Sondre Schirmer-Mikalsen, Arash Aminpour (Group 126)

Assignment 3: Clustering

Task 1: k-means Clustering

See added python-code

Task 2: Hierarchical Agglomerative Clustering (HAC)

2a)

Hierarhical Agglomerative Clustering is a clustering method which is performed in a bottom-up manner. Initially, all the data points are unique clusters, and they are later merged into larger clusters. The first step is to identify the closest pair of data points, and join them in a cluster. Then the process is repeated, and the second closest pair is clustered. One of the data points in the next closest pair is already in a cluster, the entire cluster is joined with the other one. This is why it is called hierarchical clustering - clusters are built in an arranged order/levels and can be broken down in order to find the closest pairs. The cluster joining is repeated until there is only one cluster remaining.

Both MIN-link and MAX-link join clusters in order based on which pair of clusters are closest to each other in terms of distance. The difference between them is that MIN-link compares the most optimal (shortest) distance between data points from one cluster to another, while MAX-link checks the least optimal (longest) distance.

2b)

We used EXCEL to represent the distances between the points as a distance matrix, with 0 on the diagonal, since the distance between for example A - B and B-A is the same. Further we calculate the distances between the data points using the euclidian distance function. These were written down in tables as shown in the images below. Then we followed the HAC clustering procedure step by step, by clustering the data points with the shortest distance.

MIN-LINK

table 1: shortest distance is A to D (1). A and D are clustered.

table 2: shortest distance is between cluster A,D and B. Distance between D and B is smaller than A to B, and is chosen.

table 3: shortest distance between clusters are from cluster A,D,B to C, and the shortest distance to C is from A (4.12).

table 4: finally, cluster A,D,B,C is clustered with E.

MIN-LINKS

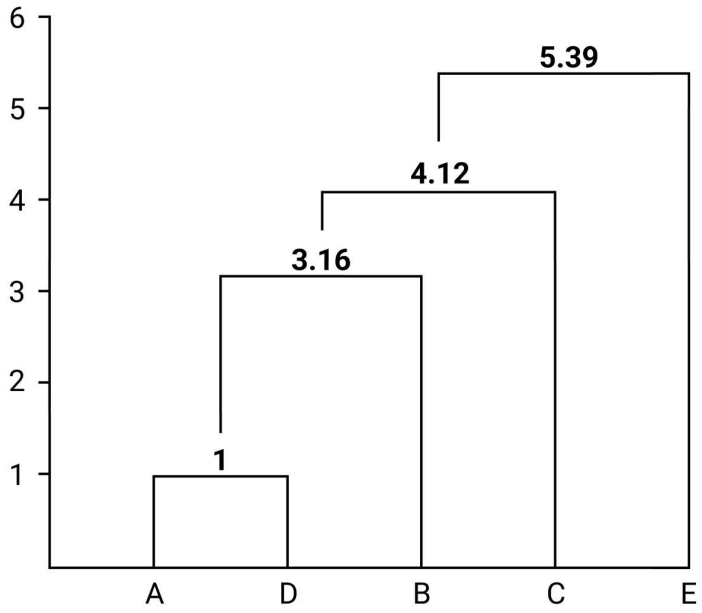
	A	B	C	D	E
A	0				
B	4,12310563	0			
C	4,12310563	7,07106781	0		
D	1	3,16227766	4,47213595	0	
E	7,21110255	7	5,38516481	6,70820393	0

	A,D	B	C	E
A,D	0			
B	3,16227766	0		
C	4,12310563	7,07106781	0	
E	6,70820393	7	5,38516481	0

	A,D,B	C	E
A,D,B	0		
C	4,12310563	0	
E	6,70820393	5,38516481	0

	A,D,B,C	E
A,D,B,C	0	
E	5,38516481	0

Dendogram for the MIN-LINK, (made in FIGMA):



MAX-LINK

The same procedure as with the Min-link was used. The only difference is that the updated distances between clusters are now based on the longest distances between data points in different clusters instead of the shortest one.

