K-Means Clustering: Mathematical Notes

K-Means Clustering is one of the most widely used unsupervised learning algorithms for partitioning a dataset into K distinct, non-overlapping subgroups (clusters). The goal is to make the intra-cluster data points as similar as possible, while keeping inter-cluster data points as dissimilar as possible. It is a centroid-based algorithm, where each cluster is represented by the mean (centroid) of its data points.

Core Concepts:

- **Unsupervised Learning:** K-Means operates on unlabeled data, meaning there are no pre-defined target variables or classes. Its goal is to discover inherent structures or groupings within the data.
- **Cluster:** A collection of data points that are similar to each other and dissimilar to data points in other clusters.
- **Centroid:** The arithmetic mean position of all data points within a cluster. It serves as the representative point for that cluster.
- **Distance Metric:** Typically, Euclidean distance is used to measure the similarity (or dissimilarity) between data points and centroids. For two points (x_1, y_1) and (x_2, y_2) in 2D, the Euclidean distance is:
$$d((x_1, y_1), (x_2, y_2)) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$
- **Inertia (Within-Cluster Sum of Squares - WCSS):** A measure of how internally coherent clusters are. It's the sum of squared distances of each point to its assigned centroid. The K-Means algorithm aims to minimize this value.
$$\text{WCSS} = \sum_{i=1}^{K} \sum_{x \in C_i} ||x - \mu_i||^2$$
where K is the number of clusters, C_i is the i-th cluster, x is a data point in cluster C_i, and μ_i is the centroid of cluster C_i.

The K-Means Clustering Algorithm

The standard K-Means algorithm (also known as Lloyd's algorithm) is an iterative process that proceeds as follows:

Step 1. Initialization:

- Choose the number of clusters, K.
- Randomly initialize K centroids $(\mu_1, \mu_2, \ldots, \mu_K)$ from the dataset's feature space. These can be actual data points or randomly generated points within the data's range.

Step 2. Assignment Step (Expectation):

- Assign each data point to the cluster whose centroid is closest to it (based on the chosen distance metric, typically Euclidean distance).
$$C_i = \{x \mid ||x - \mu_i||^2 \leq ||x - \mu_j||^2 \quad \forall j = 1, \ldots, K\}$$

Step 3. Update Step (Maximization):

- Recalculate the centroids for each cluster. The new centroid for a cluster is the mean of all data points assigned to that cluster.
$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

Step 4. Convergence Check:

- Repeat Steps 2 and 3 until the centroids no longer change significantly, or the assignments of data points to clusters no longer change, or a maximum number of iterations is reached.

Step-by-Step Calculation Example

Let's consider a simple 2D dataset and apply K-Means clustering.

Dataset: We have 7 data points: $P_1 = (1,1)$, $P_2 = (1.5,2)$, $P_3 = (3,4)$, $P_4 = (5,7)$, $P_5 = (3.5,5)$, $P_6 = (4.5,5)$, $P_7 = (3.5,4.5)$

Goal: Cluster these 7 points into $K = 2$ clusters.

Step 1: Initialization Choose $K = 2$. Randomly select two data points as initial centroids. Let's say:

- Centroid 1 (C_1) = $P_1 = (1,1)$
- Centroid 2 (C_2) = $P_4 = (5,7)$

Step 2: Iteration 1 - Assignment Step Calculate the Euclidean distance from each data point to $C_1(1,1)$ and $C_2(5,7)$. Assign each point to the cluster with the closest centroid. **Euclidean Distance:**

$$d((x_1,y_1),(x_2,y_2)) = \sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$

Point	Coordinates	Dist to C_1	Dist to C_2	Assigned Cluster
P_1	$(1,1)$	0.00	7.21	C_1
P_2	$(1.5,2)$	1.12	6.10	C_1
P_3	$(3,4)$	3.61	3.61	C_1 (Tie-break, assign to 1st)
P_4	$(5,7)$	7.21	0.00	C_2
P_5	$(3.5,5)$	4.71	2.50	C_2
P_6	$(4.5,5)$	5.31	2.06	C_2
P_7	$(3.5,4.5)$	4.30	2.92	C_2

Current Clusters:

- Cluster 1 (Red): $\{P_1(1,1), P_2(1.5,2), P_3(3,4)\}$
- Cluster 2 (Blue): $\{P_4(5,7), P_5(3.5,5), P_6(4.5,5), P_7(3.5,4.5)\}$

Step 3: Iteration 1 - Update Step Recalculate centroids:

- New C_1: $(\frac{1+1.5+3}{3}, \frac{1+2+4}{3}) = (\frac{5.5}{3}, \frac{7}{3}) \approx (1.83, 2.33)$
- New C_2: $(\frac{5+3.5+4.5+3.5}{4}, \frac{7+5+5+4.5}{4}) = (\frac{16.5}{4}, \frac{21.5}{4}) \approx (4.13, 5.38)$

Step 4: Iteration 2 - Assignment Step Calculate distances to new centroids $C_1(1.83, 2.33)$ and $C_2(4.13, 5.38)$.

Point	Coordinates	Dist to C_1	Dist to C_2	Assigned Cluster
P_1	$(1,1)$	1.57	5.38	C_1
P_2	$(1.5,2)$	0.47	4.28	C_1
P_3	$(3,4)$	2.04	1.78	C_2
P_4	$(5,7)$	5.69	1.84	C_2
P_5	$(3.5,5)$	3.15	0.73	C_2
P_6	$(4.5,5)$	3.78	0.53	C_2
P_7	$(3.5,4.5)$	2.74	1.08	C_2

Current Clusters:

- Cluster 1 (Red): $\{P_1(1,1), P_2(1.5,2)\}$
- Cluster 2 (Blue): $\{P_3(3,4), P_4(5,7), P_5(3.5,5), P_6(4.5,5), P_7(3.5,4.5)\}$

Notice P_3 moved from Cluster 1 to Cluster 2.

