Advance Data Mining-Clustering Lab

Yusur Al-Mter (yusal621), Roshni Sundaramurthy

28 January 2019

SimpleKmeans:

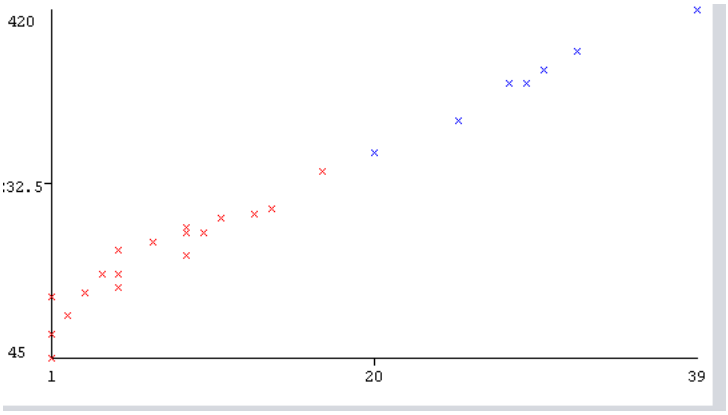
Apply "SimpleKMeans" to your data. In Weka euclidean distance is implemented in SimpleKmeans. You can set the number of clusters and seed of a random algorithm for generating initial cluster centers. Experiment with the algorithm as follows:

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 tends to not work well on mixed type data. It is meant for data where every attribute has the same relevance and scale.

The *name* attribute holds no quantitative weightage towards the analysis of the data, since it's a string that is neither numeric nor nominal and the process for simple Kmeans to cluster is calculating distances between the arbitrarily choosen objects assigned as centroids for each cluster and the closest objects to attach to those clusters. Hence using the attribute *name* is useless and meaningless and it should be ignored.

Observing the relationship between the attributes *Energy* and *Fat*, as it's shown below, where the x-axis is Fat and the y-axis is Energy, one can observe the direct effect of the increase in Fat on the increasement in Energy, for that reason both have been choosen to represent our future analysis.



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

Used attributes: Energy and Fat.

Seed value: 10.

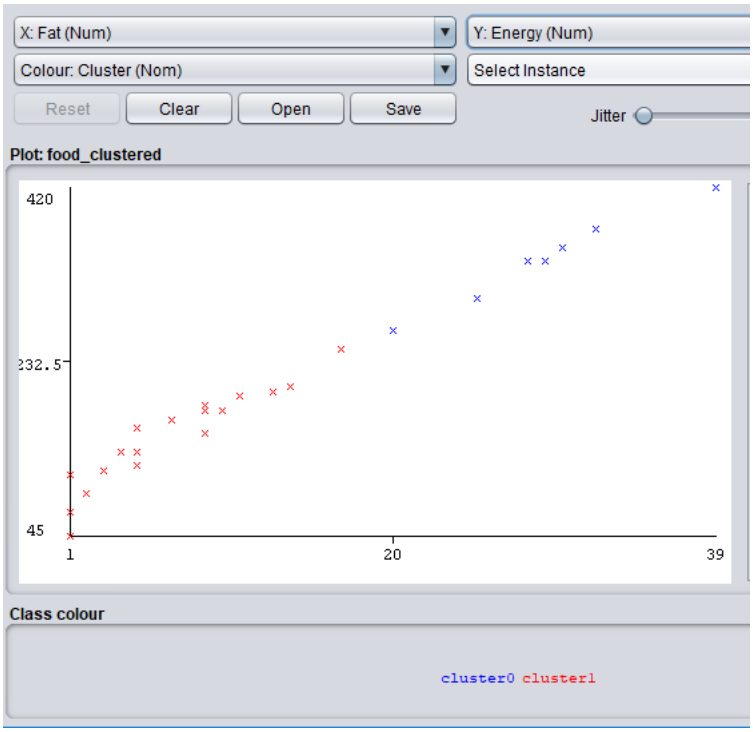
Letting k to represent the number of clusters performed on the training set, we have:

