Clustering with Different Modifications

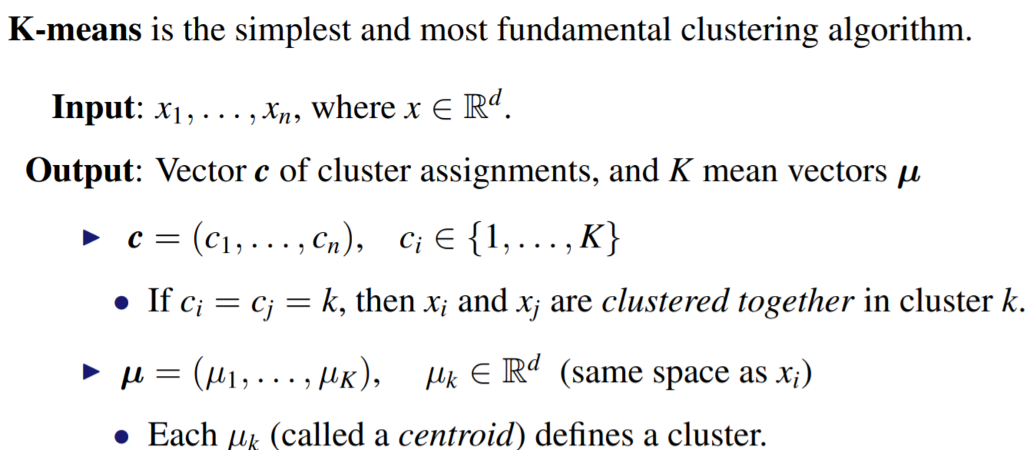
贾之远 118010114

1.Model Construction and basic implementation

1.1 Data Preprocessing

Firstly we read the data , from https: //archive.ics.uci.edu/ml/datasets/seeds. The number of clusters is set as 3.

1.2 K-means



K-means is the simplest and most fundamental clustering algorithm.

Input: $x_1, \dots, x_n$, where $x \in \mathbb{R}^d$.

Output: Vector $\boldsymbol{c}$ of cluster assignments, and K mean vectors $\boldsymbol{\mu}$

- $\boldsymbol{c} = (c_1, \dots, c_n), \quad c_i \in \{1, \dots, K\}$
  - If $c_i = c_j = k$, then x_i and x_j are *clustered together* in cluster k.
- $\boldsymbol{\mu} = (\mu_1, \dots, \mu_K), \quad \mu_k \in \mathbb{R}^d$ (same space as x_i)
  - Each μ_k (called a *centroid*) defines a cluster.

We First initialize, then iterated back and forth between two steps:

1. Given centroid， find the best indices c_i
2. Given vector c, finding the best centroid.

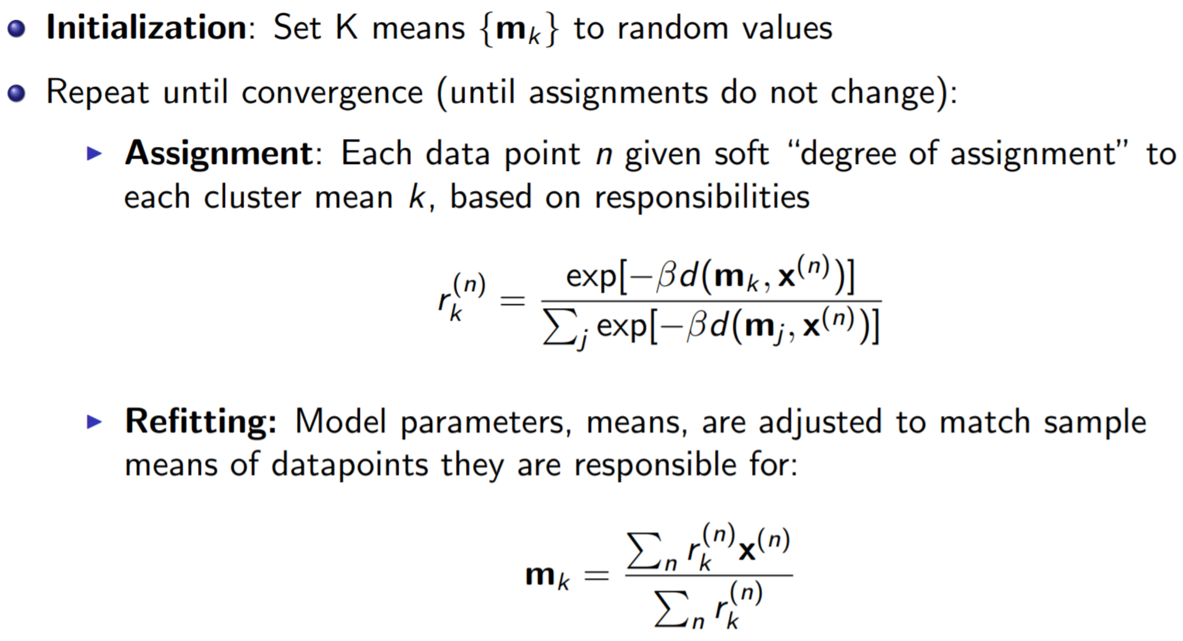
Remark that we we used the Euclidean norm. The processes will converges.