MAX-LINKS

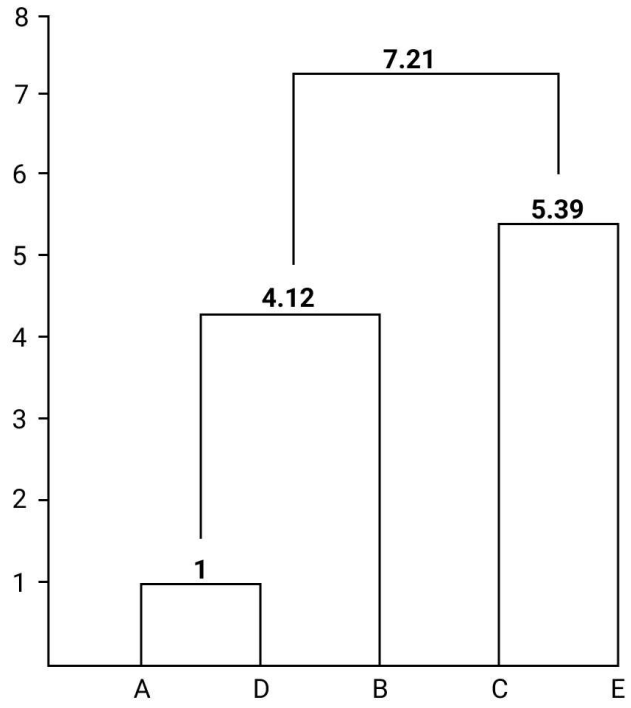
	A	B	C	D	E
A	0				
B	4,12310563	0			
C	4,12310563	7,07106781	0		
D	1	3,16227766	4,47213595	0	
E	7,21110255	7	5,38516481	6,70820393	0

	A,D	B	C	E
A,D	0			
B	4,12310563	0		
C	4,47213595	7,07106781	0	
E	7,21110255	7	5,38516481	0

	A,D,B	C	E
A,D,B	0		
C	7,07106781	0	
E	7,21110255	5,38516481	0

	A,D,B	C,E
A,D,B	0	
C,E	7,21110255	0

Dendogram for the MAX-Link, (made in FIGMA):

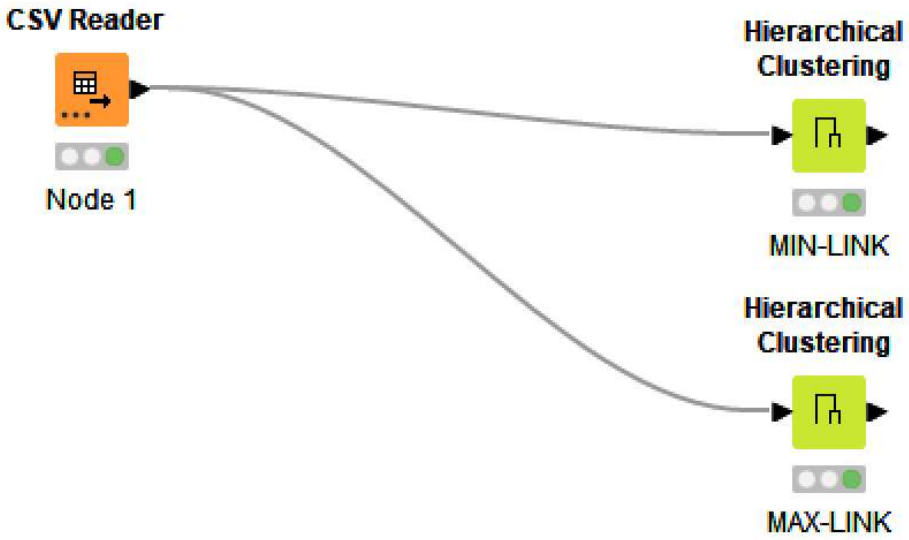


2C)

We created the following workflow in KNIME to verify our results. CVS Reader used to import the given data points. Further we added two hierarchical clusters, one for the MIN-Link and one for the MAX-Link. From the resulting dendograms given this workflow we can see that they are the same as the ones we made in Figma.

