LAB1

Andreas Stasinakis & Jesper Lindberg

January 28, 2019

Contents

Contributoros: Jesper Lindberg(jesli060) & Andreas Stasinakis(andst745)

In this Lab we implement two algorithms. In the first part we apply the $K-mean$ algorithm, while in the second one the density - based method for the $k-means$ algorithm. Both algorithms were implemented using the data mining toolkit called "Weka".

Question 1

As mentioned before, for the first part we implement the $K-means$ algorithm. Therefore from now on we will just changing parameters and attributes in order to comment the optimal case. We can now have a short description for the algorithm. The general idea for this algorithm is that given a dataset, we choose the number of clusters we want. In other words, the parameter K is the number of clusters that we choose and the algorithm will try to classify the data in K different clusters. It is obvious that choosing K is really important for the results. We also can choose different initial points which are the starting points for the algorithm(Also really crucial choice which is explained in depth later on). In general, given a number of clusters and points, $K-means$ algorithm calculates the distance(in this case "euclidean") between each point and the initial points and classify it using the smaller distance. After that it calculates the mean of each cluster, calculates again the distance between each data point and the mean and finally classify them using that distance. It is obvious that in order to use the algorithm efficiently, we have to try different values of K and different initial points.

1.1 Choose attributes

Choose a set of attributes for clustering and give a motivation. (Hint: always ignore attribute "name". Why does the name attribute need to be ignored?)

The data consist of 5 columns of numeric values("energy","protein","fat","calcium","iron") and one column with the names of 27 different kinds of food. Despite the fact that all of the five numeric features seems to be important enough for our clustering problem we will exclude Calcium in the tasks below. The reason why is that one data point of this feature is an outlier and it will effect the algorithm because K means is really sensitive to outliers. Moreover, the variable "names" is not included because it is not a numeric variable and the algorithm can not calculate the distances.

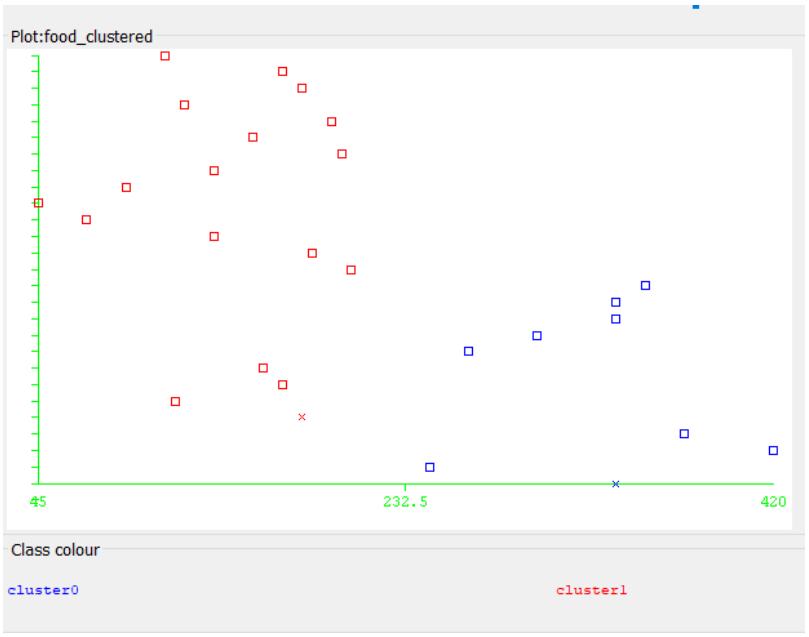
1.2 Different clusters

Experiment with at least two different numbers of clusters, e.g. 2 and 5, but with the same seed value 10.