Result: The Center is

```
[[18.72180328 16.29737705  0.88508689  6.20893443  3.72267213  3.60359016
   6.06609836]
 [14.81910448 14.53716418  0.88052239  5.59101493  3.29935821  2.70658507
   5.21753731]
 [11.98865854 13.28439024  0.85273659  5.22742683  2.88008537  4.58392683
   5.0742439 ]]
```

1.3 Soft K-means

In this assignment, instead of direct assign, we used the responsibility as probability to assign, which allows the cluster to use more information. The steps are Assignment and refitting:



- **Initialization**: Set K means $\{\mathbf{m}_k\}$ to random values
- Repeat until convergence (until assignments do not change):
  - **Assignment**: Each data point n given soft "degree of assignment" to each cluster mean k, based on responsibilities

$$r_k^{(n)} = \frac{\exp[-\beta d(\mathbf{m}_k, \mathbf{x}^{(n)})]}{\sum_j \exp[-\beta d(\mathbf{m}_j, \mathbf{x}^{(n)})]}$$

  - **Refitting:** Model parameters, means, are adjusted to match sample means of datapoints they are responsible for:

$$\mathbf{m}_k = \frac{\sum_n r_k^{(n)} \mathbf{x}^{(n)}}{\sum_n r_k^{(n)}}$$

Result: The Center is

```
[[15.25450407 14.74244842  0.87288767  5.6882123   3.3095374   3.64059351
   5.47043871]
 [14.88182935 14.57466316  0.87122872  5.63327946  3.26322658  3.6857019
   5.4117882 ]
 [14.42236686 14.36800281  0.86895601  5.56646603  3.20507726  3.77170112
   5.34441935]]
```

1.4 Accelerated K-means

Reference: https://www.aaai.org/Papers/ICML/2003/ICML03-022.pdf

With Lemma 1 and Lemma 2 reported in the paper (triangle-inequality), it avoid unnecessary distance calculations by keeping lower bounds, The alogrithms is as follows:

First, pick initial centers. Set the lower bound $l(x, c) = 0$ for each point x and center c. Assign each x to its closest initial center $c(x) = \operatorname{argmin}_c d(x, c)$, using Lemma 1 to avoid redundant distance calculations. Each time $d(x, c)$ is computed, set $l(x, c) = d(x, c)$. Assign upper bounds $u(x) = \min_c d(x, c)$ Next, repeat until convergence:

Second,

1. For all centers c and c', compute $d(c, c')$. For all centers c, compute $s(c) = \frac{1}{2}\min_{c' \neq c} d(c, c')$
2. Identify all points x such that $u(x) \leq s(c(x))$.
3. For all remaining points x and centers c such that (i) $c \neq c(x)$ and (ii) $u(x) > l(x, c)$ and (iii) $u(x) > \frac{1}{2}d(c(x), c)$

3a. If $r(x)$ then compute $d(x, c(x))$ and assign $r(x) =$ false. Otherwise, $d(x, c(x)) = u(x)$. 3b. If $d(x, c(x)) > l(x, c)$ or $d(x, c(x)) > \frac{1}{2}d(c(x), c)$ then Compute $d(x, c)$ If $d(x, c) < d(x, c(x))$ then assign $c(x) = c$.

4. For each center c, let $m(c)$ be the mean of the points assigned to c.
5. For each point x and center c, assign

$$l(x, c)=\max \{l(x, c)-d(c, m(c)), 0\}.$$

6. For each point x, assign

$$\begin{aligned} u(x) &= u(x)+d(m(c(x)), c(x)) \\ r(x) &= \text{ true. } \end{aligned}$$

7. Replace each center c by $m(c)$.

Third, we repeated the second process 1-7 until convergence.

Result: The Center is

```
[[17.80896552 15.89643678  0.88361379  6.06827586  3.62617241  3.40872414
   5.88771264]
 [14.568125   14.42        0.87998125  5.558625    3.252875    1.7567
   5.1280625 ]
 [12.48140187 13.4928972   0.85939813  5.28143925  2.96059813  4.22781308
   5.05995327]]
```

1.5 GMM with EM

In this part, we use GMM to approximate the densities of a clustering by linear combinations of Gaussian Distribution.

$$p(\mathbf{x})=\sum_{k=1}^{K} \pi_{k} \mathcal{N}\left(\mathbf{x} \mid \mu_{k}, \Sigma_{k}\right)$$

with π_k the mixing coefficients, where:

$$\sum_{k=1}^{K} \pi_{k}=1 \quad \text { and } \quad \pi_{k} \geq 0 \quad \forall k$$

To optimize the fitting, we maximize log likelihood

$\ln p(\mathbf{X} \mid \pi, \mu, \Sigma)=\sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_{k} \mathcal{N}\left(\mathbf{x}^{(n)} \mid \mu_{k}, \Sigma_{k}\right)\right)$w.r.t $\Theta=\left\{\pi_{k}, \mu_{k}, \Sigma_{k}\right\}$

Then we apply the EM algorithm, that is

E-step: Compute the posterior probability over z given our current model - i.e. how much do we think each Gaussian generates each data-point.