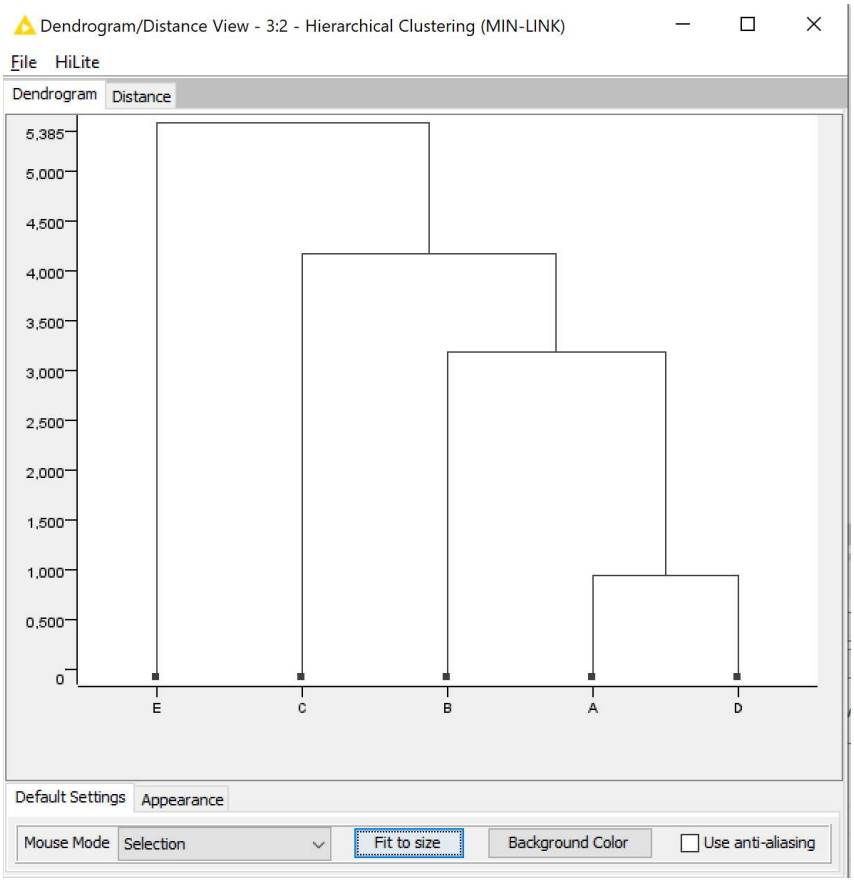
Sondre Schirmer-Mikalsen, Arash Aminpour (Group 126)

Workflow made in KNIME:

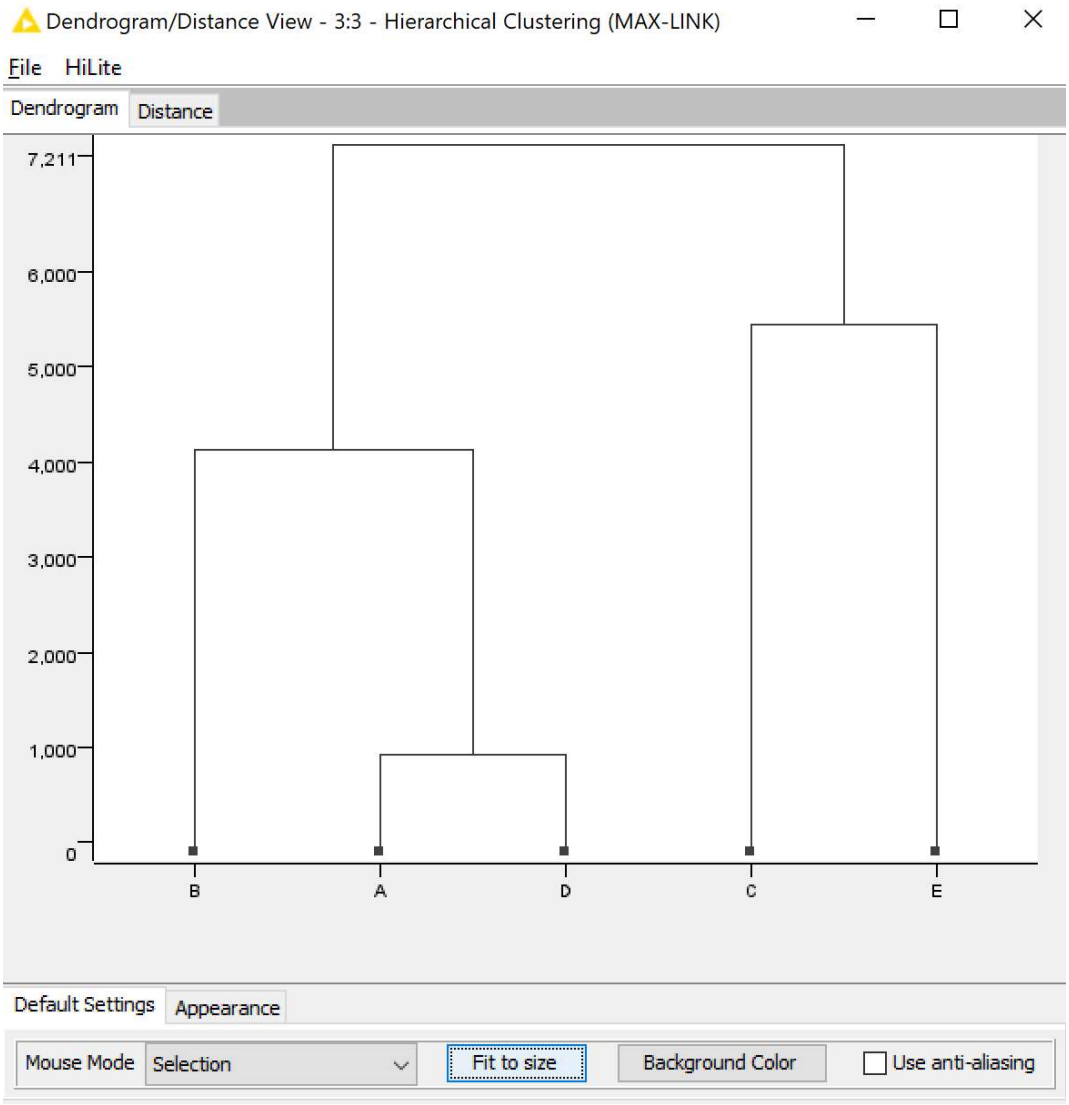


Dendograms from KNIME:

MIN-LINK:



MAX-LINK:


Dendrogram/Distance View - 3:3 - Hierarchical Clustering (MAX-LINK)
File HiLite
Dendrogram Distance
7,211
6,000
5,000
4,000
3,000
2,000
1,000
0
B
A
D
C
E
Default Settings Appearance
Mouse Mode Selection
Fit to size
Background Color
Use anti-aliasing

