1 Notations

Unless specified otherwise inside the text,

- The vector notation $x_{i:} = [x_{i1}, x_{i2}, \ldots, x_{ip}]$ specifies the i-th element of the matrix containing the data set $\mathcal{X}$ consisting of n observations and p dimensions (or features or data attributes). j is an index on the p dimensions.

- Given K number of groups (clusters) $\mathcal{C} = \{c_1, c_2, \ldots, c_K\}$, with $n_1, n_2, \ldots, n_K$ number of elements in each group respectively (n without an index will refer to the number of elements of the whole data set), the vector notation $m_{k:} = [m_{k1}, m_{k2}, \ldots, m_{kp}]$ specifies the k-th group center (centroid). This group center is the mean of the data in the group

- The notation $\mu_{1:}$ refers to the global center of the data set, which is unique, and $\mu_{1:} = [\mu_{11}, \mu_{12}, \ldots, \mu_{1p}]$.

- Given K class labels ℓ then n_ℓ are the number of elements belonging to each class and $n_\ell^{(k)}$ is the number of elements of class ℓ belonging to cluster k.

- The letters w and a are reserved to specify the weights of each dimension, i.e. $w_1, w_2, \ldots, w_p$ and $a_1, a_2, \ldots, a_p$.

- The stylized letter $\Bbbk$ is used to indicate the $\Bbbk$-fold cross validation.

- The notation,

$$\sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} x_{i:} = \sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} \sum_{j=1}^{p} x_{ij}$$

  specifies a summation of all the p-dimensional data points $x_{i:}$, $i = 1 \ldots n_k$ which belong to the k-th group ($x_{i:} \in c_k$).

-
  - WCSS = Within Clusters Sum of Squares.
  - BCSS = Between Clusters Sum of Squares.

2 K-Means

Objective function: Minimize WCSS [Jain, 2010].

$$\mathcal{J}_{kmeans} = \sum_{k=1}^{K} \sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} \sum_{j=1}^{p} (x_{ij} - m_{kj})^2 \tag{1}$$

Objective function: Maximize BCSS [Witten and Tibshirani, 2010].

$$\mathcal{J}_{BCSS} = \sum_{i=1}^{n} \sum_{j=1}^{p} (x_{ij} - \mu_{1j})^2 - \sum_{k=1}^{K} \sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} \sum_{j=1}^{p} (x_{ij} - m_{kj})^2 \tag{2}$$

Equivalent formulas [Witten and Tibshirani, 2010].

$$WCSS = \sum_{k=1}^{K} \sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} \sum_{j=1}^{p} (x_{ij} - m_{kj})^2 = \sum_{k=1}^{K} \frac{1}{2n_k} \sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} \sum_{\substack{i'=1 \\ (x_{i':} \in c_k)}}^{n_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \tag{3}$$

Minimization of Eq. (1) leads to cluster centroids.

$$\frac{\partial \mathcal{J}_{kmeans}}{\partial m_{k'j'}} = 0 \Rightarrow \frac{\partial}{\partial m_{k'j'}} \sum_{k=1}^{K} \sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} \sum_{j=1}^{p} (x_{ij} - m_{kj})^2 = 0 \Rightarrow \quad m_{k'j'} = \frac{1}{n_{k'}} \sum_{\substack{i=1 \\ (x_{i:} \in c_{k'})}}^{n_{k'}} x_{ij'} \tag{4}$$

K-Means algorithms.

- Lloyd's K-Means [Jain, 2010; Lloyd, 1982]
- MacQueen K-Means [MacQueen et al., 1967]
- Hartigan-Wong K-Means [Hartigan and Wong, 1979; Slonim et al., 2013]

Lloyd's K-Means algorithm

1. Initialise K initial centroids $M = \{m_{1j}, \ldots, m_{Kj}\}$ using some initialisation method.

2. Assign each data point to cluster c_{k^*} so that,

$$k^* = \underset{k}{argmin}\left\{ \sum_{j=1}^{p} (x_{ij} - m_{kj})^2 \right\}$$

3. Recompute the cluster centroids,

$$m_{kj} = \frac{1}{n_k} \sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} x_{ij}$$

4. Go to step 2 until converge.

The algorithm returns the final clusters (centroids and element assignments).

