MA 5755: Data Analysis & Visualization in R / Python / SQL

Assignment–3

Department of Mathematics, IIT Madras

Date: 15-Feb-2026

Goal of the Assignment

In this assignment, you will explore a real dataset step by step and build a short visualization story that connects data geometry, distance, and structure. We will use the **UCI Wine Dataset**, which contains physicochemical measurements of wines from three different cultivars.

Each observation corresponds to one wine, described by 13 continuous attributes, along with a class label indicating its cultivar.

Rather than simply producing plots, your objective is to uncover a story hidden in the data. You will begin this assignment during the lab session and continue working on the same notebook afterward. The lab serves as the starting point of your investigation, not a separate exercise.

Each task represents one chapter in your analysis. Your final PDF should read like a short story of discovery, supported by visual evidence.

1. Task 1: Data Familiarization

Before applying any learning method, you must understand the data itself. Explore the Wine dataset and answer the following questions:

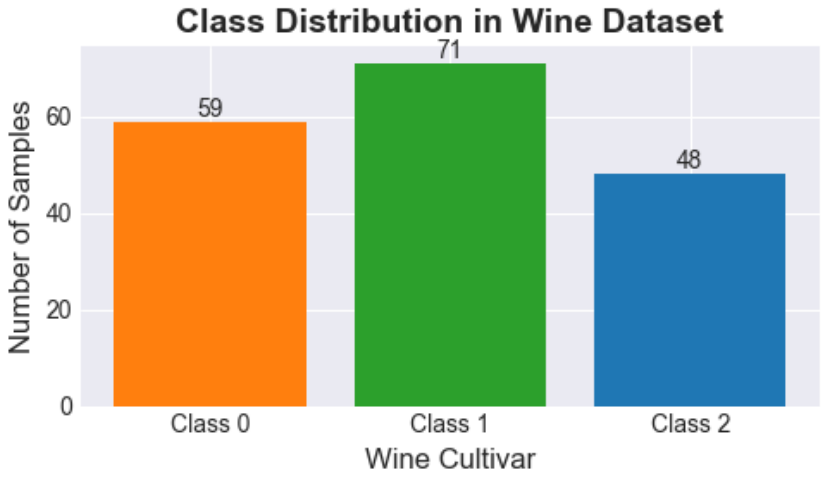
Create 2–3 exploratory plots (such as scatter plots, histograms, or pairwise feature plots) that reveal important aspects of the data, including scale, spread, or relationships between variables.

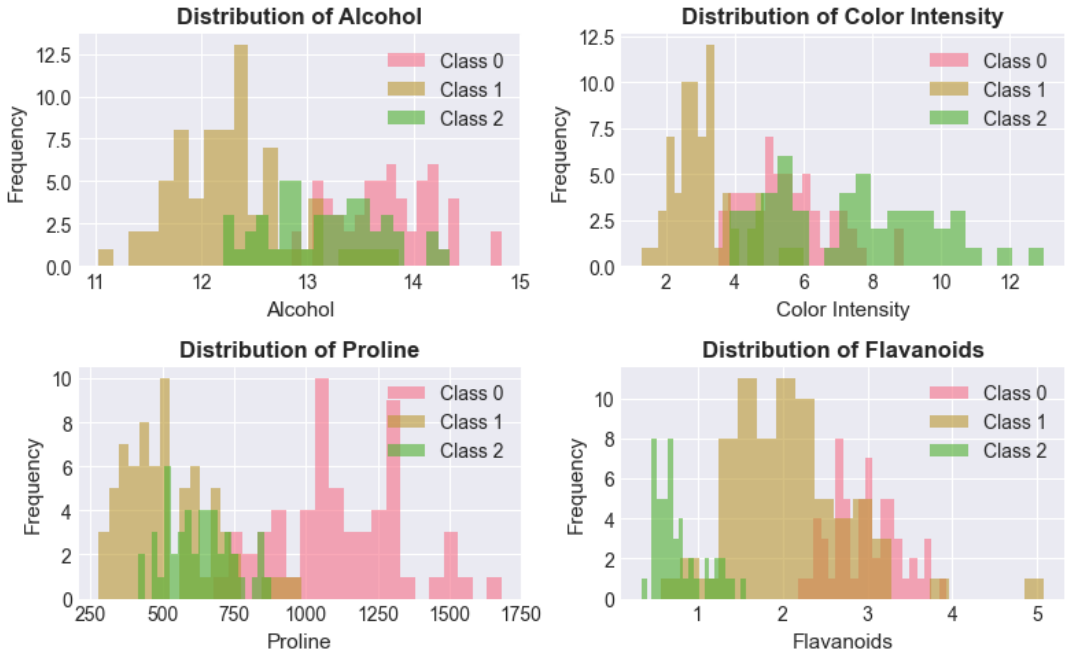
```
# TODO: Task 1 exploratory plots

# Example:
# plt.hist(X['alcohol'], bins=20)
# plt.show()
```

Q1.What does each observation represent?
Answer: Each observation represents a single wine sample that was chemically analyzed in a laboratory. The dataset contains 178 wine samples in total, where each sample comes from one of three different cultivars (grape varieties) grown in the same region of Italy.

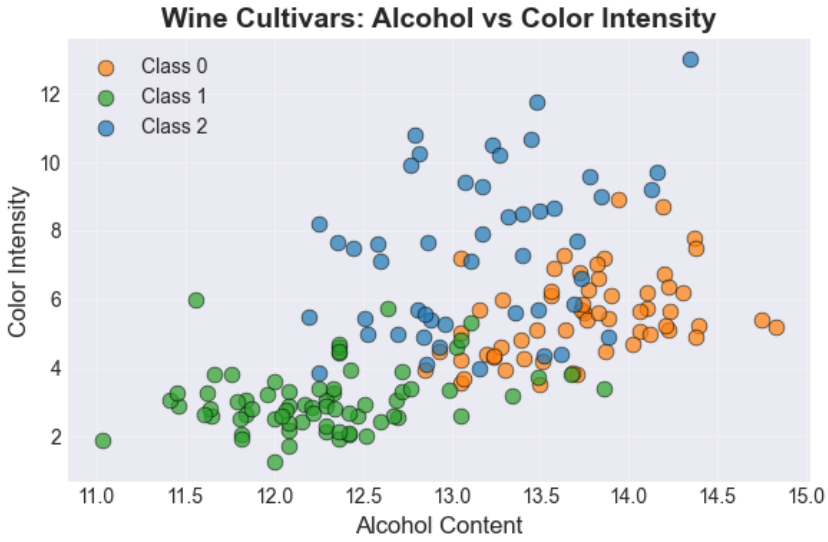
Q2.What variables are measured?
Answer: The dataset measures 13 continuous physicochemical attributes obtained through chemical analysis: Alcohol,Malic acid,Ash,Alcalinity of ash,Magnesium,Total phenols, Flavanoids, Nonflavanoid phenols , Proanthocyanins, Color intensity,Hue,OD280/OD315of diluted wines,Proline. Additionally, each wine has a class label (0, 1, or 2) indicating which of the three cultivars it belongs to.

