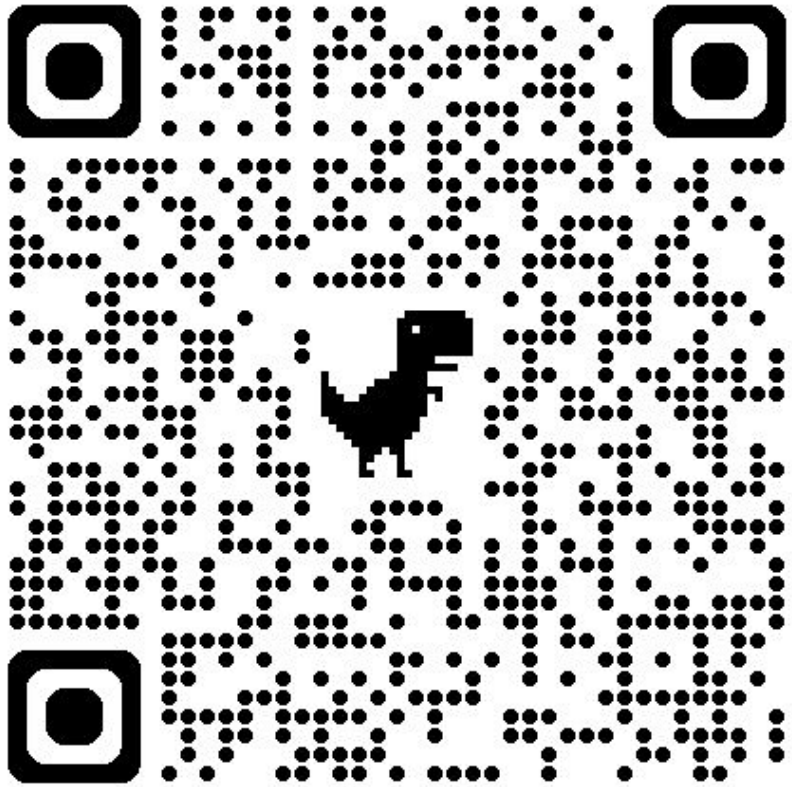
K-MEANS VS DBSCAN CLUSTERING ALGORITHMS

Presented by : Swathi Bangalore Satish, Fatima Bowers, Prapti Bhatt

Test Questions

1. Who introduced the term K-Means?
2. What is the time complexity of K-Means?
3. What are the hyperparameters for DBSCAN?



https://forms.gle/JqkVaA1xeXbaXREu8

I'm Prapti Bhatt

I'm from Rajasthan, India 🇮🇳

EDUCATION

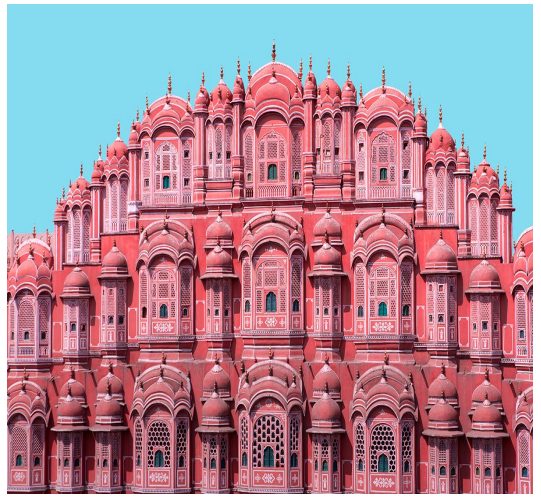
- Master's in Computer Science at UTK
- Bachelor's in CS from India
- Area of Interest: Healthcare & Technology

ABOUT RAJASTHAN

- Known as the "Land of Kings"
- Famous for Udaipur, Jaisalmer, Jaipur
- Rich culture, palaces & festivals

MY PASSIONS

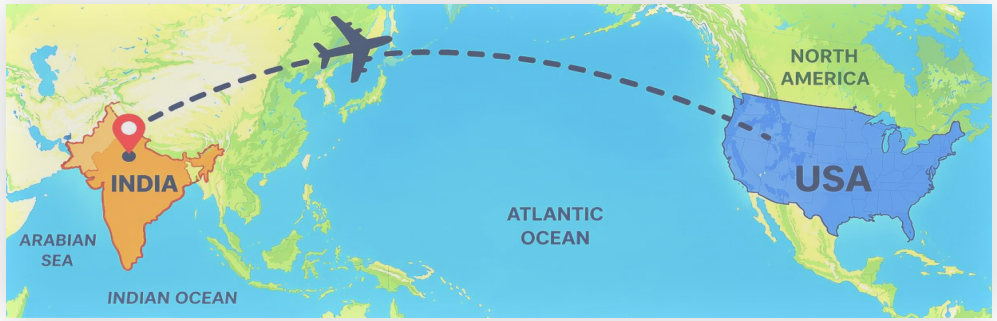
- Creating natural perfumes using flowers & herbs
- Super spicy food & exploring different sweets
- Mountain lover & paragliding enthusiast
- Skydiving is on my bucket list!



I’m Swathi,

I’m from INDIA

- I am currently pursuing my Master’s in Computer Science at UT.
- Undergrad in Electronics and Communication



About Fatima

- Hometown: Knoxville, TN
- High School: Hardin Valley Academy

Degree Program

- Pursuing M.S. in Computer Science
  - Interests: Software Development and Healthcare
- B.S. in C.S. at UTK
- GTA

Hobbies/Interests

- Baking
- Art
- Basketball
- Coffee
- Road trips

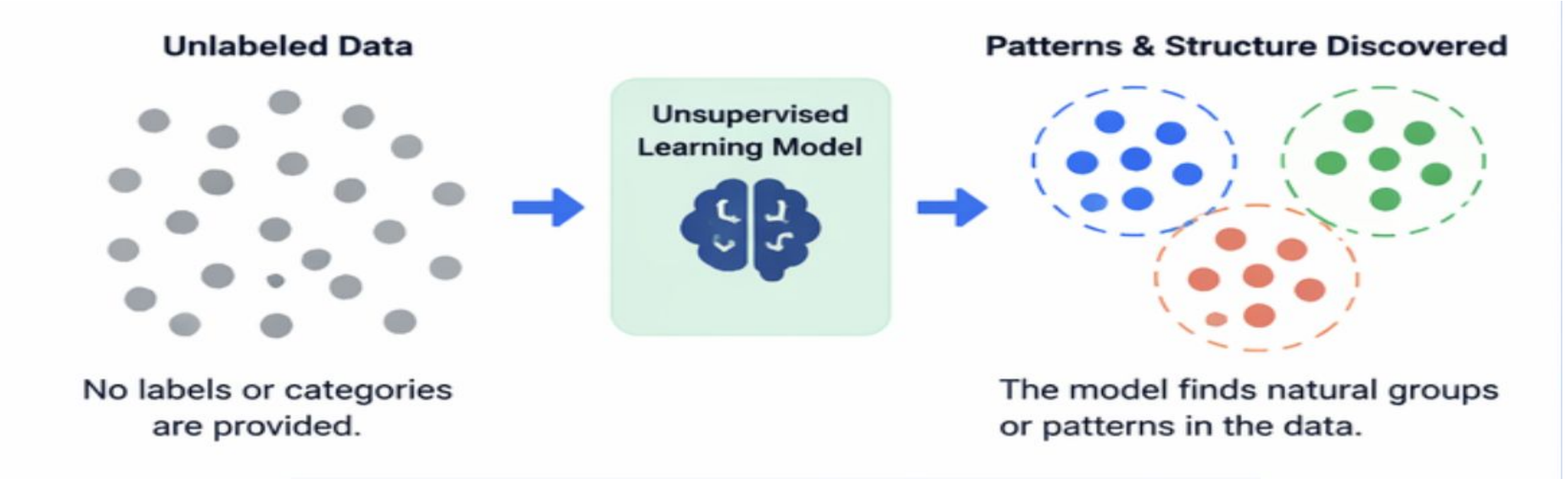
Outline

- Overview
  - What is Clustering?
  - Types of Clustering
- History of K-means and DBSCAN
- K-means Algorithm
- DBSCAN Algorithm
- Applications
- Implementation
- Comparison Results: K-means vs DBSCAN
- Open Issues, References, Discussion

Overview

Clustering

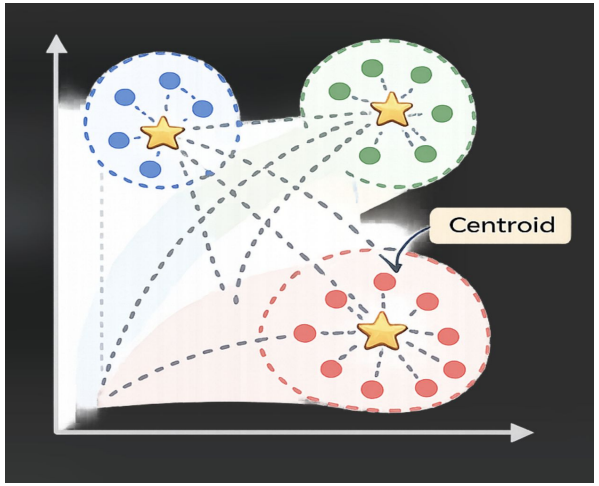
- Unsupervised Learning - A type of machine learning that finds patterns or structure in unlabeled data without using predefined output labels.
- Clustering is a type of unsupervised learning
  - It groups similar data points together.
  - Similarity is based on shared features or distance.
  - Used to discover natural groups in data.



