clusteringData

Mathieu Lagrange

November 5, 2015

Introduction

This report documents the second demonstration of the use of the expLanes framework to conduct a computational experiment. The project clusteringData is about the comparison of several algorithms that clusters synthetic data.

Design

The project is divided into two processing steps:

1. **generate**: generation of the synthetic datasets
2. **compute**: clustering of those datasets and comparison of the resulting clustering with true ones using standard performance metrics

***Generate step**: generation of synthetic datasets*

Each dataset comprises a set of points in a 2D space. Those points are grouped into classes of different shapes. The factors of variability that are to be tested are:

- the number of classes
- the shape of each clusters: spherical, spiral, and Gaussian
- the number of data points per class.

This step outputs for each dataset the location of the data points (elements) as well as the class to which they belong to (class).

***Compute step**: Computation of distances*

Once the datasets are generated, the data can be clustered into a predefined number of clusters using several algorithms. The main factor for this step is then the type of clustering algorithm. We consider three algorithms, the well known kMeans algorithm[1], the kernel kMeans one[2], and the kMedoids one[3].

For the kernel kMeans algorithm, one needs to specify the type of kernel, and for the exponential kernel, the σ has to be set. For the kMedoids algorithm, the type of similarity has to be specified. In order to have a lower bound baseline, the `chance` method is considered as a random placement of the elements into the clusters.

For all the algorithms, the clustering is run several times and the clustering with lowest intra-class distance is selected. Then, this process is run several times to gain statistical confidence in the observations by considering different initializations.

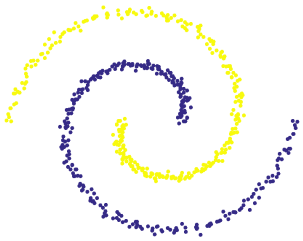

Figure 1: A dataset with spiral shaped clusters.

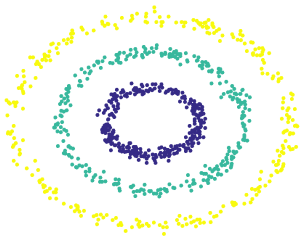

Figure 2: A dataset with spherical clusters.

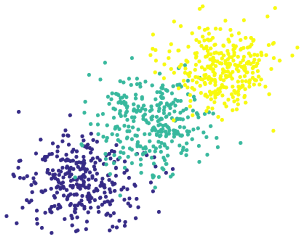

Figure 3: A dataset with Gaussian clusters.

[1]

[2]

[3]

Observations

We consider the following metrics:

1. **accuracy**: accuracy between the reference clustering and the achieved clustering maximally aligned with the reference clustering.
2. **nmi**: normalized mutual information

Definition of factors

Those factors and their corresponding modalities are defined in the file named `cldaFactors.txt` whose content is the following:

```
dataType === {'spiral', 'spherical', 'gaussian'}
nbClasses ==1/0/1,1/[2 3]/2= [2,3]
method =2== {'kMeans', 'kernelKmeans', 'kMedoids', 'chance'}
kernel =2=3/2= {'linear', 'polynomial', 'exponential'}
sigma =2=4/3= .2:.2:1
similarity =2=3/3= {'euclidean', 'seuclidean', 'cosine'}
nbElementsPerClass === 300
nbRuns =2== 4
nbReplicates =2== 5
nbIterations =2== 200
```

Please note the `2/2` between the last two equals of the factor definition of factor `kernel` which specify that this factor is needed only for the second modality of the third factor `method`. Along the same line, the `1/0/1,1/[2 3]/2` between the last two equals of the factor definition of factor `nbClasses` specify that its first modality is relevant for all the modalities of the factor `dataType`, but its second modality is relevant only for the second and third modalities of the factor `dataType`. Most the factor design discussed above is compactly displayed in Figure 4.

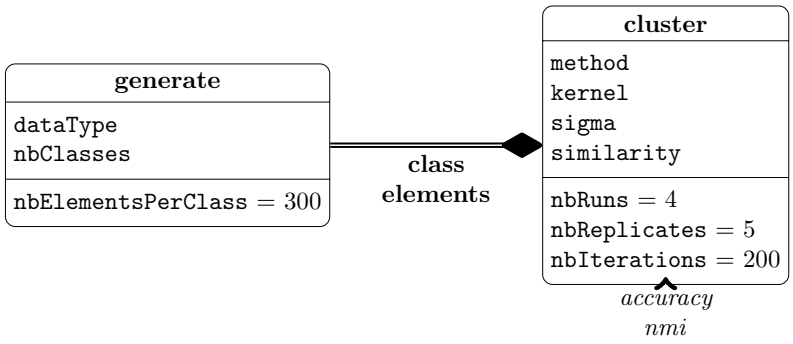

Figure 4: Factor and data flow graph.

