# Data Science - Exercise 3 - Clustering

Student:
se21m024
Thomas Stummer

# Data Sets

## Toy Data Set

Data taken from:
https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html

## Big Data Set: Census Income

Data taken from:
https://archive.ics.uci.edu/ml/datasets/Census-Income+%28KDD%29

Data Original Owner:
U.S. Census Bureau
http://www.census.gov/
United States Department of Commerce

Donor:
Terran Lane and Ronny Kohavi
Data Mining and Visualization
Silicon Graphics.
terran '@' ecn.purdue.edu, ronnyk '@' sgi.com

## Small Data Set: Heart Disease

Data taken from:
https://archive.ics.uci.edu/ml/datasets/Heart+Disease

Data Creators:

1. Hungarian Institute of Cardiology. Budapest: Andras Janosi, M.D.
2. University Hospital, Zurich, Switzerland: William Steinbrunn, M.D.
3. University Hospital, Basel, Switzerland: Matthias Pfisterer, M.D.
4. V.A. MediMcal Center, Long Beach and Cleveland Clinic Foundation: Robert Detrano, M.D., Ph.D.

# Algorithms

Three clustering algorithms were chosen based on the requirement to work with at least one partitive and one hierarchical algorithm and the fact that those algorithms seemed to be good based on the overview that can be found at the scikit learn platform https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html :

- Mini-Batch k-Means (partitive)
- Agglomerative Clustering (hierarchical)
- DBSCAN (density based)

## Mini-Batch k-Means

For the required category of partitive clustering algorithms the Mini-Batch k-Means was selected because it is a variation of the k-Means that is suitable for large data sets. Small, randomly chosen batches are processed in each iteration instead of working on the whole data set at once.

## Agglomerative Clustering

For the required category of hierarchical clustering algorithms an agglomerative algorithm was chosen. It follows a bottom-up approach, meaning that all individual data points are assigned to an individual cluster, which are then connected to one another in hierarchical manner to form a new cluster until only the desired amount of clusters remains.

## DBSCAN

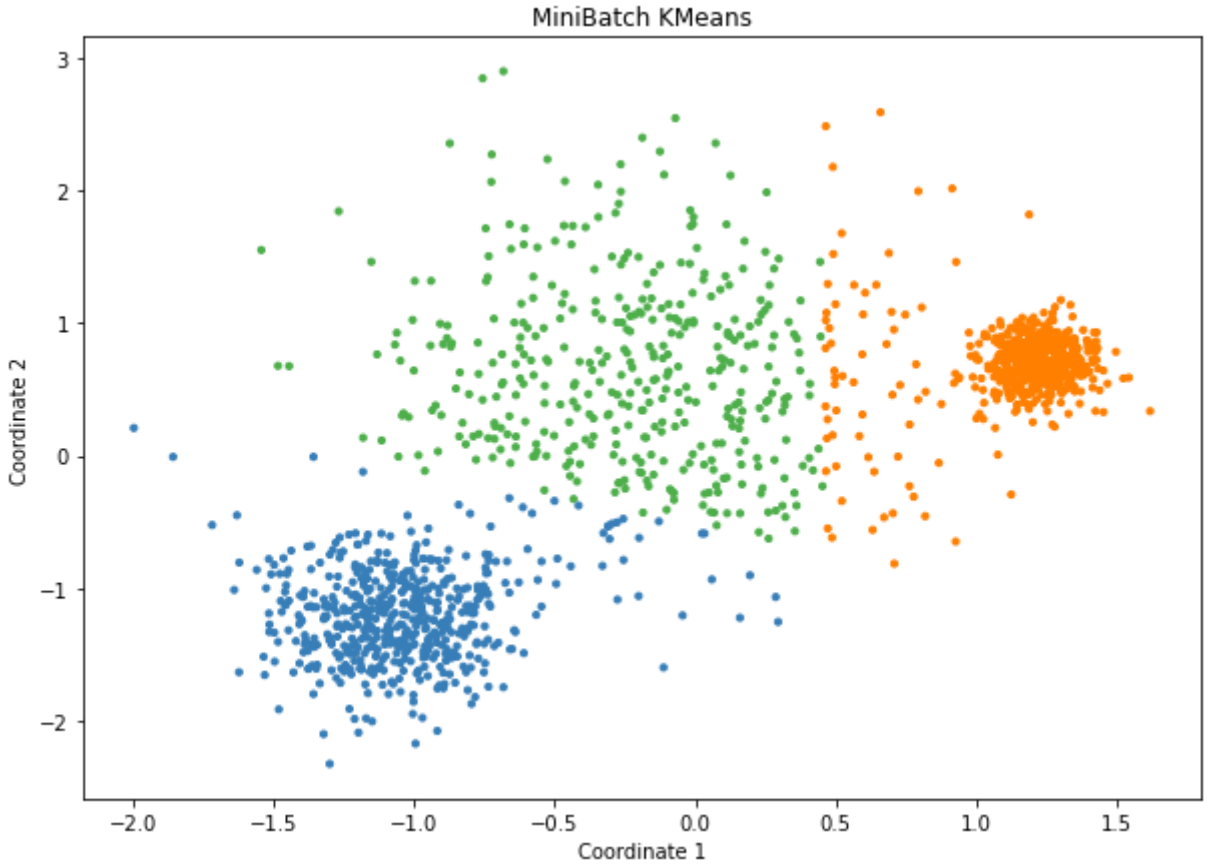
The 'density-based spatial clustering of applications with noise' (DBSCAN) algorithm groups together data points that are closely located to one another. Points with no close neighbour points are detected and marked as outliers. This algorithm was chosen for this exercise because the density based approach seems to have the potential to find structures that the two previously mentioned algorithms do not. A density based alternative would have been the 'ordering points to identify the clustering structure' (OPTICS) algorithm. The DBSCAN was favored for this assignment because it is significantly faster due to the scikit learn overview referenced above.