Centroid-based and Density-based clustering

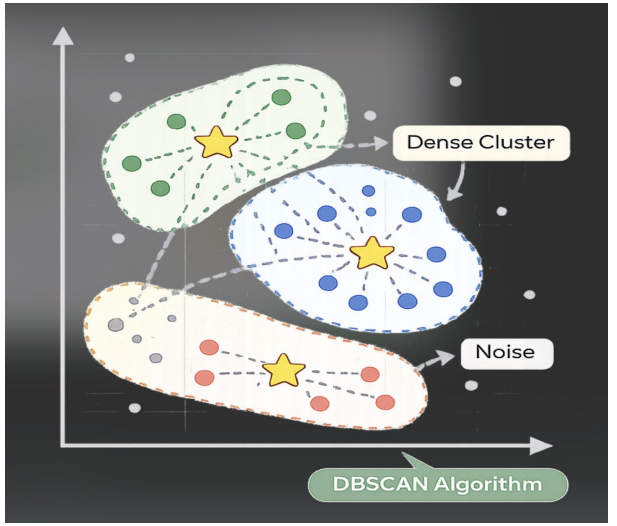
Centroid-based clustering

- Forms clusters around a central point called a centroid
- Example: K-means



Density-based clustering

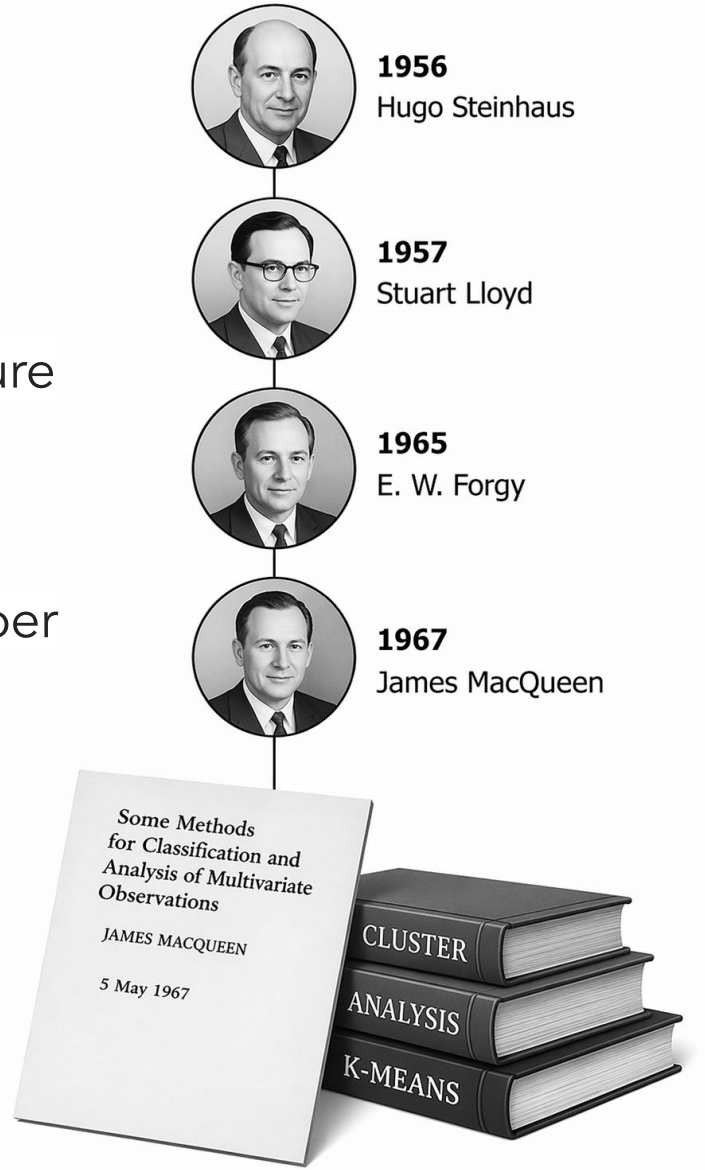
- Forms clusters based on dense regions of data points
- Example: DBSCAN



History

History of K-Means

- 1956: Hugo Steinhaus proposed an early clustering idea based on grouping points by closeness.
- 1957: Stuart Lloyd developed the iterative centroid-based procedure used in modern K-means.
- 1965: E. W. Forgy published a similar clustering method.
- 1967: James MacQueen introduced the term “K-means” in his paper Some Methods for Classification and Analysis of Multivariate Observations.
- Over time, K-means became one of the most widely used partitioning clustering algorithms in machine learning and data analysis.

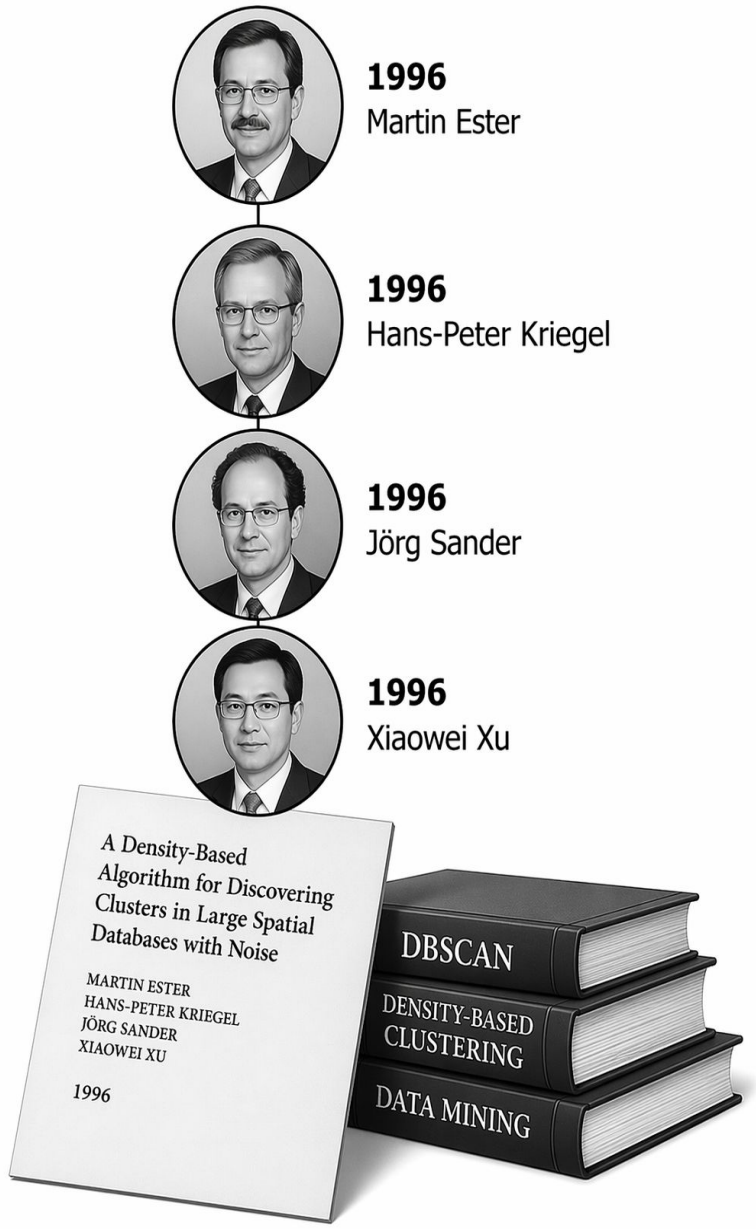


History of DBSCAN

- 1996: DBSCAN was introduced by Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu.
- It was proposed in the paper:

  "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise."

- DBSCAN was designed to solve problems where traditional clustering methods struggled with:
  - irregular cluster shapes
  - noise and outliers
  - large spatial datasets
- Unlike K-means, DBSCAN does not require the number of clusters in advance.
- It became one of the most important density-based clustering algorithms in data mining and machine learning.



Algorithms

What is K-Means?

- Centroid-Based Clustering Algorithm
- An unsupervised learning algorithm that partitions n data points into K clusters.
- Each point belongs to the cluster with the nearest centroid (mean).

Objective Function

- Minimize WCSS:

$$J = \sum_{i=1}^{K} \sum_{x \in C_i} ||x - \mu_i||^2$$

Goal: Minimize J until convergence

Key Terms

- K: Number of clusters (you choose!)
- Centroid: Center point of a cluster
- WCSS: Within-cluster sum of squares (minimize this!)

- J: Total error to minimize
- K: Number of clusters
- C_i: Points in cluster i
- x: A data point
- μ_i: Centroid of cluster i
- $||x - \mu_i||^2$: Squared Euclidean distance

How K-Means Works

Step 1: Initialize

Randomly place K centroids in the data space. These are starting guesses.

Step 2: Assign

Formula: $c_i = \text{argmin} \, ||x - \mu_j||^2 \, j \in \{1,...,K\}$

Each point calculates distance to all centroids and joins the nearest one's cluster.

Step 3: Update

Formula: $\mu_i = (1/|C_i|) \sum x \in C_i$

Recalculate each centroid as the mean (average position) of all points in its cluster.

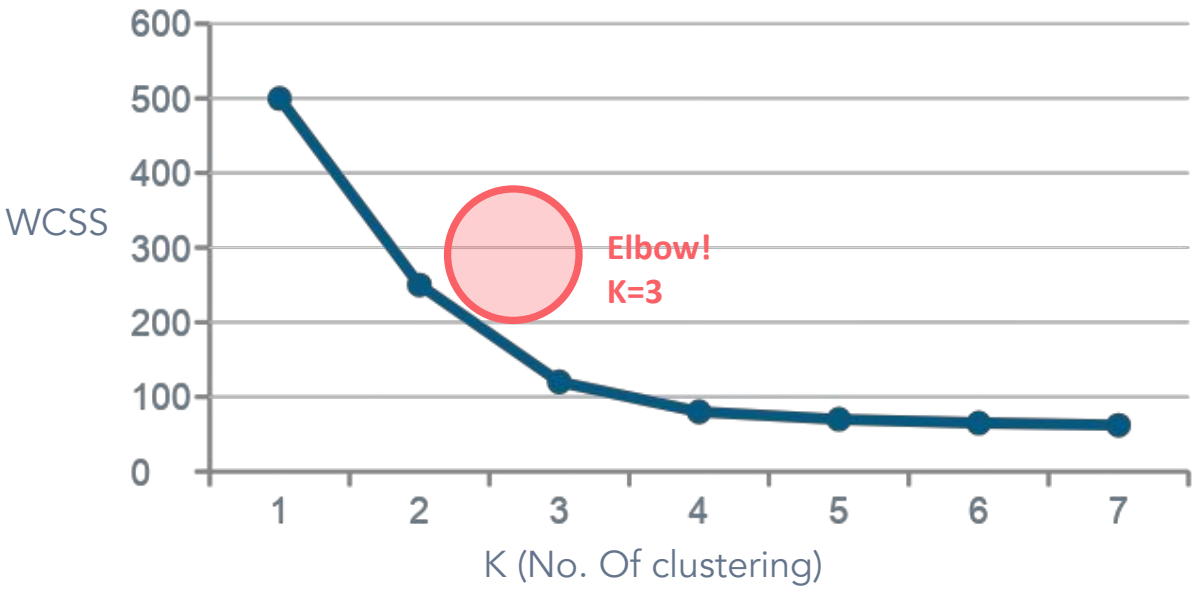
Step 4: Repeat

Go back to Step 2. Stop when centroids don't move anymore (convergence).