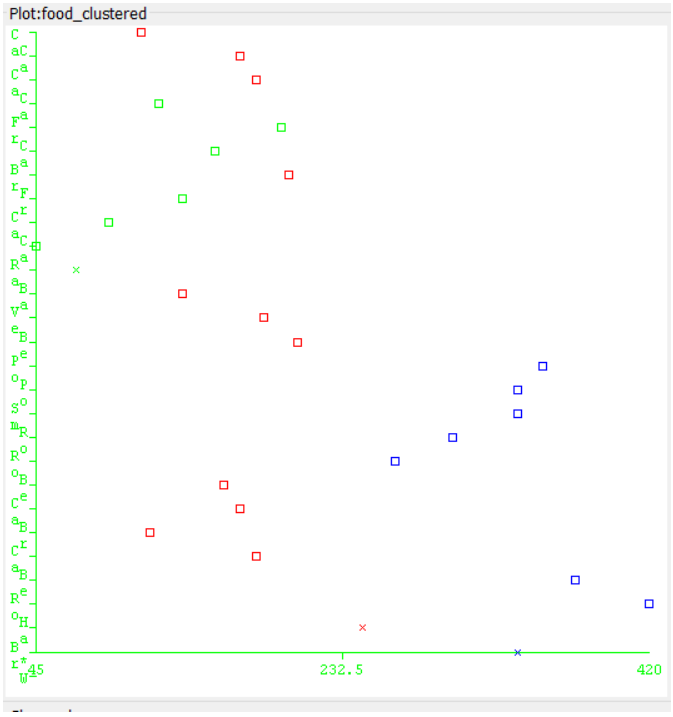
The first crucial choice we have to take is the number of clusters we want(parameter K). There are several methods in order to choose the optimal K. We can choose the *Elbow* method which seems to work properly. The problem in this case though is that we do not have enough observations so it is better to choose K using the visualization plots from Weka.

As mentioned before, the sample we have is really small. For that reason we do not check for many clusters. We check for $k = 2, 3, 4, 5, 8$. From the plots, we can conclude that the k that seems to work fine in most of the cases is 2. We plot the Name vs Energy for the different K above:

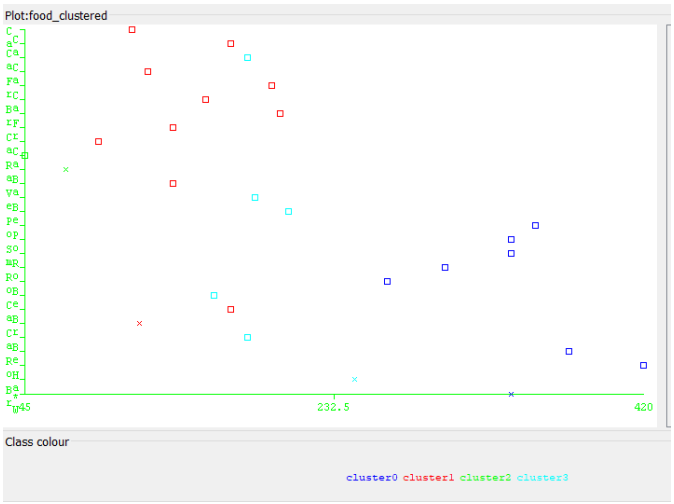
For $K = 2$


Plot:food_clustered
45
232.5
420
Class colour
cluster0
cluster1

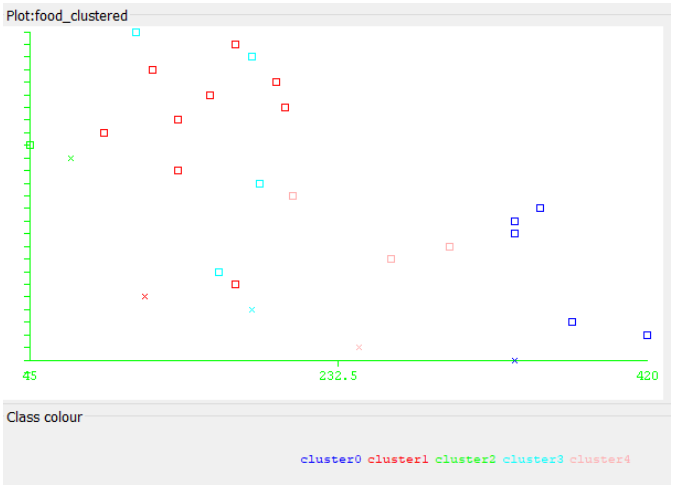
For $K = 3$


Plot:food_clustered
45
232.5
420

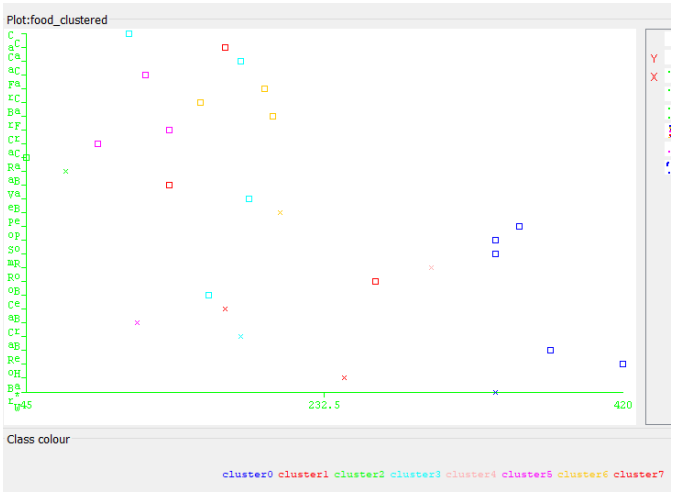
For $K = 4$



For $K = 5$



For $K = 8$



Firstly, it is obvious that for K more than 4 many of the clusters are overlapping each other. Therefore we do not need so many clusters because some observations from different clusters can be merged in the same cluster. Moreover for $k = 3$, the last cluster is overlapping with the second cluster, which mean that we do