When k=2,

Clusterer output

```
Ignored:
              Name
              Protein
              Calcium
              Iron
Test mode:    evaluate on training data


=== Clustering model (full training set) ===


kMeans
======

Number of iterations: 2
Within cluster sum of squared errors: 0.8481897660818714

Initial starting points (random):

Cluster 0: 340,28
Cluster 1: 170,7

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute    Full Data          0          1
               (27.0)      (8.0)     (19.0)
=============================================
Energy        207.4074    341.875   150.7895
Fat            13.4815     28.875          7




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

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

Clustered Instances

0       8 ( 30%)
1      19 ( 70%)
```

Visualizing the two clusters,

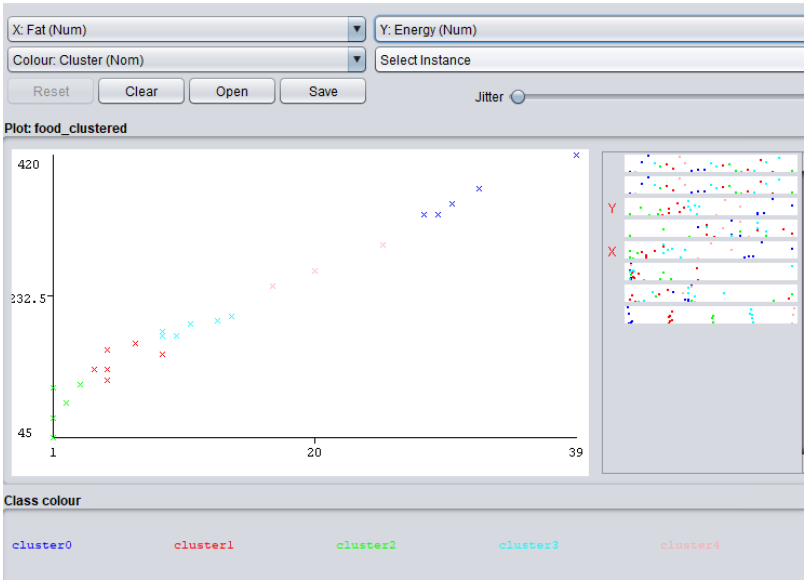

X: Fat (Num)
Y: Energy (Num)
Colour: Cluster (Nom)
Select Instance
Reset
Clear
Open
Save
Jitter
Plot: food_clustered
420
232.5
45
1
20
39
Class colour
cluster0 cluster1

When k=5,

Clusterer output

```
=== Clustering model (full training set) ===


kMeans
======

Number of iterations: 3
Within cluster sum of squared errors: 0.20447197056969912

Initial starting points (random):

Cluster 0: 340,28
Cluster 1: 170,7
Cluster 2: 90,2
Cluster 3: 180,9
Cluster 4: 300,25

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute    Full Data          0          1          2          3          4
                (27.0)      (6.0)      (7.0)      (5.0)      (6.0)      (3.0)
================================================================================
Energy        207.4074   361.6667   149.2857         86   190.8333        270
Fat            13.4815         31          6        1.6         11    20.6667



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

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

Clustered Instances

0       6 ( 22%)
1       7 ( 26%)
2       5 ( 19%)
3       6 ( 22%)
4       3 ( 11%)
```

visualizing the five clusters,



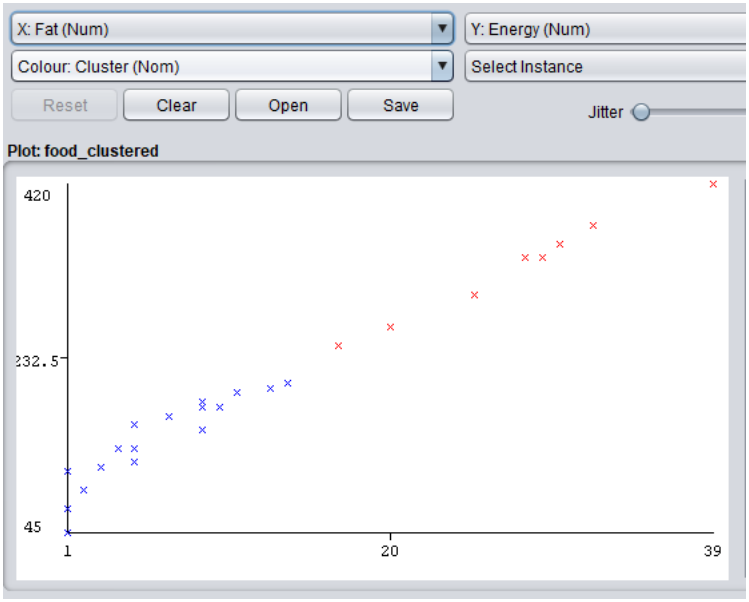
Choosing the right number for k clusters, is often ambiguous, and it depends on the user desire clustering

results and interpretations of the data set. In addition, increasing k without penalty will always reduce the amount of error in the resulting clustering, to the extreme case of zero error if each data point is considered its own cluster. And also, it can increase the risk of overfitting. Hence, choosing k clusters must be examined to reach the desired goals.