# Clustering Results

## Toy Data Set

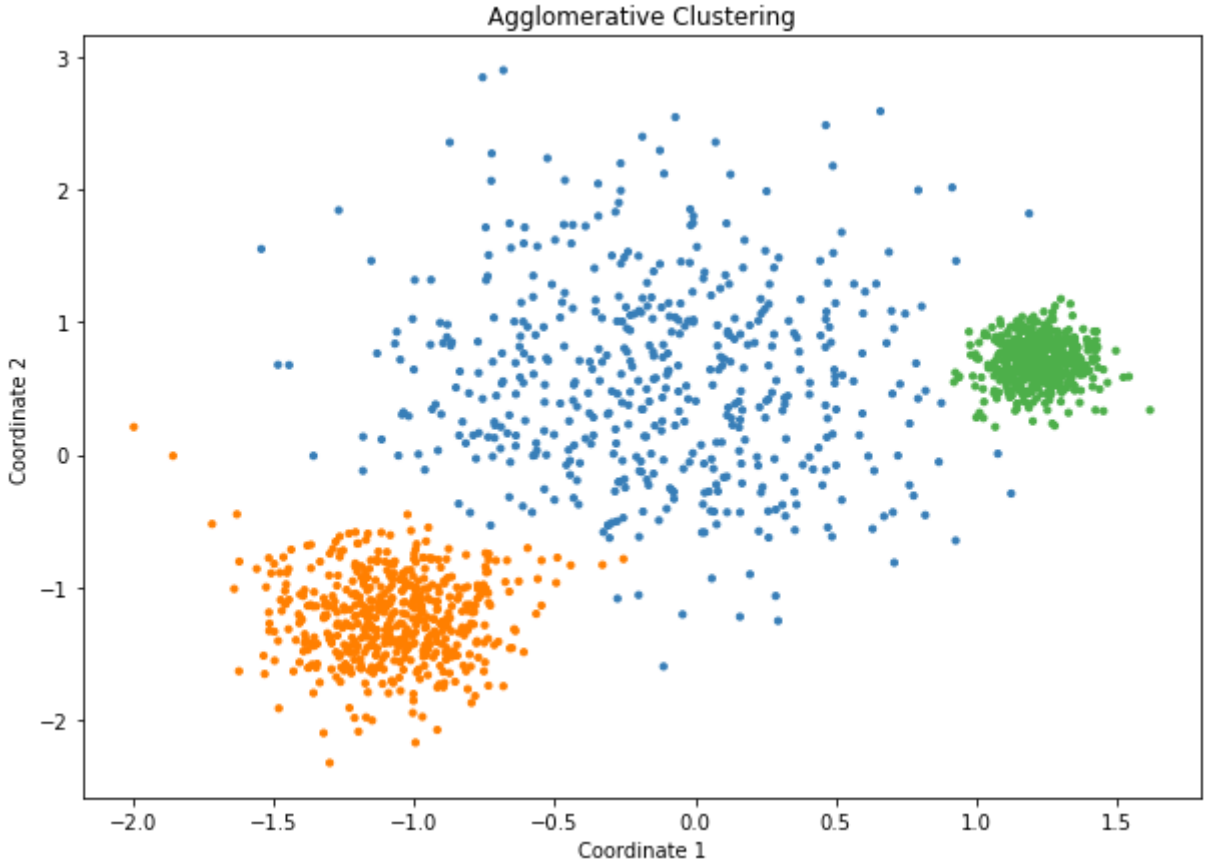
### Algorithm 1: Mini-Batch k-Means

Subjectively I would say that the toy data set can be best described with three clusters. One compact one on the left side, one compact one an the right side an a loosely coupled, big one in the middle. When setting three clusters as target, the Mini-Batch k-Means algorithm produced a result reflecting these expectations. All tightly connected points on the left side are fully captured by the blue cluster and all tightly connected points on the right side are fully captured by the orange cluster. For an ideal result, I would have expected that only the dense, ball-shaped structures are captured for the outer