Choosing K & Complexity

How to Choose K? The Elbow Method

1. Run K-Means for K = 1, 2, 3, ...
2. Plot WCSS vs K
3. Find the 'elbow' : where adding clusters stops helping



Time Complexity

$O(n \times K \times d \times i)$

Space:
$O((n + K) \times d)$

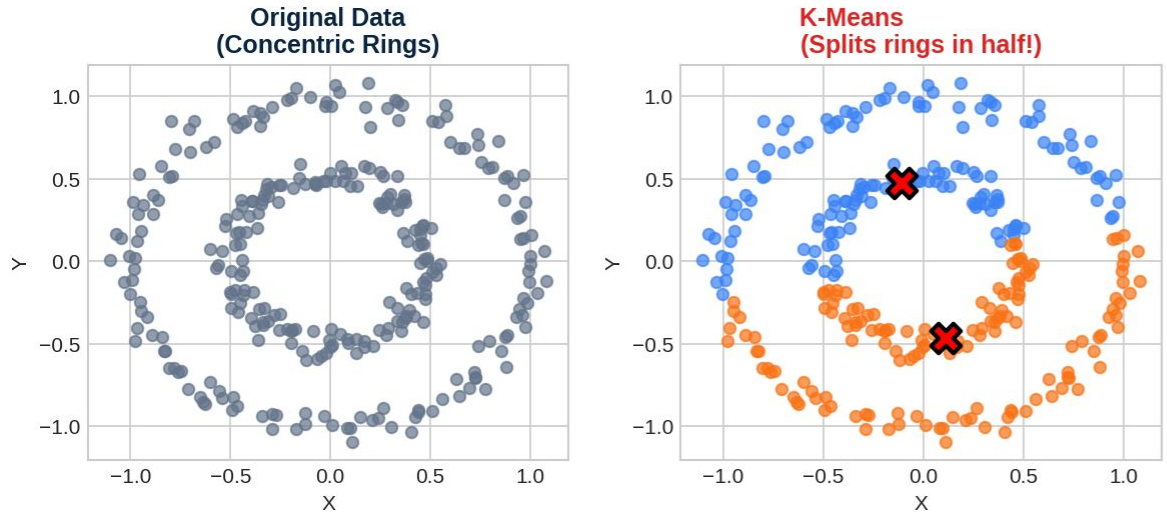
n = points, K = clusters, d = dimensions,
i = iterations

Key Properties

Fast : linear in n
Guaranteed to converge

Why K-Means Struggles

- Assumes spherical clusters only: Real data often has irregular shapes
- Must choose K upfront: How do we know how many clusters exist?
- Sensitive to outliers: One outlier can distort the centroid.
- Can't handle non-spherical data: Splits ring/moon shapes incorrectly!

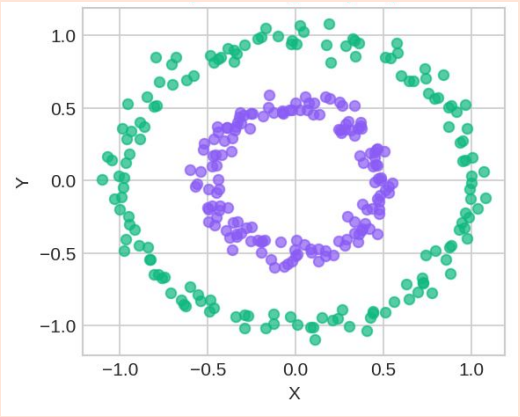


Solution: DBSCAN

Finds clusters of ANY shape
No need to specify K
Handles outliers as 'noise'

What is DBSCAN?

- Density-Based Spatial Clustering of Applications with Noise
- An unsupervised learning algorithm that clusters data into groups based on the density of data points
- Discovers arbitrary-shaped clusters and identifies outliers
- Does not require specifying number of clusters



Hyperparameters

- ε (epsilon): maximum distance between two points to be considered "neighbors"
- minPts: minimum points required to form a dense region

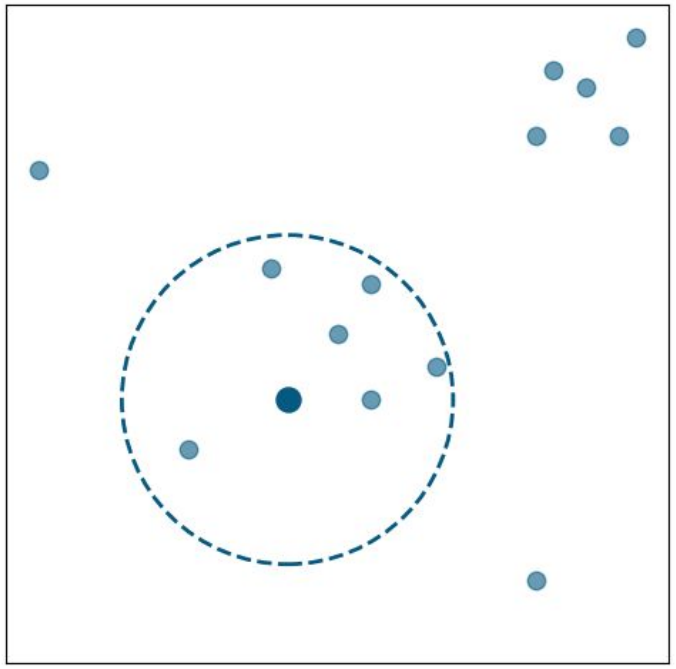
Key Terms

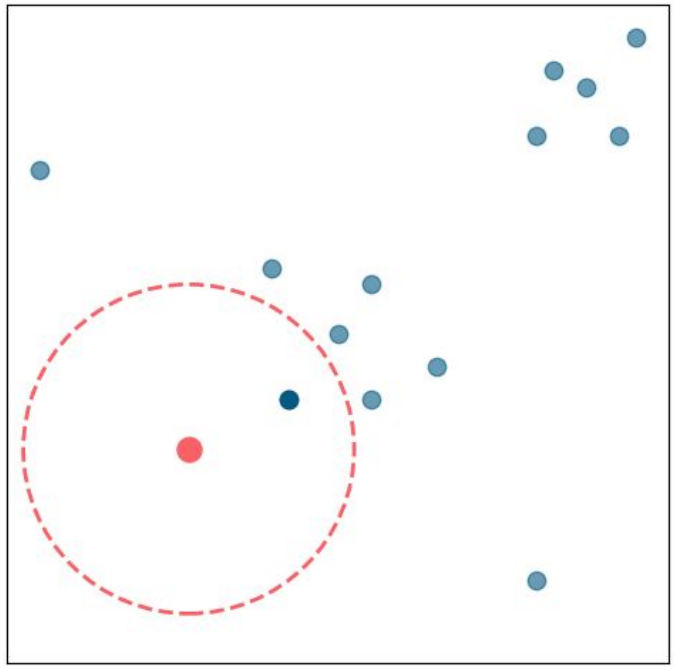
- ε-neighborhood: region around a point with a radius of ε

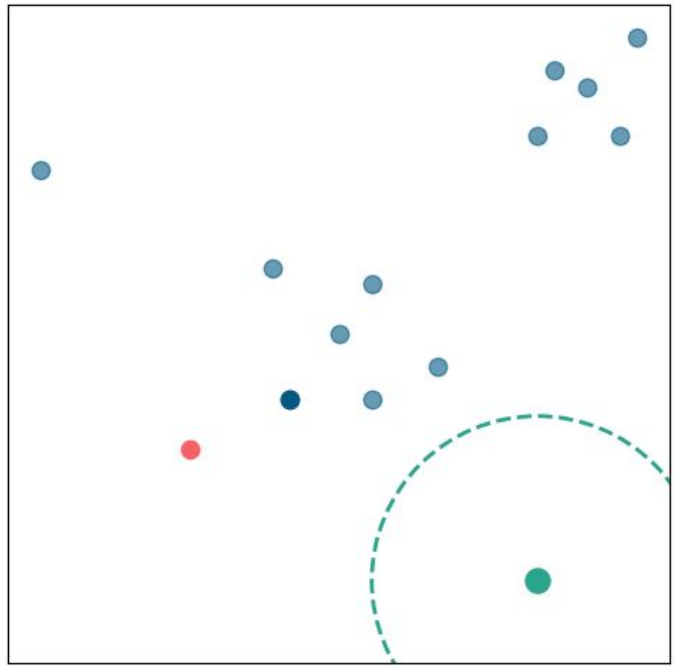
DBSCAN: Point Types

ε = 1
minPts = 5

Core Point	Border Point	Noise Point
Has at least minPts in its ε-neighborhood	Not a core point, but within ε of a core point	Not a core point or a border point







DBSCAN: How it Works

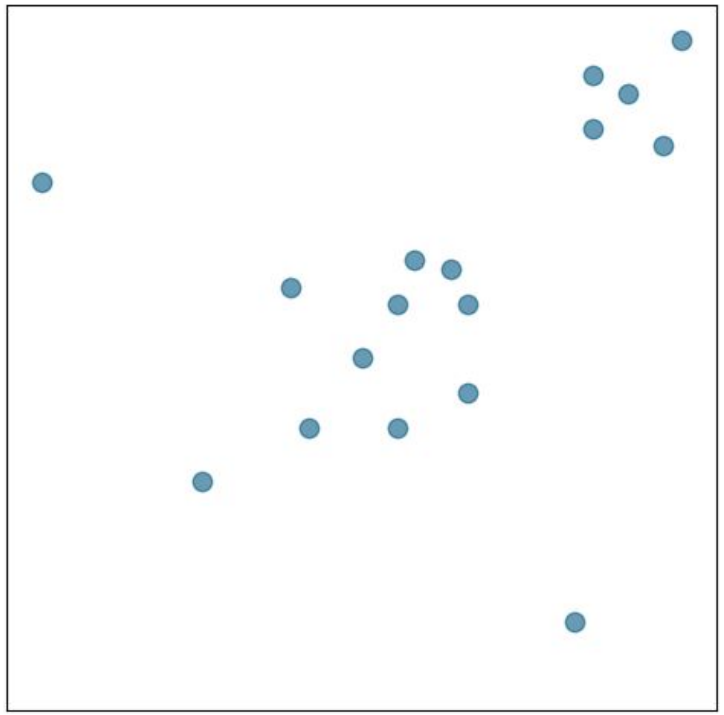
ε = 1
minPts = 5

Step 1: Point Selection
Select an unvisited point.