M-step: Assuming that the data really was generated this way, change the parameters of each Gaussian to maximize the probability that it would generate the data it is currently responsible for.

Result: The Center is

```
[[16.84695502 15.45205366  0.88329952  5.90919313  3.52285685  3.23572505
   5.65115979]
 [12.11       13.47        0.8392      5.159       3.032       1.502
   4.519     ]
 [12.04822153 13.30782864  0.85394953  5.23658141  2.88707604  4.38325053
   5.07416552]]
```

2. Comparison between Evaluation Metrics

Evaluation (or "validation") of clustering results is as difficult as the clustering itself. Popular approaches involve "*internal*" evaluation, where the clustering is summarized to a single quality score, "*external*" evaluation, where the clustering is compared to an existing "ground truth" classification, "*manual*" evaluation by a human expert, and "*indirect*" evaluation by evaluating the utility of the clustering in its intended application

In external evaluation, clustering results are evaluated based on data that was not used for clustering, such as known class labels and external benchmarks. Such benchmarks consist of a set of pre-classified items, and these sets are often created by (expert) humans. Thus, the benchmark sets can be thought of as a gold standard for evaluation.[33] These types of evaluation methods measure how close the clustering is to the predetermined benchmark classes. However, it has recently been discussed whether this is adequate for real data, or only on synthetic data sets with a factual ground truth, since classes can contain internal structure, the attributes present may not allow separation of clusters or the classes may contain anomalies.[39] Additionally, from a knowledge discovery point of view, the reproduction of known knowledge may not necessarily be the intended result.[39] In the special scenario of constrained clustering, where meta information (such as class labels) is used already in the clustering process, the hold-out of information for evaluation purposes is non-trivial.[40]

A number of measures are adapted from variants used to evaluate classification tasks. In place of counting the number of times a class was correctly assigned to a single data point (known as true positives), such *pair counting* metrics assess whether each pair of data points that is truly in the same cluster is predicted to be in the same cluster.

We consider the external:

2.1 Purity

Purity is a measure of the extent to which clusters contain a single class.[36] Its calculation can be thought of as follows: For each cluster, count the number of data points from the most common class in said cluster. Now take the sum over all clusters and divide by the total number of data points. Formally, given some set of clusters and some set of classes, both partitioning data points, purity can be defined as:

$$\frac{1}{N}\sum_{m\in M}\max_{d\in D}|m\cap d|$$

2.2 Rand Index

The Rand index computes how similar the clusters (returned by the clustering algorithm) are to the benchmark classifications. It can be computed using the following formula:

$$RI = \frac{TP+TN}{TP+FP+FN+TN}$$

TP,TN,FP,FN are the True Positive, True Negative, False Positive and False Negative .

2.3 NMI(Normalized Mutual Information)

$$\mathrm{NMI}(\Omega, C)=\frac{I(\Omega ; \mathbb{C})}{[H(\Omega)+H(\mathbb{C})] / 2}$$

where

$$I(\Omega ; \mathbb{C})=\quad \frac{\sum_k \sum_j P(\omega_k \cap c_j) \log \frac{P\left(\omega_k \cap c_j\right)}{P(\omega_k) P\left(c_j\right)}}{\sum_k \sum_j \frac{\left|\omega_k \cap c_j\right|}{N} \log \frac{N\left|\omega_k \cap c_j\right|}{\left|\omega_k\right|\left|c_j\right|}}$$

$$H(\Omega)=\frac{-\sum_k P\left(\omega_k\right) \log P\left(\omega_k\right)}{-\sum_k \frac{\left|\omega_k\right|}{N} \log \frac{\left|\omega_k\right|}{N}}$$

2.4 Comparisons:

	K-means	Soft-K-means	Acc-K-MEANS	GMM-EM
Purity	0.8904761904761904	0.7095238095238096	0.7428571428571429	0.6714285714285714
NMI	0.7100637577760451	0.5443371417462751	0.5644693562905989	0.5562826317966371
RI	0.6698564593301436	0.6698564593301436	0.6698564593301436	0.6698564593301436

Using RI, all those algorithms appears to be have similar performance.

Using Purity and NMI, K-Means is the best one, which indicated the data are well-separated.

3. Sensitivity Analysis

The Result of the Clustering is correlated to the generation of the initial value. For some Algorithms, such as K-means, the optimal may falls in the local extrema, thus considering some random initialization and its corresponding sensitivity.

At first, we can choose the first 5 centers for the initializations. For Sensitive Analysis, we applied the random initialization and repeated to see its sensitivity. We initialized each alogrithms 10 times.(appendix)

We compare the variance of the indices, and notice that for sensitivity, in this test,

Accelerated K-means is the worst, while Soft K-means is the best with least varience, which is the most Robust one.

4. Reporting of Additional Information

In this section, we answer the 4th problem by

4.1 Running Iterations

K-means	Accelerated K-means	GMM-EM	Soft-Kmeans
5	1	5 (max-restrict)	6

4.2 Running Time

K-means	Accelerated K-means	GMM-EM	Soft-K-means
0.0008	0.013	0.0043	0.002

So Accelerated used the smallest numbers of iterations, with K means the fastest.

