Pattern Recognition Assignment 3: K-Means

Yu Hao

Student ID: 1831607

December 6, 2018

1 Introduction

In this assignment, we are expected to implement a clustering algorithm. There are many kinds of methods for clustering, for example, K-Means, hierarchical clustering, DBSCAN, etc. I choose standard K-Means as the basic method and there will be some extensions to be mentioned later. As we confront a real problem from Huawei project which needs to partition dataset equally under the condition of minimizing within-cluster sum of squares (WCSS), we propose an extension of K-means to accomplish the objective.

Hence, the assignment is organized by two parts: the standard K-Means implementation and its extension. Furthermore, the second part will be filled and plump at our team's final project.

2 Algorithm Formulation

Given a set of data points $\mathbf{H} = \{\boldsymbol{h}_1, \dots, \boldsymbol{h}_N\}$, where each point is a d-dimensional real-valued vector. The aim of K-means algorithm is to divide N data points into K $(\leqslant N)$ sets $\mathbf{C} = \{C_1, \dots, C_K\}$ so as to minimize the within-cluster sum of squares (WCSS) function. Formally, the objective is to find

$$\begin{aligned} &\underset{\mathbf{C}}{\operatorname{argmin}} \sum\nolimits_{k=1}^{K} \sum\nolimits_{i \in C_k} \|\boldsymbol{h}_i - \boldsymbol{c}_k\|^2 \\ &= \sum\nolimits_i \|\boldsymbol{h}_i\|^2 - \sum\nolimits_{k=1}^{K} \frac{1}{n_k} \sum\nolimits_{i,j \in C_k} \boldsymbol{h}_i{}^T \boldsymbol{h}_j, \end{aligned} \tag{1}$$

where $\boldsymbol{c}_k$ represents the centroid of cluster C_k for $k = 1, 2, \dots, K$ and $\boldsymbol{n} = [n_1, \dots, n_K]^T \in \mathbb{R}^K$ records the number of data points for K clusters. The optimization process is known as a NP-hard problem [1], and the standard K-means algorithm [2] was proposed to provide an approximate solution. The algorithm uses an iterative refinement technique. Given an initial set of K centroids $\{\boldsymbol{c}_1^{(0)}, \boldsymbol{c}_2^{(0)}, \dots, \boldsymbol{c}_K^{(0)}\}$, the algorithm proceeds with alternating strategy between an assignment step and an update step as follows.

During the assignment step, each data point is assigned to the cluster characterized by nearest centroid, that is

$$\begin{aligned} C_k^{(t)} &= \left\{ \mathbf{h}_p : \left\| \mathbf{h}_p - \mathbf{c}_k^{(t)} \right\|^2 \leq \left\| \mathbf{h}_p - \mathbf{c}_j^{(t)} \right\|^2, 1 \leq j \leq K \right\}, \\ n_k^{(t)} &= |C_k^{(t)}|, \\ k &= 1, \ldots, K, \end{aligned} \tag{2}$$

where each data point is assigned to exactly one cluster, and this step can minimize the WCSS function.

During the update step, the new centroids can be computed with current clusters:

$$\mathbf{c}_k^{(t+1)} = \frac{1}{n_k^{(t)}} \sum_{\mathbf{h}_j \in C_k^{(t)}} \mathbf{h}_j, \ k = 1, \ldots, K. \tag{3}$$

Since the arithmetic mean is a least-squares estimator, this step also minimizes the WCSS function.

The algorithm converges when the centroids no longer change, in other words, the data points are no longer assigned to new cluster. The K-means algorithm can converge to a local optimum. Since for a given K, the result of above clustering algorithm depends solely on the initial centroids.

The computational complexity of the K-means algorithm is $O(NdKM)$, where N is the number of the d-dimensional vectors, K is the number of clusters and M is the number of iterations before reaching convergence.

3 K-Means Implementation

Three parts will be mentioned there. Similar with former assignments, I choose two datasets: The mobile prices classifications and glasses types classifications. Algorithm implementation consists of three parts, which are centroids initialization, update data points assignments, and recalculate centroids. After acquiring the result, we will compare clustering results with true labels to examine the clustering quality. Visualization is the last implementation to see the result clearly.

