Unsupervised Learning

Author: Wenxiao Yang

Institute: Department of Mathematics, University of Illinois at Urbana-Champaign

All models are wrong, but some are useful.

Contents

Chapter 1 Clustering

General Goal of **Clustering Algorithm**:

- the "similarity" of the objects in the same cluster is maximized while
- the "similarity" of objects in different clusters is minimized.

Definition 1.1

*For a given set of objects $V = \{x_1, x_2, ..., x_m\}$, we call a **cluster** $\mathbf{S_k}$ a subset of these objects, and we call a **clustering** the set of all K clusters $\{\mathbf{S_1}, \mathbf{S_2}, ..., \mathbf{S_k}\}$.* ♣

Example 1.1 Clustering of $\{x_1, x_2, x_3, x_4\}$: (1). $\{\{x_1, x_3\}, \{x_2, x_4\}\}$; (2). $\{\{x_1, x_3\}, \{x_1, x_2, x_4\}\}$; (3). $\{\{x_3\}, \{x_2, x_4\}\}$.

1.1 K-Means

1.1.1 K-Means Clustering Optimization Problem

1. **Input:**

   Desired number of clusters (ex: $K = 3$)

   Dataset of m objects $X = \{\vec{x}_1, \vec{x}_2, ..., \vec{x}_m\}$, where each object $\vec{x}_i = (x_{i1}, x_{i2}, ...\vec{x}_{in})$ has n numerical attributes. (We can also think of X as being an $m \times n$ matrix $X_{m \times n}$.)

2. **Goal of K-Means:**

   Out of all possible clusterings of $\{S_1, S_2, ..., S_K\}$ with K clusters that can be made from the m objects in X, find the optimal clustering $\{S_1^*, S_2^*, ..., S_K^*\}$ that minimizes the sum of the "distance" of each object and the centroid (the mean of the cluster that object is assigned to).

   Technically, we can write this as an optimization problem

$$\{S_1^*, S_2^*, ..., S_K^*\} = \underset{S_1, S_2, ..., S_K}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{x \in S_k} \|x - \mu_k\|^2$$

$$\text{Optimal Inertia} = \min_{S_1, S_2, ..., S_K} \sum_{k=1}^{K} \sum_{x \in S_k} \|x - \mu_k\|^2$$

   Inertia measures how well a dataset was clustered by K-Means. It is calculated by measuring the distance between each data point and its centroid, squaring this distance, and summing these squares across one cluster. A good model is one with low inertia and a low number of clusters (K).

Find the clustering $\{S_1^*, S_2^*, ..., S_K^*\}$ that provides a global minimum is **NP-hard**.

We use a heuristic algorithm to find a local minimum is good enough.

1.1.2 Lloyd's Algorithm

1. **Input:**

   Desired number of clusters (ex: $K = 3$)

   Dataset of m objects $X = \{\vec{x}_1, \vec{x}_2, ..., \vec{x}_m\}$, where each object $\vec{x}_i = (x_{i1}, x_{i2}, ...x_{in})$ has n numerical attributes. (We can also think of X as being an $m \times n$ matrix $X_{m \times n}$.)

2. **Algorithm:**

   - **Step 1: Centroid Initialization Step**

     Randomly select K centroids $\{\vec{\mu}_1, \vec{\mu}_2, ..., \vec{\mu}_K\}$, where $\vec{\mu}_k = (\mu_{k1}, \mu_{k2}, ..., \mu_{kn})$

   - **Step 2: Cluster Assignment Step**

     Assign each object x_i in the dataset to it's closest centroid (specifically the *smallest squared euclidean distance*)

   - **Step 3: Centroid Update Step**

     Find the mean of each cluster created in step 2. These means are now the new centroids.

   - **Step 4: Stopping Criterion**

     If the old centroids and the new centroids are the same, stop the algorithm. Otherwise, go back to step 2.

3. **Output:** Clustering with K clusters $\{V_1, V_2, ..., V_K\}$.

Lloyd's algorithm is known as a **non-deterministic** algorithm because, even with the same input, it can exhibit different behaviors on different runs.

1.1.3 Benefits and Drawbacks

Benefits

- Fast algorithm.
- Computationally efficient.
- It scales well as the number of objects or attributes grows really large. (However, k-means is not great for "big data".)
- One of the easiest to understand.

Drawbacks

- Only works well with some types of data.