clusters. Not only but especially regarding the orange cluster it seems that many points that are very loosely connected to the orange main accumulation are still marked orange although they visually seem to be more suitable in the green middle cluster of generally looser connected points.



## Algorithm 2: Agglomerative Clustering

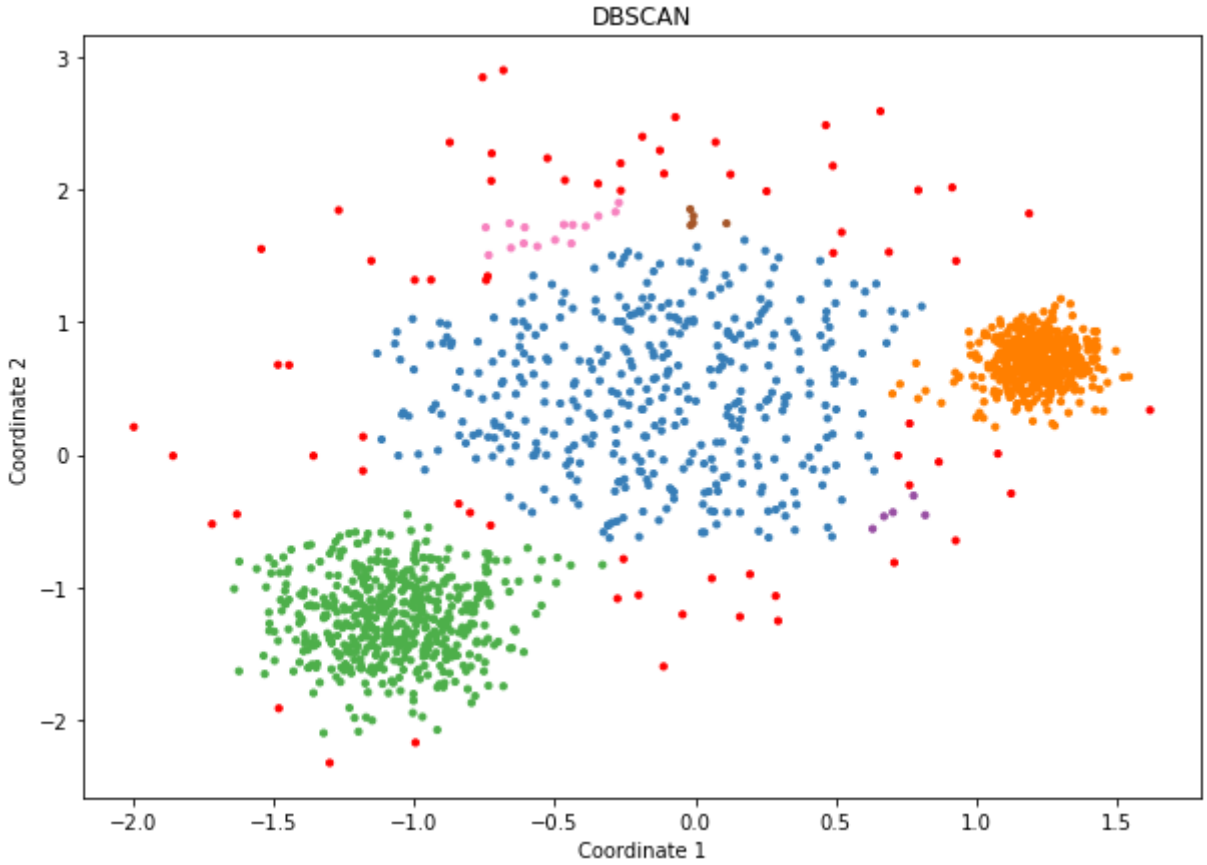
The agglomerative algorithm with ward linkage and euclidean metric produces pretty exactly the expected output. In comparison to the Mini-Batch k-Means algorithm nearly all loosely connected points are not part of the two outer clusters but of the blue cluster in the middle. Subjectively I would argue that this algorithm fits the data better than the first one.