Step 2: Point Classification
Core point: create a new cluster.
Not a core point: mark as noise (tentatively).

Step 3: Cluster Expansion (Core Pts Only)
Add neighbors to the cluster.
For each neighbor: if it's a core point, recursively add its neighbors.

Step 4: Repeat
Repeat until all points are visited.



DBSCAN: How it Works

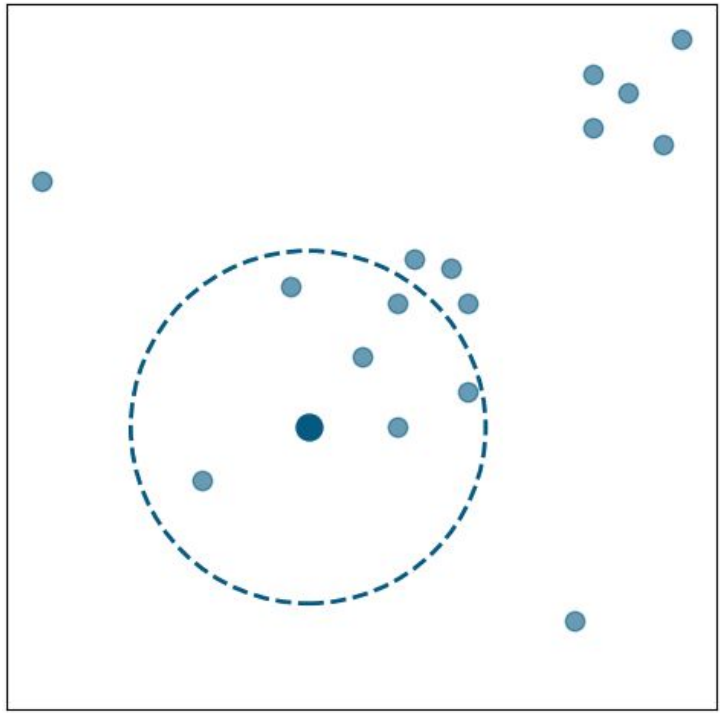
ε = 1
minPts = 5

Step 1: Point Selection
Select an unvisited point.

Step 2: Point Classification
Core point: create a new cluster.
Not a core point: mark as noise (tentatively).

Step 3: Cluster Expansion (Core Pts Only)
Add neighbors to the cluster.
For each neighbor: if it's a core point, recursively add its neighbors.

Step 4: Repeat
Repeat until all points are visited.



Step 1-2

DBSCAN: How it Works

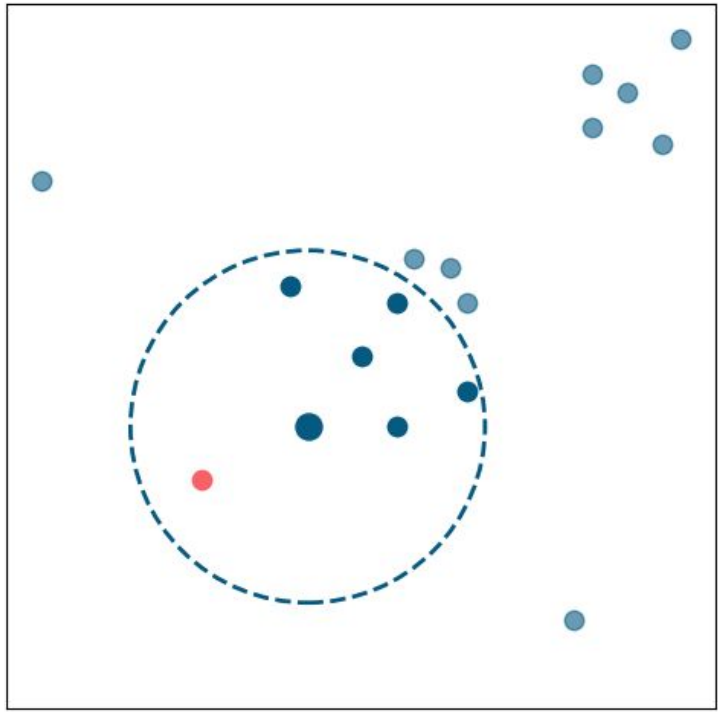
ε = 1
minPts = 5

Step 1: Point Selection
Select an unvisited point.

Step 2: Point Classification
Core point: create a new cluster.
Not a core point: mark as noise (tentatively).

Step 3: Cluster Expansion (Core Pts Only)
Add neighbors to the cluster.
For each neighbor: if it's a core point, recursively add its neighbors.

Step 4: Repeat
Repeat until all points are visited.



Step 3

DBSCAN: How it Works

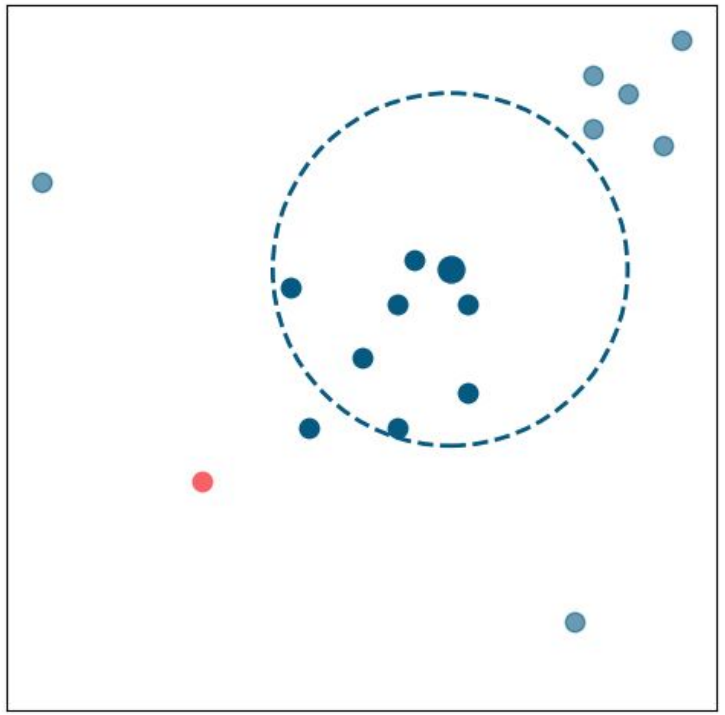
ε = 1
minPts = 5

Step 1: Point Selection
Select an unvisited point.

Step 2: Point Classification
Core point: create a new cluster.
Not a core point: mark as noise (tentatively).

Step 3: Cluster Expansion (Core Pts Only)
Add neighbors to the cluster.
For each neighbor: if it's a core point, recursively add its neighbors.

Step 4: Repeat
Repeat until all points are visited.



Step 3

DBSCAN: How it Works

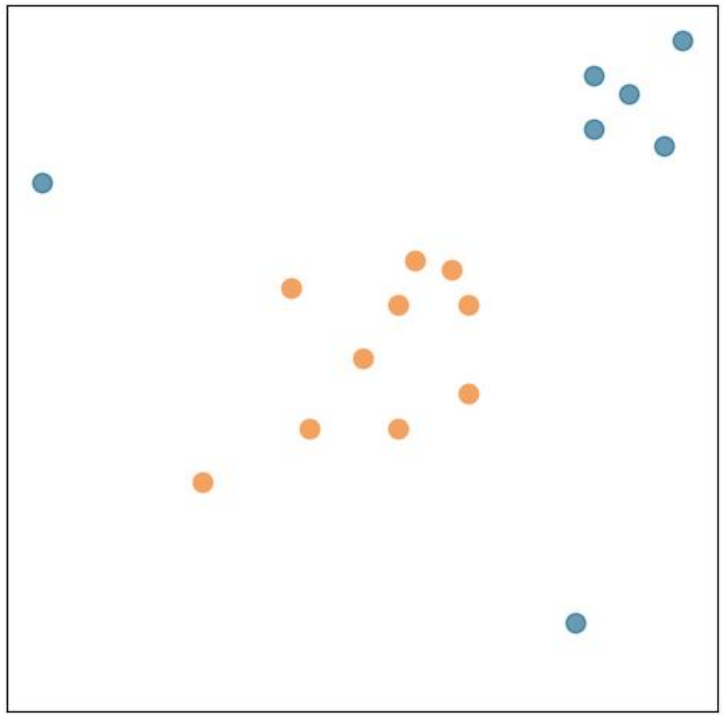
ε = 1
minPts = 5

Step 1: Point Selection
Select an unvisited point.

Step 2: Point Classification
Core point: create a new cluster.
Not a core point: mark as noise (tentatively).

Step 3: Cluster Expansion (Core Pts Only)
Add neighbors to the cluster.
For each neighbor: if it's a core point, recursively add its neighbors.

Step 4: Repeat
Repeat until all points are visited.