Task 3: DBSCAN Clustering

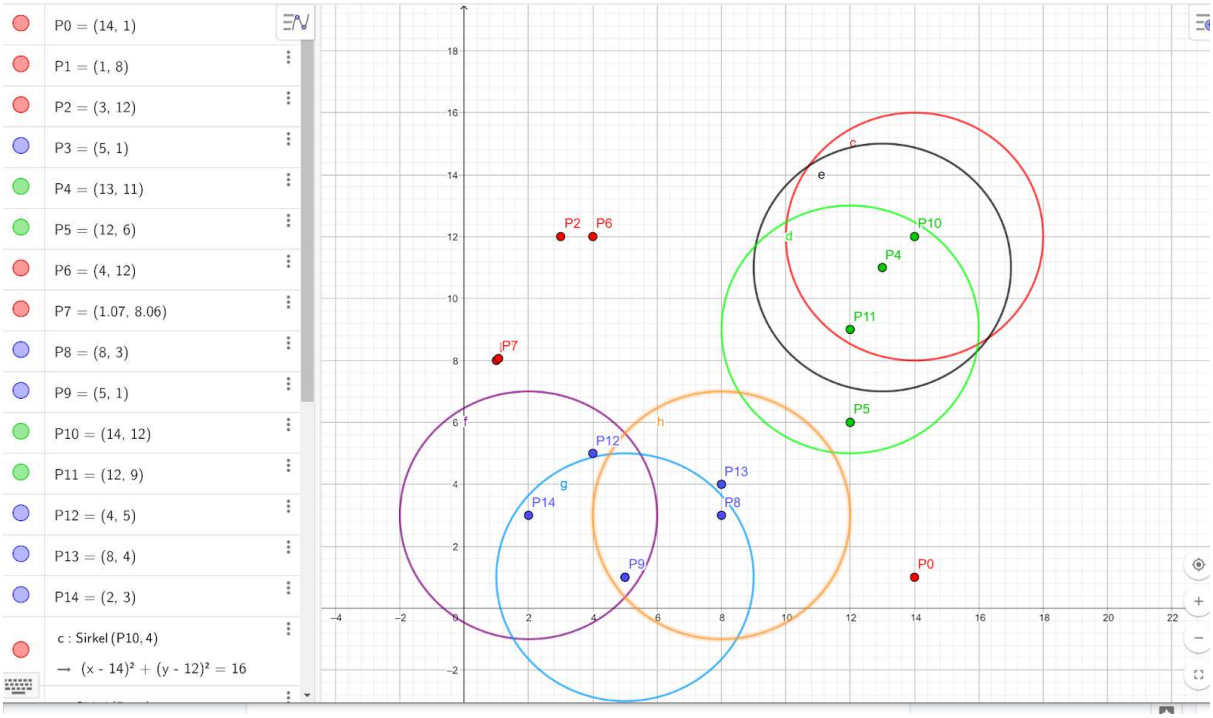
3a)

In order to solve this task, we plotted the data points in GeoGebra and generated circles around every data point with a radius of 4 (eps). From here we can figure out whether the point is a core point, border point or noise, based on minPts = 3 and Eps = 4.

- Core point has at least the specified number of points (in this case 3) within its Eps (4), when we count the point itself.
- Border point has less than the specified number of points in its range (less than 3), but is yet a part of another core points neighborhood.
- Noise point is any point that is not a core point or a border point.

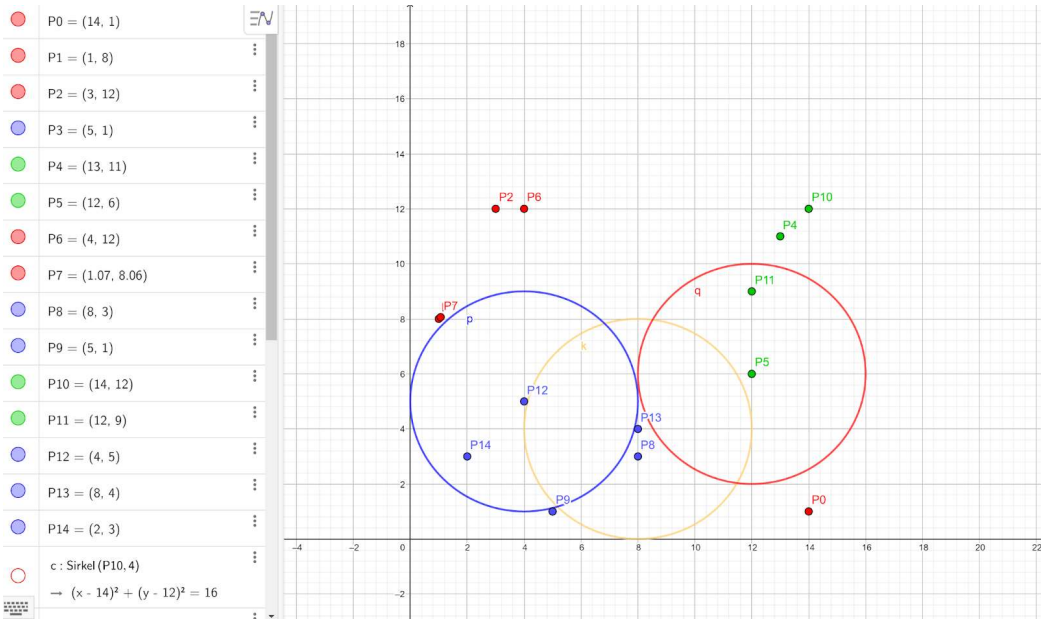
Core points: P10, P4, P11, P8, P9, P3, P14

