# K-means Clustering Tutorial

September 17, 2025

## 1 Introduction

K-means clustering partitions observations into $K$ disjoint groups by minimizing within-cluster variance. The algorithm alternates between assigning each point to its nearest centroid and updating centroids as the mean of assigned points. Because K-means assumes roughly spherical clusters of similar density, it performs best on standardized numeric features and can struggle with elongated or irregular shapes.

## 2 Theory and Formulas

### 2.1 Objective Function

Given data matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top$, K-means minimizes

$$\min_{\{C_k\},\{\boldsymbol{\mu}_k\}} \sum_{k=1}^{K} \sum_{\mathbf{x}_i \in C_k} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|_2^2, \tag{1}$$

where $C_k$ denotes the index set of cluster $k$ and $\boldsymbol{\mu}_k$ its centroid. This is equivalent to maximizing between-cluster dispersion when total dispersion is fixed.

### 2.2 Lloyd's Algorithm

The iterative refinement procedure typically used is:

1. **Initialization**: choose $K$ starting centroids, often via k-means++ to promote separation.
2. **Assignment**: allocate each point to the cluster with the nearest centroid under the chosen distance metric (commonly Euclidean).
3. **Update**: recompute each centroid as the mean of its assigned points.
4. Repeat steps 2–3 until assignments stop changing or the centroid shifts fall below a tolerance.

Although convergence is guaranteed, it may reach a local minimum, so multiple restarts with different seeds are recommended.

### 2.3 Relationship to Variance Decomposition

K-means implicitly minimizes the trace of within-cluster scatter matrix $\mathbf{S}_W$. For standardized data, total scatter $\mathbf{S}_T$ decomposes as $\mathbf{S}_T = \mathbf{S}_W + \mathbf{S}_B$, with between-cluster scatter $\mathbf{S}_B$ capturing centroid separation. Lower within-cluster inertia therefore corresponds to higher between-cluster separation.

## 3 Applications and Tips

- **Customer segmentation**: group customers by purchasing patterns or engagement metrics to tailor marketing strategies.
- **Vector quantization**: compress signals or images by mapping observations to a limited set of prototype vectors.
- **Feature engineering**: use cluster IDs as categorical features for downstream supervised models.
- **Best practices**: scale features, inspect inertia and silhouette for model selection, and watch out for sensitivity to outliers.

## 4 Python Practice

The script `gen_clustering_k_means_figures.py` synthesizes three Gaussian blobs, applies K-means, and saves both the labeled scatter plot and an elbow curve of inertia versus $K$.

Listing 1: Excerpt from gen$_{c}lustering_{km}eans_{f}igures.py$

```python
from sklearn.cluster import KMeans

kmeans = KMeans(n_clusters=3, init="k-means++", n_init=10, max_iter=300,
                random_state=42)
labels = kmeans.fit_predict(points)

inertias = []
for k in range(1, 9):
    model = KMeans(n_clusters=k, init="k-means++", n_init=10,
                   max_iter=300, random_state=42)
    model.fit(points)
    inertias.append(model.inertia_)
```