Appendix: Some using Output

```
Name:  <class 'clusters.KMeans'>
Fit time 0.0008523464202880859
Iteration:  5
centriods:
[[18.72180328 16.29737705  0.88508689  6.20893443  3.72267213  3.60359016
   6.06609836]
 [14.81910448 14.53716418  0.88052239  5.59101493  3.29935821  2.70658507
   5.21753731]
 [11.98865854 13.28439024  0.85273659  5.22742683  2.88008537  4.58392683
   5.0742439 ]]
Purity:  0.8904761904761904
NMI:  0.7100637577760451
RI:  0.6698564593301436
---
Name:  <class 'clusters.SoftKMeans'>
Fit time 0.002033233642578125
Iteration:  6
centriods:
[[15.25450407 14.74244842  0.87288767  5.6882123   3.3095374   3.64059351
   5.47043871]
 [14.88182935 14.57466316  0.87122872  5.63327946  3.26322658  3.6857019
   5.4117882 ]
 [14.42236686 14.36800281  0.86895601  5.56646603  3.20507726  3.77170112
   5.34441935]]
Purity:  0.7095238095238096
NMI:  0.5443371417462751
RI:  0.6698564593301436
---
Name:  <class 'clusters.AccelaratedKMeans'>
Fit time 0.013448953628540039
Iteration:  1
centriods:
[[17.80896552 15.89643678  0.88361379  6.06827586  3.62617241  3.40872414
   5.88771264]
 [14.568125   14.42        0.87998125  5.558625    3.252875    1.7567
   5.1280625 ]
 [12.48140187 13.4928972   0.85939813  5.28143925  2.96059813  4.22781308
   5.05995327]]
Purity:  0.7428571428571429
NMI:  0.5644693562905989
RI:  0.6698564593301436
---
Name:  <class 'clusters.GMM'>
Fit time 0.004369735717773437 5
Iteration:  5
centriods:
```

```
[[16.84695502 15.45205366  0.88329952  5.90919313  3.52285685  3.23572505
   5.65115979]
 [12.11       13.47        0.8392      5.159       3.032       1.502
   4.519     ]
 [12.04822153 13.30782864  0.85394953  5.23658141  2.88707604  4.38325053
   5.07416552]]
Purity:  0.6714285714285714
NMI:  0.5562826317966371
RI:  0.6698564593301436
```

```
#### Sensitive analysis

----------------------------------------------------------------------
Ran 1 test in 0.000s

OK (skipped=1)
test_acc (test.TestClusters) ... skipped ''
test_gmm (test.TestClusters) ... skipped ''
test_k_means (test.TestClusters) ... skipped ''
test_report (test.TestClusters) ... Name:  <class 'clusters.KMeans'>
Fit time 0.0019948482513427734
Iteration:  5
centriods:
[[18.72180328 16.29737705  0.88508689  6.20893443  3.72267213  3.60359016
   6.06609836]
 [14.81910448 14.53716418  0.88052239  5.59101493  3.29935821  2.70658507
   5.21753731]
 [11.98865854 13.28439024  0.85273659  5.22742683  2.88008537  4.58392683
   5.0742439 ]]
Purity:  0.8904761904761904
NMI:  0.7100637577760451
RI:  0.6698564593301436
---
Name:  <class 'clusters.SoftKMeans'>
Fit time 0.003025531768798828
Iteration:  6
centriods:
[[15.25450407 14.74244842  0.87288767  5.6882123   3.3095374   3.64059351
   5.47043871]
 [14.88182935 14.57466316  0.87122872  5.63327946  3.26322658  3.6857019
   5.4117882 ]
 [14.42236686 14.36800281  0.86895601  5.56646603  3.20507726  3.77170112
   5.34441935]]
Purity:  0.7095238095238096
NMI:  0.5443371417462751
RI:  0.6698564593301436
---
Name:  <class 'clusters.AccelaratedKMeans'>
Fit time 0.01995086669921875
Iteration:  1
centriods:
[[17.80896552 15.89643678  0.88361379  6.06827586  3.62617241  3.40872414
   5.88771264]
 [14.568125   14.42        0.87998125  5.558625    3.252875    1.7567
   5.1280625 ]
 [12.48140187 13.4928972   0.85939813  5.28143925  2.96059813  4.22781308
   5.05995327]]
Purity:  0.7428571428571429
NMI:  0.5644693562905989
RI:  0.6698564593301436
---
Name:  <class 'clusters.GMM'>
Fit time 0.0069828033447265625
Iteration:  5
centriods:
[[16.84695502 15.45205366  0.88329952  5.90919313  3.52285685  3.23572505
   5.65115979]
 [12.11       13.47        0.8392      5.159       3.032       1.502
   4.519     ]
 [12.04822153 13.30782864  0.85394953  5.23658141  2.88707604  4.38325053
   5.07416552]]
Purity:  0.6714285714285714
NMI:  0.5562826317966371
RI:  0.6698564593301436
---
ok

----------------------------------------------------------------------
Ran 4 tests in 0.610s

OK (skipped=3)
test_acc (test.TestClusters) ... skipped ''
test_gmm (test.TestClusters) ... skipped ''