Step 4

DBSCAN: How it Works

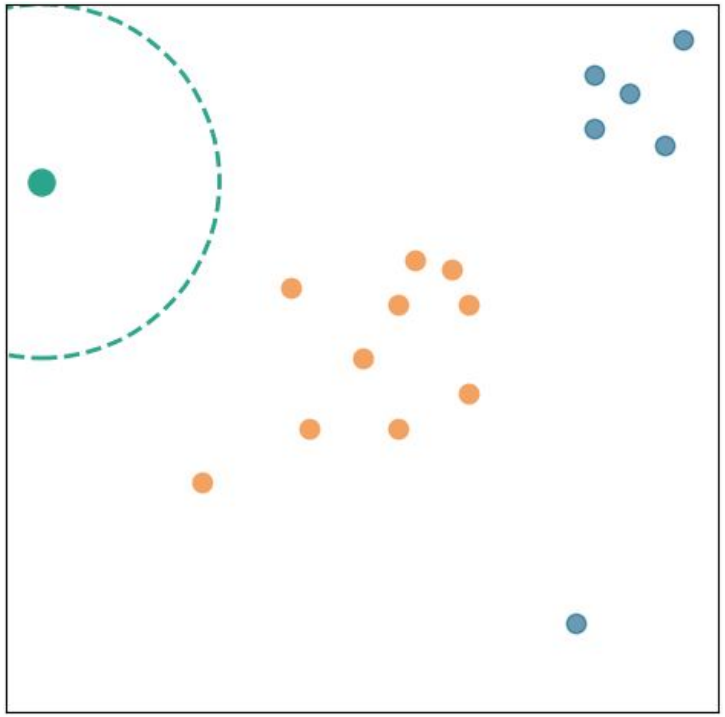
ε = 1
minPts = 5

Step 1: Point Selection
Select an unvisited point.

Step 2: Point Classification
Core point: create a new cluster.
Not a core point: mark as noise (tentatively).

Step 3: Cluster Expansion (Core Pts Only)
Add neighbors to the cluster.
For each neighbor: if it's a core point, recursively add its neighbors.

Step 4: Repeat
Repeat until all points are visited.



Steps 1-2

DBSCAN: How it Works

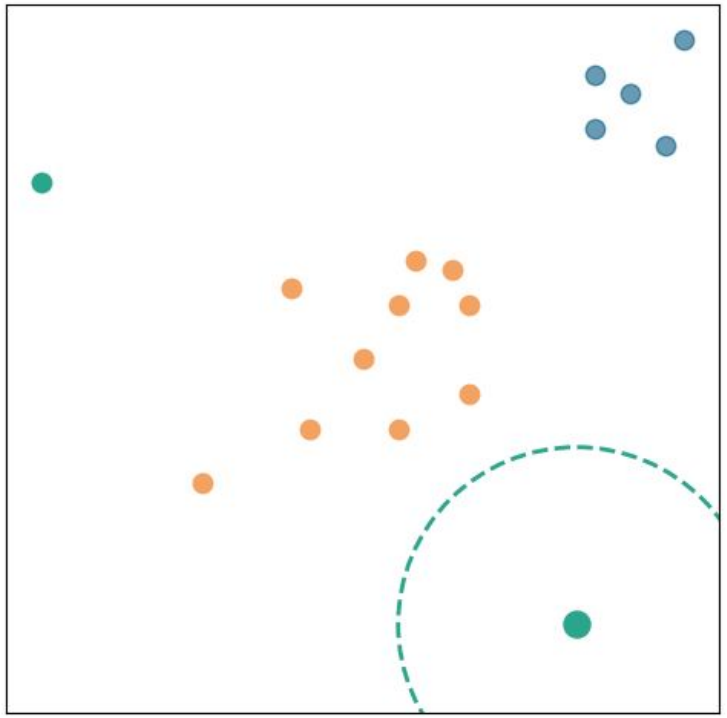
ε = 1
minPts = 5

Step 1: Point Selection
Select an unvisited point.

Step 2: Point Classification
Core point: create a new cluster.
Not a core point: mark as noise (tentatively).

Step 3: Cluster Expansion (Core Pts Only)
Add neighbors to the cluster.
For each neighbor: if it's a core point, recursively add its neighbors.

Step 4: Repeat
Repeat until all points are visited.



Steps 1-2

DBSCAN: How it Works

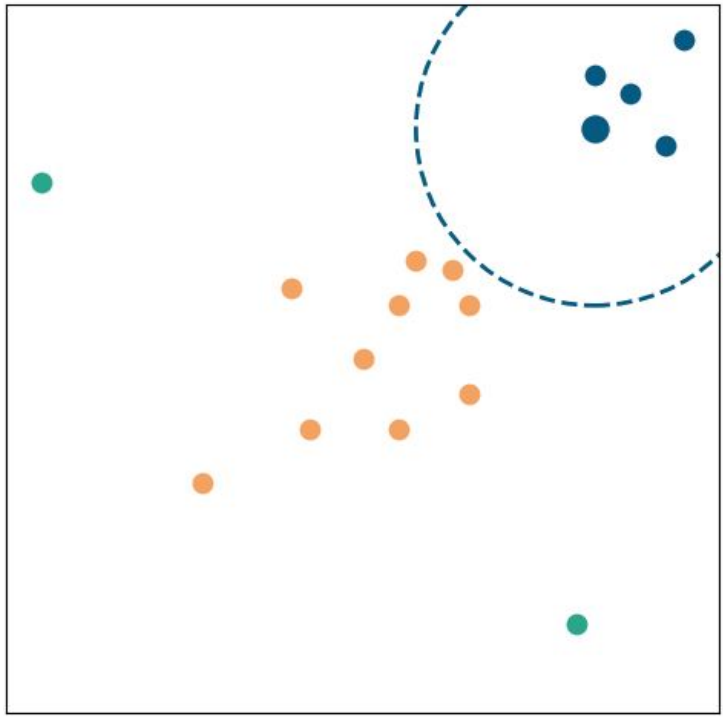
ε = 1
minPts = 5

Step 1: Point Selection
Select an unvisited point.

Step 2: Point Classification
Core point: create a new cluster.
Not a core point: mark as noise (tentatively).

Step 3: Cluster Expansion (Core Pts Only)
Add neighbors to the cluster.
For each neighbor: if it's a core point, recursively add its neighbors.

Step 4: Repeat
Repeat until all points are visited.



Steps 1-3

DBSCAN: How it Works

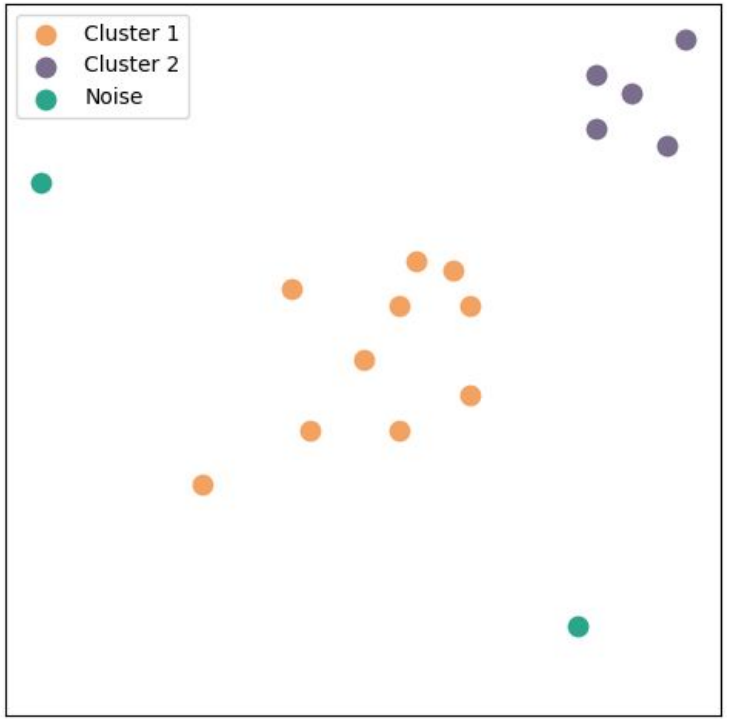
ε = 1
minPts = 5

Step 1: Point Selection
Select an unvisited point.

Step 2: Point Classification
Core point: create a new cluster.
Not a core point: mark as noise (tentatively).

Step 3: Cluster Expansion (Core Pts Only)
Add neighbors to the cluster.
For each neighbor: if it's a core point, recursively add its neighbors.

Step 4: Repeat
Repeat until all points are visited.



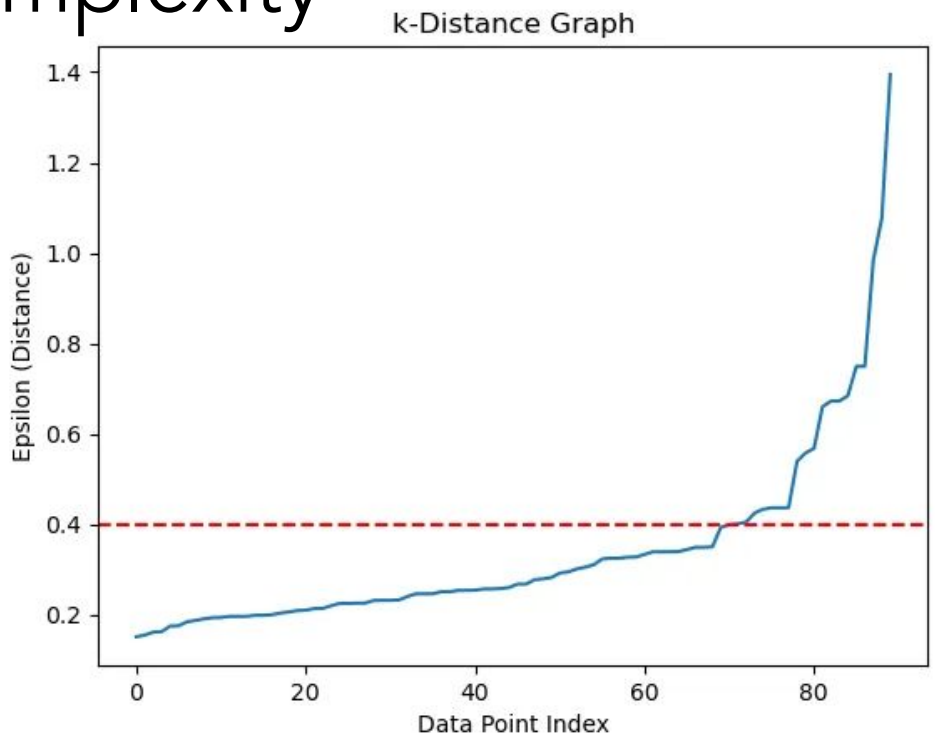
Choosing Hyperparameters & Complexity

ε (epsilon)