## Algorithm 3: DBSCAN

The DBSCAN algorithm produces a result similar to the agglomerative clustering algorithm. The two main outer clusters are clearly separated to the less dense blue cluster in the middle. In contrast to the agglomerative algorithm, where the number of clusters is fixed, three additional clusters are detected. They are all very small compared to the three main clusters and can be seen in the top middle (pink and brown) and in the bottom right (purple). Furthermore, outliers that are far away from the main clusters are detected and marked in red.

Overall I would argue that the DBSCAN performs best in discovering the structure of the toy data set. The three main clusters are clearly captured and outliers are identified as such. If all data points should be assigned to one of the clusters, I would argue that the agglomerative clustering algorithm is the next best choice.



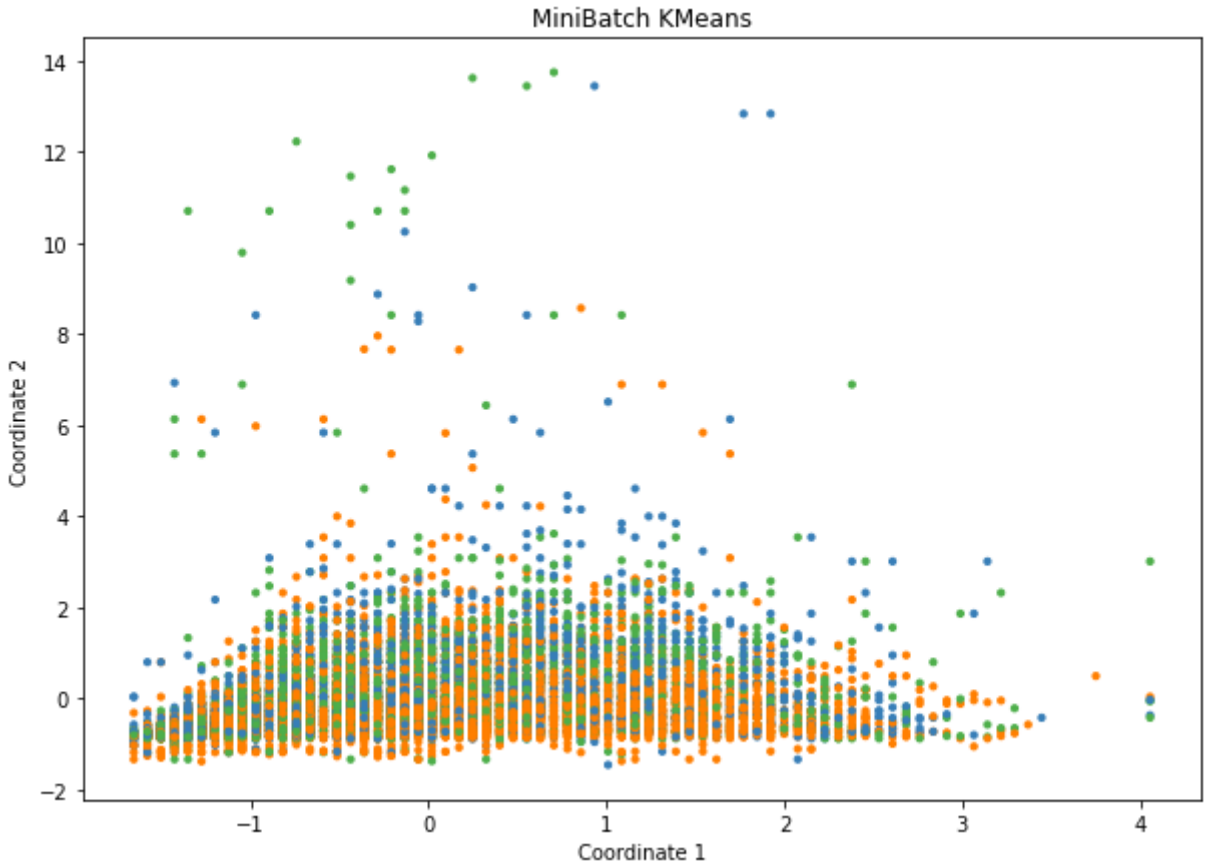
# Big Data Set: Census Income

Because the data projection exercise did not indicate a suitable number of clusters, various target numbers of clusters were tried. For the two algorithms requiring a number of clusters beforehand an amount of three clusters was chosen. Two clusters seemed to be to coarse-grained but four and five clusters seemed to make it even harder to deduce a meaningful interpretation of the clustering calculated that three target clusters already do. E.g. with more than three target clusters the agglomerative algorithm on the big data set made it harder to distinguish visually between points belonging to one or another cluster.