test_k_means (test.TestClusters) ... skipped ''
test_report (test.TestClusters) ... Name:  <class 'clusters.KMeans'>
Fit time 0.001994609832763672
Iteration:  5
centriods:
[[18.72180328 16.29737705  0.88508689  6.20893443  3.72267213  3.60359016
   6.06609836]
 [14.81910448 14.53716418  0.88052239  5.59101493  3.29935821  2.70658507
   5.21753731]
 [11.98865854 13.28439024  0.85273659  5.22742683  2.88008537  4.58392683
   5.0742439 ]]
Purity:  0.8904761904761904
NMI:  0.7100637577760451
RI:  0.6698564593301436
---
Name:  <class 'clusters.SoftKMeans'>
Fit time 0.003989219665527344
Iteration:  6
centriods:
[[15.25450407 14.74244842  0.87288767  5.6882123   3.3095374   3.64059351
   5.47043871]
 [14.88182935 14.57466316  0.87122872  5.63327946  3.26322658  3.6857019
   5.4117882 ]
 [14.42236686 14.36800281  0.86895601  5.56646603  3.20507726  3.77170112
   5.34441935]]
Purity:  0.7095238095238096
NMI:  0.5443371417462751
RI:  0.6698564593301436
---
Name:  <class 'clusters.AccelaratedKMeans'>
```

```
Fit time 0.017918825149536133
Iteration:  1
centriods:
[[17.80896552 15.89643678  0.88361379  6.06827586  3.62617241  3.40872414
   5.88771264]
 [14.568125   14.42        0.87998125  5.558625    3.252875    1.7567
   5.1280625 ]
 [12.48140187 13.4928972   0.85939813  5.28143925  2.96059813  4.22781308
   5.05995327]]
Purity:  0.7428571428571429
NMI:  0.5644693562905989
RI:  0.6698564593301436
---
Name:  <class 'clusters.GMM'>
Fit time 0.005984067916870117
Iteration:  5
centriods:
[[16.84695502 15.45205366  0.88329952  5.90919313  3.52285685  3.23572505
   5.65115979]
 [12.11       13.47        0.8392      5.159       3.032       1.502
   4.519     ]
 [12.04822153 13.30782864  0.85394953  5.23658141  2.88707604  4.38325053
   5.07416552]]
Purity:  0.6714285714285714
NMI:  0.5562826317966371
RI:  0.6698564593301436
---
ok

----------------------------------------------------------------------
Ran 4 tests in 0.587s

OK (skipped=3)
test_acc (test.TestClusters) ... skipped ''
test_gmm (test.TestClusters) ... skipped ''
test_k_means (test.TestClusters) ... skipped ''
test_report (test.TestClusters) ... Name:  <class 'clusters.KMeans'>
Fit time 0.0019941329956054688
Iteration:  5
centriods:
[[18.72180328 16.29737705  0.88508689  6.20893443  3.72267213  3.60359016
   6.06609836]
 [14.81910448 14.53716418  0.88052239  5.59101493  3.29935821  2.70658507
   5.21753731]
 [11.98865854 13.28439024  0.85273659  5.22742683  2.88008537  4.58392683
   5.0742439 ]]
Purity:  0.8904761904761904
NMI:  0.7100637577760451
RI:  0.6698564593301436
---
Name:  <class 'clusters.SoftKMeans'>
Fit time 0.0029931068420410156
Iteration:  6
centriods:
[[15.25450407 14.74244842  0.87288767  5.6882123   3.3095374   3.64059351
   5.47043871]
 [14.88182935 14.57466316  0.87122872  5.63327946  3.26322658  3.6857019
   5.4117882 ]
 [14.42236686 14.36800281  0.86895601  5.56646603  3.20507726  3.77170112
   5.34441935]]
Purity:  0.7095238095238096
NMI:  0.5443371417462751
RI:  0.6698564593301436
---
Name:  <class 'clusters.AccelaratedKMeans'>
Fit time 0.01795339584350586
Iteration:  1
centriods:
[[17.80896552 15.89643678  0.88361379  6.06827586  3.62617241  3.40872414
   5.88771264]
 [14.568125   14.42        0.87998125  5.558625    3.252875    1.7567
   5.1280625 ]
 [12.48140187 13.4928972   0.85939813  5.28143925  2.96059813  4.22781308
   5.05995327]]
Purity:  0.7428571428571429
NMI:  0.5644693562905989
RI:  0.6698564593301436
---
Name:  <class 'clusters.GMM'>
Fit time 0.014958858489990234
Iteration:  5
centriods:
[[16.84695502 15.45205366  0.88329952  5.90919313  3.52285685  3.23572505
   5.65115979]
 [12.11       13.47        0.8392      5.159       3.032       1.502
   4.519     ]
 [12.04822153 13.30782864  0.85394953  5.23658141  2.88707604  4.38325053
   5.07416552]]
Purity:  0.6714285714285714
NMI:  0.5562826317966371
RI:  0.6698564593301436
---ok

----------------------------------------------------------------------

Ran 4 tests in 0.581s

OK (skipped=3)
test_acc (test.TestClusters) ... skipped ''
test_gmm (test.TestClusters) ... skipped ''