- Too small → most points become noise
- Too large → clusters merge together
- k-distance graph
  - plots distance from every point to its k-th nearest neighbor (k = minPts - 1)
  - elbow point is optimal ε

minPts

- Too small → many small clusters
- Too large → few large clusters
- Common methods:
  - Lower bound: minPts ≥ D + 1
  - Start with minPts = 2 x D



Time Complexity

- With spatial index: O(n log n)
- No index or in high dimensions: $O(n^2)$

Space Complexity

- O(n)

Applications

K-means

Image Compression

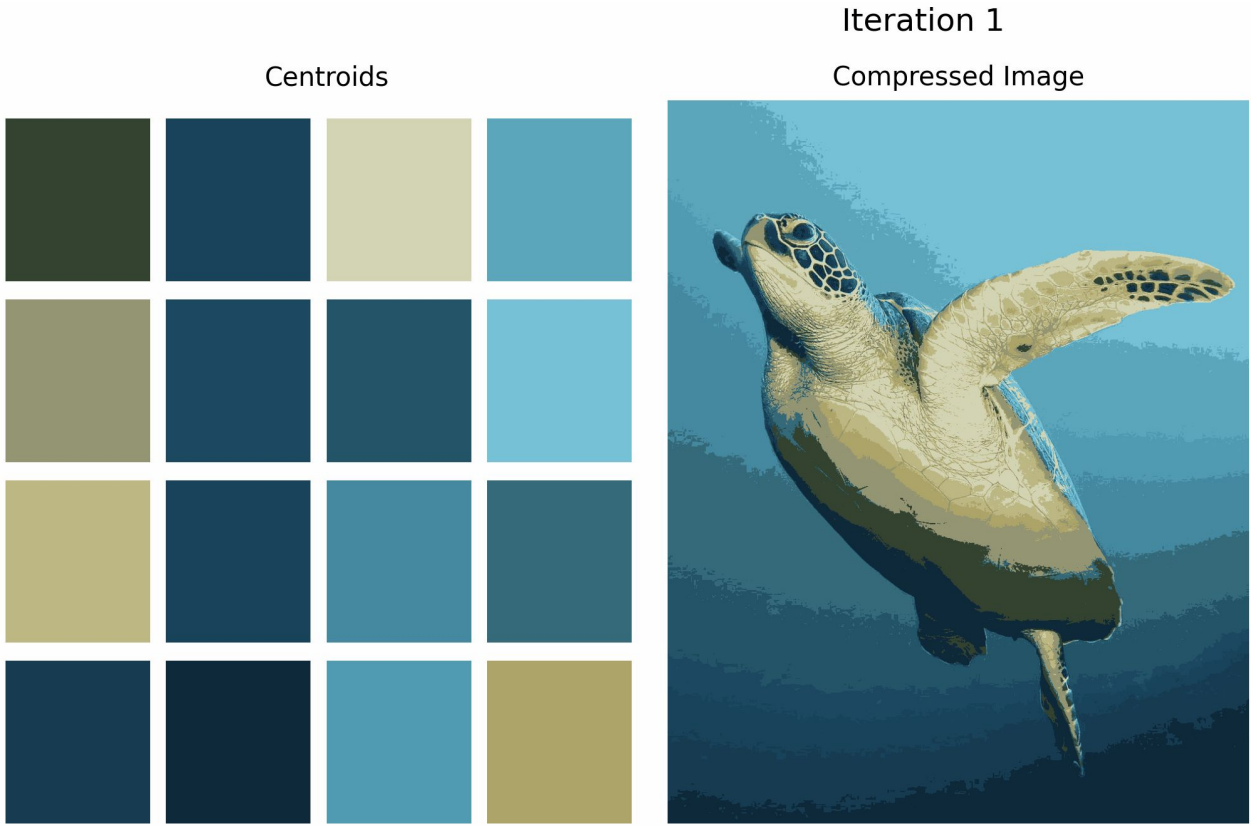
- Reduce colors by replacing similar pixel colors with a single color (centroid)

Customer Segmentation (Marketing)

- Group customers by purchase behavior, demographics, or website activity

Document Classification (NLP)

- Groups documents by word frequency or similar themes



DBSCAN

Medical Imaging Segmentation

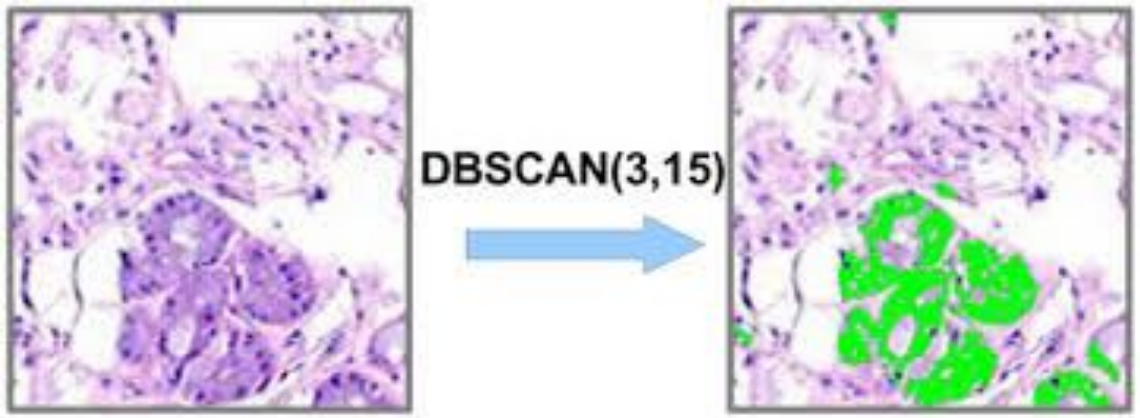
- Detect cancerous cells based on the density of tumor masses

Geospatial Analysis

- Identify high-density spatial regions

Network Intrusion Detection

- Flags unusual behavior as noise points



Implementation

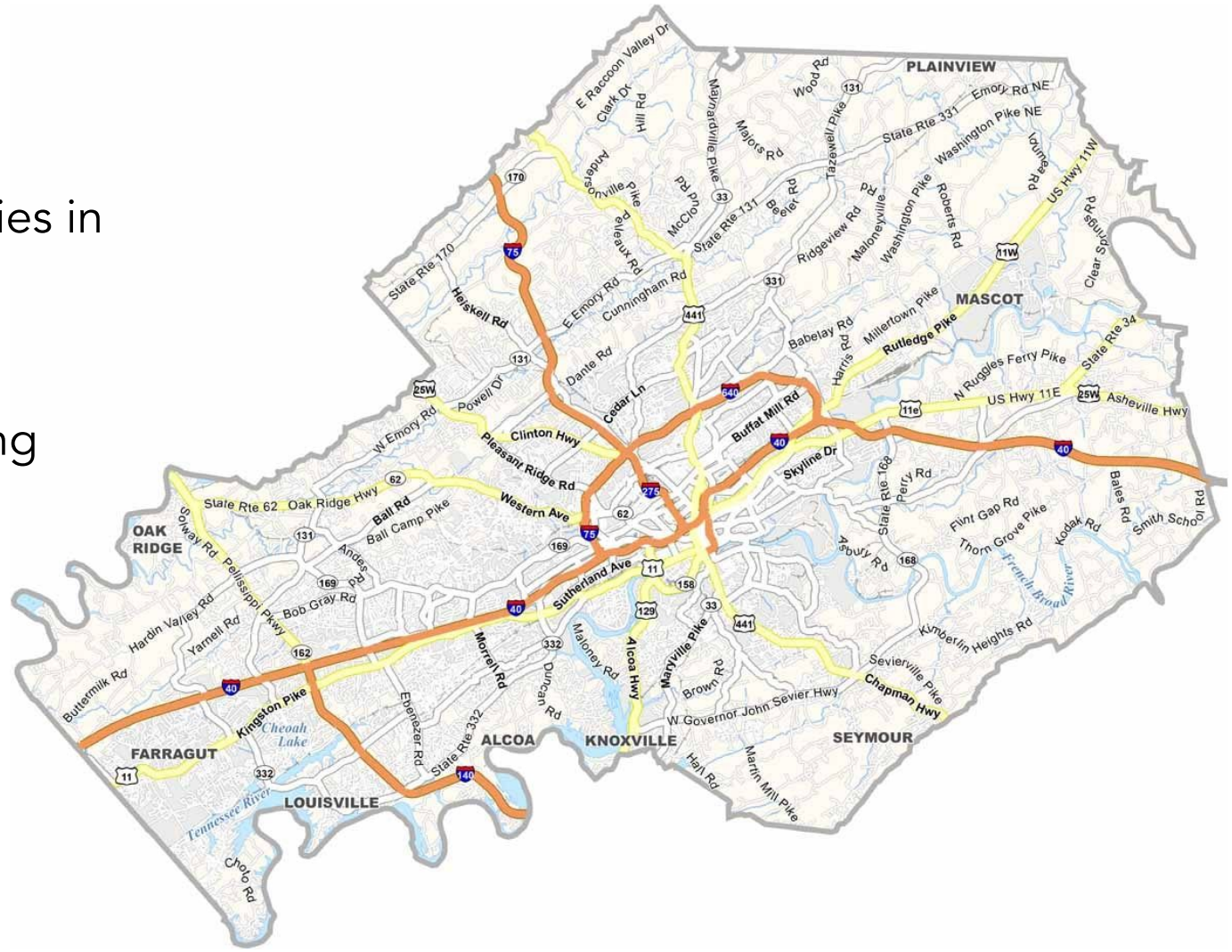
Objective

Goal

- Perform geospatial clustering of amenities in Knoxville, TN
- Identify high-density activity areas
- Compare K-Means vs DBSCAN clustering