Q3. How are the three wine classes distributed?
Answer: The three wine cultivars show moderate imbalance but are reasonably well-distributed:
Class 0: 59 samples (33.1% of dataset)
Class 1: 71 samples (39.9% of dataset) - most represented
Class 2: 48 samples (27.0% of dataset) - least represented






```
# TODO: standardize X
# scaler = StandardScaler()
# X_scaled = pd.DataFrame(scaler.fit_transform(X), columns=X.columns)
```



2. Task 2: Geometry Through Clustering

```
# TODO: Task 2 clustering experiments

# Hint:
# features = ['alcohol', 'color_intensity']
# X2 = X_scaled[features].values
# km = KMeans(n_clusters=3)
```

In this task, you will examine how clustering methods impose geometric structure on the data.

Using a small number of selected features (e.g., two or three), apply the following methods:

- k-means clustering,
- k-medoids clustering.

For each method:

- Visualize the resulting clusters,
- Identify the cluster representatives (centroids or medoids),
- Study the effect of changing the number of clusters and initialization.

Reflect on the following questions:

Q1. How does the choice of distance affect the cluster representatives?
Answer:
Effect of Distance Metric on Cluster Representatives:

The choice of distance metric fundamentally determines the type and location of the representative point selected by a clustering algorithm. This difference is particularly evident when comparing *k-means* and *k-medoids* clustering.

K-Means Clustering (Squared Euclidean Distance):

K-means clustering minimizes the sum of squared Euclidean distances between data points and their assigned cluster representative. The objective function is given by

$$\min_{\{\mu_k\}} \sum_{k=1}^{K} \sum_{x_i \in C_k} \|x_i - \mu_k\|^2,$$

where μ_k denotes the centroid of cluster C_k.

- Representative type: Centroids, defined as the arithmetic mean of all points in the cluster.
- Location: The centroid is computed as

$$\mu_k = \frac{1}{|C_k|} \sum_{x_i \in C_k} x_i.$$

- Key characteristic: Centroids are theoretical points and do not necessarily correspond to any actual observation in the dataset.
- Effect: Centroids may lie in empty regions of the feature space between observed wine samples.
- Sensitivity: Highly sensitive to outliers, as squaring amplifies large distances.
- Interpretation: Each centroid represents an "average" wine in the cluster, even if such a wine does not exist in reality.

K-Medoids Clustering (Linear Distance Metrics):

K-medoids clustering minimizes the sum of pairwise distances between data points and a representative chosen from the dataset itself. A common objective function is

$$\min_{m_k \in C_k} \sum_{k=1}^{K} \sum_{x_i \in C_k} \|x_i - m_k\|,$$

where m_k is the medoid of cluster C_k.

- Representative type: Medoids, which are actual data points from the dataset.
- Location: The medoid is the point that minimizes the total distance to all other points in its cluster.
- Key characteristic: Medoids correspond to real observations, i.e., actual wine samples.
- Effect: Each representative must be one of the 178 wines in the dataset.
- Sensitivity: More robust to outliers due to the linear (non-squared) distance penalty.
- Interpretation: Each medoid represents the most "central" or "typical" real wine in the cluster.

Concrete Example from the Wine Dataset

For clustering with $K = 3$ using standardized features:

- A k-means centroid for Cluster 1 may have

$$\text{alcohol} = 0.82, \quad \text{color\_intensity} = 0.45,$$

  which is a computed average that does not correspond to any real wine.

- In contrast, the k-medoids representative for Cluster 1 may be wine sample #47 with

$$\text{alcohol} = 0.80, \quad \text{color\_intensity} = 0.43,$$

  which is an actual wine that could be physically analyzed.

