Data Mining Lab 01

Anubhav Dikshit (anudi287) and Nahid Farazmand (nahfa911)

February 7, 2019

Algorithm 1: Simple K-means

Apply "SimpleKMeans" to your data. In Weka euclidean distance is implemented in SimpleKmeans. You can set the number of clusters and seed of a random algorithm for generating initial cluster centers. Experiment with the algorithm as follows:

1. 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?)

K-means algorithm cannot handle categorical data (attributes) so we have to ignore all categorical attributes when we want to use this algorithm for clustering.

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

Number of clusters = 2 and seed = 10

a. Buffer results

```
kMeans
======

Number of iterations: 2
Within cluster sum of squared errors: 5.069321339929419

Initial starting points (random):

Cluster 0: 340,20,28,9,2.6
Cluster 1: 170,25,7,12,1.5

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute    Full Data          0          1
                (27.0)      (9.0)     (18.0)
=============================================
Energy        207.4074   331.1111   145.5556
Protein             19         19         19
Fat            13.4815    27.5556     6.4444
Calcium         43.963     8.7778    61.5556
Iron            2.3815     2.4667     2.3389



Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       9 ( 33%)
1      18 ( 67%)
```

We can see that sum of squared error within the clusters is around 5.07 and the division rate in clusters are 67% and 33% in cluster 1and 0, respectively.