3.1 Data Preprocessing

We have two datasets and both of them are numerical, so I just need to normalize the features. The purpose of normalization is to avoid the influences of features' order of magnitudes, rather focus on their distributions and relative distances. Standardizing is very simple:

$$z = \frac{x - \mu}{\sigma}, \tag{4}$$

where x is the number to be standardized, μ is the feature's mean, and σ is the standard deviation.

3.2 Fit

This module is to do the clustering. Firstly we need to choose K centroids as the initial centroids. I choose the corresponding data points from the dataset as the centroids. There are some input checks. The algorithm cannot handle the circumstance that the number of clusters bigger than data points'. And obviously, the clusters number needs to be a positive integer.

Then we iterate the next two steps until the **centroids** don't change anymore:

- Update assignments. I calculate the Euclidean distances from all data points to all centroids and choose the closest centroid to be the representative of each point. After accomplishing the calculation, I record the result to an array: **clustering_results**.
- Recalculate centroids. For the data points belong to the same cluster, they need a new representative. For simplicity, the mean of these data points is a suitable choice. I denote the result as **centroids**.

3.3 Result Comparison

As the dataset has true labels, we can examine the quality of clustering using Adjusted Rand index. The Rand index in data clustering is a measure for calculating the similarity between two data clusterings. From a mathematical standpoint, the Rand index is related to the accuracy but is applicable even when class labels are not used.

3.3.1 Rand index

Given a set of N elements $\mathbf{S} = \{s_1, \ldots, s_N\}$ and two partitions of $\mathbf{S}$ to compare, $\mathbf{X} = \{X_1, \ldots, X_K\}$, a partition of $\mathbf{S}$ into K subsets, and $\mathbf{Y} = \{Y_1, \ldots, Y_Q\}$, a partition of $\mathbf{S}$ into Q subsets. define the following:

- a, the number of pairs of elements in $\mathbf{S}$ that are in the same subset in $\mathbf{X}$ and in the same subset in $\mathbf{Y}$;
- b, the number of pairs of elements in $\mathbf{S}$ that are in different subsets in $\mathbf{X}$ and in different subsets in $\mathbf{Y}$;
- c, the number of pairs of elements in $\mathbf{S}$ that are in the same subset in $\mathbf{X}$ and in different subsets in $\mathbf{Y}$;
- b, the number of pairs of elements in $\mathbf{S}$ that are in different subsets in $\mathbf{X}$ and in the same subset in $\mathbf{Y}$.

The Rand index, R, is:

$$R = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{N}{2}}. \tag{5}$$

Intuitively, $a+b$ can be considered as the number of agreements between $\mathbf{X}$ and $\mathbf{Y}$, and $c+d$ as the number of disagreement between them. Since the denominator is the total

	Y_1	Y_2	$\cdots$	Y_Q	Sums
X_1	n_{11}	n_{12}	$\cdots$	n_{1Q}	a_1
X_2	n_{21}	n_{22}	$\cdots$	n_{2Q}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
X_K	n_{K1}	n_{K2}	$\cdots$	n_{KQ}	a_K
Sums	b_1	b_2	$\cdots$	b_Q	

Table 1: contingency table

number of pairs, the Rand index represents the frequency of occurrence of agreement over the total pairs or the probability that **X** and **Y** will agree on a randomly chosen pair.

3.3.2 Adjusted Rand index

The adjusted Rand index is the corrected-for-chance version of the Rand index. Such a correction for chance establishes a baseline by using the expected similarity of all pair-wise comparisons between clusterings specified by a random model. Traditionally, the Rand Index was corrected using the Permutation Model for clusterings (the number and size of clusters within a clustering are fixed, and all random clusterings are generated by shuffling the elements between the fixed clusters). However, the premises of the permutation model are frequently violated; in many clustering scenarios, either the number of clusters or the size distribution of those clusters vary drastically. For example, consider that in K-Means the number of clusters is fixed by the practitioner, but the sizes of those clusters are inferred from the data. Variations of the adjusted Rand Index account for different models of random clusterings.