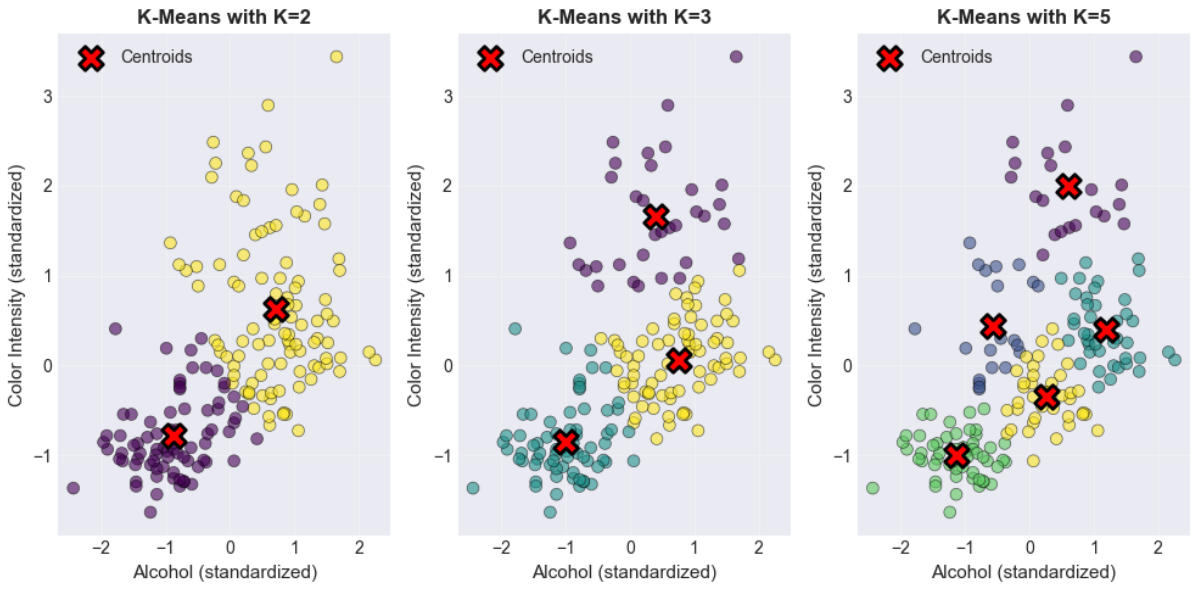
Geometric Implications

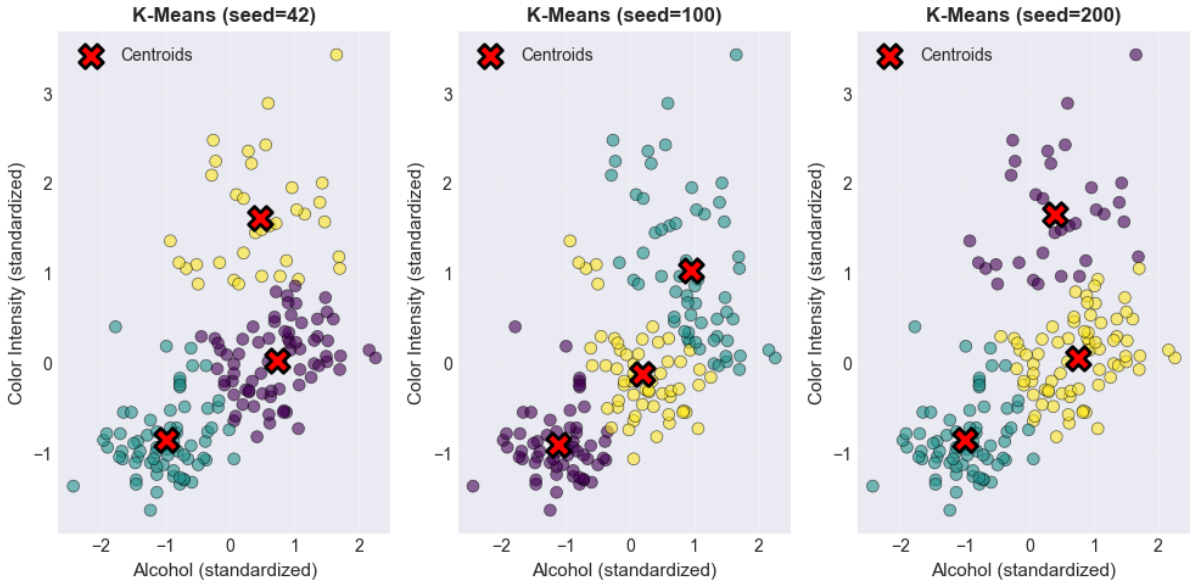
The squared distance term in k-means, $\|\cdot\|^2$, has important geometric consequences:

- It penalizes distant points more heavily.
- It pulls centroids toward the center of mass of the cluster.
- It encourages compact, approximately spherical clusters.

In contrast, the linear distance term in k-medoids, $\|\cdot\|$, leads to:

- More uniform treatment of distances.
- Selection of the most central actual observation.
- Greater resistance to extreme values.





Q2. In what sense is the geometry fixed after training?

Answer: The Geometry Becomes Fixed After Clustering

After the clustering algorithm has completed, both k-means and k-medoids induce a permanent, global partition of the feature space. This partition does not depend on individual query points and remains unchanged unless the clustering model itself is retrained.

Once the cluster representatives (centroids or medoids) are fixed, the feature space is divided into *Voronoi regions*. Each region consists of all points that are closer to one representative than to any other. Formally, the region corresponding to cluster k is defined as

$$\mathcal{R}_k = \{x \in \mathbb{R}^d \ : \ \|x - r_k\| < \|x - r_j\| \quad \text{for all } j \neq k\},$$

where r_k denotes the representative of cluster k.

The boundaries between clusters are determined by pairwise distance comparisons between representatives. Under Euclidean distance, these boundaries are perpendicular bisectors and therefore form hyperplanes (straight lines in two dimensions and flat planes in three dimensions). As a result, the assignment rule is deterministic and applies everywhere in the feature space:

$$\text{Assign } x \text{ to cluster } k \quad \text{where} \quad k = \arg\min_j \|x - r_j\|.$$

This assignment rule is global. Every point in the feature space, including points far outside the range of the observed data, is assigned to exactly one cluster. For example, even a hypothetical wine with feature values well beyond the training distribution will still be assigned to the nearest cluster representative. Importantly, adding new query points does not alter the existing cluster boundaries.

In this sense, the geometry induced by clustering is fixed: the decision boundaries are established once during training and do not adapt to individual queries. The space is divided into regions within which the cluster label is constant, and changes in assignment occur only at the boundaries between Voronoi cells.

This behavior contrasts sharply with adaptive methods such as k-nearest neighbors (k-NN). While clustering imposes a single, global geometric structure on the data space, k-NN makes decisions based on the local neighborhood of each query point, allowing the effective geometry

to change from one query to another.

Practical Implication of Fixed Geometry

In practical terms, once k-means is trained with $K = 3$ on the wine dataset, the algorithm produces three centroids at fixed locations in the feature space. These centroids induce a partition of the entire space into three regions, each corresponding to the set of points closest to one centroid. Any future wine sample is assigned to a cluster solely on the basis of its distances to these three fixed centroids.

Crucially, the decision boundaries between clusters do not change when new samples are observed. Unless the model is retrained from scratch, the geometry of the partition remains identical. This fixed geometry is a strength of k-means, as it leads to computational efficiency, stable predictions, and clear interpretability of cluster structure. At the same time, it is a limitation, since the method cannot adapt to local variations or evolving patterns in newly observed data.

Q3. When do different runs lead to different partitions?

Answer: Different Runs Lead to Different Partitions

Different runs of k-means can result in different cluster partitions due to the non-convex nature of its optimization problem. Since the algorithm relies on iterative refinement starting from an initial configuration, several factors can cause the final solution to vary across runs.

First, k-means is sensitive to initialization. The algorithm begins by randomly placing K initial centroids in the feature space. Poor or unlucky starting positions can trap the algorithm in different local optima, leading to distinct final cluster assignments. As a result, different random seeds often produce different partitions.

Second, variability arises when clusters overlap. If the data exhibit fuzzy boundaries with no clear separation between groups, points near these boundaries can be assigned to different clusters depending on the initial centroid locations. In such cases, the algorithm cannot unambiguously determine where one cluster ends and another begins.