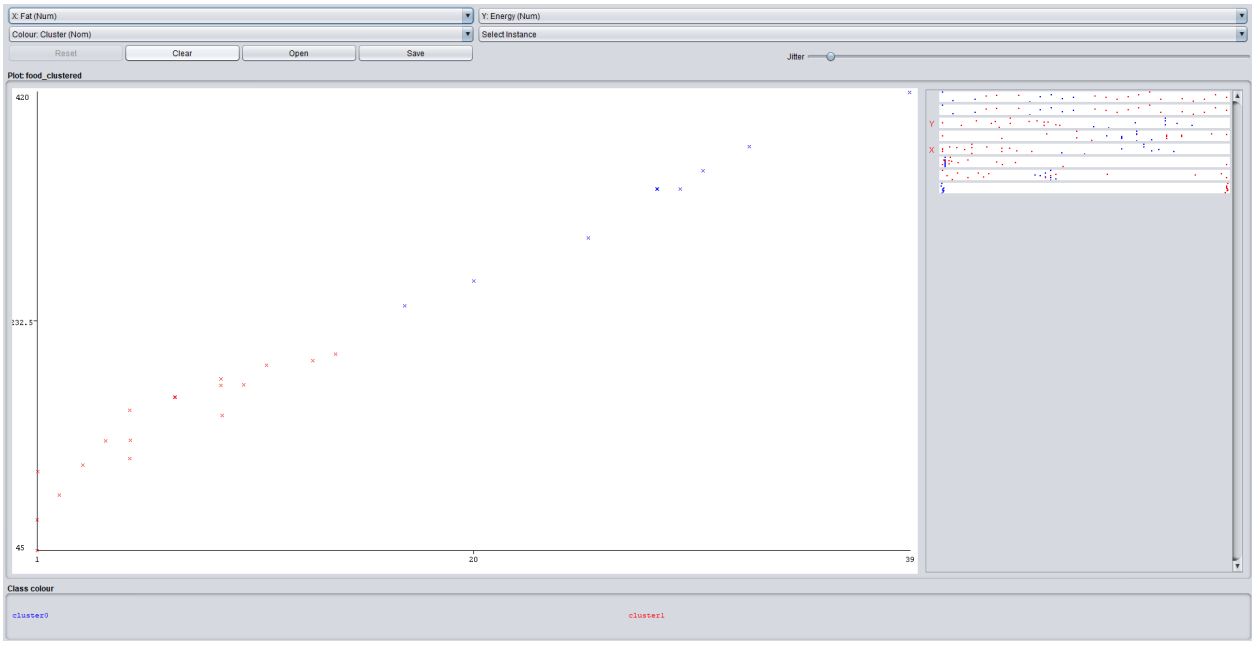
b. Clusters

	Instance_number	Energy	Protein	Fat	Calcium	Iron	Cluster
1	0	340	20	28	9	2.6	cluster0
2	1	245	21	17	9	2.7	cluster0
3	2	420	15	39	7	2.0	cluster0
4	3	375	19	32	9	2.6	cluster0
9	8	265	20	20	9	2.6	cluster0
10	9	300	18	25	9	2.3	cluster0
11	10	340	20	28	9	2.5	cluster0
12	11	340	19	29	9	2.5	cluster0
13	12	355	19	30	9	2.4	cluster0
5	4	180	22	10	17	3.7	cluster1
6	5	115	20	3	8	1.4	cluster1
7	6	170	25	7	12	1.5	cluster1
8	7	160	26	5	14	5.9	cluster1
14	13	205	18	14	7	2.5	cluster1
15	14	185	23	9	9	2.7	cluster1
16	15	135	22	4	25	0.6	cluster1
17	16	70	11	1	82	6.0	cluster1
18	17	45	7	1	74	5.4	cluster1
19	18	90	14	2	38	0.8	cluster1
20	19	135	16	5	15	0.5	cluster1
21	20	200	19	13	5	1.0	cluster1
22	21	155	16	9	157	1.8	cluster1
23	22	195	16	11	14	1.3	cluster1
24	23	120	17	5	159	0.7	cluster1
25	24	180	22	9	367	2.5	cluster1
26	25	170	25	7	7	1.2	cluster1
27	26	110	23	1	98	2.6	cluster1

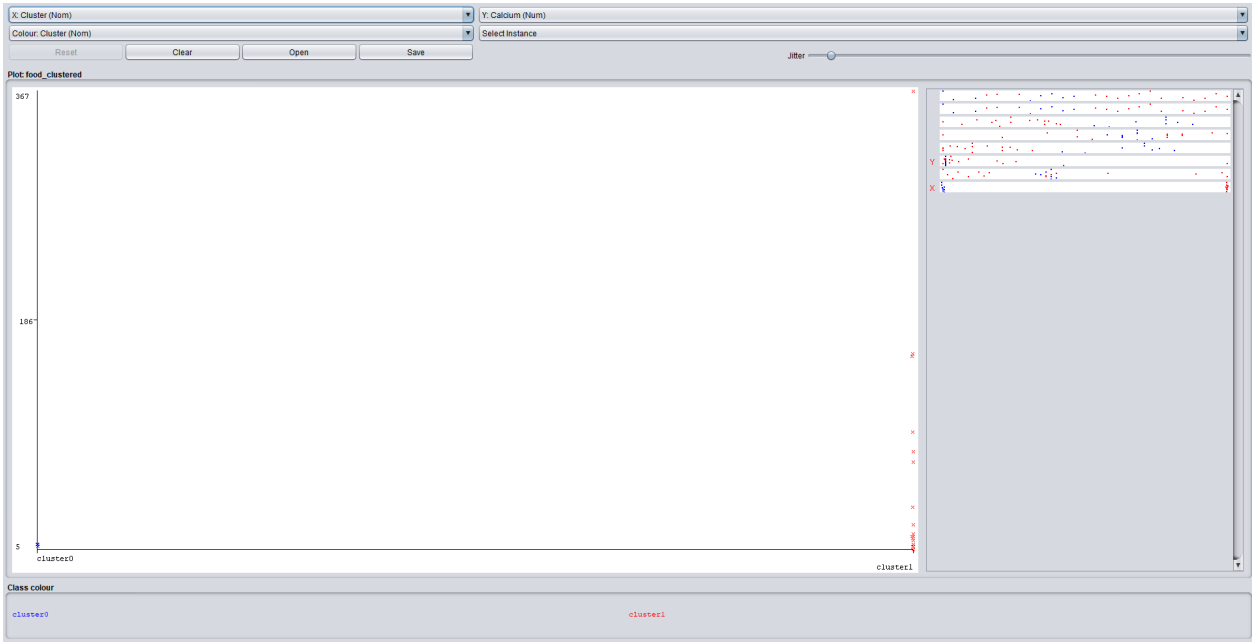
By choosing Cluster = 2 and seed = 10, the initial points are observations number 1 and 7.

c. Visualization

In this step we can check the relation between different attributes and the effectiveness of different attributes on finding clusters. The most observations of this step were as bellow:



It can be clearly seen that there is a strong correlation between energy and fat attributes. So we can choose just one of this attributes and this can help us to reduce the space dimension.