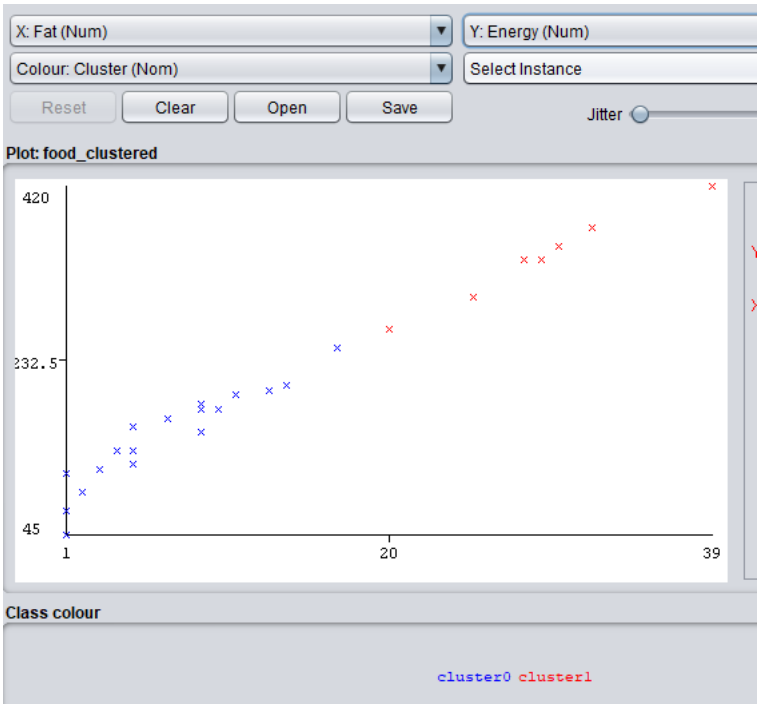
3. Then try with a different seed value, i.e. different initial cluster centers. Compare the results with the previous results. Explain what the seed value controls.

Used attributes: Energy and Fat.

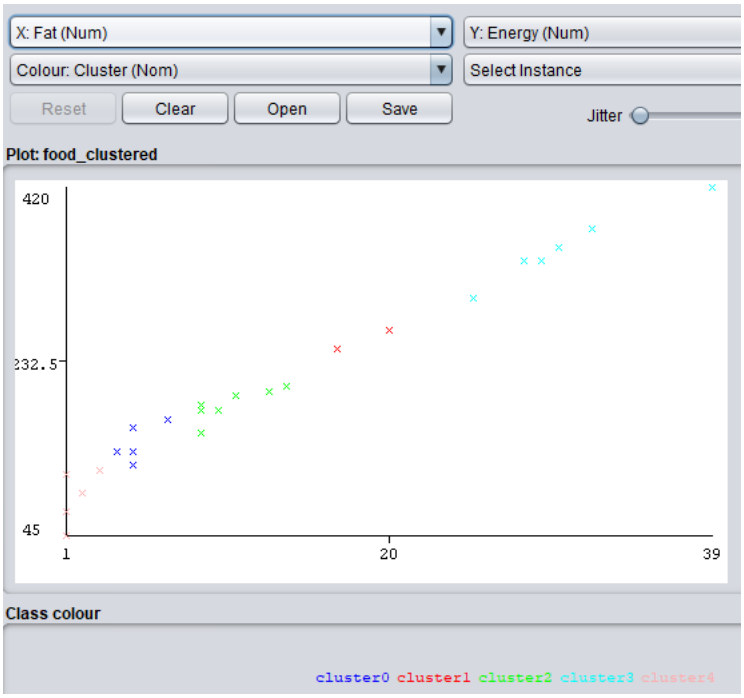
Visualizing the two clusters with seed =100,