The core points are illustrated below. They have at least 3 data points within their radius (circles). P3 and P9 share the same coordinates and share the same circle.



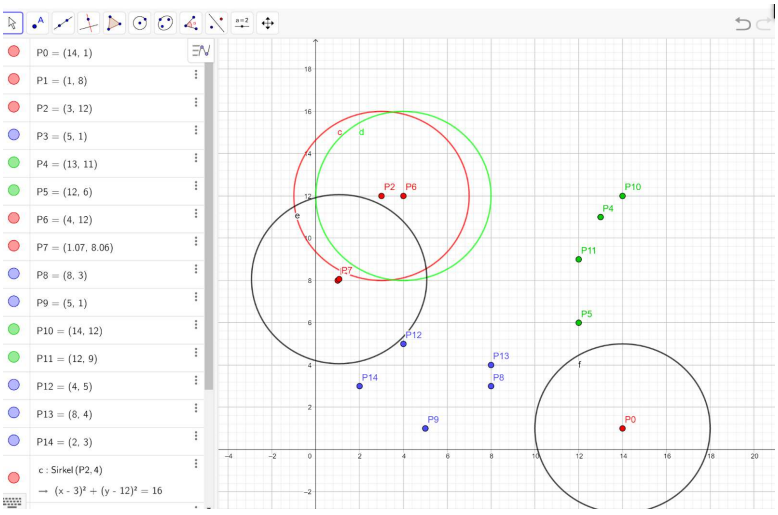
Border points: P12, P13,P5

The border points do not have at least 3 data points within their radius, but are contained within the circle of a core point nearby. The border points are illustrated below.



Noise points: P2, P6, P7, P0, P1 (highlighted in red).

The points that are not core points or border points. P0 and P7 have the same coordinates and share the same circle.

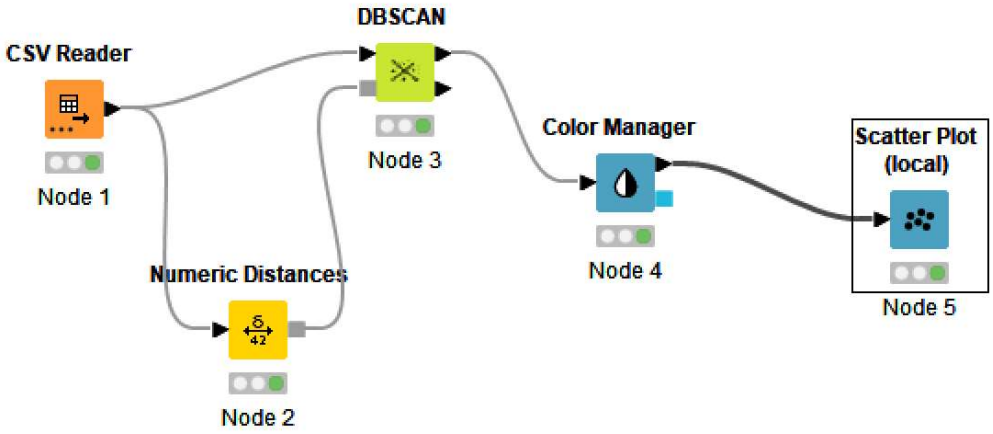


3b)