Potential Applications

- Urban planning and growth analysis
- Infrastructure development
- Resource allocation



Dataset

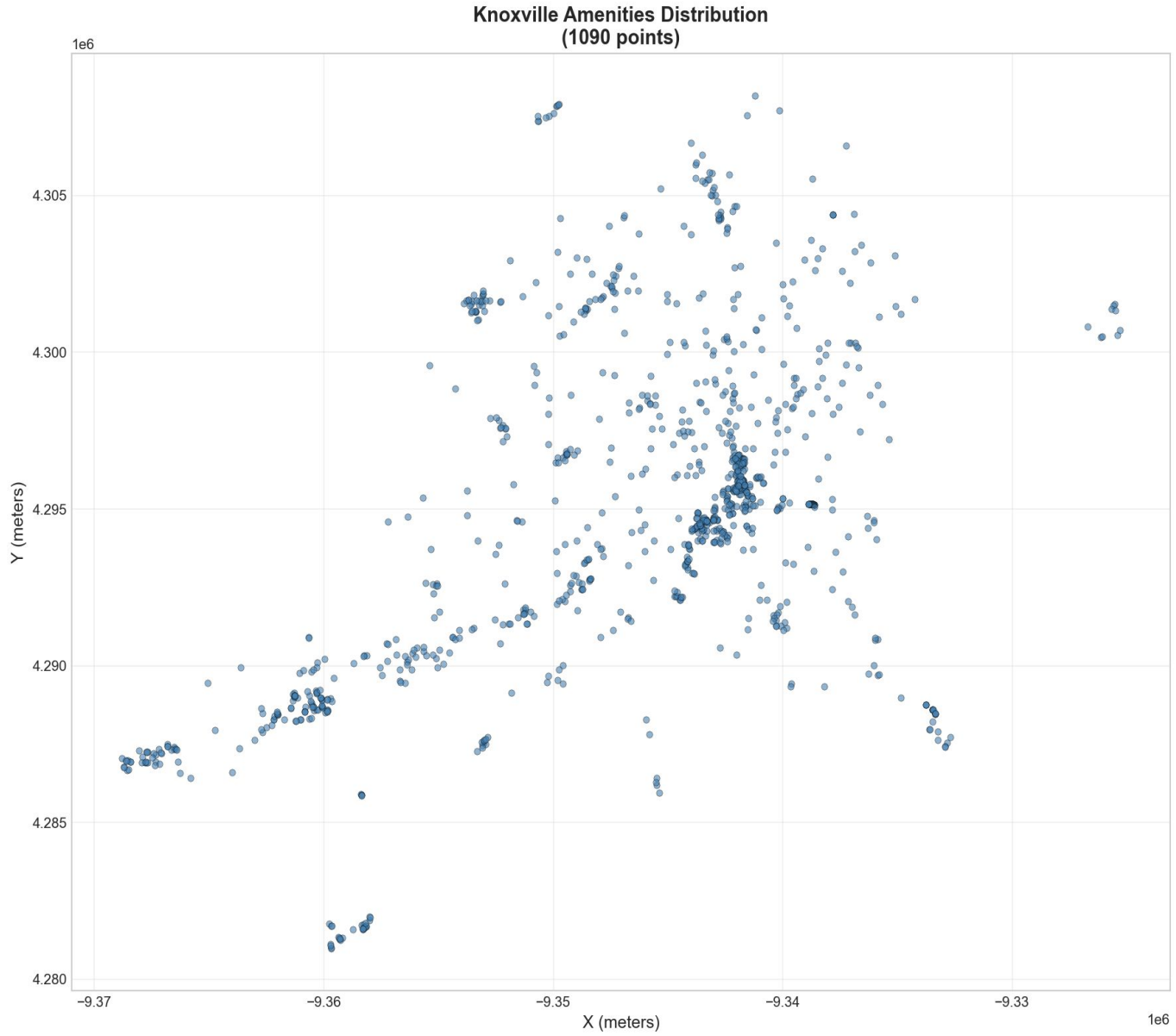
- OpenStreetMap: open, community-maintained map of the whole world
- Python package: OSMnx

Features Used

- Latitude and longitude coordinates
  - converted to projected coordinates (in meters)
- Points of Interest (Amenities)
  - businesses, facilities, or services that people use

Amenity Examples

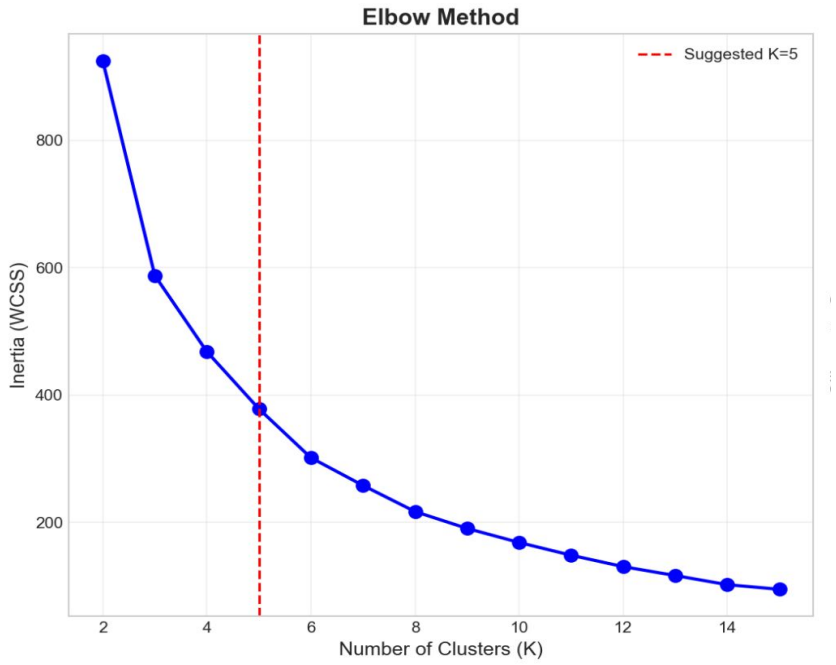
- Restaurants, Places of Worship, Gas Stations, Schools, Hospitals