not need both of them. Of course this is only for the specific feature, but we can make some conclusions in order to continue the tasks. Finally, from now on we will use 2 clusters, but also $K = 3$ does not seem like a bad choice.

1.3 Different seed values

Then try with a different seed value, i.e. different initial cluster centers. Compare the results with the previous results. Explain what the seed value controls.

Another factor which plays important role to the clustering problem is the starting point. Most of the times we choose random points, we run the algorithm for many different set of points and in the end choose the one which seems optimal. Of course there are some other methods in order to choose the starting points but they are not going to be discussed in this report.

In this case, the randomness is about the starting centroids. Every time we change the seed value, we take different starting points and this may give different cluster if the decision boundaries of the observations are not that clear. If we do not specify the seed though, we will take different values every time, something we do not want. From the table bellow we can make conclusions about the optimal choice of seed value. It seems that the error does not change that much in most of the cases for that reason we will set the seed equal to 10.

Table 1: Table for Errors and Seed value

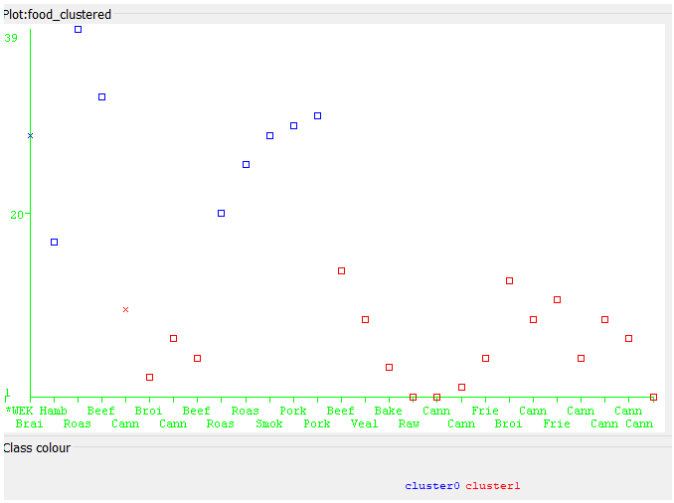
seed_value	error
10	3.98
100	3.99
123	5.35
12345	3.99
12345678	3.98
123456789	3.98
1234567890	3.99

In general, we set the seed when we have some randomness but we want to have the same output. For example, in this case we have to set the seed in order our generator give the same starting centroids, otherwise we can not make comparisons.

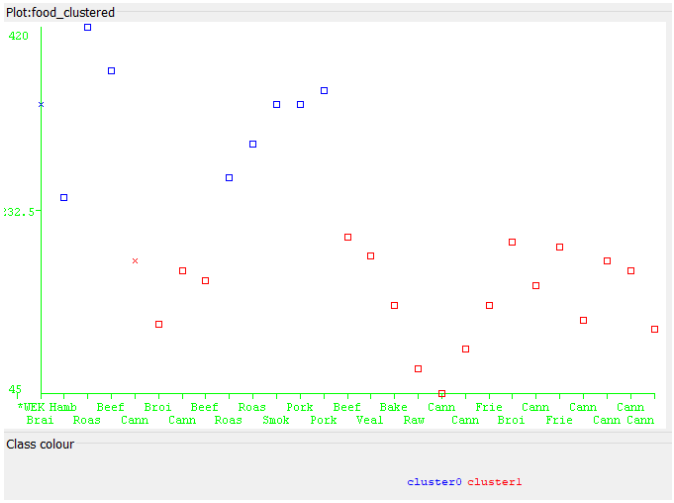
1.4 Comment on the clusters

Do you think the clusters are "good" clusters? (Are all of its members "similar" to each other? Are members from different clusters dissimilar?)