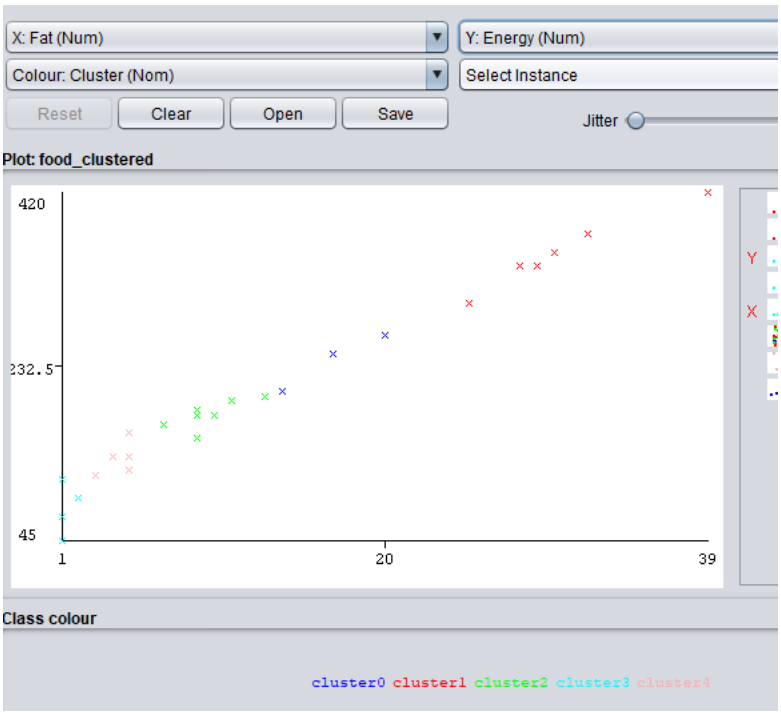
Visualizing the two clusters with seed =400,



Visualizing the five clusters with seed =100,



Visualizing the five clusters with seed =400,



The K-means algorithm for clustering is very much dependent on the initial seed values, the seed value changes the starting value (centroids) of the analysis which could result in different elements falling into different clusters due to different random starting points in the data set. And because K-means is sensitive to initial points, one should try experimentation on the stability of the clusters with different seeds, with better seeds, k-means converges faster and the quality of the clusters will be good.

Observing the behaviour of the above graphs for both 2 and 5 clusters with seed equal to 100 and 400, the kmeans procedure seems more unstable for k =5, while for k=2 the members in clusters are more similar.

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

Cluster analysis aims at sorting different objects into groups in a way that the degree of association between two objects is maximal if they belong to the same group and minimal otherwise. Energy and fat being on same range of values, the clusters seem to be good and the observations within the clusters are similar when compared to other attributes.

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

Two clusters with seed equal to 10 seems more reliable since it has more apparent visual separation, while increasing the number of clusters didn't help improving our results. The obtained clusters can be labeled as High Energy/Fat cluster and Low Energy/Fat cluster.

Make Density Based Clusters

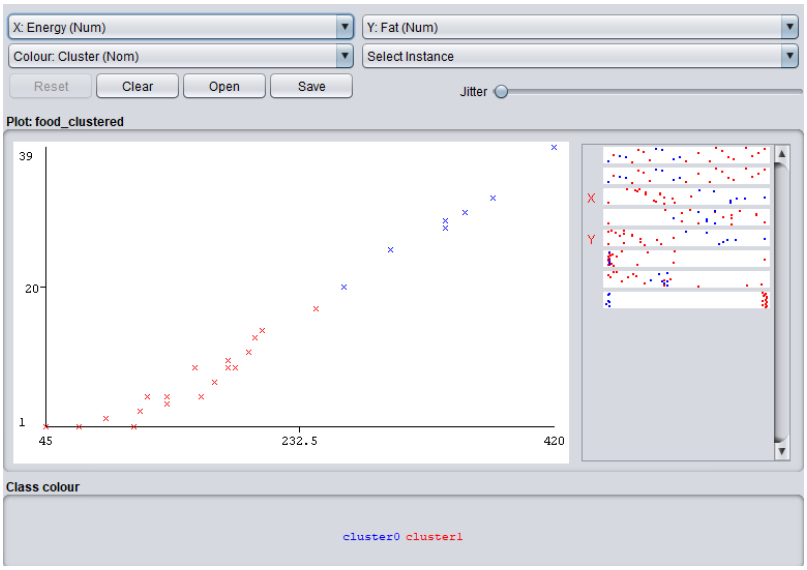
1. Use the Simple KMeans clusterer which gave the result you haven chosen in 5).