## Algorithm 1: Mini-Batch k-Means

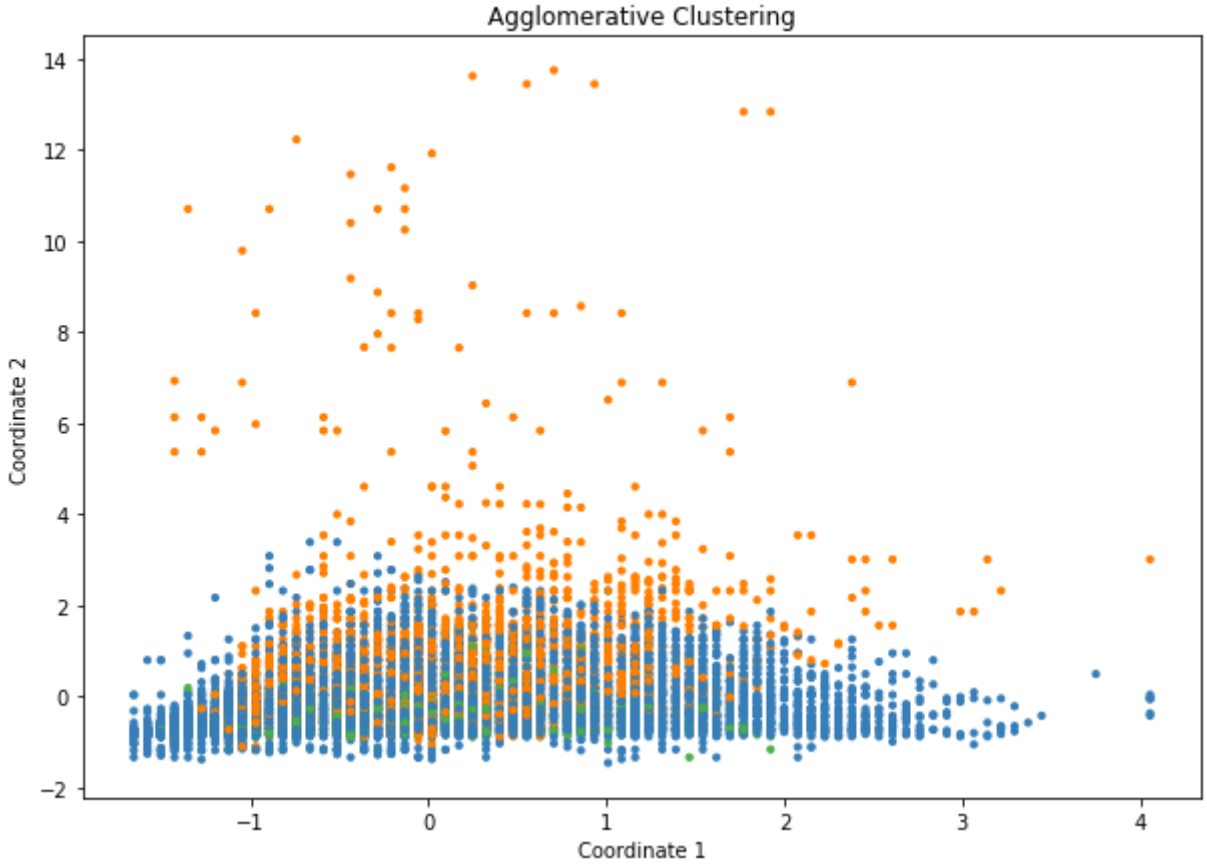
Although the clustering result is not visually satisfying it does not seem surprising. The plotted data does not indicate a distinct 'cut' through the data set along cluster boarders because the data set basically appears as one cohesive chunk with a couple of outliers. Nevertheless I would argue that the Mini-Batch k-Means algorithm did find some inner structure in the data. When e.g. inspecting the outliers on the top left of the plot, one can clearly see that they are all assigned to the green cluster. I therefore suppose that the green points indeed have one or more (hidden) attributes in common because cluster on the top but also in the main body there are plenty of green points. Orange points on the other hand only appear under a virtual border along the horizontal axis and are not present in the upper third of the plot at all. Also the orange points seem to be more common on the very bottom of the plot.

I experimented with plotting the data along different axis but was not able to find a view where the intra-cluster connection was comprehensibly observable.



