Evaluation of unsupervised classification algorithms on the MNIST dataset

Asier Barrasa

Abstract

In this document I report my proposal for the application of the different clustering algorithms for handwritten image recognition. The algorithms that I have selected for the clustering tasks have been: K Means, Gaussian Mixture based model, and Birth. I have implemented the clustering process using the scikit-learn [1], keras [2] and numpy [3] library. After making the clusters, I have visualized it using matplotlib [4]. From our results the best accuracy has been produced by the Birch clustering algorithm.

1 Description of the problem

The database used in this project is MNIST [5], MNIST is a handwritten digits database. The database has 70.000 examples, 60.000 for training and 10.000 for testing, each example is composed of 28x28 pixels. In other words a 28x28 array, which can be flattened on a 784 dimensional array. Each point has a value between 0 and 256, this value tell us how dark is each pixel. The goal of the project is to cluster data and see the differences between the selected clustering algorithms.

The database has the following characteristics:

- 784 attributes, one per pixel. 28x28 matrix.
- 10 classes: 0,1,2,3,4,5,6,7,8,9.
- 1000 selected instances, of the 7000 that has the database .
- The data is balanced, I have picked 100 instances of each class.

2 Description of our approach

We organized the implementation of the project according to the tasks:

1. Preprocessing of the dataset.
2. Define the clustering algorithms, and group data on clusters.
3. Visualize the clusters.
4. Design the validation method to evaluate the accuracy of the proposed clusters.

2.1 Preprocessing

In order to select 1.000 examples to make faster calculations, I decided to get, at first a random sample of 1.000 elements, after watching the result, I decided to take a random sample, of each class, in other words, I take a random sample of 100 elements from the class 0, other random sample of 100 elements from the class 1 this way until the class 9. Thusly, I obtained the same examples of each class, which allowed me to have more accurate results.

After selecting which data will I use, I reshaped the data in order to obtain a 784 array from each example. This way I obtained an array with shape of (n_samples, n_features)

2.2 Classifiers

This, was probably one of the most difficult tasks of the projects, firstly I made a fast search on the internet and I find several types of algorithms, I looked for more information of those that I found most interesting for the project and finally I decided for these three. I principally selected this one, because you can tell to the algorithm the number of clusters to search. [1]

1. K Means with parameters:
   - n_clusters=10
   - init=k-means++/ random / setting the centers with PCA
   - n_init=10
   - max_iter=300,
   - tol=0.0001
   - precompute_distances=auto
   - verbose=0,
   - random_state=None
   - copy_x=True
   - n_jobs=None
   - algorithm=auto
2. Gaussian Mixture with parameters:
   - n_components=10
   - covariance_type=diag
   - tol=0.001
   - reg_covar=1e-06
   - max_iter=100
   - n_init=1
   - init_params=kmeans
   - weights_init=None
   - means_init=None
   - precisions_init=None
   - random_state=None
   - warm_start=False
   - verbose=0
   - verbose_interval=10
3. Birch with parameters
   - threshold=0.5
   - branching_factor=1000
   - n_clusters=10
   - compute_labels=True
   - copy=True

2.2.1 K Means

K Means is probably the most popular clustering algorithm. K-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean. We can init the clusters on a random position, on k means++, in wick sklearn selects initial cluster centers for k-mean clustering in a smart way to speed up convergence, also you can use PCA. I have tried with all of them. The one that gave the best results was by far the ones which initialize the centers with PCA algorithm.

[1] Most of the parameters are the default parameters from sklearn

2.2.2 Gaussian Mixture

Is a probabilistic model, with give us the probability of a data belonging to a class. It has as many Gaussian distributions as clusters, each one with his mean and standard deviation, and it mixes all of them, to assign the probability.

2.2.3 Birch

Birch in contrast to his brother k-means use itheratical clustering. The algorithm, assign each sample to its own cluster, then at each step, the two clusters that are the most similar are merged; the algorithm continues until all of the clusters have been merged. Also you can told to sklearn, how many clusters do you need.

3 Visualization

One of the biggest problems of this dataset has been to visualize the results, due to the large number of dimensions of the data.

In order to solve this problem, I have used PCA (Principal component analysis) to minimize the dimension of the data, with the smallest loss of information possible.

PCA [6] makes linear dimensionality reduction using Singular Value Decomposition of the data to project it to a lower dimensional space.

```
reduced_data = PCA(n_components=2).fit_transform(data)
```

We reduce the components to two, the two witch more information give us. I didn't make an analysis of how many information, gives each variable, so I have used the PCA library on sklearn with default values and 2 components.