We can see that Calcium attribute does not play and effective role in clustering and all observations except 2 observations belong to 1 cluster!

Number of clusters = 5 and seed = 10

Now we will try another number of clusters but without changing the seed. The result will be as follow:

a. Buffer results

```
kMeans
======

Number of iterations: 4
Within cluster sum of squared errors: 2.750432407251998

Initial starting points (random):

Cluster 0: 340,20,28,9,2.6
Cluster 1: 170,25,7,12,1.5
Cluster 2: 90,14,2,38,0.8
Cluster 3: 180,22,9,367,2.5
Cluster 4: 300,18,25,9,2.3

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute    Full Data          0          1          2          3          4
                (27.0)      (7.0)      (8.0)      (6.0)      (1.0)      (5.0)
================================================================================
Energy        207.4074   352.8571   153.125      102.5        180        222
Protein             19    18.5714      23.25       13.5         22       18.8
Fat            13.4815    30.1429       5.75     3.8333          9         15
Calcium         43.963     8.7143      23.75       87.5        367        8.8
Iron            2.3815     2.4143       2.45     2.5333        2.5       2.02



Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       7 ( 26%)
1       8 ( 30%)
2       6 ( 22%)
3       1 (  4%)
4       5 ( 19%)
```

Here by increasing the number of clusters, the rate of sum of squared error decreased to about half of that of the previous choice.

b. Clusters

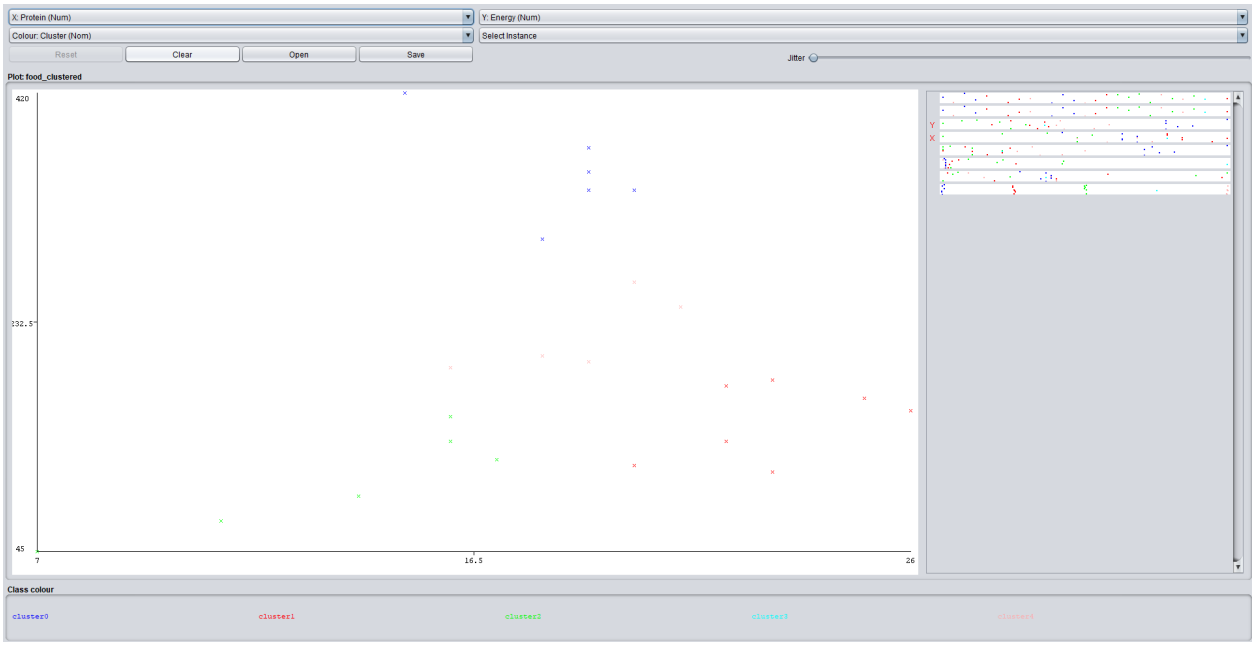
```
##    Instance_number Energy Protein Fat Calcium Iron  Cluster
## 1                0    340      20  28       9  2.6 cluster0
## 3                2    420      15  39       7  2.0 cluster0
## 4                3    375      19  32       9  2.6 cluster0
## 10               9    300      18  25       9  2.3 cluster0
## 11              10    340      20  28       9  2.5 cluster0
## 12              11    340      19  29       9  2.5 cluster0
## 13              12    355      19  30       9  2.4 cluster0
## 5                4    180      22  10      17  3.7 cluster1
## 6                5    115      20   3       8  1.4 cluster1
## 7                6    170      25   7      12  1.5 cluster1
## 8                7    160      26   5      14  5.9 cluster1
## 15              14    185      23   9       9  2.7 cluster1
## 16              15    135      22   4      25  0.6 cluster1
## 26              25    170      25   7       7  1.2 cluster1
## 27              26    110      23   1      98  2.6 cluster1
## 17              16     70      11   1      82  6.0 cluster2
## 18              17     45       7   1      74  5.4 cluster2
```

```
## 19          18     90      14   2      38  0.8 cluster2
## 20          19    135      16   5      15  0.5 cluster2
## 22          21    155      16   9     157  1.8 cluster2
## 24          23    120      17   5     159  0.7 cluster2
## 25          24    180      22   9     367  2.5 cluster3
## 2            1    245      21  17       9  2.7 cluster4
## 9            8    265      20  20       9  2.6 cluster4
## 14          13    205      18  14       7  2.5 cluster4
## 21          20    200      19  13       5  1.0 cluster4
## 23          22    195      16  11      14  1.3 cluster4
```