Plot for fat:



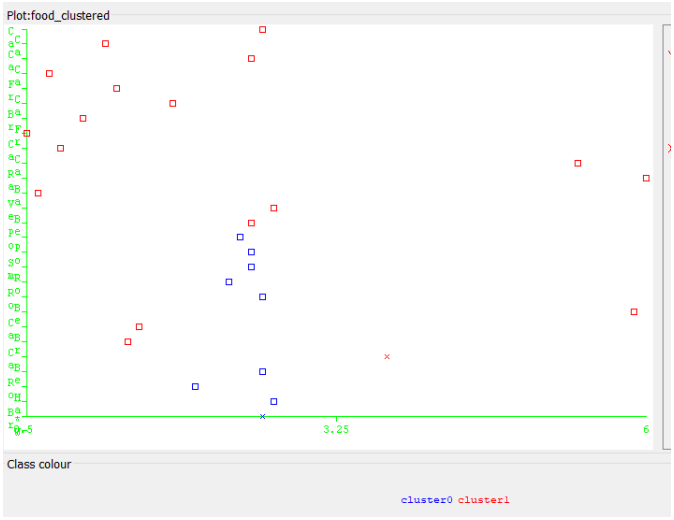
Plot for Energy:



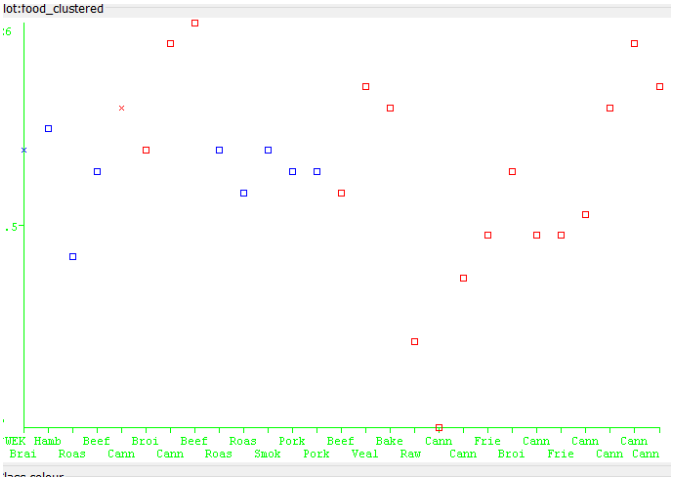
The figures is the output of the *Weka* tool for $K = 2$ and seed value 10. It can be said that the cluster do decent job. In most of the cases the decision boundaries are pretty clear.

There are some exceptions but it is logical because we have two clusters representing food and some of the features are in high percentage in both clusters. For example protein or iron(plots below) is really high in most of the observations and it seems from the plot that for this values the clusters do not have clear boundaries.

Plot for Iron:



Plot for Protein:



1.5 Conclusions

What does each cluster represent? Choose one of the results. Make up labels (words or phrases in English) which characterize each cluster.

As mentioned before, all the data represent food. After, experiment with the data for many different number of clusters and all plots from different features we choose $K = 2$, we set the seed equal to 10 and we exclude Calcium of the procedure because of one outlier which could make our results more complicated.

The first cluster cluster(cluster 0) has roasted, boiled and smoked meat but not fish at all. More specific, we can find all the types of meats and ways of cooking them. The second cluster(cluster 1) consists of both canned food and also fishes cooked in different ways. So in general, cluster 0 has only meat while cluster 1 meat and fish. This is also the reason why 18 of the total 27 observations belong to cluster 1. One can say that using 3 clusters can divide cluster 1 into one for meats and one for fish but this is not possible because they have high comprehensiveness of protein and iron so their boundaries are not clear.

Question 2 MakeDensityBasedClusters

Now with MakeDensityBasedClusters, SimpleKMeans is turned into a denstiy-based clusterer. You can set the minimum standard deviation for normal density calculation. Experiment with the algorithm as the follows:

Use the SimpleKMeans clusterer which gave the result you haven chosen in 5).

Experiment with at least two different standard deviations. Compare the results. (Hint: Increasing the standard deviation to higher values will make the differences in different runs more obvious and thus it will be easier to conclude what the parameter does)

In the MakeDensityBasedClusters we use the same model in tasks above and we also set the standard deviation. For *logical* values of standard deviation, the algorithm returns similar results with the simple K means. The SD parameters are changing the SD for each cluster distribution, note that it is only changing the minimum sd for a cluster. So if we set it to 10, then no cluster will have a distribution with a SD lower than 10. The SD in a cluster is how close the data is clustered together. Just like the normal distribution when changing the sd, higher sd is flatter curve. We can see that same curve representing how close the data points are clustered, high sd, very flat (far from each other), low sd, very tight cluster. Therefore, the larger standard deviation's value the less clusters we will have in general.