  The K-means algorithm works best for data when "the underlying clustering" of the data has the following properties:

  (1). Each cluster has roughly the same number of objects;

  (2). The clusters are spherical;

  (3). The clusters have the same sparsity;

  (4). There is good separation between the clusters;

  (5). You know the right number of clusters to ask for;

  (6). Attributes are numerical (non-categorical);

  (7). Data does not have a lot of noise or outliers.

  (**Caveat:** Just because some of these assumptions are not met does not mean necessarily the algorithm will perform worse.)

- Need to know the "right" number of clusters to ask for in advance. (We use k-means elbow plot method)
- It is a non-deterministic algorithm.

1.1.4 Elbow Method

Elbow Plot

1. For $k = 1$ to K:

   [a] Cluster the data several times into k clusters.

   [b] Calculate the average inertia of these resulting clusterings.

2. Plot "k vs. average inertia".

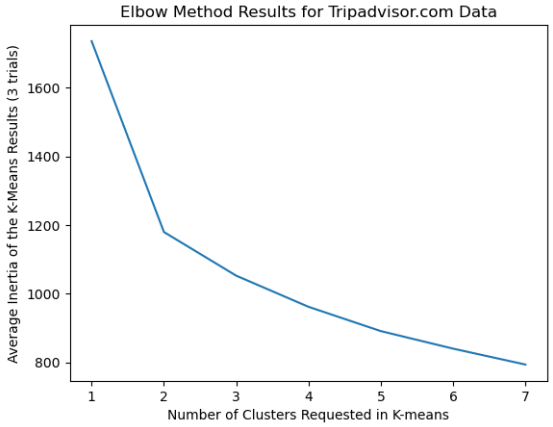

Figure 1.1: Elbow Plot Example

Interpretation of Elbow Plot

1. If there is not a dramatic elbow, then this suggests that either:
    1. The dataset is not clusterable or
    2. K-means is not a suitable algorithm for detecting the underlying clusters.
2. If there is a dramatic elbow, then this suggests that:
    1. There is a clustering structure and
    2. The k-means clustering algorithm is suggesting that there are about K clusters where the plot levels off.

In the example of the figure

1. We see a somewhat dramatic elbow in the plot. This suggests that there is some clustering structure in the dataset and that k-means is capable of identifying some clustering structure.

2. We see that that plot levels off dramatically at k=2 clusters. So this suggests that asking the k-means algorithm to return k=2 clusters will be the most insightful.

1.2 Types of Clusters Definitions

As we know the K-means algorithm can only work well with data that fulfills specific properties, we define some common **types of clusters** that could be considered in a numerical dataset to help introduce our new algorithms.

Definition 1.2 (Well-Separated Cluster)

*A **well-separated cluster** defines a cluster only when the data contains natural clusters that are far apart from each other. (This definition is vague in how far apart do clusters have to be.)*

♣

Why K-means may not work well?: Well-Separated Cluster can be non-spherical.

Definition 1.3 (Density-Based Cluster)

*A **density-based cluster** defines a cluster as a dense region of objects that is surrounded by a region of lower density. (This definition is vague in how dense it needs to be considered a cluster.)*

♣

Why K-means may not work well?: Density-Based Cluster can have noise.

Definition 1.4 (Graph-Based Cluster)

***Graph-based cluster** is a group of objects that are connected to one another, but have no connection to objects outside the group. (This definition is vague in how do we decide objects are connected.)*

♣

Why K-means may not work well?: Graph-based cluster can be non-spherical and not well separated.

Definition 1.5 (Contiguity-Based Cluster (a type of graph-based cluster definition))

*In **contiguity-based cluster** (a type of graph-based cluster definition), two objects are connected only if they are within a specified distance of one another.*

♣

Types of contiguity-based clustering algorithms: spectral clustering.