Here we can see that the observation 1, 7, 10, 19 and 25 have been chosen as the initial centers for clusters. It should be noticed that two point are the same as points which were the initial points for k = 2. Another result that should be considered is that point number 25 has been defined as an outlier. However, this observation has been introduced as an cluster, because k-mean algorithm is as partitioning method and in these methods all points should belong to an cluster.

c. Visualization

As we said before, energy and fat are correlated. Besides, calcium and Iron also are not as effective as energy and protein, so here we have chosen energy and protein to show the clusters:



We can see that the shape of clusters are convex and the border between clusters are obvious, however some points are far from the centroids but still have been assigned to the clusters because as we said before, all points should belong to a cluster.

3. 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.

In this step we chose k = 5 and run the clustering algorithm for different seeds.

a. seed = 1000

Buffer results

```
kMeans
======

Number of iterations: 5
Within cluster sum of squared errors: 1.8730334748591244

Initial starting points (random):

Cluster 0: 70,11,1,82,6
Cluster 1: 120,17,5,159,0.7
Cluster 2: 245,21,17,9,2.7
Cluster 3: 180,22,10,17,3.7
Cluster 4: 155,16,9,157,1.8

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute    Full Data          0          1          2          3          4
                (27.0)     (2.0)      (7.0)      (8.0)      (8.0)      (2.0)
================================================================================
Energy       207.4074       57.5   141.4286    341.875    178.125      167.5
Protein            19          9    17.7143      18.75     22.875         19
Fat           13.4815          1     6.1429     28.875      8.75          9
Calcium        43.963         78    37.7143       8.75     21.625        262
Iron           2.3815        5.7        0.9     2.4375       2.85       2.15



Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       2 (  7%)
1       7 ( 26%)
2       8 ( 30%)
3       8 ( 30%)
4       2 (  7%)
```