## Algorithm 2: Agglomerative Clustering

The agglomerative algorithm with ward linkage and euclidean metric delivered a more satisfying result that the Mini-Batch k-Means. It clearly indicates that the loosely connected data points in the top are assigned to one (orange) cluster. The dense main structure on the bottom on the other hand does mostly contain data points assigned to the blue cluster. A data point in this main body seems to be more likely assigned to the orange cluster instead of the blue cluster, the higher it is located and the more close to the (horizontal) center it is. Additionally individual points in the mid center of the main body seem to form a green cluster.

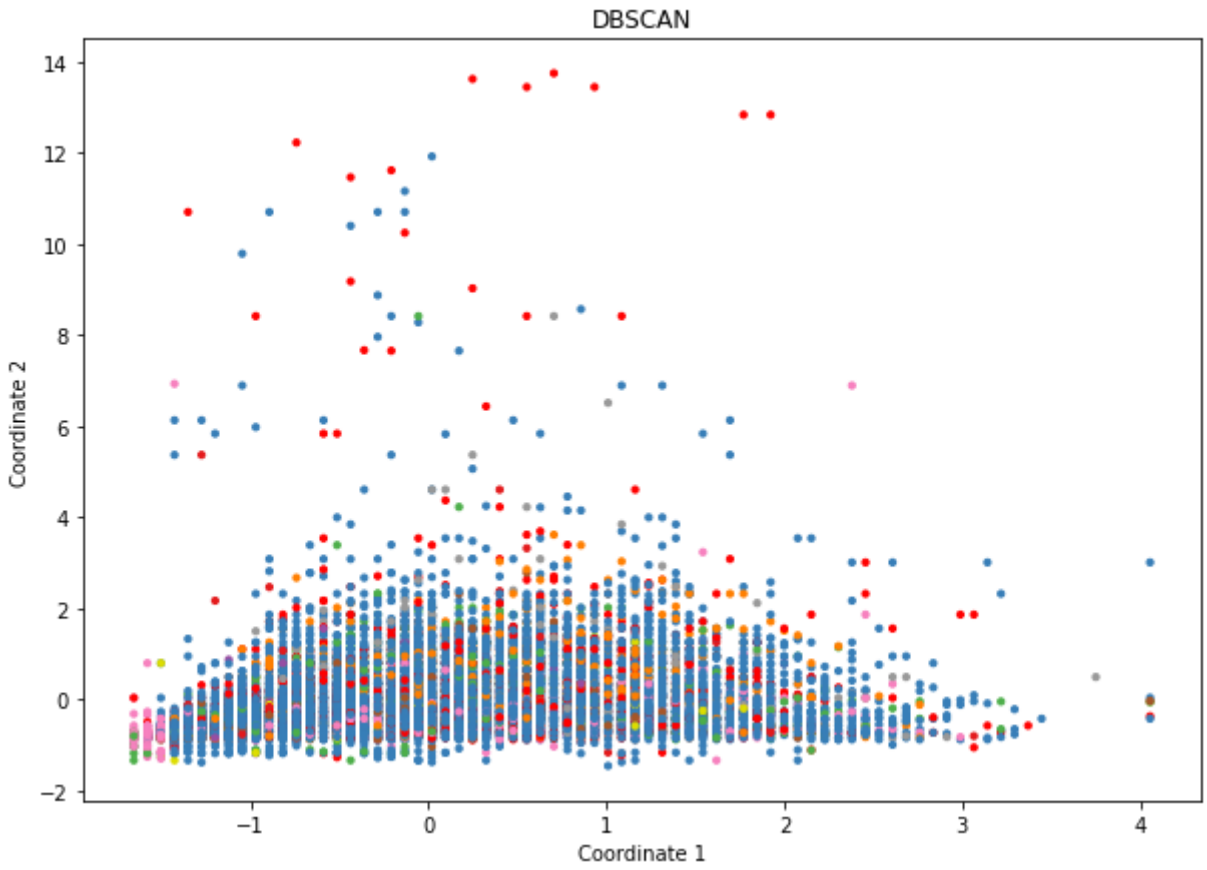


## Algorithm 3: DBSCAN

The DBSCAN algorithm did not deliver a result that provided any insights in the structure of the data set. It was actually quite challenging to find a combination of parameters that produced a clustering that was not either one mono-colored structure or a completely red plot, meaning that all points were considers as outliers. The parameters chosen for this plot were eps=7.0 (maximum distance between two points to be considered as neighbors), min_samples=20 (number of points in the neighbourhood of a point to be considered as core point) and metric="cityblock".