Lloyd's K-Means algorithm

1. Initialise K initial centroids using some initialisation method.

2. Assign each data point to its nearest centroid.

3. Recompute the cluster centroids by taking the mean of the data points belonging to each cluster.

4. Go to step 2 until converge.

The algorithm returns the final clusters (centroids and element assignments).

Hartigan-Wong's K-Means algorithm

1. Initialise K initial centroids $M = \{m_{1j}, \ldots, m_{Kj}\}$ using some initialisation method.

2. Assign each data point to cluster k' so that,
$$k' = \underset{k}{argmin}\left\{ \sum_{j=1}^{p} (x_{ij} - m_{kj})^2 \right\}$$

3. Recompute the cluster centroids,
$$m_{kj} = \frac{1}{n_k} \sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} x_{ij}$$

4. Set an indicator $s = 0$.

5. For each data point $x_{i:}$:

   (a) Remove it from its cluster $c_{k'}$ and compute $WCSS_{k'}$. Set $WCSS_{min} = WCSS_{k'}$.

   (b) For each cluster $c_t \neq c_{k'}$, $t = 1, \cdots, K$

      i. Assign it to c_t and compute $WCSS_t$

      ii. If $WCSS_t < WCSS_{min}$, $WCSS_{min} = WCSS_t$ and assign $x_{i:}$ to cluster c_t.

   (c) If $WCSS_{min} \neq WCSS_{k'}$, set $s = 1$ and update $m_{k':}$ and $m_{min:}$. Else $WCSS_{min} = WCSS_{k'}$ assign $x_{i:}$ to its original cluster $c_{k'}$.

6. If $s = 1$, set $s = 0$ and go to step 5 else terminate

The algorithm returns the final clusters (centroids and element assignments).

Hartigan-Wong's K-Means algorithm

1. Initialise K initial centroids using some initialisation method.

2. Assign each data point to its nearest centroid.

3. Recompute the cluster centroids by taking the mean of the data points belonging to each cluster.

4. For each data point $x_{i:}$:

   (a) Remove it from its cluster and compute the WCSS of that cluster.

   (b) Compute the WCSS of each other cluster if $x_{i:}$ was assigned them.

   (c) Assign $x_{i:}$ to the cluster with the minimum WCSS.

   (d) Update the old and the new cluster centroids of $x_{i:}$.

5. If no data points were changed clusters terminate else go to step 4.

The algorithm returns the final clusters (centroids and element assignments).

MacQueen's K-Means algorithm

1. Initialise K initial centroids $M = \{m_{1j}, \ldots, m_{Kj}\}$ using some initialisation method.
2. Assign each data point to cluster c_{k^*} so that,
$$k^* = \underset{k}{argmin}\left\{ \sum_{j=1}^{p} (x_{ij} - m_{kj})^2 \right\}$$
3. Recompute the cluster centroids,
$$m_{kj} = \frac{1}{n_k} \sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} x_{ij}$$
4. Set an indicator $s = 0$.
5. For each data point $x_{i:}$:
   (a) Assign it from cluster $c_{k'}$ to cluster c_{k^*} so that,
$$k^* = \underset{k}{argmin}\left\{ \sum_{j=1}^{p} (x_{ij} - m_{kj})^2 \right\}$$
   (b) If $c_{k'} \neq c_{k^*}$ update $m_{k':}$ and $m_{*:}$ and set $s = 1$.
6. If $s = 1$, set $s = 0$ and go to step 5 else terminate

The algorithm returns the final clusters (centroids and element assignments).

MacQueen's K-Means algorithm

1. Initialise K initial centroids using some initialisation method.
2. Assign each data point to its nearest centroid.
3. Recompute the cluster centroids by taking the mean of the data points belonging to each cluster.
4. For each data point $x_{i:}$:
   (a) Assign it to its nearest centroid.
   (b) Update the old and the new cluster centroids of $x_{i:}$.
5. If no data points were changed clusters terminate else go to step 4.

The algorithm returns the final clusters (centroids and element assignments).

3 K-Medians

Objective function [Aggarwal, 2014].

$$\mathcal{J}_{kmedians} = \sum_{k=1}^{K} \sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} \sum_{j=1}^{p} |x_{ij} - m_{kj}| \tag{5}$$

K-Medians algorithm

1. Initialise K initial centroids $M = \{m_{1j}, \ldots, m_{Kj}\}$ using some initialisation method.

2. Assign each data point to cluster k^* so that,

$$k^* = \underset{k}{argmin} \left\{ \sum_{j=1}^{p} (x_{ij} - m_{kj})^2 \right\}$$

3. Recompute the cluster centroids,

$$m_{kj} = \begin{cases} x_{ij} & , \quad i = (n_k + 1)/2 \\ (x_{ij} + x_{i'j})/2 & , \quad i = n_k/2, \qquad i' = (n_k + 1)/2 \end{cases}$$

4. Go to step 2 until converge.

The algorithm returns the final clusters (centroids and element assignments).