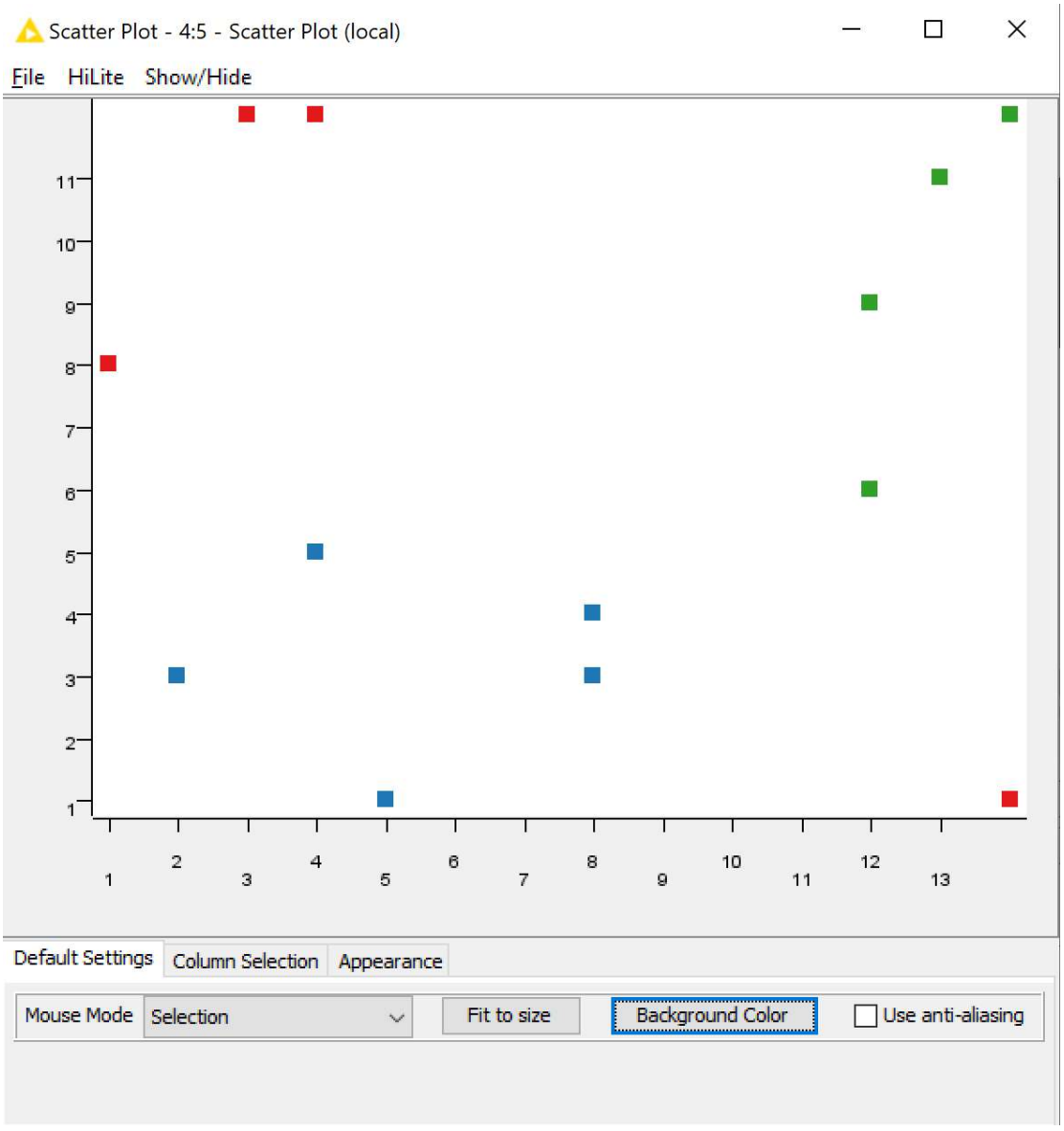
To verify our results we created the following workflow in KNIME. We used CSV Reader to import the given file with the data points. The points and euclidian distances

are given as input to the DBSCAN, which creates an output. The color manager and scatter plot were then used to create the diagram.

KNIME-Workflow:



Scatterplot from KNIME:



In the scatterplot the color coding is as follows:

- RED = Noise
- BLUE = Cluster 1
- Green = Cluster 2