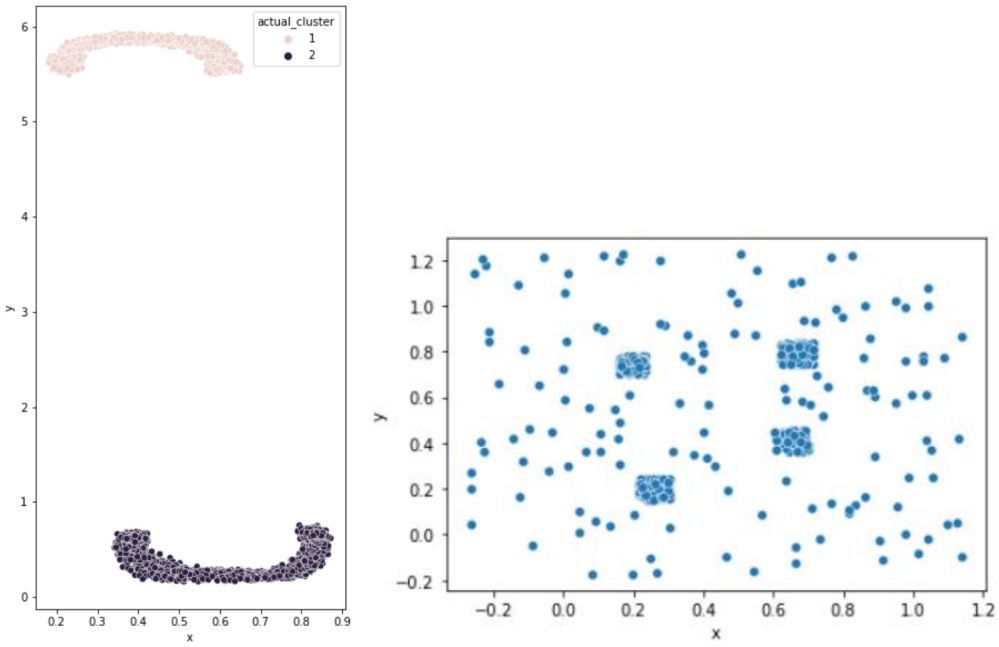

Figure 1.2: (1). Well-Separated Cluster; (2). Density-Based Cluster

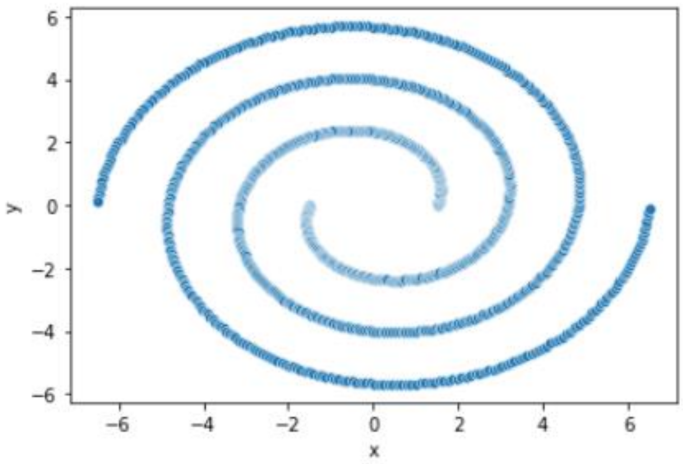

Figure 1.3: Contiguity-Based Cluster

Definition 1.6 (Prototype-Based Cluster)

*A **prototype-based cluster** defines a cluster as a set of objects in which each object is closer (or more similar) to the prototype (e.g. mean, median) that defines the cluster than to the prototype of any other cluster.*

♣

Why K-means may not work well?: Prototype-Based Cluster may be not well-separated and have outliers.

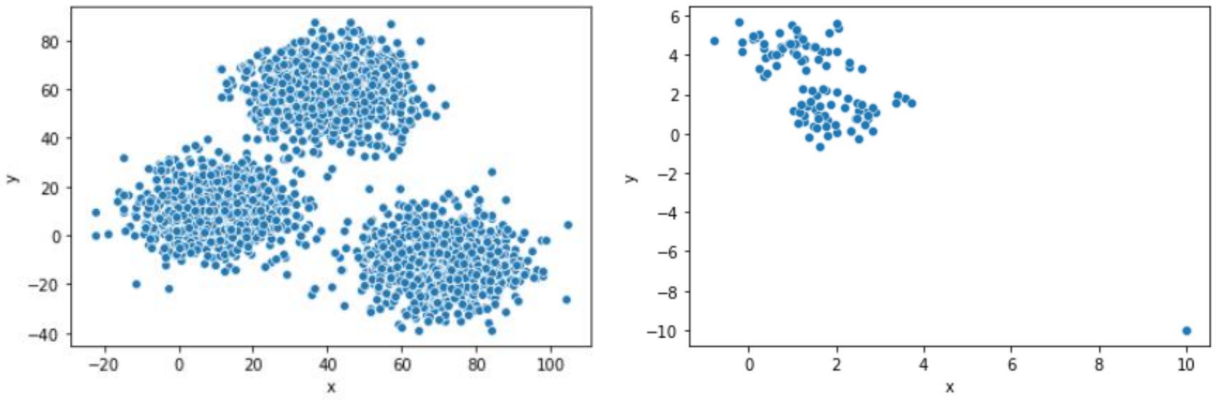

Figure 1.4: Prototype-Based Cluster

Types of prototype-based clustering algorithms:

- **K-means**: Prototype is the mean of the cluster.
- **K-median**: Prototype is the median of the cluster.

