: K-means

شرح خورجی اجرای k-means (فایل kmeans.py ضمیمه شده) بر روی دیتای MNST با K (تعداد کلاستر) های مختلف :

Number of Clusters: 2
Inertia: 2986892.6226943154
Purity: 0.06823960443954137
Rand Index: 0.045682856939595354
Accuracy: 0.20548333333333332

Number of Clusters: 5
Inertia: 2670930.0110975984
Purity: 0.28702154860050655
Rand Index: 0.221811722576066
Accuracy: 0.3945666666666667

Number of Clusters: 8
Inertia: 2441899.18719378
Purity: 0.41294841417872097
Rand Index: 0.31290768560790005
Accuracy: 0.48788333333333334

Number of Clusters: 10
Inertia: 2377369.0306059984
Purity: 0.48592276262737916
Rand Index: 0.38222969298288073
Accuracy: 0.5817333333333333

Number of Clusters: 15
Inertia: 2297861.0767535707
Purity: 0.5134246733774221
Rand Index: 0.31883942415164646
Accuracy: 0.6063166666666666

Number of Clusters: 20
Inertia: 2218226.7408089465
Purity: 0.5774451269297105
Rand Index: 0.3449066512623065
Accuracy: 0.6766333333333333

Number of Clusters: 36
Inertia: 1969159.784972142
Purity: 0.6737473174058259
Rand Index: 0.260705304067344
Accuracy: 0.74195

Number of Clusters: 64
Inertia: 1960817.4830404054
Purity: 0.6878766763218052
Rand Index: 0.21440666431808972
Accuracy: 0.7654833333333333

Number of Clusters: 200
Inertia: 1565086.193382992
Purity: 0.8204608067887437
Rand Index: 0.08060568919462752
Accuracy: 0.8764

Number of Clusters: 500
Inertia: 1393425.9467612035
Purity: 0.8737673408038801

Rand Index: 0.03668896859020881
Accuracy: 0.9175333333333333

Number of Clusters: 1000
Inertia: 1271909.7924474673
Purity: 0.8991890380638158
Rand Index: 0.02032901596322402
Accuracy: 0.9317333333333333

نتیجه:

با افزایش تعداد کلاستر ها تا مرزی بسته به داده ها دقت کلاسترینگ افزایش میابد اما زمان اجرای آن زیاد میشود، Purity افزایش پیدا میکند و Rand Index ابتدا افزایشی است و از مرحله ای به بعد با افزایش تعداد کلاستر ها کاهش میابد.

روش مورد نیاز برای اجرای k-means توسط MLP:

1 Background Knowledge for Clustering

In semi-unsupervised clustering, background knowledge refers to the available knowledge concerning either pair-wise (must-link or cannot-link) constrains between data items or class labels for some items. In current work, we will focus on using constrains between data items. Two types of pairwise constrains will be considered:

- *Must-link constrains* specify that two instances have to be in the same cluster.
- *Cannot-link constrains* specify that two instances must not be placed in the same cluster.

Must-link and Cannot-link are Boolean function. Assuming S is the given data set and P , Q are data instances, $P, Q \in S$. If P and Q belong to same class, $Must-link\ (P, Q) = True$. Otherwise, $Cannot-link\ (P, Q) = True$. Table 1 shows that pairwise constraints have two properties: symmetric and transitive.

Table 1. Properties of pairwise constraints

Symmetric: if $P, Q \in S$,
$Must-link\ (P, Q) \Leftrightarrow Must-link\ (Q, P)$
$Cannot-link\ (P, Q) \Leftrightarrow Cannot-link\ (Q, P)$
Transitive: if $P, Q, R \in S$,
$Must-link\ (P, Q)\ \&\&Must-link\ (Q, R) \Rightarrow Must-link\ (P, R)$
$Must-link(P, Q)\ \&\&Cannot-link(Q, R) \Rightarrow Cannot-link(P, R)$

2 MLP-KMEANS

2.1 K-means Clustering

K-means clustering [11] is a method commonly used to automatically partition a data set into k groups. It proceeds by selecting k initial cluster centers and then iteratively refining them as follows:

1. Each instance d_i is assigned to its closest cluster center.

2. Each cluster center C_j is updated to be the mean of its constituent instances.

The algorithm converges when there is no further change in assignment of instances to clusters. In this work, we initialize the clusters using instances chosen at random from the data set. The data sets we used are composed solely of numeric features. Euclidean distance is used as measure of similarity between two data instances.

Table 2. MLP-KMEANS

Algorithm: MLP-KMEANS

Input: data set D must-link constrains $C_{m-link} \subseteq D \times D$

cannot-link constrains $C_{no-link} \subseteq D \times D$

Output: Partitions of instances in D

Stage 1: K-means clustering

1. Let $C_1 ... C_k$ be the initial cluster centers.
2. For each point d_i in D, assign it to the closet cluster C_j.
3. For each cluster C_j, update its center by averaging all of the points d_j that have been assigned to it.
4. Iterate between (2) and (3) until convergence.
5. Return $\{C_1 ... C_k\}$.

Stage 2: Violate-Constraints Test

6. $\{C_1 ... C_k\}$ makes new constrains $C_{k-m-link}$ and $C_{k-no-link}$
7. For instances d_i and d_j, if they have consistent constrains in original and new constraints, their labels generated by K-means are thought reliable. D_r includes all the instances with reliable labels.

Stage 3: MLP Training

8. MLP is trained by error back propagation (EBP) algorithm. Only D_r and corresponding labels are used for training.

Stage 4: Clustering using MLP

9. D is inputted into MLP to cluster.

2.2 Combining MLP and K-means for Clustering

Table 1 contains the algorithm MLP-KMEANS. The algorithm takes in a data set (D), a set of must-link constraints (C_{m-link}), and a set of cannot-link constrains $(C_{no-link})$. It returns a partition of the instances in D that satisfied all specified constrains.

In MLP-KMEANS, clustering consists of four stages. In the first stage, D is partitioned by K-means. K clusters $C_1 ... C_k$ are generated. The second step is Violate-Constraints test. The key idea of clustering in MLP-KMEANS is that MLP is trained using the output of K-means algorithm. So if the output of K-means clustering is not correct, MLP cannot be trained well. In turn, MLP cannot achieve high clustering accuracy. This step is used to filter out those samples whose labels generated by K-means might not be correct by violate-constraints test. Violate-constraints is Boolean function. For any two data instances P, Q, if $VC(P, Q) = True$, then P, Q are though mis-clustered by K-means. In detail, new constraints are generated based on the output of K-means. We call them k-must-link constraints $(C_{k-m-link})$ and k-cannot-link constraints $(C_{k-no-link})$. For P, Q, $VC(P, Q) = True$ in the following situations:

1) $Must - link(P, Q) \ \& \& K - Cannot - link(P, Q) = True$
2) $Cannot - link(P, Q) \ \& \& K - Must - link(P, Q) = True$

After Violate-Constrains test, the instances with $VC(P, Q) = False$ are gathered into D_r .Stage 3 is MLP training using D_r and corresponding labels. After training, in stage 4, MLP can be used for clustering instead of K-means.

Source: http://uclab.khu.ac.kr/ar/~donghai/pdfs/c5.pdf