Though the Rand Index may only yield a value between 0 and +1, the adjusted Rand index can yield negative values if the index is less than the expected index.

Given a set S of N elements, and two groupings or partitions (e.g. clusterings) of these elements, namely $\mathbf{X} = \{X_1, X_2, \ldots, X_K\}$ and $\mathbf{Y} = \{Y_1, Y_2, \ldots, Y_Q\}$, the overlap between X and Y can be summarized in a contingency table $[n_{ij}]$ where each entry n_{ij} denotes the number of objects in common between X_i and Y_j: $n_{ij} = |X_i \cap Y_j|$.

Dataset	Number of dataset	Number of clusters	ARI
mobile prices classes	2000	4	0.466
glasses types	142	7	0.598

Table 2: ARI of two datasets

The original Adjusted Rand Index using the Permutation Model is

$$\begin{aligned} ARI &= \frac{Index - ExpectedIndex}{MaxIndex - ExpectedIndex} \\ Index &= \sum_{ij} \binom{n_{ij}}{2} \\ ExpectedIndex &= \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{N}{2}} \\ MaxIndex &= \frac{\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}}{2} \end{aligned} \tag{6}$$

Hence, the *ARI* can be an effective index to examine the similarity between the clustering result and true classification. For the two datasets I mentioned, table 2 shows simulation result.

3.4 Visualization

Here I draw the true classifications and clustering (predicting) results in pictures using different colors to classify corresponding clusters.

Compare figure 1 and 2, we can see that the true result is a parallel-shaped figure with overlaps whereas K-Means cannot partition like that. In fact, I think it's a challenge for most clustering methods. For the sparser dataset, glasses, it has some obvious centers in distribution. Intuitively, it is easier to be partitioned, and the *ARI* proves the assumption.

4 Improvement of K-Means

K-Means has been proposed for many years and it has a lot of perfection versions now. We just list some representative works here.

4.1 Advantages and Disadvantages of K-Means

4.1.1 Advantages

- Easy to implement.
- With a large number of variables, K-Means may be computationally faster than hierarchical clustering (if K is small).

