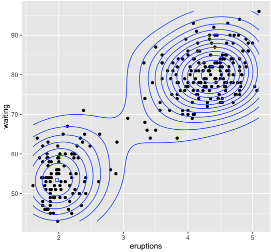
Unsupervised Methods and k-means Clustering

Dec, 2017

Introduction

- Up until now we've looked at supervised methods for classification and regression
- **Unsupervised methods** are a class of statistical and machine learning algorithms that do not require labels
- Unsupervised methods infer patterns and relationships from the data itself
- Cluster analysis or clustering groups is an exploratory technique on a set of items X such that items in the same group (a cluster) are more *similar* to each other than to those in other groups
- *Chicken or Egg problem* for number of clusters



A survey of clustering algorithms

- **Centroid based methods**
  - k-means
  - k-means++
  - k-medoids
- **Hierarchical methods**
  - Agglomerative clustering
  - Divisive clustering
- **Density based methods**
  - DBSCAN
- **Mixture models**
  - Gaussian mixture model (GMM)
- **Other methods**
  - Kohonen map/SOM
  - HDBSCAN

Hard Assignment

- **Objective: partition data set into k clusters, where k is given.**
- Suppose we have $\{x_1 ... x_n\}$ consisting of N observations of a D-dimensional variable x.
- Let $r_{nk} \in \{0, 1\}$ be an indicator variable, where $k = 1...K$ denotes which cluster to assign x_n, so $r_{nk} = 1$ if x_n is assigned to cluster k.
- Determine the sum of squares of the distances of each data point to its assigned vector μ_k , where μ_k is the centroid (or mean) of the kth cluster.
- How do we find values for r_{nk} and μ_k that minimize J?

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||x_n - \mu_k||^2$$

k-means algorithm

K-means algorithm

K-MEANS $(\mathbf{D}, k, \epsilon)$:

$t = 0$

Randomly initialize k centroids: $\boldsymbol{\mu}_1^t, \boldsymbol{\mu}_2^t, \ldots, \boldsymbol{\mu}_k^t \in \mathbb{R}^d$

repeat

  $t \leftarrow t + 1$

  $C_j \leftarrow \emptyset$ for all $j = 1, \cdots, k$

  `// Cluster Assignment Step`

  foreach $\mathbf{x}_j \in \mathbf{D}$ **do**

    $j^* \leftarrow \arg\min_i \left\{ \left\| \mathbf{x}_j - \boldsymbol{\mu}_i^t \right\|^2 \right\}$ `// Assign` $\mathbf{x}_j$ `to closest centroid`

    $C_{j^*} \leftarrow C_{j^*} \cup \{\mathbf{x}_j\}$

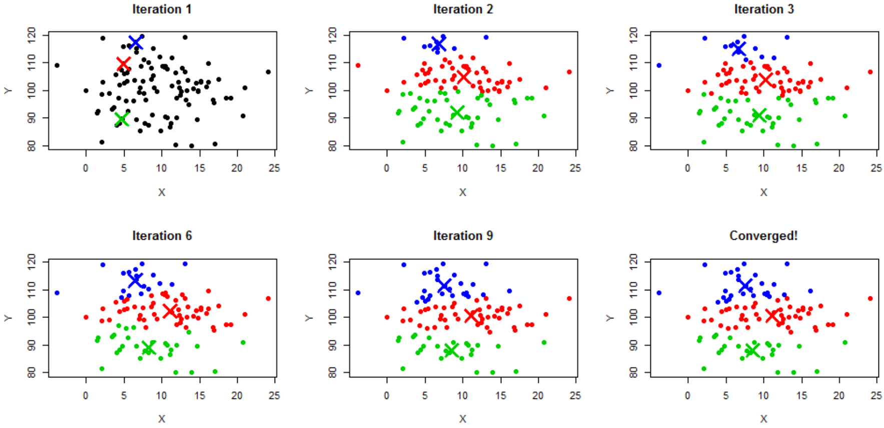
  `// Centroid Update Step`

  foreach $i = 1$ *to* k **do**

    $\boldsymbol{\mu}_i^t \leftarrow \frac{1}{|C_i|} \sum_{\mathbf{x}_j \in C_i} \mathbf{x}_j$

until $\sum_{i=1}^k \left\| \boldsymbol{\mu}_i^t - \boldsymbol{\mu}_i^{t-1} \right\|^2 \leq \epsilon$

k-means algorithm



k-means++

- k-means is highly sensitive to its cluster initializations. A much better way to do this is using an algorithm called k-means++.
- **K-means++**:
  1. Choose one center uniformly at random from among the data points.
  2. For each data point x, compute $D(x)$, the distance between x and the nearest center that has already been chosen.
  3. Choose one new data point at random as a new center, using a weighted probability distribution where a point x is chosen with probability proportional to $D(x)^2$.
  4. Repeat Steps 2 and 3 until k centers have been chosen.
  5. Now that the initial centers have been chosen, proceed using standard k-means clustering.

k-medoids

- k-medoids is less sensitive to outliers than k-means
- A **medoid** can be defined as the object of a cluster whose average dissimilarity to all the objects in the cluster is minimal: it is a most centrally located point in the cluster.
- **Partitioning Around Medoids (PAM) algorithm**
- PAM uses a greedy search which may not find the optimum solution, but it is faster than exhaustive search.

```
1. Initialize: randomly select k of the n data points as the medoids
2. Associate each data point to the closest medoid.
3. While the cost of the configuration decreases:
        1. For each medoid m, for each non-medoid data point o:
                1. Swap medoid m and o, recompute the cost (sum of distances of points to their
                    ↪ medoid)
                2. If the total cost of the configuration increased in the previous step, undo
                    ↪ the swap
```

k-means with different distance metrics

- k-means with L_1 distance is known as k-medians (not to be confused with k-medoids).
- k-means minimizes within-cluster variance, which equals squared Euclidean distances.
- k-medians minimizes absolute deviations, which equals Manhattan distance: more resiliant against outliers.
- Chebyshev and Minkowski distance can also be used.
- See https://www.ijircce.com/upload/2014/january/7_A%20comparative.pdf for more details.