Moreover, as we can see in the plots below, SD = 10 we take exactly the same clustering as before.

Plots using density and simple means methods:

```
0 0 | *WEKA*DUMMY*STRING*FOR*STRING*ATTRIBUTES*
1 0 | Braised beef
1 0 | Hamburger
1 0 | Roast beef
1 0 | Beefsteak
0 1 | Canned beef
0 1 | Broiled chicken
0 1 | Canned chicken
0 1 | Beef heart
1 0 | Roast lamb leg
1 0 | Roast lamb shoulder
1 0 | Smoked ham
1 0 | Pork roast
1 0 | Pork simmered
0 1 | Beef tongue
0 1 | Veal cutlet
0 1 | Baked bluefish
0 1 | Raw clams
0 1 | Canned clams
0 1 | Canned crabmeat
0 1 | Fried haddock
0 1 | Broiled mackerel
0 1 | Canned mackerel
0 1 | Fried perch
0 1 | Canned salmon
0 1 | Canned sardines
0 1 | Canned tuna
0 1 | Canned shrimp
```

```
0 0 | *WEKA*DUMMY*STRING*FOR*STRING*ATTRIBUTES*
1 0 | Braised beef
1 0 | Hamburger
1 0 | Roast beef
1 0 | Beefsteak
0 1 | Canned beef
0 1 | Broiled chicken
0 1 | Canned chicken
0 1 | Beef heart
1 0 | Roast lamb leg
1 0 | Roast lamb shoulder
1 0 | Smoked ham
1 0 | Pork roast
1 0 | Pork simmered
0 1 | Beef tongue
0 1 | Veal cutlet
0 1 | Baked bluefish
0 1 | Raw clams
0 1 | Canned clams
0 1 | Canned crabmeat
0 1 | Fried haddock
0 1 | Broiled mackerel
0 1 | Canned mackerel
0 1 | Fried perch
0 1 | Canned salmon
0 1 | Canned sardines
0 1 | Canned tuna
0 1 | Canned shrimp

Cluster 0 <-- Braised beef
Cluster 1 <-- Canned beef

Incorrectly clustered instances :       25.0     92.5926 %
```

We also run the algorithm for different values(plots below). In general, there are not many differences between the K mean results and this one. For SD = 100, we have only 2 changes. More specific, Roast lamb leg and hamburger are changing cluster. This change does not make that sense because those observations are cluster as canned meat. While SD is increasing we can observe that all the observations from the cluster 0(meat cluster) are moving to the second cluster. Finally, after SD = 236, we have only one cluster which consists of all the data. Finally, for extreme values of standard deviation, for example 10000, nothing is changing and only one cluster exists. In conclusion, we can say that the K means does a really good job and we can choose SD = 10 in order to take the same results.

For $SD = 235$

```
0 1  <-- assigned to cluster
0 0 | *WEKA*DUMMY*STRING*FOR*STRING*ATTRIBUTES*
0 1 | Braised beef
0 1 | Hamburger
1 0 | Roast beef
0 1 | Beefsteak
0 1 | Canned beef
0 1 | Broiled chicken
0 1 | Canned chicken
0 1 | Beef heart
0 1 | Roast lamb leg
0 1 | Roast lamb shoulder
0 1 | Smoked ham
0 1 | Pork roast
0 1 | Pork simmered
0 1 | Beef tongue
0 1 | Veal cutlet
0 1 | Baked bluefish
0 1 | Raw clams
0 1 | Canned clams
0 1 | Canned crabmeat
0 1 | Fried haddock
0 1 | Broiled mackerel
0 1 | Canned mackerel
0 1 | Fried perch
0 1 | Canned salmon
0 1 | Canned sardines
0 1 | Canned tuna
0 1 | Canned shrimp
```

For $SD = 236$

```
1  <-- assigned to cluster
0 | *WEKA*DUMMY*STRING*FOR*STRING*ATTRIBUTES*
1 | Braised beef
1 | Hamburger
1 | Roast beef
1 | Beefsteak
1 | Canned beef
1 | Broiled chicken
1 | Canned chicken
1 | Beef heart
1 | Roast lamb leg
1 | Roast lamb shoulder
1 | Smoked ham
1 | Pork roast
1 | Pork simmered
1 | Beef tongue
1 | Veal cutlet
1 | Baked bluefish
1 | Raw clams
1 | Canned clams
1 | Canned crabmeat
1 | Fried haddock
1 | Broiled mackerel
1 | Canned mackerel
1 | Fried perch
1 | Canned salmon
1 | Canned sardines
1 | Canned tuna
1 | Canned shrimp
```