clusters

	Instance_number	Energy	Protein	Fat	Calcium	Iron	Cluster
17	16	70	11	1	82	6.0	cluster0
18	17	45	7	1	74	5.4	cluster0
6	5	115	20	3	8	1.4	cluster1
16	15	135	22	4	25	0.6	cluster1
19	18	90	14	2	38	0.8	cluster1
20	19	135	16	5	15	0.5	cluster1
21	20	200	19	13	5	1.0	cluster1
23	22	195	16	11	14	1.3	cluster1
24	23	120	17	5	159	0.7	cluster1
1	0	340	20	28	9	2.6	cluster2
3	2	420	15	39	7	2.0	cluster2
4	3	375	19	32	9	2.6	cluster2
9	8	265	20	20	9	2.6	cluster2
10	9	300	18	25	9	2.3	cluster2
11	10	340	20	28	9	2.5	cluster2
12	11	340	19	29	9	2.5	cluster2
13	12	355	19	30	9	2.4	cluster2
2	1	245	21	17	9	2.7	cluster3

```
## 5                 4     180       22  10       17  3.7 cluster3
## 7                 6     170       25   7       12  1.5 cluster3
## 8                 7     160       26   5       14  5.9 cluster3
## 14               13     205       18  14        7  2.5 cluster3
## 15               14     185       23   9        9  2.7 cluster3
## 26               25     170       25   7        7  1.2 cluster3
## 27               26     110       23   1       98  2.6 cluster3
## 22               21     155       16   9      157  1.8 cluster4
## 25               24     180       22   9      367  2.5 cluster4
```

b. seed = 10000

Buffer results

```
kMeans
======

Number of iterations: 5
Within cluster sum of squared errors: 1.776241584499267

Initial starting points (random):

Cluster 0: 180,22,9,367,2.5
Cluster 1: 180,22,10,17,3.7
Cluster 2: 70,11,1,82,6
Cluster 3: 135,16,5,15,0.5
Cluster 4: 115,20,3,8,1.4

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute    Full Data          0          1          2          3          4
               (27.0)     (1.0)      (8.0)      (2.0)      (8.0)      (8.0)
================================================================================
Energy        207.4074        180   341.875       57.5    143.125    178.125
Protein             19         22    18.75          9       17.5     22.875
Fat            13.4815          9   28.875          1        6.5      8.75
Calcium         43.963        367     8.75         78     52.625     21.625
Iron            2.3815        2.5   2.4375        5.7     1.0125       2.85



Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       1 (  4%)
1       8 ( 30%)
2       2 (  7%)
3       8 ( 30%)
4       8 ( 30%)
```

clusters

```
##    Instance_number Energy Protein Fat Calcium Iron  Cluster
## 25              24    180      22   9     367  2.5 cluster0
## 1                0    340      20  28       9  2.6 cluster1
## 3                2    420      15  39       7  2.0 cluster1
## 4                3    375      19  32       9  2.6 cluster1
## 9                8    265      20  20       9  2.6 cluster1
```

```
## 10               9    300      18  25       9  2.3 cluster1
## 11              10    340      20  28       9  2.5 cluster1
## 12              11    340      19  29       9  2.5 cluster1
## 13              12    355      19  30       9  2.4 cluster1
## 17              16     70      11   1      82  6.0 cluster2
## 18              17     45       7   1      74  5.4 cluster2
## 6                5    115      20   3       8  1.4 cluster3
## 16              15    135      22   4      25  0.6 cluster3
## 19              18     90      14   2      38  0.8 cluster3
## 20              19    135      16   5      15  0.5 cluster3
## 21              20    200      19  13       5  1.0 cluster3
## 22              21    155      16   9     157  1.8 cluster3
## 23              22    195      16  11      14  1.3 cluster3
## 24              23    120      17   5     159  0.7 cluster3
## 2                1    245      21  17       9  2.7 cluster4
## 5                4    180      22  10      17  3.7 cluster4
## 7                6    170      25   7      12  1.5 cluster4
## 8                7    160      26   5      14  5.9 cluster4
## 14              13    205      18  14       7  2.5 cluster4
## 15              14    185      23   9       9  2.7 cluster4
## 26              25    170      25   7       7  1.2 cluster4
## 27              26    110      23   1      98  2.6 cluster4
```

Seed parameter controls the initial selected centroids so we can see that when we change seed, the initial selected points will change. Here we considered 2 different choices:

1. K = 5 and seed = 1000
2. k = 5 and seed = 10000

By choosing these amounts we can see that compared to the previous step with k = 5 and seed = 10, clusters are different. However, if we compare the results of these two cases we can see that most of the clusters are similar (similar clusters for 2 different seeds):

```
##   seed_1000               seed_10000
## 1         2                        1
## 2         0                        2
## 3         1    3 plus observation 22
## 4         3                        4
## 5         4 0 without observation 22
```

Comparing sum of squared error rates, seed = 10000 has the least amount of error.