1.3 K-Medians

1.3.1 K-Medians Clustering Optimization Problem

Goal of K-Means:

Out of all possible clustering of $\{S_1, S_2, ..., S_K\}$ with K clusters that can be made from the m objects in X, find the optimal clustering $\{S_1^*, S_2^*, ..., S_K^*\}$ that minimizes the sum of the Manhattan distances (i.e., L_1 distances) of each object and the centroid (the median of the cluster that object is assigned to).

Technically, we can write this as an optimization problem

$$\{S_1^*, S_2^*, ..., S_K^*\} = \underset{S_1, S_2, ..., S_K}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{x \in S_k} \|x - c_k\|_1$$

$$\text{Optimal Inertia} = \min_{S_1, S_2, ..., S_K} \sum_{k=1}^{K} \sum_{x \in S_k} \|x - c_k\|_1$$

1.3.2 K-Medians Heuristic Algorithm

1. **Input:**

   Desired number of clusters (ex: $K = 3$)

   Dataset of m objects $X = \{\vec{x}_1, \vec{x}_2, ..., \vec{x}_m\}$, where each object $\vec{x}_i = (x_{i1}, x_{i2}, ...x_{in})$ has n numerical attributes. (We can also think of X as being an $m \times n$ matrix $X_{m \times n}$.)

2. **Algorithm:**

   - **Step 1: Centroid Initialization Step**

     Randomly select K centroids $\{\vec{c}_1, \vec{c}_2, ..., \vec{c}_K\}$, where $\vec{c}_k = (c_{k1}, c_{k2}, ..., c_{kn})$

   - **Step 2: Cluster Assignment Step**

     Assign each object x_i in the dataset to its closest centroid (specifically the *smallest Manhattan distance*)

   - **Step 3: Centroid Update Step**

     Find the median of each cluster created in step 2. These medians are now the new centroids.

   - **Step 4: Stopping Criterion**

     If the old centroids and the new centroids are the same, stop the algorithm. Otherwise, go back to step 2.

3. **Output:** Clustering with K clusters $\{V_1, V_2, ..., V_K\}$.

1.4 K-Medoids

Definition 1.7 (Medoid)

*In the context of clustering, we define a **medoid** as an actual object in a cluster whose sum of distance to all the objects in the cluster is minimal.*

♣

1.4.1 K-Medoids Clustering Optimization Problem

Goal of K-Medoids:

Out of all possible clustering of $\{S_1, S_2, ..., S_K\}$ with K clusters that can be made from the m objects in X, find the optimal clustering $\{S_1^*, S_2^*, ..., S_K^*\}$ that minimizes the sum of the distances (any distance metric) of each object and the centroid (the medoid of the cluster that object is assigned to).

Technically, we can write this as an optimization problem

$$\{S_1^*, S_2^*, ..., S_K^*\} = \underset{S_1,S_2,...,S_K}{\text{argmin}} \sum_{k=1}^{K} \sum_{x \in S_k} \text{dist}(x, c_k)$$

$$\text{Optimal Inertia} = \min_{S_1,S_2,...,S_K} \sum_{k=1}^{K} \sum_{x \in S_k} \text{dist}(x, c_k)$$

1.4.2 K-Medoids Clustering Algorithm

1. **Input:**

   Desired number of clusters (ex: $K = 3$)

   Dataset of m objects $X = \{\vec{x}_1, \vec{x}_2, ..., \vec{x}_m\}$, where each object $\vec{x}_i = (x_{i1}, x_{i2}, ...x_{in})$ has n numerical attributes. (We can also think of X as being an $m \times n$ matrix $X_{m \times n}$.)

2. **Algorithm:**

   - **Step 1: Centroid Initialization Step**

     Randomly select K centroids $\{\vec{c}_1, \vec{c}_2, ..., \vec{c}_K\}$, where $\vec{c}_k = (c_{k1}, c_{k2}, ..., c_{kn})$

   - **Step 2: Cluster Assignment Step**

     Assign each object x_i in the dataset to its closest centroid (specifically the *using distance metric you've chosen*)

   - **Step 3: Centroid Update Step**

     Find the medoid of each cluster created in step 2. These medians are now the new centroids.