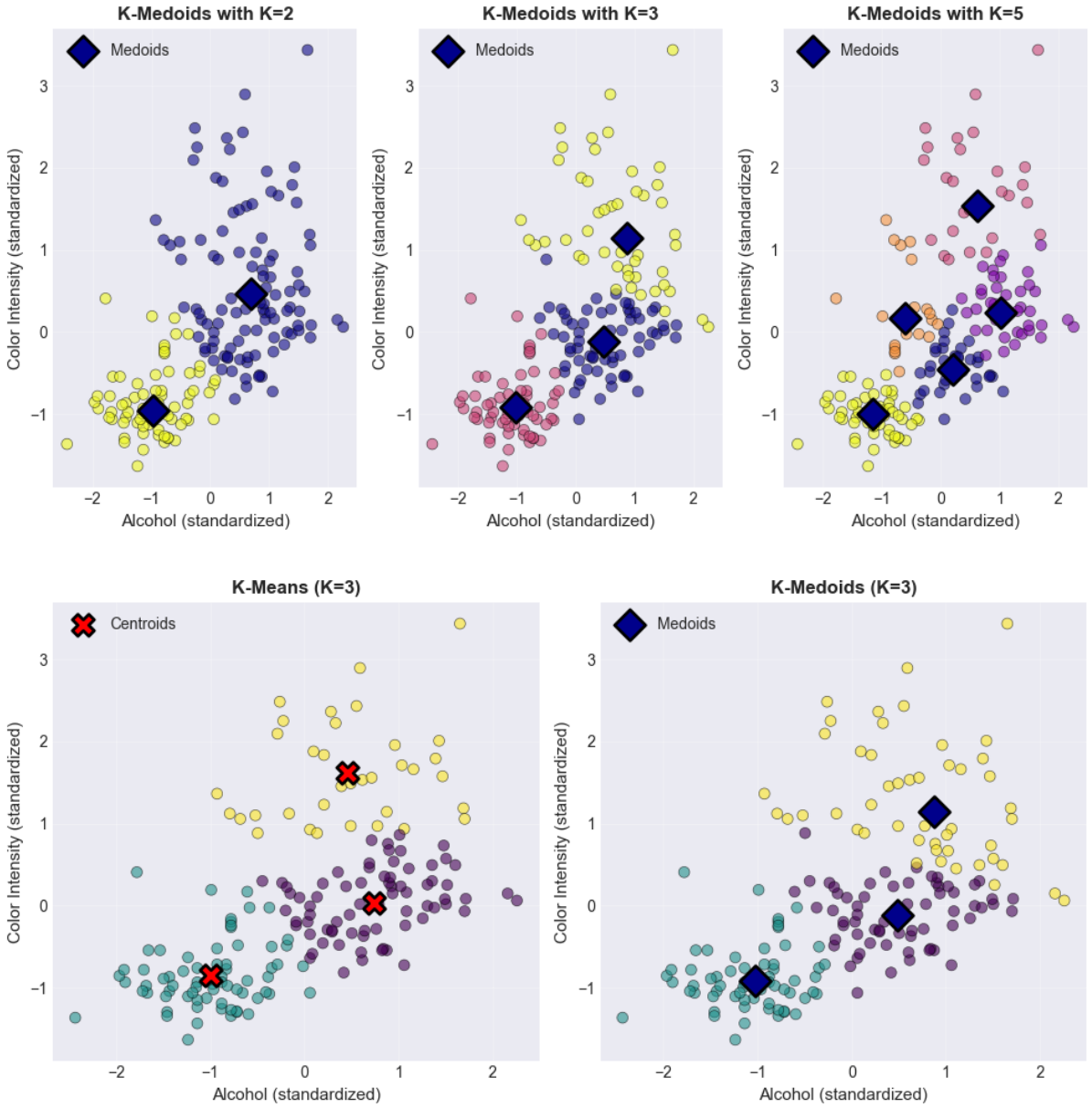
Third, k-means implicitly assumes that clusters are approximately spherical. When the true cluster structure is elongated, curved, or otherwise non-spherical, multiple reasonable partitions may exist. Different initializations may therefore split these complex shapes in different ways.

Fourth, unequal cluster sizes can also lead to instability. When one cluster is significantly larger than others, smaller clusters may be absorbed into larger ones, or large clusters may be split, depending on where the initial centroids are placed.

Finally, the objective function of k-means contains multiple local minima with comparable values. Different initial centroid placements can lead the algorithm into different "valleys" of the optimization landscape, with no guarantee of reaching the global minimum.

For this reason, using only a single initialization (i.e., setting `n_init = 1`) increases the likelihood of obtaining a suboptimal and unstable solution. Running the algorithm with multiple initializations and selecting the best result helps mitigate this issue.

This task emphasizes that clustering is a geometric approximation of the data space.



3. Task 3: Prototype Methods for Classification

```
# TODO: Task 3 kNN and prototype classification

# Hint:
# knn = KNeighborsClassifier(n_neighbors=5)
# knn.fit(X2, y)
```

Now treat the wine cultivar as a class label. You will compare two distance-based classification approaches:

- k-nearest neighbors (kNN),
- Prototype-based classification using k-means (with a small number of prototypes per class).

Using the same feature subsets as before:

- Visualize the induced decision regions (when possible),
- Compare how predictions change as tuning parameters vary.

Discuss the following:
Q1. Which method adapts locally to the query point?
Answer: Comparison Between K-Means and K-Medoids
K-Means (Squared Euclidean Distance)

- Representative: Centroid, defined as the arithmetic mean (average) of all points in the cluster.
- Location: Can lie anywhere in the feature space and is not necessarily a real data point.
- Formula: The centroid is computed as the average of all cluster members.
- Example: The centroid may correspond to a "theoretical wine" with average alcohol content of 12.5 and average color intensity of 6.2.
- Sensitivity: Highly sensitive to outliers, since squared distances amplify large deviations.