test_k_means (test.TestClusters) ... skipped ''
test_report (test.TestClusters) ... Name:  <class 'clusters.KMeans'>
Fit time 0.001992940902709961
Iteration:  5
centriods:
[[18.72180328 16.29737705  0.88508689  6.20893443  3.72267213  3.60359016
   6.06609836]
 [14.81910448 14.53716418  0.88052239  5.59101493  3.29935821  2.70658507
   5.21753731]
 [11.98865854 13.28439024  0.85273659  5.22742683  2.88008537  4.58392683
   5.0742439 ]]
Purity:  0.8904761904761904
NMI:  0.7100637577760451
RI:  0.6698564593301436
---
Name:  <class 'clusters.SoftKMeans'>
```

```
Fit time 0.002991914749145508
Iteration:  6
centriods:
[[15.25450407 14.74244842  0.87288767  5.6882123   3.3095374   3.64059351
   5.47043871]
 [14.88182935 14.57466316  0.87122872  5.63327946  3.26322658  3.6857019
   5.4117882 ]
 [14.42236686 14.36800281  0.86895601  5.56646603  3.20507726  3.77170112
   5.34441935]]
Purity:  0.7095238095238096
NMI:  0.5443371417462751
RI:  0.6698564593301436
---
Name:  <class 'clusters.AccelaratedKMeans'>
Fit time 0.019956588745117188
Iteration:  1
centriods:
[[17.80896552 15.89643678  0.88361379  6.06827586  3.62617241  3.40872414
   5.88771264]
 [14.568125   14.42        0.87998125  5.558625    3.252875    1.7567
   5.1280625 ]
 [12.48140187 13.4928972   0.85939813  5.28143925  2.96059813  4.22781308
   5.05995327]]
Purity:  0.7428571428571429
NMI:  0.5644693562905989
RI:  0.6698564593301436
---
Name:  <class 'clusters.GMM'>
Fit time 0.006937265396118164
Iteration:  5
centriods:
[[16.84695502 15.45205366  0.88329952  5.90919313  3.52285685  3.23572505
   5.65115979]
 [12.11       13.47        0.8392      5.159       3.032       1.502
   4.519     ]
 [12.04822153 13.30782864  0.85394953  5.23658141  2.88707604  4.38325053
   5.07416552]]
Purity:  0.6714285714285714
NMI:  0.5562826317966371
RI:  0.6698564593301436
---
ok

----------------------------------------------------------------------
Ran 4 tests in 0.584s

OK (skipped=3)
test_acc (test.TestClusters) ... skipped ''
test_gmm (test.TestClusters) ... skipped ''
test_k_means (test.TestClusters) ... skipped ''
test_report (test.TestClusters) ... Name:  <class 'clusters.KMeans'>
Fit time 0.008041143417358398
Iteration:  5
centriods:
[[18.72180328 16.29737705  0.88508689  6.20893443  3.72267213  3.60359016
   6.06609836]
 [14.81910448 14.53716418  0.88052239  5.59101493  3.29935821  2.70658507
   5.21753731]
 [11.98865854 13.28439024  0.85273659  5.22742683  2.88008537  4.58392683
   5.0742439 ]]
Purity:  0.8904761904761904
NMI:  0.7100637577760451
RI:  0.6698564593301436
---
Name:  <class 'clusters.SoftKMeans'>
Fit time 0.002992391586303711
Iteration:  6
centriods:
[[15.25450407 14.74244842  0.87288767  5.6882123   3.3095374   3.64059351
   5.47043871]
 [14.88182935 14.57466316  0.87122872  5.63327946  3.26322658  3.6857019
   5.4117882 ]
 [14.42236686 14.36800281  0.86895601  5.56646603  3.20507726  3.77170112
   5.34441935]]
Purity:  0.7095238095238096
NMI:  0.5443371417462751
RI:  0.6698564593301436
---
Name:  <class 'clusters.AccelaratedKMeans'>
Fit time 0.026928424835205078
Iteration:  1
centriods:
[[17.80896552 15.89643678  0.88361379  6.06827586  3.62617241  3.40872414
   5.88771264]
 [14.568125   14.42        0.87998125  5.558625    3.252875    1.7567
   5.1280625 ]
 [12.48140187 13.4928972   0.85939813  5.28143925  2.96059813  4.22781308
   5.05995327]]
Purity:  0.7428571428571429
NMI:  0.5644693562905989
RI:  0.6698564593301436
---
Name:  <class 'clusters.GMM'>
Fit time 0.005985975265502930
Iteration:  5
centriods:
[[16.84695502 15.45205366  0.88329952  5.90919313  3.52285685  3.23572505
   5.65115979]
 [12.11       13.47        0.8392      5.159       3.032       1.502
   4.519     ]
 [12.04822153 13.30782864  0.85394953  5.23658141  2.88707604  4.38325053
   5.07416552]]
Purity:  0.6714285714285714
NMI:  0.5562826317966371
RI:  0.6698564593301436
---
ok

----------------------------------------------------------------------
Ran 4 tests in 0.650s

OK (skipped=3)
test_acc (test.TestClusters) ... skipped ''