K-Medians algorithm

1. Initialise K initial centroids using some initialisation method.

2. Assign each data point to its nearest centroid.

3. Recompute the cluster centroids by taking the median of the data points belonging to each cluster.

4. Go to step 2 until converge.

The algorithm returns the final clusters (centroids and element assignments).

4 Geometric K-Medians

Objective function [Whelan et al., 2015].

$$\mathcal{J}_{gkmedians} = \sum_{k=1}^{K} \sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} \left| \sum_{j=1}^{p} (x_{ij} - m_{kj}) \right| \tag{6}$$

Cluster centroids.

$$\frac{\partial \mathcal{J}_{gkmedians}}{\partial m_{k'j'}} = \frac{\partial}{\partial m_{k'j'}} \sum_{k=1}^{K} \sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} \left| \sum_{j=1}^{p} (x_{ij} - m_{kj}) \right| = 0 \Rightarrow m_{k'j'} = \frac{\sum_{\substack{i=1 \\ (x_{i:} \in c_{k'})}}^{n_{k'}} \frac{x_{ij'}}{\sqrt{(x_{ij'} - m_{k'j'})^2}}}{\sum_{\substack{i=1 \\ (x_{i:} \in c_{k'})}}^{n_{k'}} \frac{1}{\sqrt{(x_{ij'} - m_{k'j'})^2}}} \tag{7}$$

Weiszfeld's algorithm

1. Initialise K initial centroids $M = \{m_{1j}, \ldots, m_{Kj}\}$ using some initialisation method.

2. Assign each data point to cluster k^* so that,

$$k^* = \underset{k}{argmin} \left\{ \sum_{j=1}^{p} (x_{ij} - m_{kj})^2 \right\}$$

3. Recompute the cluster centroids using the Weiszfeld's formula,

   (a) For each cluster k and dimension j:

   (b) Initialise the k-th centroid,

$$m_{kj} = \frac{1}{n_k} \sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} x_{ij}$$

   (c) Update the centroid estimation, $m_{kj}^{(l)} = \frac{\sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} \frac{x_{ij}}{\sqrt{\sum_{j=1}^{p} (x_{ij} - m_{kj}^{(l)})^2}}}{\sum_{\substack{i=1 \\ (x_{i:} \in c_k)}}^{n_k} \frac{1}{\sqrt{\sum_{j=1}^{p} (x_{ij} - m_{kj}^{(l)})^2}}}$,

   $l = 1, \ldots, n_k$

4. Go to step 2 until converge.

The algorithm returns the final clusters (centroids and element assignments).

5 Sparse K-Means

Objective function [Witten and Tibshirani, 2010].

$$\mathcal{J}_{skmeans} = \sum_{i=1}^{n}\sum_{j=1}^{p} w_j(x_{ij} - \mu_{1j})^2 - \sum_{k=1}^{K}\sum_{\binom{i=1}{x_{i:}\in c_k}}^{n_k}\sum_{j=1}^{p} w_j(x_{ij} - m_{kj})^2 \Rightarrow$$

$$\mathcal{J}_{skmeans} = \sum_{j=1}^{p} w_j\gamma_j \ \text{, with}\ \ \gamma_j = \sum_{i=1}^{n}(x_{ij} - \mu_{1j})^2 - \sum_{k=1}^{K}\sum_{\binom{i=1}{x_{i:}\in c_k}}^{n_k}(x_{ij} - m_{kj})^2 \tag{8}$$

$$\text{subject to} \qquad \sum_{j=1}^{p} w_j^2 \leq 1,\ \ \sum_{j=1}^{p}|w_j| \leq s,\ \ w_j \geq 0\ \ \forall j$$

Weights optimization.

$$\underset{w_j}{maximize}\left\{\sum_{j=1}^{p} w_j\gamma_j\right\} \ \text{subject to} \qquad \sum_{j=1}^{p} w_j^2 \leq 1,\ \ \sum_{j=1}^{p}|w_j| \leq s,\ \ w_j \geq 0\ \ \forall j \Rightarrow$$

$$w_j = \frac{sign(\gamma_j)(|\gamma_j| - \Delta)_+}{\sqrt{\sum_{j'=1}^{p}\left(sign(\gamma_{j'})(|\gamma_{j'}| - \Delta)\right)^2}} \tag{9}$$

where $x_+ = \mathscr{H}(x)\cdot x$, $\mathscr{H}$ is the Heaviside function and $x \in \mathbb{R}$. We assume that γ_j has a unique maximum and that $1 \leq s \leq \sqrt{p}$.