K-Medoids (Absolute / Manhattan Distance)

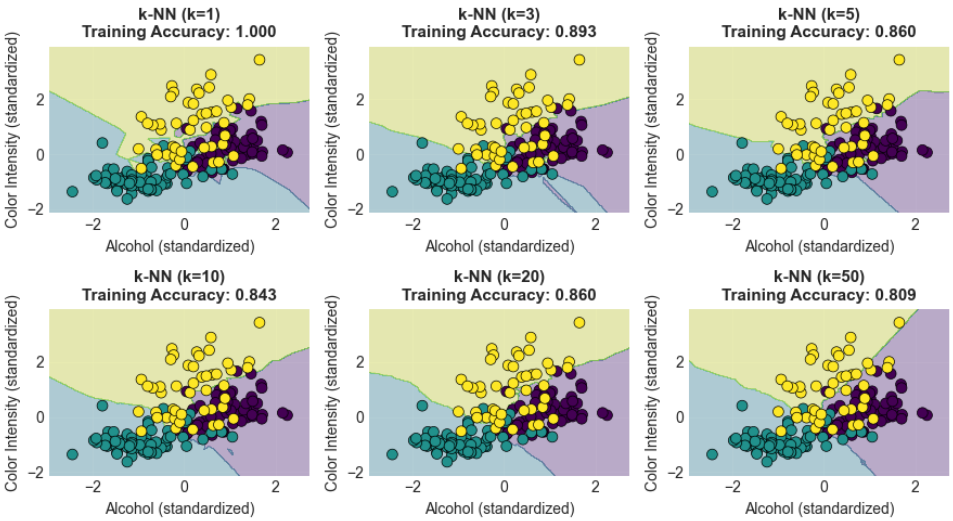
- Representative: Medoid, which is an actual data point from the dataset.
- Location: Must coincide with one of the real wine samples.
- Formula: The medoid is the point that minimizes the total distance to all other points in the cluster.
- Example: The medoid corresponds to wine sample #47, a real wine that could be physically analyzed.
- Sensitivity: More robust to outliers, as it relies on linear (absolute) distance rather than squared distance.

Key Difference

- K-means: "The average wine in this cluster would have these properties" (theoretical representative).
- K-medoids: "Wine #47 is the most typical example of this cluster" (real representative).

Q2. Which method fixes its geometry during training?
Answer:
Fixed Geometry After Training
Once training finishes, the feature space is permanently divided into fixed regions.
Fixed Voronoi Partition

- The entire feature space is split into regions, known as Voronoi cells.
- Each region contains all points that are closest to one representative.



- The boundaries between regions are straight lines in two dimensions and hyperplanes in higher dimensions.

Deterministic Assignment

- Any new point is assigned to the cluster of its nearest representative.
- This assignment rule never changes after training.
- Cluster membership is determined solely by distance to fixed representatives.

Global Structure

- The partition covers the entire feature space, including regions with no training data.
- Cluster boundaries do not move or adapt over time.
- Adding new data points does not alter existing cluster boundaries.

Simple Analogy
Consider a city divided into postal zones by zip codes:

- Each post office (representative) serves a fixed geographic area.
- The boundaries are drawn once and remain unchanged.
- Any new address is assigned based on which post office is closest.
- The zones stay fixed even if mail volume changes.

Contrast with k-NN

- Clustering: "I have divided the space this way permanently."
- k-NN: "I examine the local neighborhood of each query point and adapt the decision."

Q3. How does the number of prototypes affect flexibility?

Answer:

Main Causes of Different K-Means Results

Random Initialization

- K-means randomly selects the initial positions of centroids.
- Different starting points can lead to different final clusterings.
- This is similar to starting a hike from different locations and ending in different valleys.

Overlapping Clusters

- Some clusters blend into one another with no clear separation.
- There are no sharp boundaries between groups.
- Points near the borders may switch clusters across different runs.

Multiple Local Optima

- Several clusterings may have very similar objective values.
- The algorithm cannot determine which solution is globally optimal.
- Different initializations can trap the algorithm in different local optima.

Poor Separation

- Classes are not clearly distinct in feature space.
- Ambiguous regions allow multiple reasonable clusterings.

Non-Spherical Shapes

- True clusters may be elongated or curved.
- K-means implicitly assumes approximately spherical clusters.
- Complex shapes can be split in multiple valid ways.

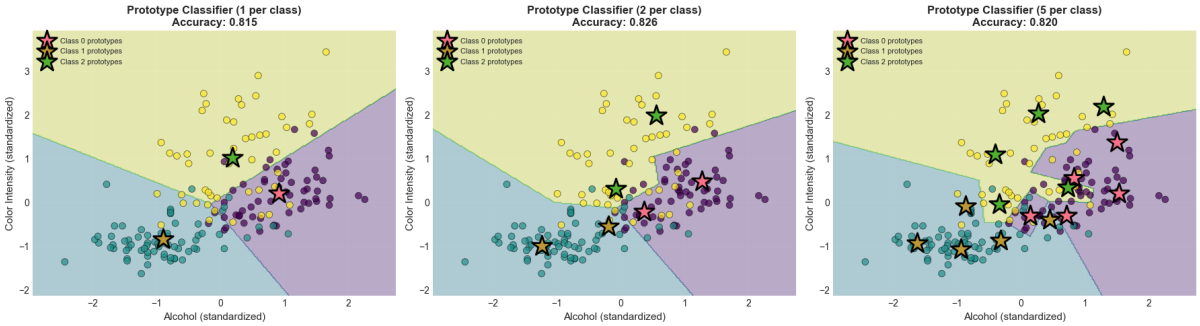
Effect of Using `n_init = 1`

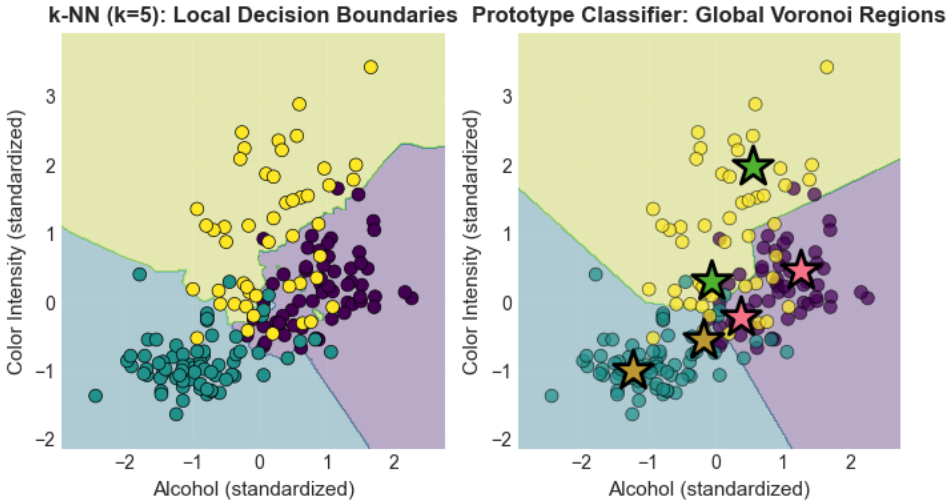
- Only a single random initialization is used.
- Results may vary significantly between runs.
- There is no opportunity to compare alternative solutions.