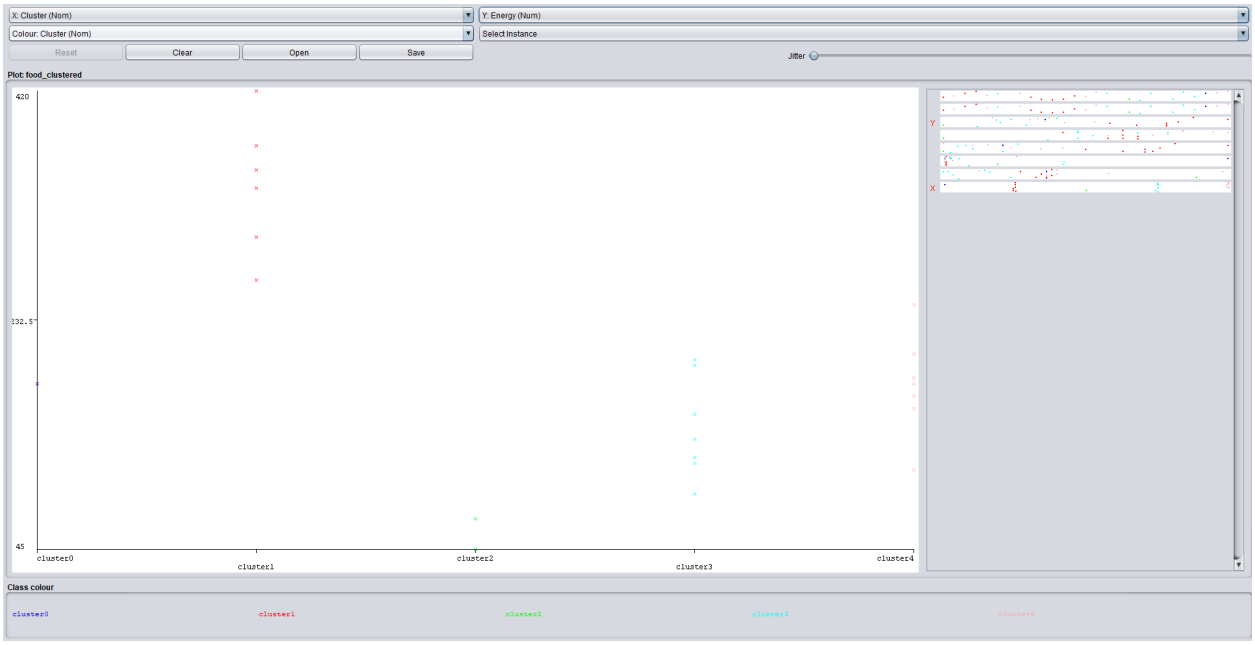
So the initial centroids can result in different clusters and if we want to do the analysis, we should try different seeds to be able to get the result which is the nearest to the optimum.