The result achieved is a blue cluster that contains the vast majority of the data points. Red outliers are spread all over the plot and various tiny clusters in can also be seen in different spots. The only reasonable non-blue cluster I could spot is a small pink cluster on the very left side.

Comparing all three algorithms, I would argue that the agglomerative clustering algorithm seems to be the most suitable one for this data set.

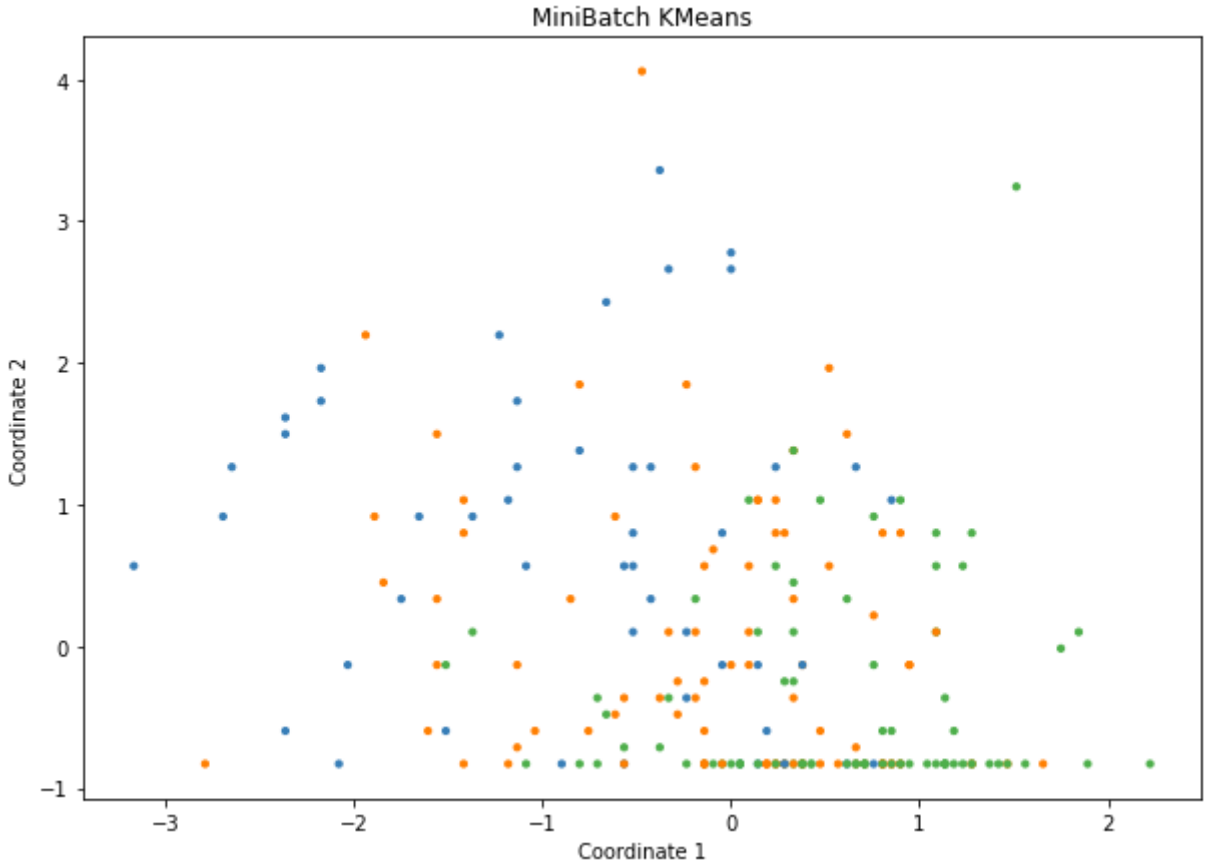


# Small Data Set: Heart Disease

## Algorithm 1: Mini-Batch k-Means

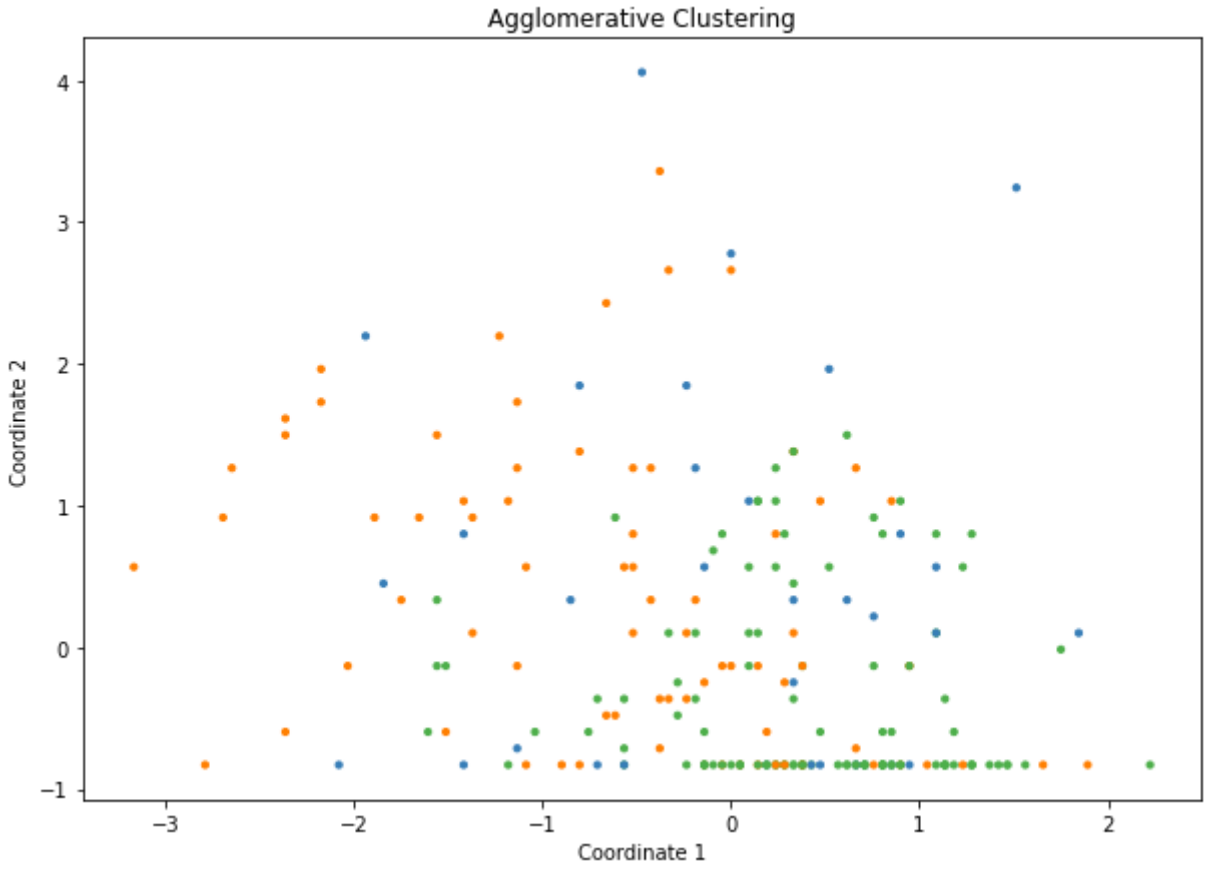
For the small data set I struggled to find a suitable view to visualize the clustering result. I ended up with the view that can be seen below. Similar to the big data set I decided to set the number of target clusters to three.

The overall structure appears to be least dense in the top and left of the plot and most dense in the bottom right. Analogous to this viewable progression of density regarding the chosen coordinates for the plot, points in the dense bottom right area are mainly assigned to the green cluster with one exceptional outliner at the very top right. In this area some points where assigned to the orange cluster. When considering the less dense area a mixture of orange and blue data points can be observed, with a progressing likelihood for a point to be blue, the more it is in the left and top area of the plot.



## Algorithm 2: Agglomerative Clustering

The agglomerative algorithm with ward linkage and euclidean metric proposed a quite similar structure of the data set as the Mini-Batch k-Means algorithm. Points in the bottom right are most likely assigned to the green cluster, especially in the very dense region at the very bottom. In the remaining area of the plot the orange cluster contains most of the data points. The blue cluster contains points that are distributed all over the plot but less points than the orange cluster (especially when compared to the Mini-Batch k-Means algorithm). This indicates that the algorithm the same dense structure in the top right as did the k-Means algorithm but delivers a distinguish result for the other two clusters.



## Algorithm 3: DBSCAN

Like the two algorithms mentioned before, the DBSCAN algorithm seems to put the vast majority of points located in the dense bottom right region into one (blue) cluster. Again points in the top left tend to be assigned to a different (orange) cluster compared to those in the bottom right. Red outliers are distributed all over the plot. The parameters used for this plot where eps=9.0, min_samples=5, metric="cityblock".

For the small data set I find it very challenging to vote for either one of the clustering algorithms applied because the results do not seem to differ that much.