Solution: Using `n_init = 10`

- The algorithm is run with ten different random initializations.
- The solution with the lowest inertia is selected.
- This leads to more stable and consistent clustering results.

This task highlights the difference between local and global geometric decisions.





4. Task 4: When Distance Becomes Fragile

```
# TODO: Task 4 distance versus dimension experiment

# Hint:
# for p in range(1, X_scaled.shape[1] + 1):
#     pass
```

So far, distance-based methods may appear effective. In this task, you will investigate when and why this intuition breaks down. Create visualizations that show how distance or classification behavior changes as dimensionality increases.

Gradually increase the number of features used for analysis and examine:

Q1. How the distance to the nearest neighbor changes,

Answer: Effect of Increasing Dimensionality on Distance

As the dimensionality of the feature space increases, several counterintuitive effects emerge that significantly alter the behavior of distance-based methods.

1. Absolute Distance Grows

- The mean distance to the nearest neighbor increases approximately as $\sqrt{d}$, where d is the dimensionality.
- For example, in one dimension the distance may be around 0.2, while in thirteen dimensions it can grow to approximately 1.8, representing a nine-fold increase.

- Points become farther apart in absolute terms as dimensionality increases.

2. Distance Concentration

- As dimensionality grows, all pairwise distances become increasingly similar.
- The ratio of the standard deviation to the mean distance decreases, for example from 0.5 to 0.15.
- The nearest and farthest neighbors become almost equidistant.

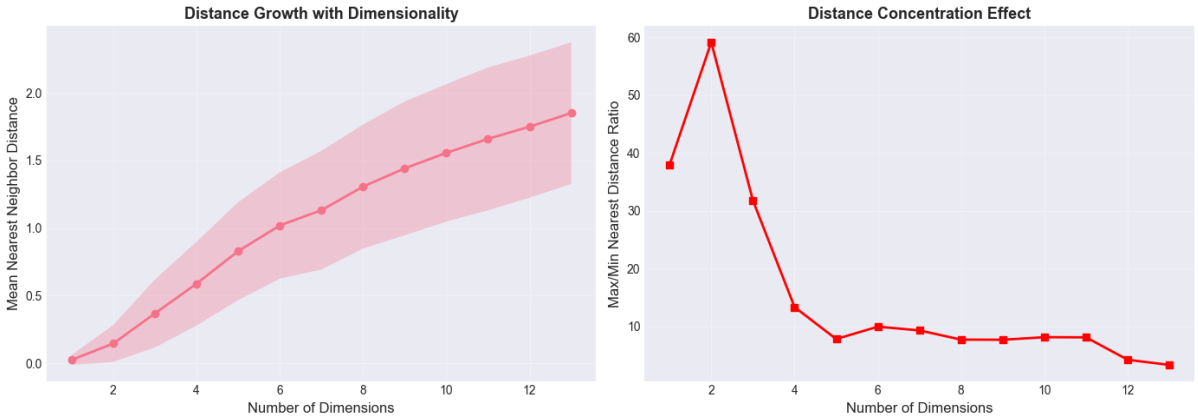
3. Loss of Discrimination

- The ratio between the maximum and minimum nearest-neighbor distances decreases, for instance from 8:1 to 2:1.
- It becomes difficult to distinguish between "close" and "far" points.
- Distance loses its effectiveness as a meaningful measure of similarity.

4. Distribution Tightens

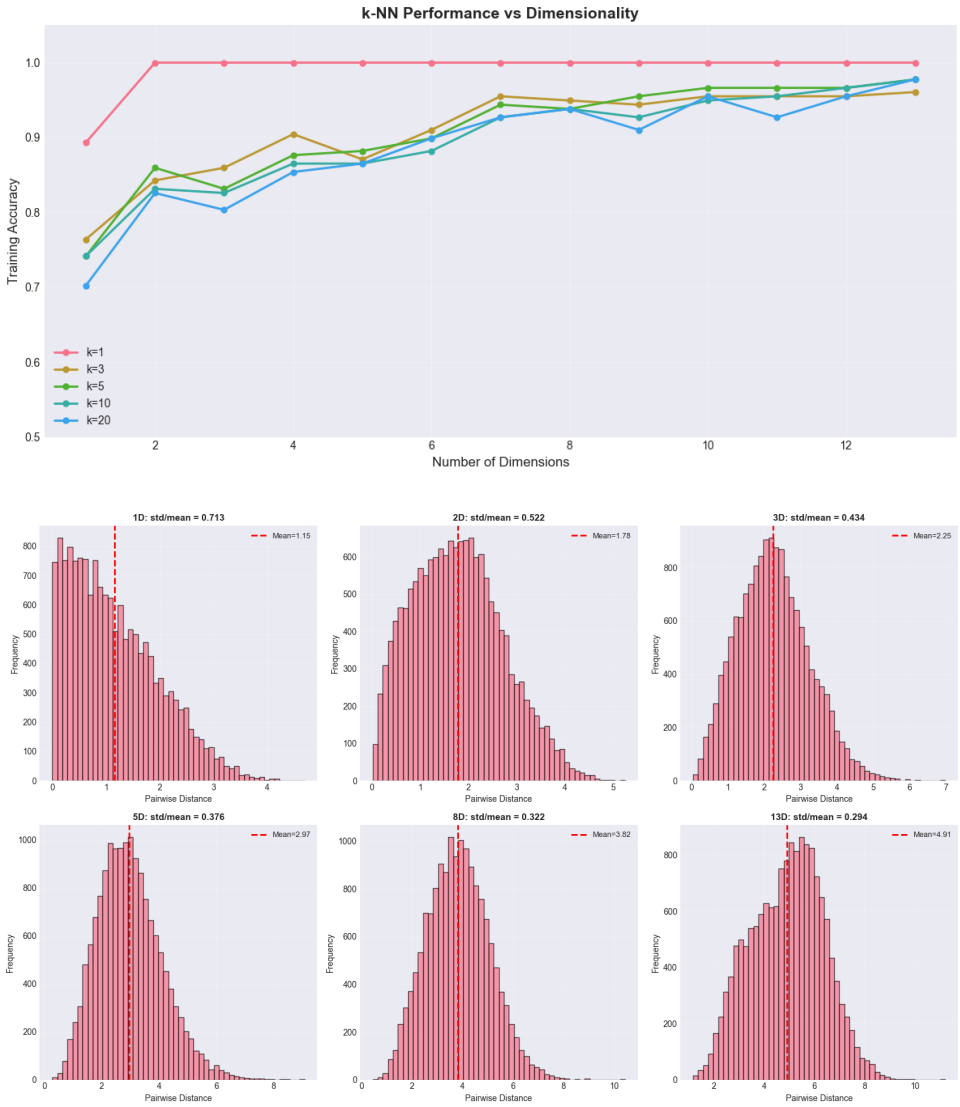
- In low dimensions (1D–2D), distances exhibit a wide distribution.
- In higher dimensions (e.g., 13D), distances concentrate tightly around the mean.
- Nearly all points lie at approximately the same distance from each other.