# 5 Result

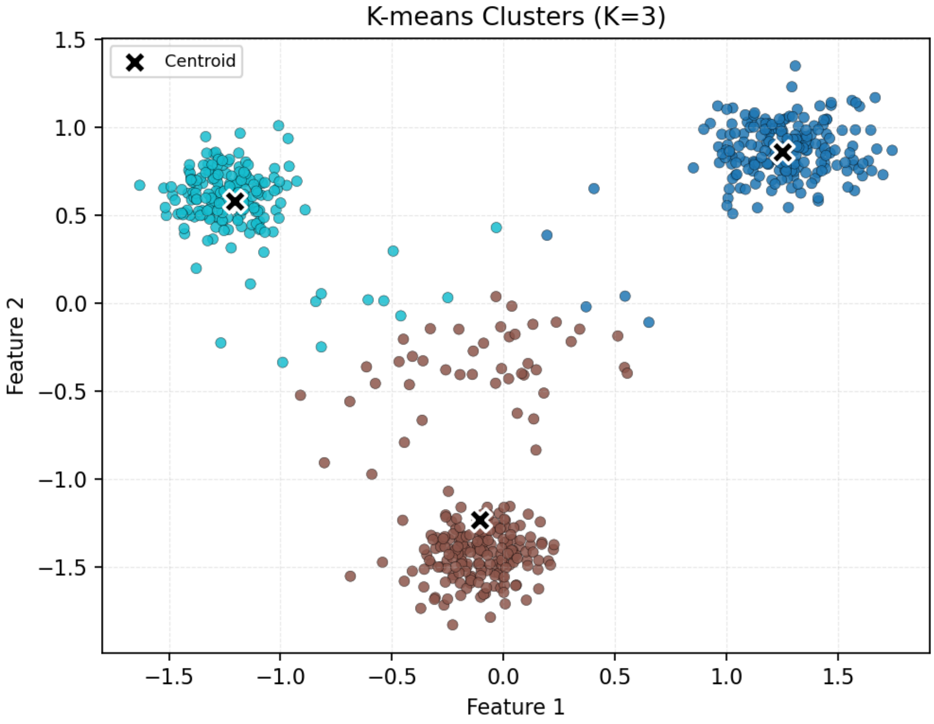

Figure 1: K-means clustering of synthetic blobs ($K = 3$)

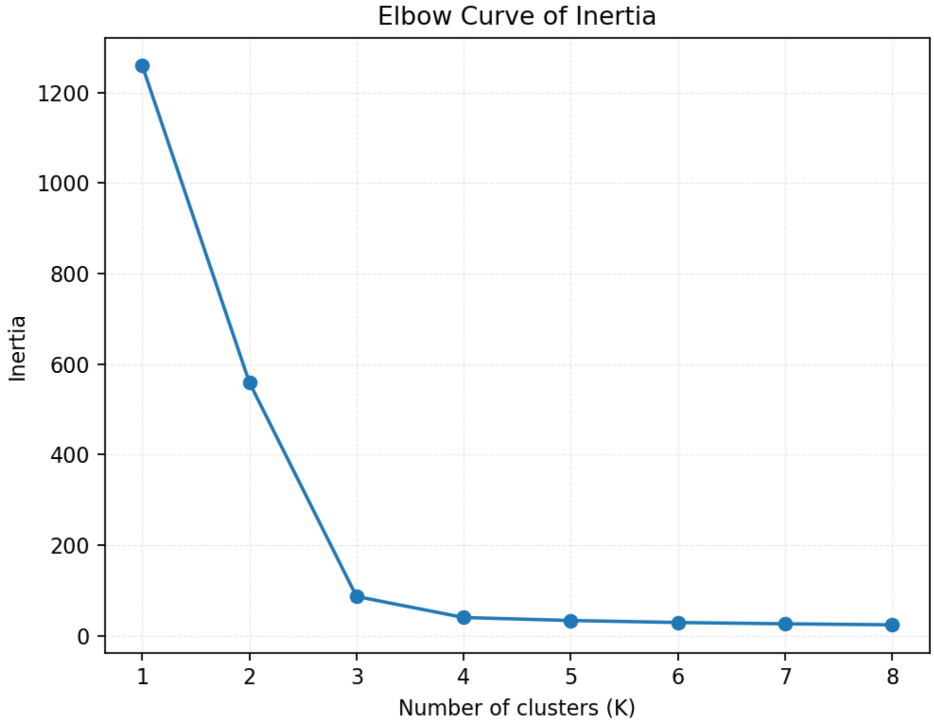

Figure 2: Elbow curve of inertia versus number of clusters

# 6 Summary

K-means offers a fast and scalable method for partitioning standardized numeric data when clusters are roughly spherical. Proper initialization, multiple restarts, and diagnostic plots such as the elbow method help mitigate local minima and over- or undersegmentation. The synthetic example demonstrates a typical workflow from clustering to inertia analysis.