Step 5: Iteration 2 - Update Step Recalculate centroids:

- New C_1: $(\frac{1+1.5}{2}, \frac{1+2}{2}) = (\frac{2.5}{2}, \frac{3}{2}) = (1.25, 1.5)$
- New C_2: $(\frac{3+3.5+4.5+3.5}{5}, \frac{4+7+5+5+4.5}{5}) = (\frac{19.5}{5}, \frac{25.5}{5}) = (3.9, 5.1)$

Step 6: Iteration 3 - Assignment Step Calculate distances to new centroids $C_1(1.25, 1.5)$ and $C_2(3.9, 5.1)$.

Point	Coordinates	Dist to C_1	Dist to C_2	Assigned Cluster
P_1	$(1, 1)$	0.56	5.02	C_1
P_2	$(1.5, 2)$	0.56	3.92	C_1
P_3	$(3, 4)$	3.05	1.42	C_2
P_4	$(5, 7)$	6.66	2.19	C_2
P_5	$(3.5, 5)$	4.16	0.41	C_2
P_6	$(4.5, 5)$	4.78	0.61	C_2
P_7	$(3.5, 4.5)$	3.75	0.72	C_2

Current Clusters:

- Cluster 1 (Red): $\{P_1(1, 1), P_2(1.5, 2)\}$
- Cluster 2 (Blue): $\{P_3(3, 4), P_4(5, 7), P_5(3.5, 5), P_6(4.5, 5), P_7(3.5, 4.5)\}$

No points changed clusters compared to the end of Iteration 1. The centroids also converged to stable positions. Therefore, the algorithm terminates.

Final Clusters and Centroids:

- **Cluster 1:** $\{P_1(1, 1), P_2(1.5, 2)\}$ with centroid $(1.25, 1.5)$
- **Cluster 2:** $\{P_3(3, 4), P_4(5, 7), P_5(3.5, 5), P_6(4.5, 5), P_7(3.5, 4.5)\}$ with centroid $(3.9, 5.1)$

Choosing the Optimal Number of Clusters (K)

One of the main challenges in K-Means is determining the optimal value for K. Here are common methods:

1. **Elbow Method:**
   - Run K-Means for a range of K values (e.g., from 1 to 10).
   - For each K, calculate the WCSS (Inertia).
   - Plot WCSS against K. The "elbow" point on the plot, where the rate of decrease in WCSS sharply changes (flattens out), is often considered the optimal K. This signifies that adding more clusters beyond this point does not significantly reduce the within-cluster variance.
2. **Silhouette Method:**
   - For each data point, calculate its silhouette coefficient, which measures how similar that point is to its own cluster (cohesion) compared to other clusters (separation). The coefficient ranges from -1 to +1.
   - Average silhouette scores are computed for different K values.
   - The K value that yields the highest average silhouette score is often considered optimal, as it indicates well-separated and cohesive clusters.
3. **Gap Statistic Method:**
   - Compares the WCSS of the clustered data to that of a reference random distribution.
   - The optimal K is the one that maximizes the gap between the WCSS of the actual data and the expected WCSS from the random data.
4. **Domain Knowledge:** Often, prior knowledge about the data or the problem domain can provide insights into a reasonable number of clusters.

Advantages of K-Means Clustering:

- **Simplicity and Efficiency:** Relatively easy to understand and implement, and computationally efficient, especially for large datasets.
- **Scalability:** Can handle large datasets with a reasonable number of clusters and features. Its time complexity is approximately $O(n \cdot K \cdot d \cdot i)$, where n is data points, K is clusters, d is dimensions, and i is iterations.
- **Interpretability:** The cluster centroids are easy to interpret as they represent the average characteristics of the points in their respective clusters.

Disadvantages of K-Means Clustering:

- **Requires Pre-specifying K:** The number of clusters (K) must be chosen beforehand, which is often difficult without prior knowledge or empirical methods.
- **Sensitive to Initial Centroids:** The final clustering result can be highly dependent on the initial random placement of centroids, potentially leading to suboptimal (local optima) solutions. This is often mitigated by running the algorithm multiple times with different initializations (e.g., K-Means++).
- **Sensitive to Outliers:** Outliers can significantly distort cluster centroids, pulling them away from the true center of the cluster. Pre-processing steps like outlier detection/removal are often necessary.
- **Assumes Spherical Clusters:** K-Means implicitly assumes that clusters are spherical and of similar size and density. It struggles with clusters of irregular shapes, varying densities, or overlapping boundaries.
- **Not Suitable for Categorical Data:** Primarily designed for numerical data. Adaptations like K-Modes are needed for categorical data.

Applications of K-Means Clustering:

K-Means is a versatile algorithm with numerous real-world applications across various domains:

- **Customer Segmentation:** Grouping customers based on purchasing behavior, demographics, or activity patterns for targeted marketing.
- **Image Compression/Quantization:** Reducing the number of distinct colors in an image by clustering similar pixel colors, leading to smaller file sizes. **Document Clustering:** Organizing large collections of text documents into themes or topics.
- **Anomaly Detection:** Identifying unusual patterns or outliers in data (e.g., fraudulent transactions, network intrusions) by flagging points far from any cluster centroid.
- **Geospatial Analysis:** Clustering geographic locations, such as identifying optimal locations for new stores or delivery hubs.
- **Genomics/Bioinformatics:** Grouping genes with similar expression patterns.

Prepared By:
Md. Atikuzzaman
Lecturer
Department of Computer Science and Engineering
Green University of Bangladesh
Email: atik@cse.green.edu.bd