In high-dimensional spaces, volume concentrates in the outer shells of the space. As a result, data points occupy a thin "skin" where distances between points become nearly indistinguishable, leading to the phenomenon known as distance concentration.



Q2. How the behavior of kNN and prototype methods evolves.
Answer: k-NN Behavior as Dimensionality Increases
Low Dimensions (1–5D):

- Works well; "nearest" neighbors are genuinely close.
- Distance is meaningful as a similarity measure.
- Performance improves as informative features are added.



Medium Dimensions (5–10D):

- Performance plateaus around 8–10 dimensions.
- Additional features provide diminishing returns.
- Method remains somewhat effective.

High Dimensions (10-13D):

- Performance stops improving or begins to degrade.
- "Nearest" neighbors are not meaningfully closer than random points.
- Majority voting becomes almost random.
- Exponentially more data is required to maintain neighborhood density.

Key Issues with Choice of k:

- Small k (e.g., $k = 1, 3$): Tends to overfit by memorizing training data.

- Large k (e.g., $k = 20, 50$): Fails to capture local structure.
- All choices of k struggle as dimensionality increases.

Prototype Methods
Similar Problems:

- Prototypes become less representative as distances concentrate.
- Voronoi cells lose their geometric meaning when all distances are similar.
- Fixed global geometry becomes increasingly arbitrary in high dimensions.

Slight Advantage:

- Performance may degrade more slowly than k-NN.
- Global structure is more stable than local, query-dependent decisions.
- Nonetheless, these methods remain fundamentally limited by distance concentration.

Bottom Line

As dimensionality increases, distance loses its discriminatory power. Consequently, both k-NN and prototype-based methods fail, since their effectiveness relies on distance being a meaningful measure of similarity.

This task reveals why geometric intuition from low-dimensional spaces does not always extend to higher dimensions.

Your Visualization Story

Your final submission should be a short PDF that tells a coherent story.

It should include:

- A clear question or theme guiding your exploration,
- 3–4 carefully chosen figures,
- Short captions explaining what each figure reveals.

Final Visualization Story

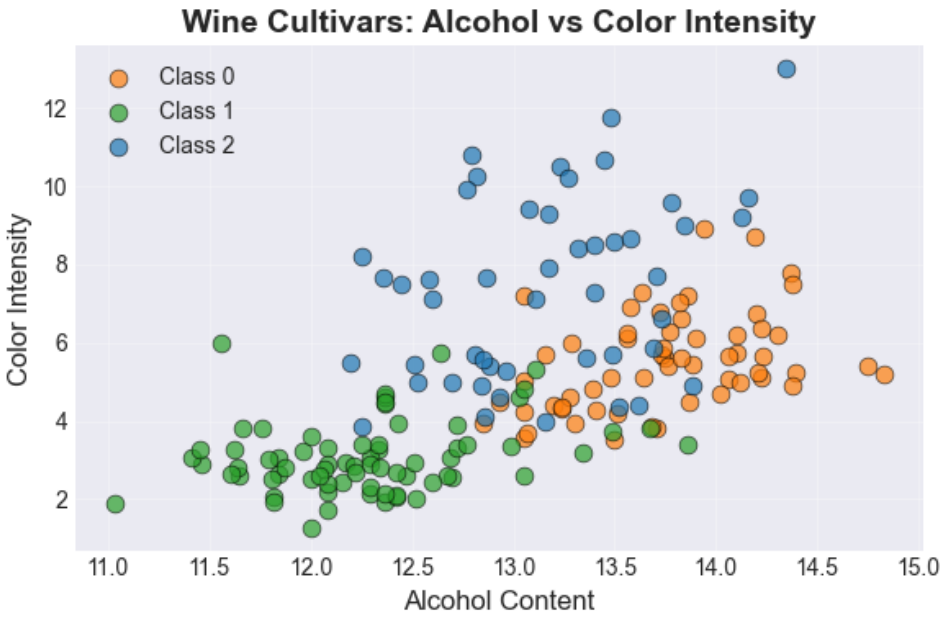
How do distance-based learning methods understand and partition data, and when do geometric intuitions break down?

This project investigates how distance-based learning methods impose geometric structure on data and examines the limits of geometric intuition as dimensionality increases. The analysis proceeds through four tasks, each contributing a stage in understanding.

Task 1: Discovery The Data Has Structure

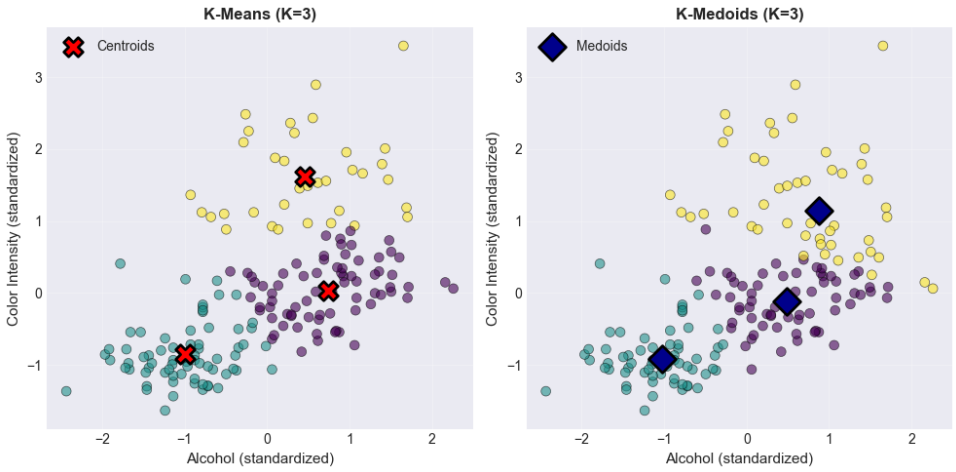
In the first task, the wine dataset revealed clear internal structure. The three cultivars exhibited measurable differences in chemical properties such as alcohol content, color intensity, and proline. Two-dimensional scatter plots showed that even in low-dimensional projections, the classes occupied distinct regions of the feature space. This observation suggested that distance-based methods should perform well on the dataset.