Results

The results discussed next are generated as follows:

```
for k=length(expFactorValues(config, 'dataType')):-1:1
    config=expExpose(config, 't', 'mask', {k 1}, 'obs', [1 2], ...
        'fontSize', 'small', 'percent', 0, 'uncertainty', 1);
end
```

Gaussian clusters

Gaussian shaped clusters are notoriously easy, so the performance are quite good for this dataset, see Table 1. In this case, the kMeans and the kernel kMeans algorithm with linear, polynomial and exponential kernels generates equivalent clustering solutions.

Spherical clusters

The spherical shaped clusters are not linearly separable, thus the kMeans produces results that are close to chance, see Table 2. The kMedoids approach does not help either, as for the kernelKmeans with linear and polynomial kernel. Only an exponential kernel with a rather specific σ value is able to increase the performance above chance.

Spiral shaped clusters

The spiral shaped clusters are also not linearly separable. The same conclusions can be drawn, expect that the choice of the σ value appears as less critical, see Table 3.

Concerning the observations metrics, the accuracy is an interesting metric as it relates directly to the task at hand. That said, it has to be compared to the chance baseline in order to reduce the impact of the topology of the dataset considered. In this respect, the nmi is an interesting alternative that appears to be, for those simple datasets, well correlated with the accuracy, while being less influenced by the topology of the dataset.

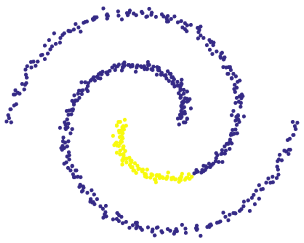

Figure 5: Clustering of the spiral shaped dataset using the exponential kernel.

Computation time

The average computation time for the different clustering methods can be displayed using:

```
config=expExpose(config, 't', 'obs', 3, 'integrate', [1 2 4 5 6], 'highlight', -1, 'sort', 1, ...
    'caption', 'Execution time in seconds');
```

The results shown on Table 4 reflect well the ranking of the different methods in terms of computational complexity.

method	kernel	sigma	similarity	accuracy (%)	nmi (%)
kernelKmeans	exponential	0.2		59±7	4
kernelKmeans	exponential	0.4		79±11	39
kernelKmeans	exponential	0.6		93±3	68
kernelKmeans	exponential	0.8		**97±0**	**81**
kernelKmeans	exponential	1.0		**97±0**	**81**
kernelKmeans	linear			**97±0**	**81**
kernelKmeans	polynomial			**97±0**	**81**
kMedoids			euclidean	96±1	76
kMedoids			seuclidean	96±0	77
kMedoids			cosine	53±0	0
kMeans				**97±0**	**81**
chance				52±2	0

Table 1: dataType: gaussian, nbClasses: 2

method	kernel	sigma	similarity	accuracy (%)	nmi (%)
kernelKmeans	exponential	0.2		**58±5**	3
kernelKmeans	exponential	0.4		**68±12**	**21**
kernelKmeans	exponential	0.6		52±1	**0**
kernelKmeans	exponential	0.8		**62±18**	**15**
kernelKmeans	exponential	1.0		51±1	**0**
kernelKmeans	linear			51±0	**0**
kernelKmeans	polynomial			51±1	**0**
kMedoids			euclidean	52±1	**0**
kMedoids			seuclidean	51±1	**0**
kMedoids			cosine	51±1	**0**
kMeans				52±1	**0**
chance				52±2	**0**

Table 2: dataType: spherical, nbClasses: 2

method	kernel	sigma	similarity	accuracy (%)	nmi (%)
kernelKmeans	exponential	0.2		56±2	1
kernelKmeans	exponential	0.4		**59±5**	4
kernelKmeans	exponential	0.6		**65±0**	**21**
kernelKmeans	exponential	0.8		**63±5**	**13**
kernelKmeans	exponential	1.0		54±3	1
kernelKmeans	linear			50±0	0
kernelKmeans	polynomial			50±0	0
kMedoids			euclidean	51±0	0
kMedoids			seuclidean	50±0	0
kMedoids			cosine	50±0	0
kMeans				50±0	0
chance				52±2	0

Table 3: dataType: spiral, nbClasses: 2

method	time
kMedoids	3.90±4.85
kernelKmeans	0.30±0.28
kMeans	0.18±0.06
chance	0.07±0.03

Table 4: Execution time in seconds