Sparse K-Means algorithm [Witten and Tibshirani, 2010]

1. Initialise K initial centroids $M = \{m_{1j}, \ldots, m_{Kj}\}$ using some initialisation method and the feature weights as $w_1 = \cdots = w_p = \frac{1}{\sqrt{p}}$.

2. Holding the weights fixed, maximize 8 with respect to M. This can be achieved by performing K-Means on the scaled data, i.e. multiply each feature j with $\sqrt{w_j}$.

3. Holding M fixed optimize equation 8 with respect to the weights applying the proposition given in equation 9. Choose $\Delta = 0$ if that leads to $\sum_{j=1}^{p}|w_j| \leq s$, otherwise find $\Delta > 0$ that results in $\sum_{j=1}^{p}|w_j| = s$. To find Δ the Bisection algorithm can be used.

4. Go to step 2 until the convergence criterion in equation 10

$$\frac{\sum_{j=1}^{p}|w_j^r - w_j^{r-1}|}{w_j^{r-1}} < 10^{-4} \ \text{, if}\ \ r > 1 \tag{10}$$

   where r refers to the current iteration, and w_j^{r-1} to the weights of the previous iteration.

The algorithm returns the final clusters (centroids and elements) and the weight of each feature.

Bisection algorithm

1. Assume $lim_1 < \Delta < lim_2$, $lim_1 = 0$ and $lim_2 = max(\gamma_1, ..., \gamma_p)$
2. Compute $\Delta = \frac{lim_1 + lim_2}{2}$ and set

$$\begin{Bmatrix} lim_2 = \Delta & , \text{ if } & \sum_{j=1}^{p} |w_j| < s \\ lim_1 = \Delta & , \text{ if } & \sum_{j=1}^{p} |w_j| \geq s \end{Bmatrix}$$

3. If $lim_2 - lim_1 \geq 10^{-4}$ go to step 2.

Publications & Online Material

- [Vouros et al., 2019]
- [Vouros and Vasilaki, 2020]

Bibliography

Aggarwal, C. C. [2014], *Data classification: algorithms and applications*, CRC Press.

Hartigan, J. A. and Wong, M. A. [1979], 'Algorithm as 136: A k-means clustering algorithm', *Journal of the Royal Statistical Society. Series C (Applied Statistics)* **28**(1), 100–108.

Jain, A. K. [2010], 'Data clustering: 50 years beyond k-means', *Pattern recognition letters* **31**(8), 651–666.

Lloyd, S. [1982], 'Least squares quantization in pcm', *IEEE transactions on information theory* **28**(2), 129–137.

MacQueen, J. et al. [1967], Some methods for classification and analysis of multivariate observations, *in* 'Proceedings of the fifth Berkeley symposium on mathematical statistics and probability', Vol. 1, Oakland, CA, USA, pp. 281–297.

Slonim, N., Aharoni, E. and Crammer, K. [2013], Hartigan's k-means versus lloyd's k-means-is it time for a change?, *in* 'IJCAI', pp. 1677–1684.

Vouros, A., Langdell, S., Croucher, M. and Vasilaki, E. [2019], 'An empirical comparison between stochastic and deterministic centroid initialisation for k-means variations', *arXiv preprint arXiv:1908.09946* .

Vouros, A. and Vasilaki, E. [2020], 'A semi-supervised sparse k-means algorithm', *arXiv preprint arXiv:2003.06973* .

Whelan, C., Harrell, G. and Wang, J. [2015], Understanding the k-medians problem, *in* 'Proceedings of the International Conference on Scientific Computing (CSC)', The Steering Committee of The World Congress in Computer Science, Computer . . . , p. 219.

Witten, D. M. and Tibshirani, R. [2010], 'A framework for feature selection in clustering', *Journal of the American Statistical Association* **105**(490), 713–726.