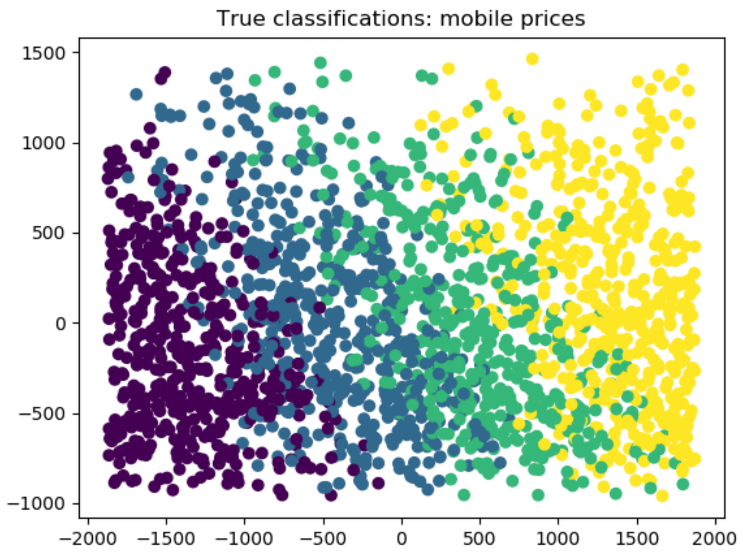

Figure 1: Mobile prices classes: True result

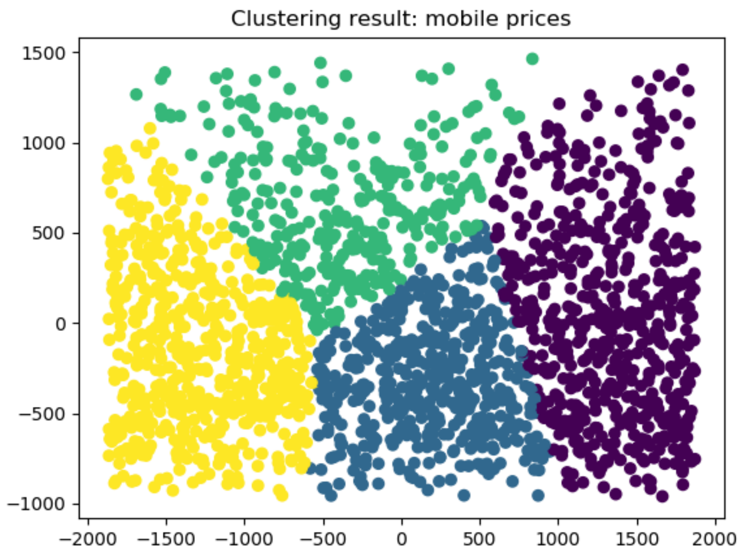

Figure 2: Mobile prices classes: Predicting result

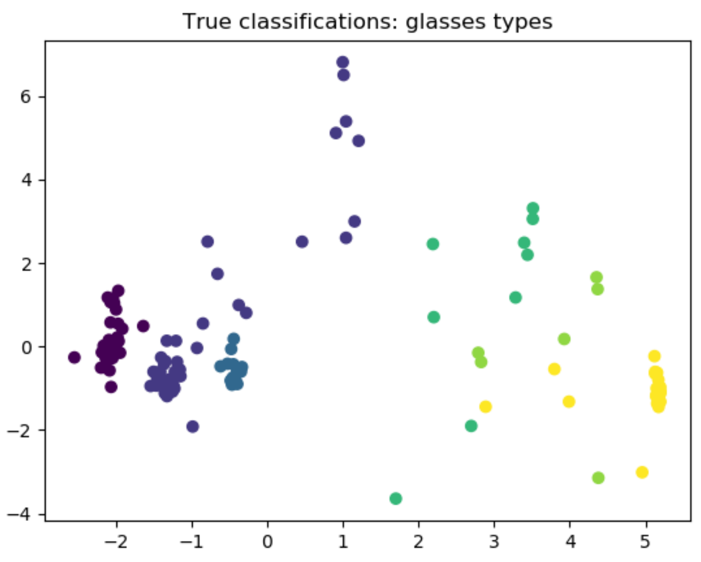

Figure 3: Glasses types: True result

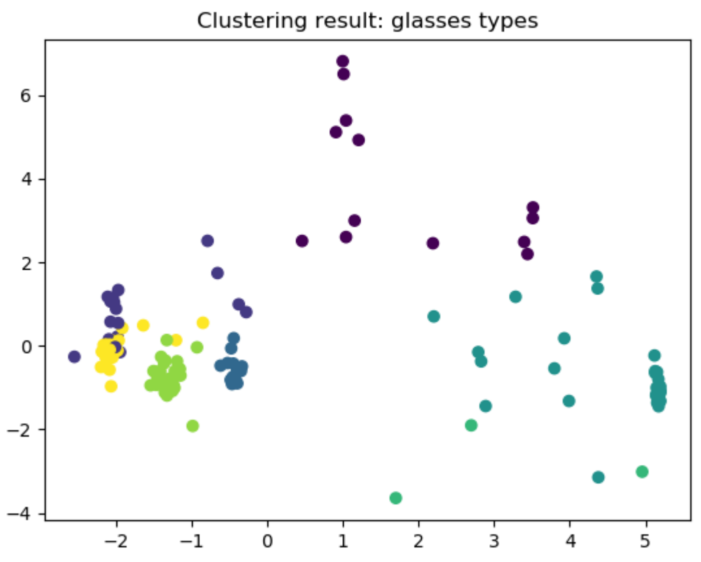

Figure 4: Glasses types: Predicting result

- k-Means may produce tighter clusters than hierarchical clustering.
- An instance can change cluster (move to another cluster) when the centroids are recomputed.