test_gmm (test.TestClusters) ... skipped ''
test_k_means (test.TestClusters) ... skipped ''
test_report (test.TestClusters) ... Name:  <class 'clusters.KMeans'>
```

```
Fit time 0.001984834671020508
Iteration:  5
centriods:
[[18.72180328 16.29737705  0.88508689  6.20893443  3.72267213  3.60359016
   6.06609836]
 [14.81910448 14.53716418  0.88052239  5.59101493  3.29935821  2.70658507
   5.21753731]
 [11.98865854 13.28439024  0.85273659  5.22742683  2.88008537  4.58392683
   5.0742439 ]]
Purity:  0.8904761904761904
NMI:  0.7100637577760451
RI:  0.6698564593301436
---
Name:  <class 'clusters.SoftKMeans'>
Fit time 0.0029935836791992188
Iteration:  6
centriods:
[[15.25450407 14.74244842  0.87288767  5.6882123   3.3095374   3.64059351
   5.47043871]
 [14.88182935 14.57466316  0.87122872  5.63327946  3.26322658  3.6857019
   5.4117882 ]
 [14.42236686 14.36800281  0.86895601  5.56646603  3.20507726  3.77170112
   5.34441935]]
Purity:  0.7095238095238096
NMI:  0.5443371417462751
RI:  0.6698564593301436
---
Name:  <class 'clusters.AccelaratedKMeans'>
Fit time 0.021973848342895508
Iteration:  1
centriods:
[[17.80896552 15.89643678  0.88361379  6.06827586  3.62617241  3.40872414
   5.88771264]
 [14.568125   14.42        0.87998125  5.558625    3.252875    1.7567
   5.1280625 ]
 [12.48140187 13.4928972   0.85939813  5.28143925  2.96059813  4.22781308
   5.05995327]]
Purity:  0.7428571428571429
NMI:  0.5644693562905989
RI:  0.6698564593301436
---
Name:  <class 'clusters.GMM'>
Fit time 0.009959220886230469
Iteration:  5
centriods:
[[16.84695502 15.45205366  0.88329952  5.90919313  3.52285685  3.23572505
   5.65115979]
 [12.11       13.47        0.8392      5.159       3.032       1.502
   4.519     ]
 [12.04822153 13.30782864  0.85394953  5.23658141  2.88707604  4.38325053
   5.07416552]]
Purity:  0.6714285714285714
NMI:  0.5562826317966371
RI:  0.6698564593301436
ok---


----------------------------------------------------------------------
```

```
Ran 4 tests in 0.574s

OK (skipped=3)
test_acc (test.TestClusters) ... skipped ''
test_gmm (test.TestClusters) ... skipped ''
test_k_means (test.TestClusters) ... skipped ''
test_report (test.TestClusters) ... Name:  <class 'clusters.KMeans'>
Fit time 0.001001119613647461
Iteration:  5
centriods:
[[18.72180328 16.29737705  0.88508689  6.20893443  3.72267213  3.60359016
   6.06609836]
 [14.81910448 14.53716418  0.88052239  5.59101493  3.29935821  2.70658507
   5.21753731]
 [11.98865854 13.28439024  0.85273659  5.22742683  2.88008537  4.58392683
   5.0742439 ]]
Purity:  0.8904761904761904
NMI:  0.7100637577760451
RI:  0.6698564593301436
---
Name:  <class 'clusters.SoftKMeans'>
Fit time 0.004992246627807617
Iteration:  6
centriods:
[[15.25450407 14.74244842  0.87288767  5.6882123   3.3095374   3.64059351
   5.47043871]
 [14.88182935 14.57466316  0.87122872  5.63327946  3.26322658  3.6857019
   5.4117882 ]
 [14.42236686 14.36800281  0.86895601  5.56646603  3.20507726  3.77170112
   5.34441935]]
Purity:  0.7095238095238096
NMI:  0.5443371417462751
RI:  0.6698564593301436
---
Name:  <class 'clusters.AccelaratedKMeans'>
Fit time 0.017951488494873047
Iteration:  1
centriods:
[[17.80896552 15.89643678  0.88361379  6.06827586  3.62617241  3.40872414
   5.88771264]
 [14.568125   14.42        0.87998125  5.558625    3.252875    1.7567
   5.1280625 ]
 [12.48140187 13.4928972   0.85939813  5.28143925  2.96059813  4.22781308
   5.05995327]]
Purity:  0.7428571428571429
NMI:  0.5644693562905989
RI:  0.6698564593301436
---
Name:  <class 'clusters.GMM'>
Fit time 0.00997614860534668
Iteration:  5
centriods:
[[16.84695502 15.45205366  0.88329952  5.90919313  3.52285685  3.23572505
   5.65115979]
 [12.11       13.47        0.8392      5.159       3.032       1.502
   4.519     ]
 [12.04822153 13.30782864  0.85394953  5.23658141  2.88707604  4.38325053
   5.07416552]]
```

```
Purity:  0.6714285714285714
NMI:  0.5562826317966371
RI:  0.6698564593301436
---
ok

----------------------------------------------------------------------
Ran 4 tests in 0.589s

OK (skipped=3)
test_acc (test.TestClusters) ... skipped ''