   - **Step 4: Stopping Criterion**

     If the old centroids and the new centroids are the same, stop the algorithm. Otherwise, go back to step 2.

3. **Output:** Clustering with K clusters $\{V_1, V_2, ..., V_K\}$.

1.4.3 K-Medoids vs. K-Means

Benefit of K-Medoids over K-Means:

1. The medoid is more robust to outliers.
2. Guaranteed to converge using any distance metric we want (K-means has to use squared euclidean distance).

Benefit of K-Means over K-Medoids:

K-Medoids is more computationally complex than k-means:

1. K-means: O (number of objects $\times$ number of attributes $\times$ number of clusters $\times$ number of iterations)
2. K-medoids: O $\left((\text{number of objects})^2 \times \text{number of attributes} \times \text{number of clusters} \times \text{number of iterations}\right)$

1.5 Types of Clustering Algorithms Results

1.5.1 Partitional vs. Hierarchical Clustering Results

Definition 1.8 (Partitional Clustering)

*We call a **partitional clustering** a division of the set of data objects into k subsets (clusters) such that each object is in exactly one subset.*

♣

Example 1.2 $\{1, 2, 8\}, \{3, 7\}, \{4, 5, 6\}$

Definition 1.9 (Hierarchical Clustering)

*In a **hierarchical clustering** we allow for clusters to have nested subclusters.*

*A hierarchical clustering is displayed as a set of nested clusters displayed as a **dendrogram** tree. The dendrogram can reflect which objects and clusters are closer to each other than others.*

♣

Example 1.3 $\{\{1, 3\}, \{\{\{6, 9\}, 10\}, \{11, 15\}\}\}, \{4, \{12, 19\}\}, \{\{\{2, 14\}, \{\{17, 20\}, 18\}\}, \{5, 8\}\}, \{7, \{13, 16\}\}$.

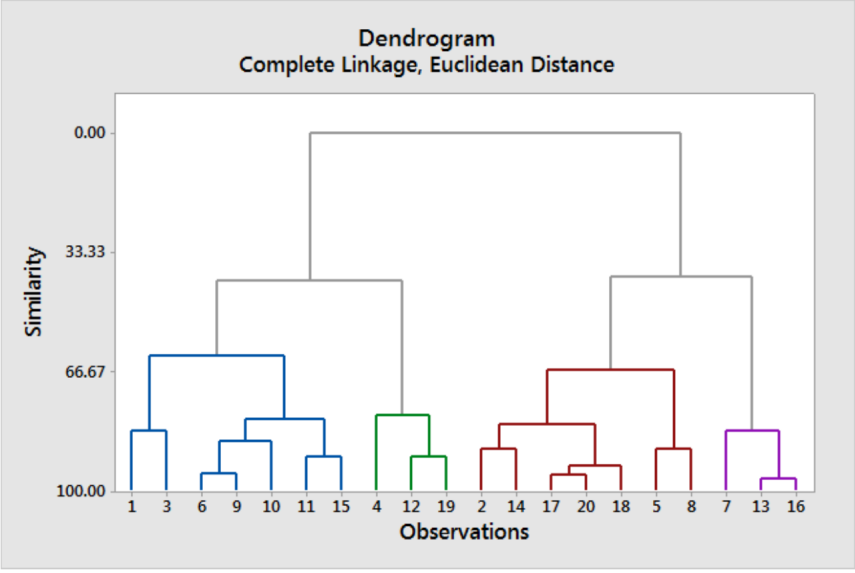

Figure 1.5: Hierarchical Clustering Example

1.5.2 Exclusive vs. Overlapping vs. Fuzzy Clustering Results

Definition 1.10

Exclusive Clustering *will assign an object to an exactly one cluster.*

♣

Example 1.4 $\{1, 3, 5\}, \{2, 4\}$.

Definition 1.11

Overlapping Clustering *can allow for an object to be assigned to more than one cluster.*

♣

Example 1.5 $\{1, 3, 5\}, \{2, 4, 5\}$

Definition 1.12

In a ***Fuzzy Clustering*** *every object belongs to every cluster with a membership weight that is between* 0 *(absolutely doesn't belong to the cluster) to* 1 *(absolutely belongs).*

1. *Usually the sum of each object's weights must sum to* 1.
2. w_{ij} = *the probability that object* i *belongs to cluster* j.

♣