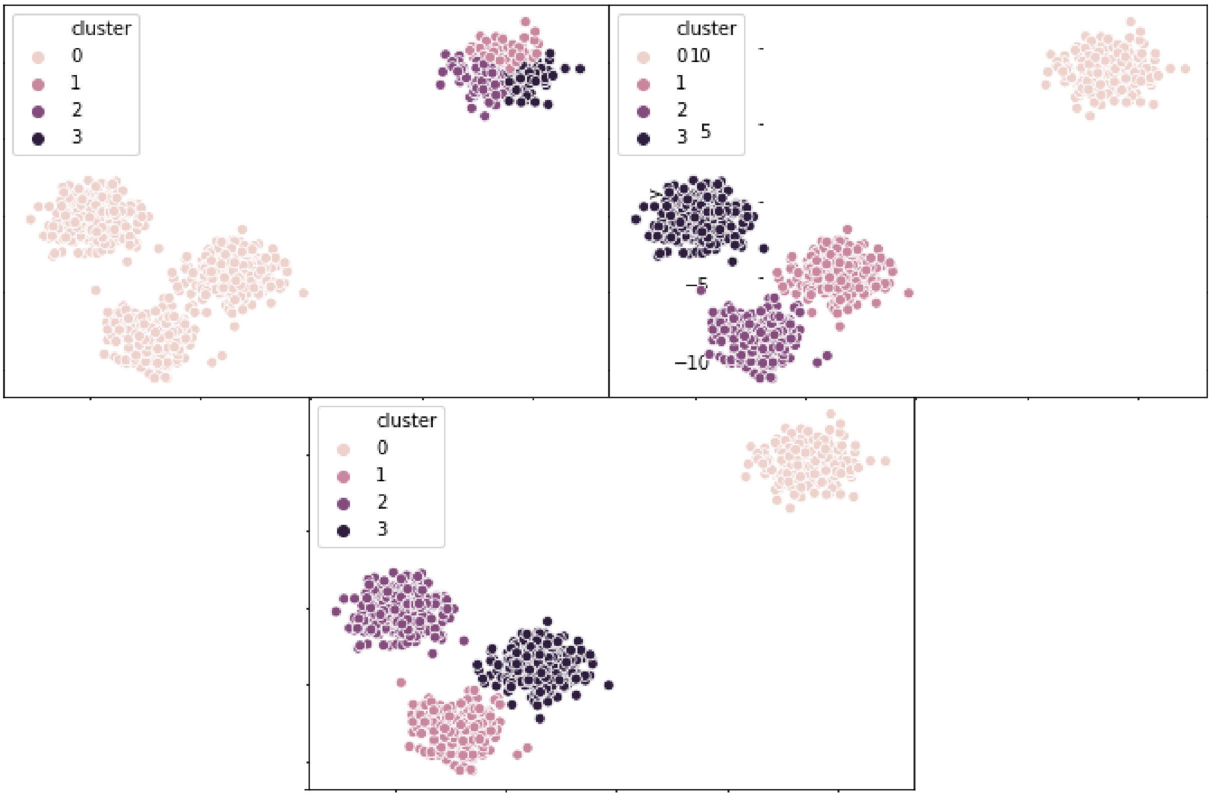

Figure 1: 3 executions of K-Means

2 Real example

We also tried out our implementations on a dataset obtained from Kaggle[1]. This dataset is an example of data which could be used to segment customers into multiple baskets based on some pieces of information. Although the quality of the data is not that high (only 3 usable features) and the clusters are clearly visible when visualizing (as there are only 3 features) we showed an

[1] https://www.kaggle.com/vjchoudhary7/customer-segmentation-tutorial-in-python

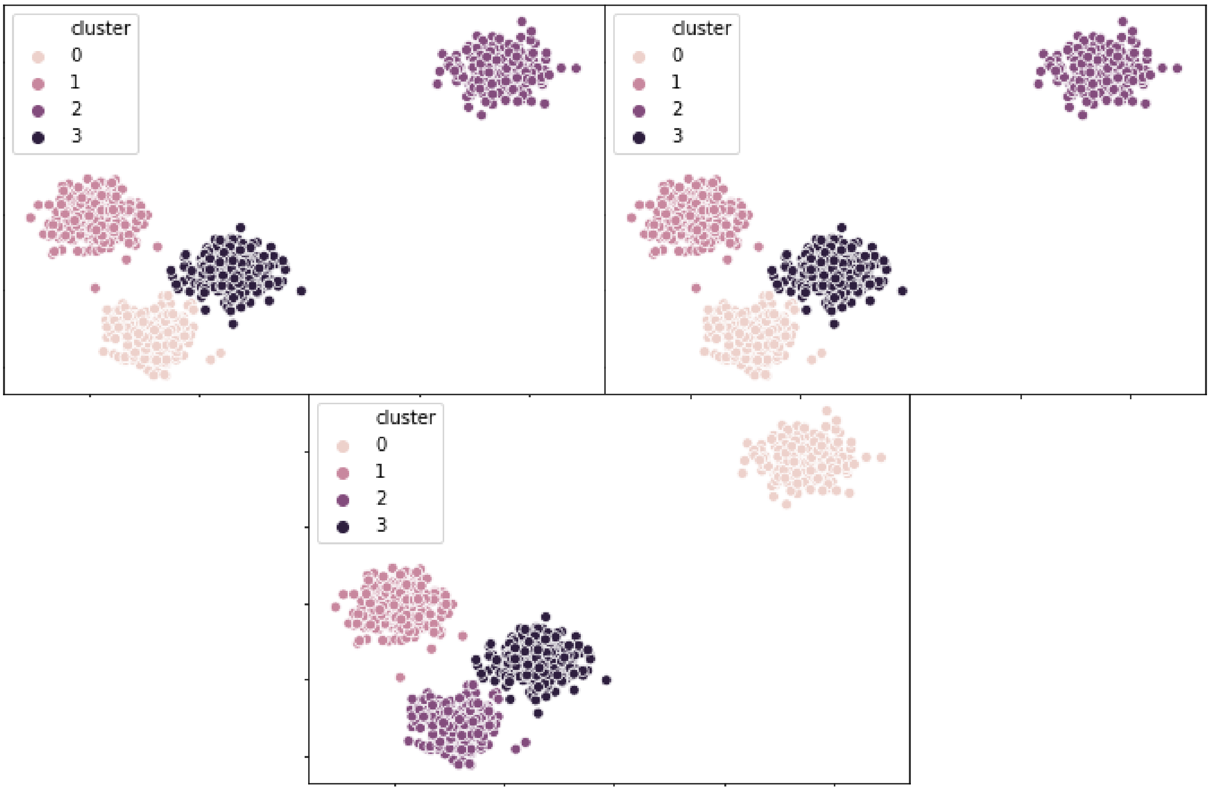

Figure 2: 3 executions of PAM

example of a clustering workflow. The whole workflow is commented and available in attached jupyter notebook and there is also a pdf export of it.

3 Conclusion

During this assignment, we implemented two simple unsupervised methods. Both methods are used for solving the clustering problem. We discussed the differences between these two methods and also tried simple machine learning workflow on a simple dataset.