After reducing the data we can plot every point, with its corresponding predicted class.To visualize the data, I have used *matplotlib* to plot the points on a 2D graph

4 Validation

1. inertia
2. homogeneity score
3. completeness score
4. V measure
5. adjusted Rand index
6. adjusted mutual information
7. silhouette coefficient
8. contingency matrix

4.1 Inertia

In case of K Means algorithm we can use inertia, Inertia can be recognized as a measure of how internally coherent clusters are.

The problem is that inertia is not a normalized metric: we just know that lower values are better and zero is optimal. But in very high-dimensional spaces, Euclidean distances tend to become inflated, we can use PCA to minimize this problem.

4.2 Homogeneity score, completeness score and V measure

Having truth class assignments of the samples, it is possible to define some intuitive metric using conditional entropy analysis.

There are two objectives, every cluster assignment want to fulfill:

- **homogeneity:** each cluster contains only members of a single class.
- **completeness:** all members of a given class are assigned to the same cluster.

The first one is, calculated, on sklearn with the function: `homogeneity_score()`

The second one with: `completeness_score`

The V-measure is the harmonic mean between homogeneity and completeness

$$v = 2 \times \frac{(homogeneity \times completeness)}{(homogeneity + completeness)}$$

4.3 Adjusted Rand index and Adjusted mutual information

Mutual information is a symmetric measure for the degree of dependency between the clustering and the manual classification

Rand index is defined as the number of pairs of objects that are either in the same group or in different groups in both partitions divided by the total number of pairs of objects. The Rand index lies between 0 and 1. When two partitions agree perfectly, the Rand index achieves the maximum value 1.

$$ARI = \frac{(RI - Expected\_RI)}{(max(RI) - Expected\_RI)}$$

Both are based on the notion of cluster purity.

4.4 Silhouette score

Compute the mean Silhouette Coefficient of all samples, in other words, the mean of the distances from a point on a given cluster from nearest cluster that the sample is not a part of. The values oscillate between -1 and 1, 0 means overlapping clusters.

4.5 Contingency matrix

This is a good way to see how many different elements of each class are on each cluster. [2]

On the columns are represented the clusters and on the rows, the classes

Labls/Clusters	0	1	2	3	4	5	6	7	8	9
0	1	0	2	0	0	34	0	60	3	0
1	0	24	0	76	0	0	0	0	0	0
2	59	15	7	4	3	2	4	0	2	4
3	1	3	10	9	2	3	71	0	0	1
4	0	13	0	0	52	1	0	0	4	30
5	0	22	11	10	8	13	28	0	0	8
6	14	2	1	9	0	1	0	0	73	0
7	0	11	1	3	19	0	0	0	0	66
8	1	11	63	6	4	1	11	0	0	3
9	1	2	1	0	43	0	0	1	0	52

Table 1: Contingency matrix produced by the Birch clustering algorithm

5 Implementation

All the project has been implemented in Python. We used *Keras* for loading the dataset, *scikit-learn* for the classification tasks, *matplotlib* for plotting the graphs, and *numpy* to process and normalize the data. I have illustrated how the implementation works in the Python notebook:

[2]The confusion matrix change on every execution, because each execution takes a random sample of each class.

ML_P57.ipynb.

6 Results

As we can see on the metrics, and the contingency table, the purity of the clusters are near the 50%. Also, we can see in the 2D plots and on the silhouette score value, the clusters are overlapped. We can also see in the 2D plots, how the points are spread on each cluster, and the area of each cluster.Birch is the algorithm that achieves the most homogeneous clusters, but k means, with the centroids initialized using PCA is the fastest.

7 Conclusions

In conclusion, we can assure, use clustering algorithms, without Neuronal Network is definitely not the best option if you need high accuracy in the results. But could not be a bad idea, to use K Means with PCA to seed the centers in a deterministic way. Hence we can run K Means with only one iteration and get very similar results, on a much younger time.

Finally, the most accurate algorithm turned out to be **Birch**.

References

[1] scikit-learn. http://scikit-learn.org/stable/modules/clustering.html.

[2] Keras. https://keras.io/datasets.

[3] NumPy. http://cs231n.github.io/python-numpy-tutorial.

[4] Matplotlib. https://matplotlib.org/api/_as_gen/matplotlib.pyplot.subplots.html.

[5] MNIST. http://yann.lecun.com/exdb/mnist.

[6] Christopher Olah. http://colah.github.io/posts/2014-10-Visualizing-MNIST, 9 October 2014.