As specified, we used the Simple kmeans clusterer with seed value=10 and k=2 and obtained the following results.

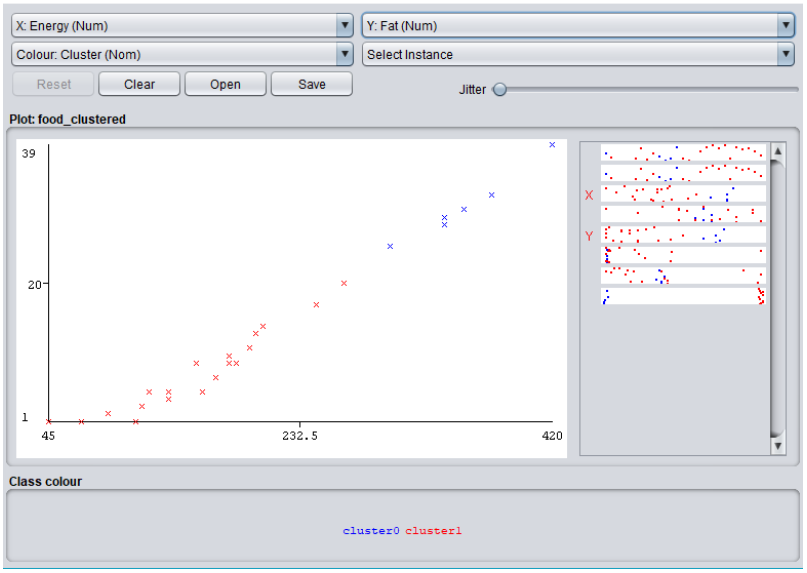
2. Experiment with at least two different standard deviations. Compare the results. (Hint: Increasing the standard deviation to higher values will make the differences in different runs more obvious and thus it will be easier to conclude what the parameter does)

With sd: 1.0E-6: log likelihood value= -8.84267 With sd: 100: log likelihood value= -11.59844 With sd: 1000: log likelihood value= -15.6623 The log likelihood value decreases with increase in standard deviation value.

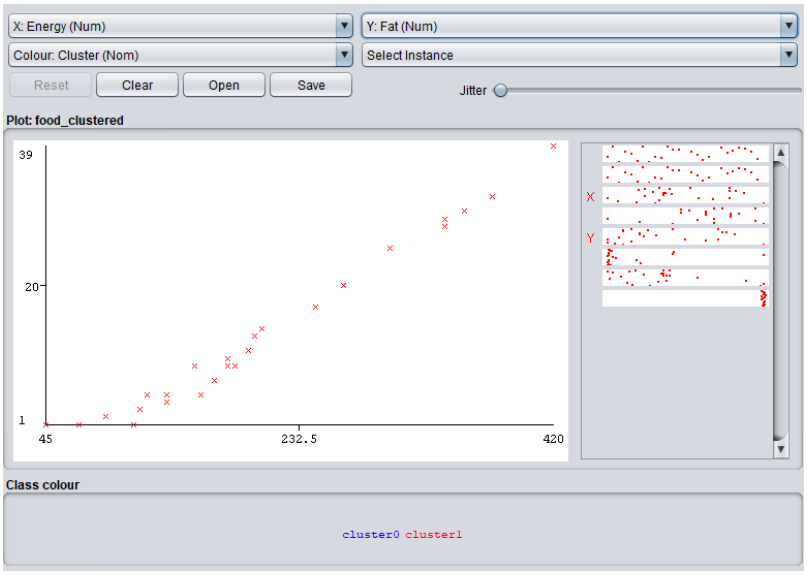
When sd: 1.0E-6



When sd: 100



When sd: 1000



When we see the above images, we can conclude that, on increasing the standard deviation, the data points tends to be included in a single cluster rather than being in different clusters. So, we need to choose the standard deviation to be minimum to obtain the good cluster quality.