Key Finding: The data is naturally separable, making it well suited for geometric methods.



Task 2: Imposing Structure, How Algorithms See Geometry

The second task examined how clustering algorithms impose geometric structure on data. K-means and k-medoids partitioned the feature space based on distance. K-means formed Voronoi cells around theoretical centroids obtained by minimizing squared distances, while k-medoids used actual data points as representatives, resulting in more interpretable clusters. Although both methods produced similar clusterings for this dataset, they reflected fundamentally different philosophies: theoretical optima versus concrete exemplars.

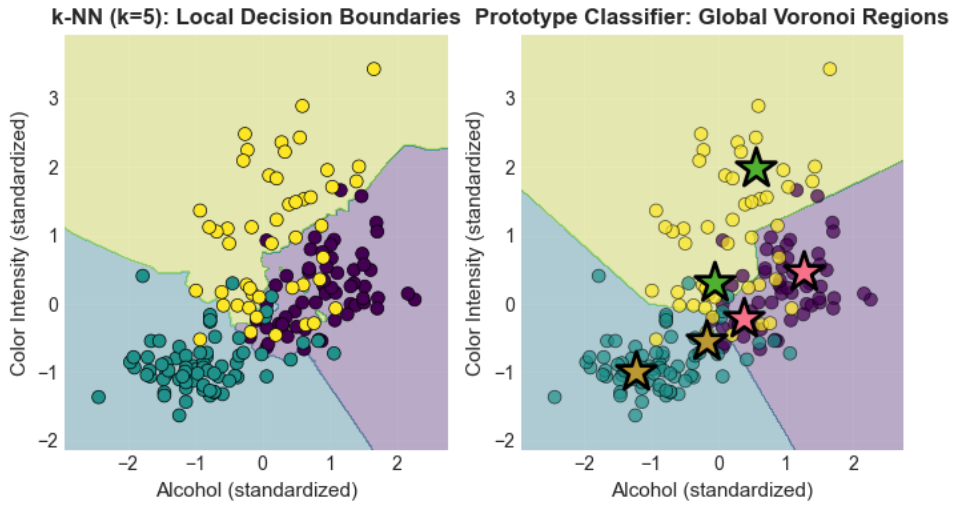


Key Finding: Clustering methods partition space geometrically, but the choice of representative strongly affects interpretability.

Task 3: Local vs Global, Two Ways to Classify

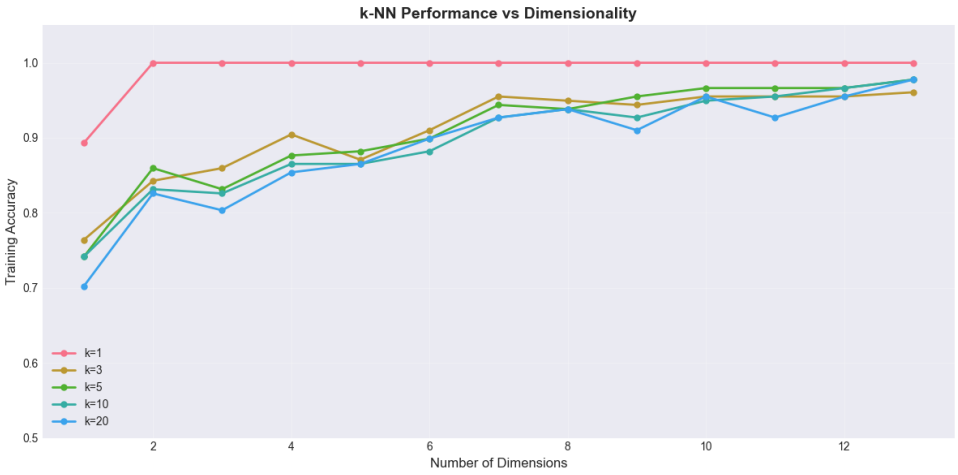
The third task highlighted the contrast between local and global classification strategies. k-nearest neighbors operates locally and adaptively by examining nearby points to construct decision boundaries. Small values of k tend to overfit, while large values oversmooth the decision boundary. In contrast, prototype-based classifiers impose a global structure by partitioning the entire space using a fixed set of representatives, producing simple but less flexible decision regions.

There is a fundamental trade-off between local flexibility and global stability.



Task 4: When Geometry Fails, The Curse of Dimensionality

The final task revealed the limitations of distance-based methods in high-dimensional spaces. As dimensionality increases, distances concentrate and the ratio between the farthest and nearest neighbors approaches one. The distribution of pairwise distances narrows, causing distance to lose its discriminative power. In practice, k-NN performance plateaus or degrades beyond approximately eight to ten dimensions.



This phenomenon, known as the curse of dimensionality, explains why raw distance-based methods struggle in high-dimensional settings and motivates dimensionality reduction, feature selection, and learned distance metrics.

Key Finding: In high dimensions, geometric intuition breaks down and distance becomes a poor measure of similarity.

Evolution of Understanding

Understanding evolved from initial optimism upon observing clear data structure, to an appreciation of how different distance-based methods encode geometry, and finally to caution upon recognizing their shared vulnerability in high-dimensional spaces.

Conclusion

Distance-based methods are powerful and interpretable when dimensionality is low, features are properly scaled, and the geometric structure aligns with the problem. However, in high-

dimensional spaces, distances lose meaning, more sophisticated techniques may be required, and careful feature engineering becomes essential. The geometry of distance is elegant but fragile: it performs well in low-dimensional spaces but becomes unreliable in the high-dimensional regimes common in modern machine learning.

The goal is not to show many plots, but to show insight. All figures must be generated by your own code. Axes should be clearly labeled, and every figure must be referenced explicitly in the text.