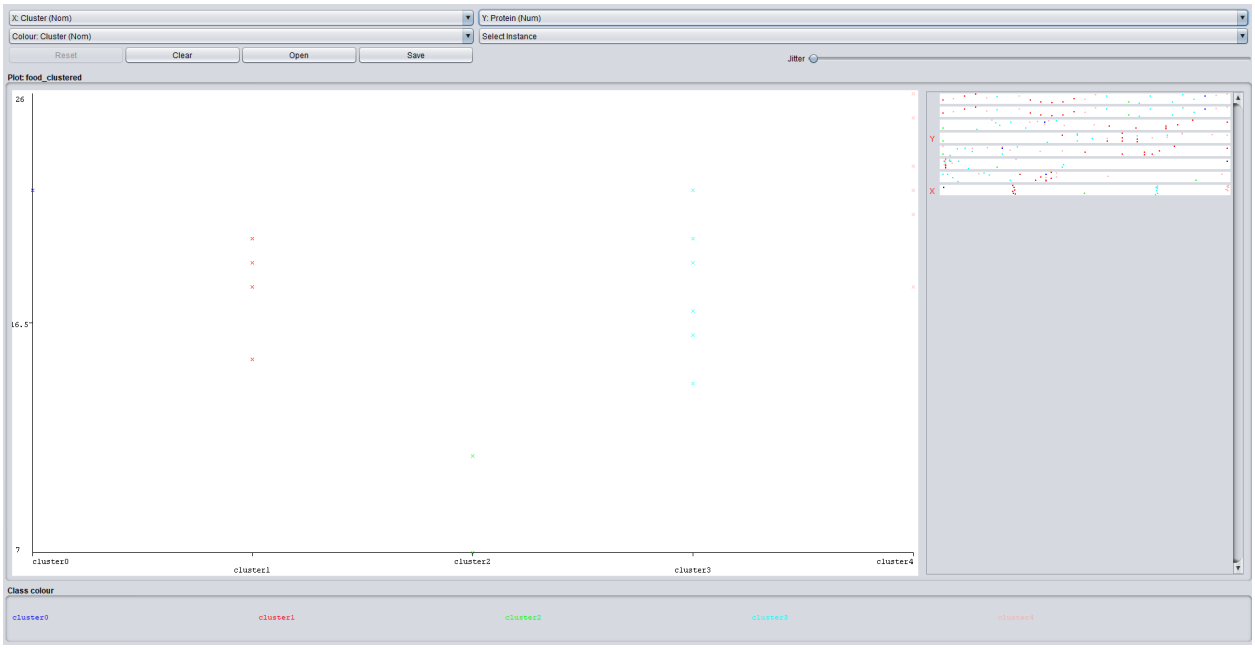
4. Do you think the clusters are "good" clusters? (Are all of its members "similar" to each other? Are members from different clusters dissimilar?)

We would say that considering energy, fat and protein, clusters are good. We can differentiate clusters based on these attributes and we can say that based on selected attributes, observations within the clusters are similar.

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

Considering the sum of squared error rates and similarity of observations within the clusters, we will choose the model with k = 5 and seed = 10000. By choosing this model we can label clusters as bellow:





Cluster 0: Outlines

Cluster 1: High Energy - Moderate Protein

Cluster 2: Outlines

Cluster 3: Low Energy - Moderate Protein

Cluster 4: Low Energy - High Protein

Algorithm 2: Density-Based Cluster

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

1. Use the Simple K-Means clusterer which gave the result you haven chosen in 5.

The parameters of K-means algorithm: k = 5 and seed = 10000

2. 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 this algorithm, in the first step, clusters will be formed by using k-means algorithm with defined parameters. Then each cluster will be adjusted based on the Gaussian distribution. Here we have chosen different standard deviation for the algorithm and the results have been as bellow:

sd = 1E-6

```
=== Model and evaluation on training set ===

Clustered Instances

0       1 (  4%)
1       9 ( 33%)
2       2 (  7%)
3       8 ( 30%)
4       7 ( 26%)


Log likelihood: -15.83785
```

sd = 1

```
=== Model and evaluation on training set ===

Clustered Instances

0       1 (  4%)
1       8 ( 30%)
2       2 (  7%)
3       7 ( 26%)
4       9 ( 33%)

Log likelihood: -17.85566
```

sd = 10

```
=== Model and evaluation on training set ===

Clustered Instances

0       1 (  4%)
1       8 ( 30%)
2       2 (  7%)
3       6 ( 22%)
4      10 ( 37%)

Log likelihood: -21.05685
```

sd = 100

```
=== Model and evaluation on training set ===

Clustered Instances

0       1 (  4%)
1       8 ( 30%)
3       9 ( 33%)
4       9 ( 33%)


Log likelihood: -28.43759
```

sd = 1000

```
=== Model and evaluation on training set ===

Clustered Instances

1       8 ( 30%)
3      10 ( 37%)
4       9 ( 33%)


Log likelihood: -39.14921
```

Based on the results we can see that if we increase the standard deviation it may decrease the number of clusters. And we can see that log-likelihood also decreases along with standard deviation growth. So we should choose standard deviation as small as we can to be able to get the best result.