test_gmm (test.TestClusters) ... skipped ''
test_k_means (test.TestClusters) ... skipped ''
test_report (test.TestClusters) ... Name:  <class 'clusters.KMeans'>
Fit time 0.003997325897216797
Iteration:  5
centriods:
[[18.72180328 16.29737705  0.88508689  6.20893443  3.72267213  3.60359016
   6.06609836]
 [14.81910448 14.53716418  0.88052239  5.59101493  3.29935821  2.70658507
   5.21753731]
 [11.98865854 13.28439024  0.85273659  5.22742683  2.88008537  4.58392683
   5.0742439 ]]
Purity:  0.8904761904761904
NMI:  0.7100637577760451
RI:  0.6698564593301436
---
Name:  <class 'clusters.SoftKMeans'>
Fit time 0.0029835700988769 53
Iteration:  6
centriods:
[[15.25450407 14.74244842  0.87288767  5.6882123   3.3095374   3.64059351
   5.47043871]
 [14.88182935 14.57466316  0.87122872  5.63327946  3.26322658  3.6857019
   5.4117882 ]
 [14.42236686 14.36800281  0.86895601  5.56646603  3.20507726  3.77170112
   5.34441935]]
Purity:  0.7095238095238096
NMI:  0.5443371417462751
RI:  0.6698564593301436
---
Name:  <class 'clusters.AccelaratedKMeans'>
Fit time 0.019947290420532227
Iteration:  1
centriods:
[[17.80896552 15.89643678  0.88361379  6.06827586  3.62617241  3.40872414
   5.88771264]
 [14.568125   14.42        0.87998125  5.558625    3.252875    1.7567
   5.1280625 ]
 [12.48140187 13.4928972   0.85939813  5.28143925  2.96059813  4.22781308
   5.05995327]]
Purity:  0.7428571428571429
NMI:  0.5644693562905989
RI:  0.6698564593301436
---
Name:  <class 'clusters.GMM'>
Fit time 0.008974313735961914
Iteration:  5
```

```
centriods:
[[16.84695502 15.45205366  0.88329952  5.90919313  3.52285685  3.23572505
   5.65115979]
 [12.11       13.47        0.8392      5.159       3.032       1.502
   4.519     ]
 [12.04822153 13.30782864  0.85394953  5.23658141  2.88707604  4.38325053
   5.07416552]]
Purity:  0.6714285714285714
NMI:  0.5562826317966371
RI:  0.6698564593301436
ok

---
----------------------------------------------------------------------
Ran 4 tests in 0.780s

OK (skipped=3)
test_acc (test.TestClusters) ... skipped ''
test_gmm (test.TestClusters) ... skipped ''
test_k_means (test.TestClusters) ... skipped ''
test_report (test.TestClusters) ... Name:  <class 'clusters.KMeans'>
Fit time 0.0009975433349609375
Iteration:  5
centriods:
[[18.72180328 16.29737705  0.88508689  6.20893443  3.72267213  3.60359016
   6.06609836]
 [14.81910448 14.53716418  0.88052239  5.59101493  3.29935821  2.70658507
   5.21753731]
 [11.98865854 13.28439024  0.85273659  5.22742683  2.88008537  4.58392683
   5.0742439 ]]
Purity:  0.8904761904761904
NMI:  0.7100637577760451
RI:  0.6698564593301436
---
Name:  <class 'clusters.SoftKMeans'>
Fit time 0.0059528350830078125
Iteration:  6
centriods:
[[15.25450407 14.74244842  0.87288767  5.6882123   3.3095374   3.64059351
   5.47043871]
 [14.88182935 14.57466316  0.87122872  5.63327946  3.26322658  3.6857019
   5.4117882 ]
 [14.42236686 14.36800281  0.86895601  5.56646603  3.20507726  3.77170112
   5.34441935]]
Purity:  0.7095238095238096
NMI:  0.5443371417462751
RI:  0.6698564593301436
---
Name:  <class 'clusters.AccelaratedKMeans'>
Fit time 0.016971349716186523
Iteration:  1
centriods:
[[17.80896552 15.89643678  0.88361379  6.06827586  3.62617241  3.40872414
   5.88771264]
 [14.568125   14.42        0.87998125  5.558625    3.252875    1.7567
   5.1280625 ]
 [12.48140187 13.4928972   0.85939813  5.28143925  2.96059813  4.22781308
   5.05995327]]
```

```
Purity:  0.7428571428571429
NMI:  0.5644693562905989
RI:  0.6698564593301436
---
Name:  <class 'clusters.GMM'>
Fit time 0.00997161865234375
Iteration:  5
centriods:
[[16.84695502 15.45205366  0.88329952  5.90919313  3.52285685  3.23572505
   5.65115979]
 [12.11       13.47        0.8392      5.159       3.032       1.502
   4.519     ]
 [12.04822153 13.30782864  0.85394953  5.23658141  2.88707604  4.38325053
   5.07416552]]
Purity:  0.6714285714285714
NMI:  0.5562826317966371
RI:  0.6698564593301436
ok

---
----------------------------------------------------------------------
Ran 4 tests in 0.583s

OK (skipped=3)
```