4.1.2 Disadvantages

- Difficult to predict the number of clusters (K-Value).
- Initial seeds have a strong impact on the final results.
- The order of the data has an impact on the final results.
- Sensitive to scale: rescaling your datasets (normalization or standardization) will completely change results. While this itself is not bad, not realizing that you have to spend extra attention (on to scaling your data might be bad).

4.2 Variations of the K-Means Algorithm

The K-Means algorithm can be classified into center-based clustering algorithms. These clustering algorithms were developed to improve the performance of the standard K-Means algorithm[3]. We will address some of these algorithms in subsequent sections.

4.2.1 The Continuous K-Means Algorithm

The continuous K-Means algorithm, proposed by Faber, is faster than the standard K-Means algorithm. It is different from the standard K-Means algorithm in the following aspects. Firstly, in the continuous K-Means algorithm, the prototypes (or reference points) are chosen as a random sample from the whole database, while in the standard K-Means algorithm the initial points are chosen arbitrarily. Secondly, the data points are treated differently. During each complete iteration, the continuous K-Means algorithm examines only a sample of the data points, while the standard K-Means algorithm examines all the data points in sequence.

4.2.2 The Compare-means Algorithm

In order to accelerate the K-Means algorithm, the algorithm compare-means uses a simple approach to avoid many unnecessary comparisons. Since the number of clusters K is usually small, distances of all pairs of means are precomputed before each iteration. Then, before comparing a point x to a mean μ_j , the above test is performed using the closest known mean to x.

4.2.3 The Sort-means Algorithm

The algorithm sort-means is an extension of compare-means. In this algorithm, the means are sorted in order of increasing distance from each mean in order to obtain a further speedup.

4.2.4 Acceleration of the K-Means Algorithm with the kd-tree

Pelleg and Moore proposed an algorithm for the K-Means clustering problem using the *kd*-tree data structure. The *kd*-tree data structure, can be used to reduce the large number of nearest-neighbor queries issued by the traditional K-Means algorithm. Hence, an analysis of the geometry of the current cluster centers can lead to a great reduction in the work needed to update the cluster centers. In addition, the initial centers of the K-Means algorithm can be chosen by the *kd*-tree efficiently. One way to use the *kd*-tree in the inner loop of the K-Means algorithm is to store the centers in the tree; another way is to store the whole data set in the tree. To describe the application of the *kd*-tree in the K-Means algorithm, let us start with an iteration of the K-Means algorithm. Let $C^{(i)}$ denote the set of centroids after the $i - th$ iteration. Before the first iteration, $C^{(0)}$ is initialized to a set of random values. The stop criterion of the algorithm is that $C^{(i)}$ and $C^{(i-1)}$ are identical. In each iteration of the algorithm, the following two steps are performed:

1. For each data point x, find the center in $C^{(i)}$ that is closest to x and associate x with this center.
2. Update $C^{(i)}$ to $C^{(i+1)}$ by taking, for each center, the center of mass of all the data points associated with this center.

4.3 Our project's requirement

The variations for K-Means above are mostly performing in the speed or the quality. We consider that K-Means cannot obtain the results with some constraints, like sizes. However, we need to partition our project's large-scale dataset into some uniform clusters so that we could solve the subproblems with similar scales concurrently. We cannot partition our original dataset directly as the elements in the set are relational. Hence, we have to adjust current methods to meet our requirement. K-Means has good expandability, which can be our foundation.

5 Looking Forward: Clustering with Size Constraints

This section will be our team's final project and the outline is disclosed in this assignment. Look forward to the main body at the final report and presentation.

- Matrix form of K-Means
- Constraints description
- Problem derivation
- Problem optimization
- Parallel solving
- Simulation and comparison
- Perfection under large-scale data

References

[1] M. Garey, D. Johnson, and H. Witsenhausen, “The complexity of the generalized lloyd-max problem (corresp.),” *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 255–256, 1982.

[2] J. A. Hartigan, “Clustering algorithms,” 1975.

[3] G. Gan, C. Ma, and J. Wu, *Data clustering: theory, algorithms, and applications.* Siam, 2007, vol. 20.