Methods to Evaluate & Choose K

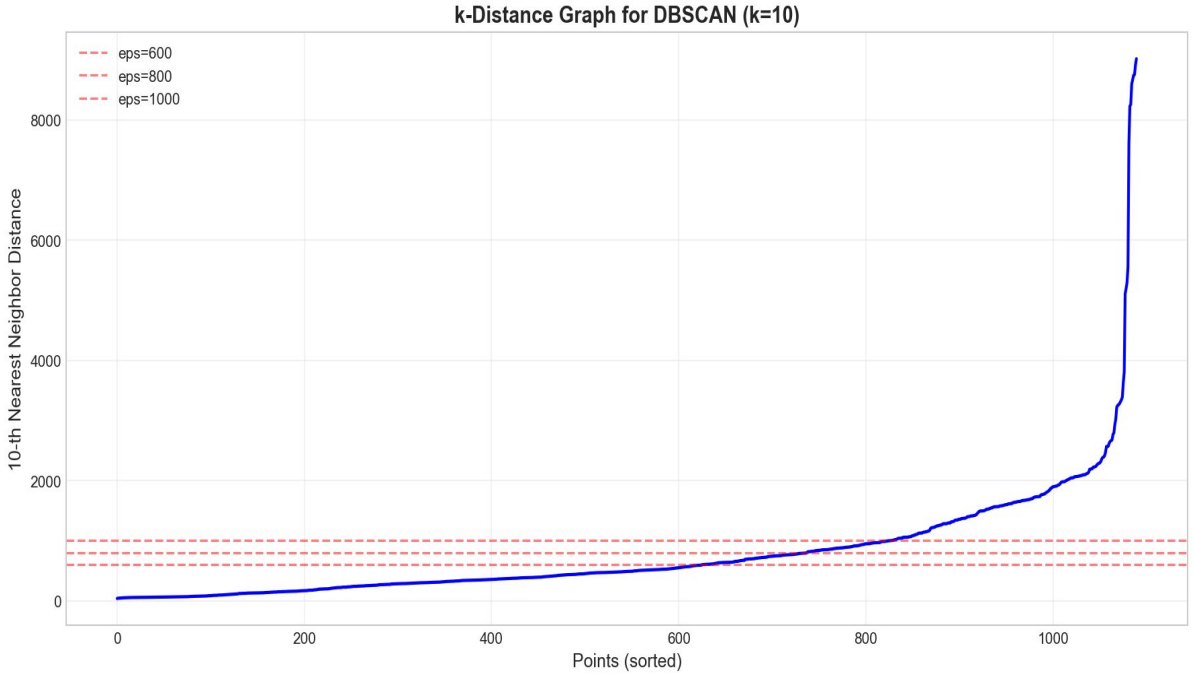
K-MEANS

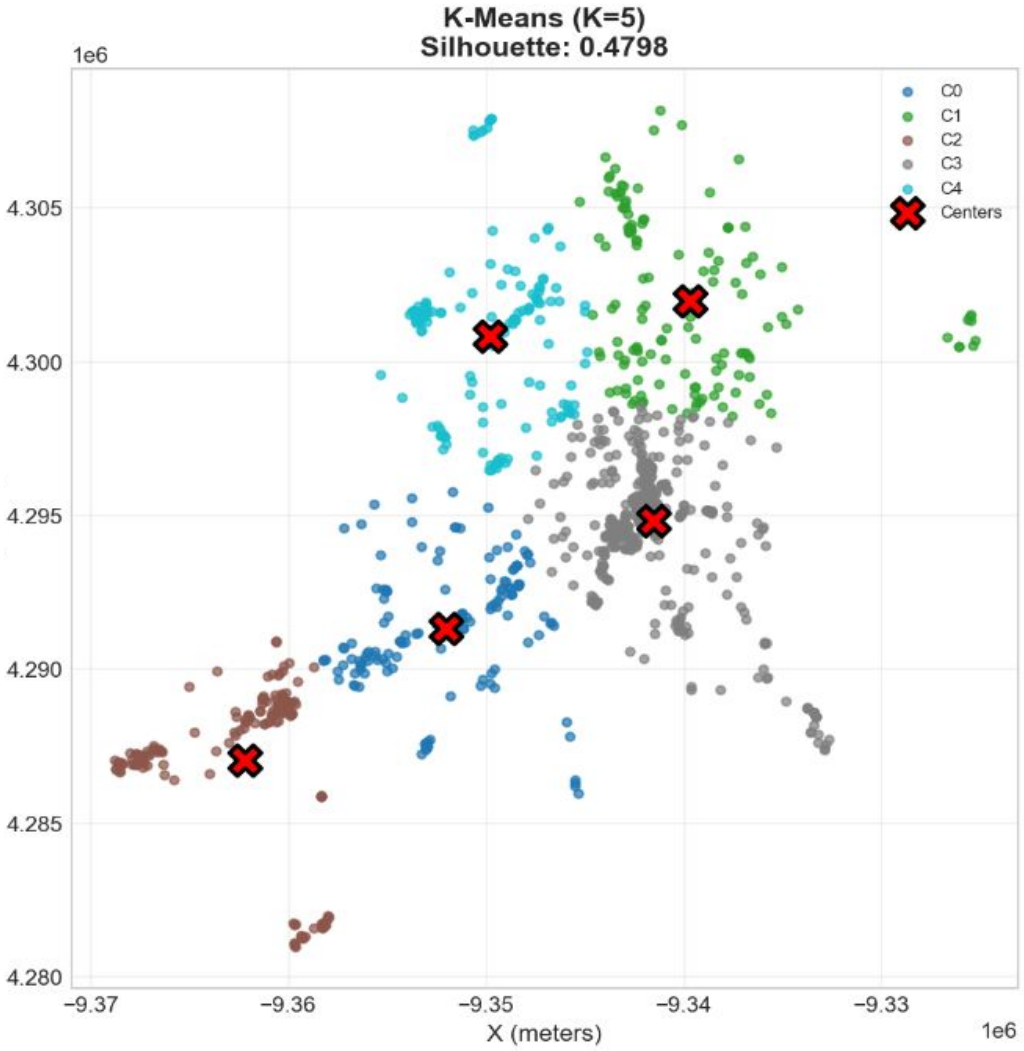
Parameter: K = 5 (Elbow Method)
Library: scikit-learn
Function: KMeans(n_clusters=5)

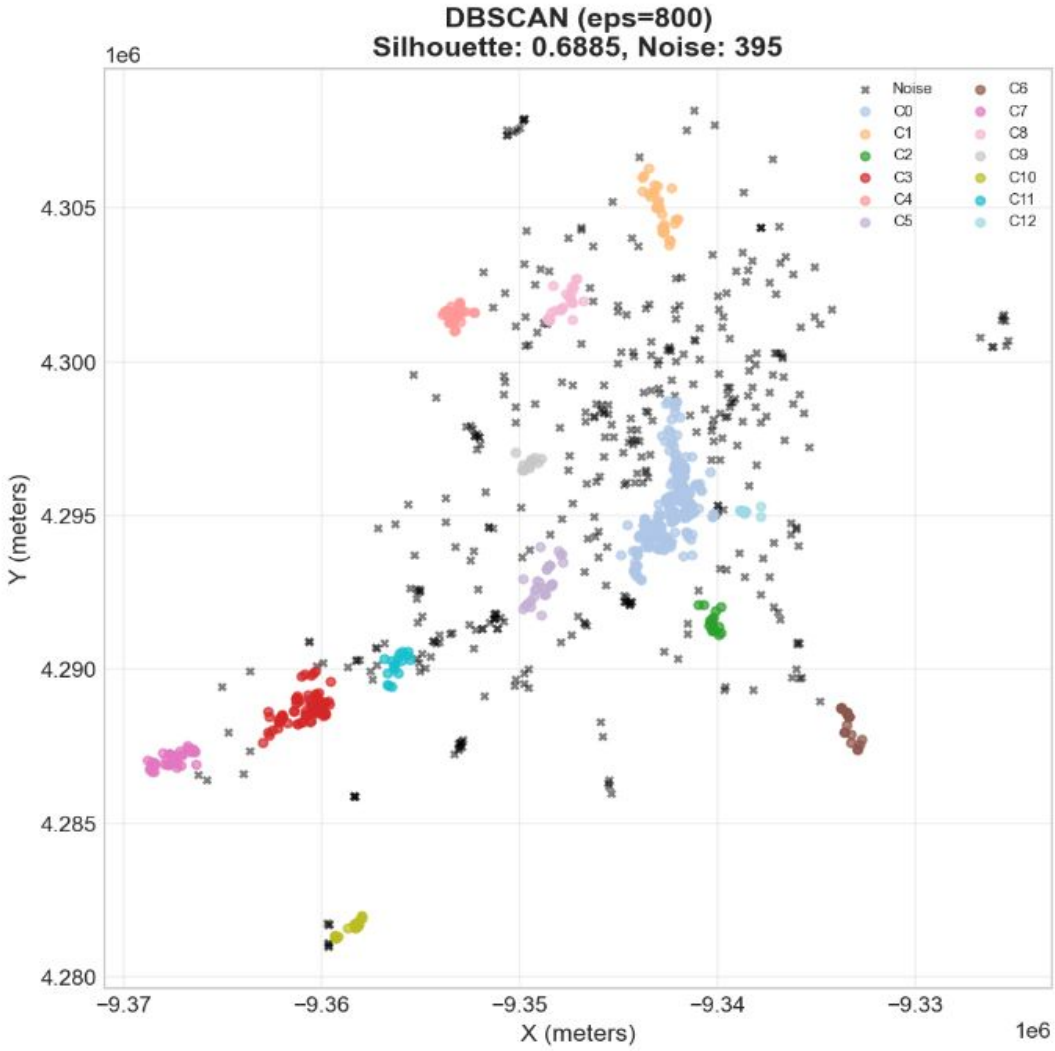


DBSCAN

Parameters: ε = 800m, minPts = 15
Method 1: k-Distance Graph → ε
Method 2: Grid Search → minPts
Library: scikit-learn
Function: DBSCAN(eps=800, min_samples=15)







- K-Means: Forces ALL 1090 points into 5 clusters (no noise detection)
- DBSCAN: Finds 13 natural clusters + identifies 395 isolated amenities as noise
- Silhouette Score: K-Means = 0.48, DBSCAN = 0.69 (44% better!)

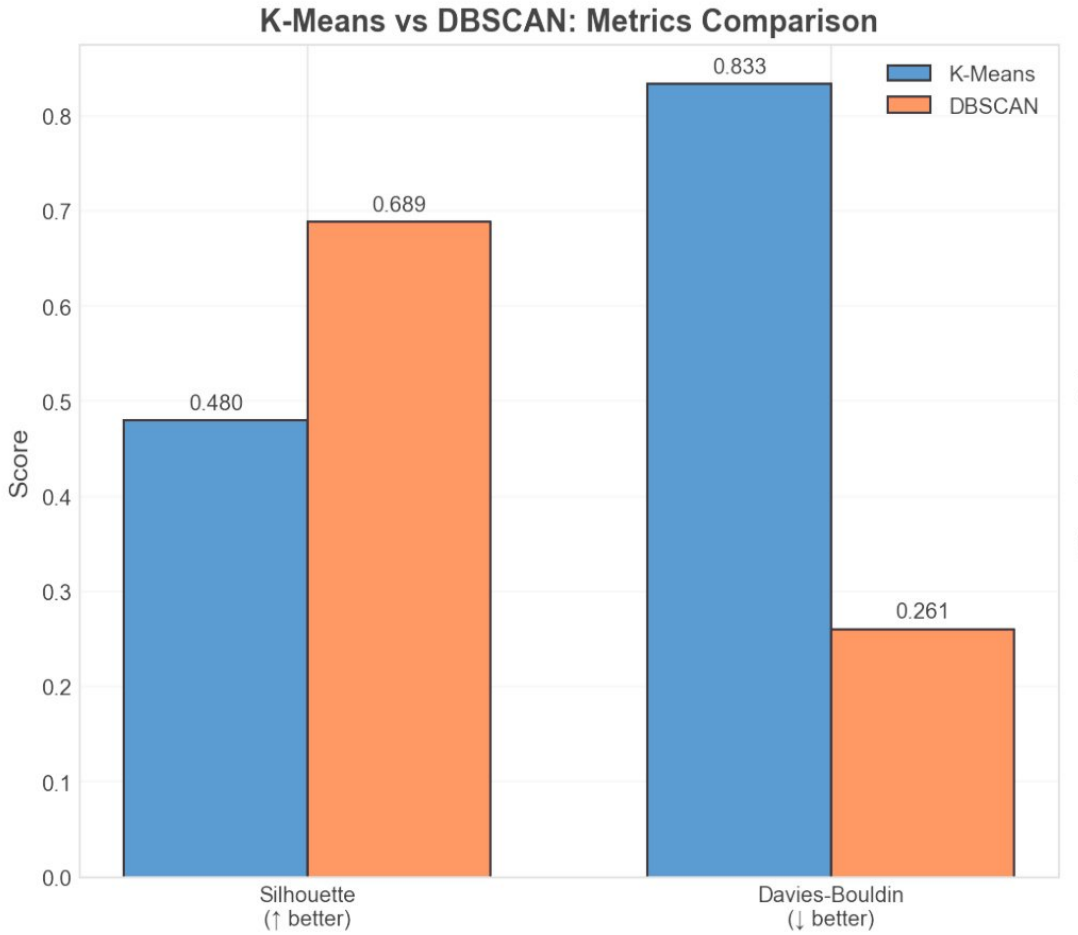
Evaluation: Methods & Results

Evaluation Methods

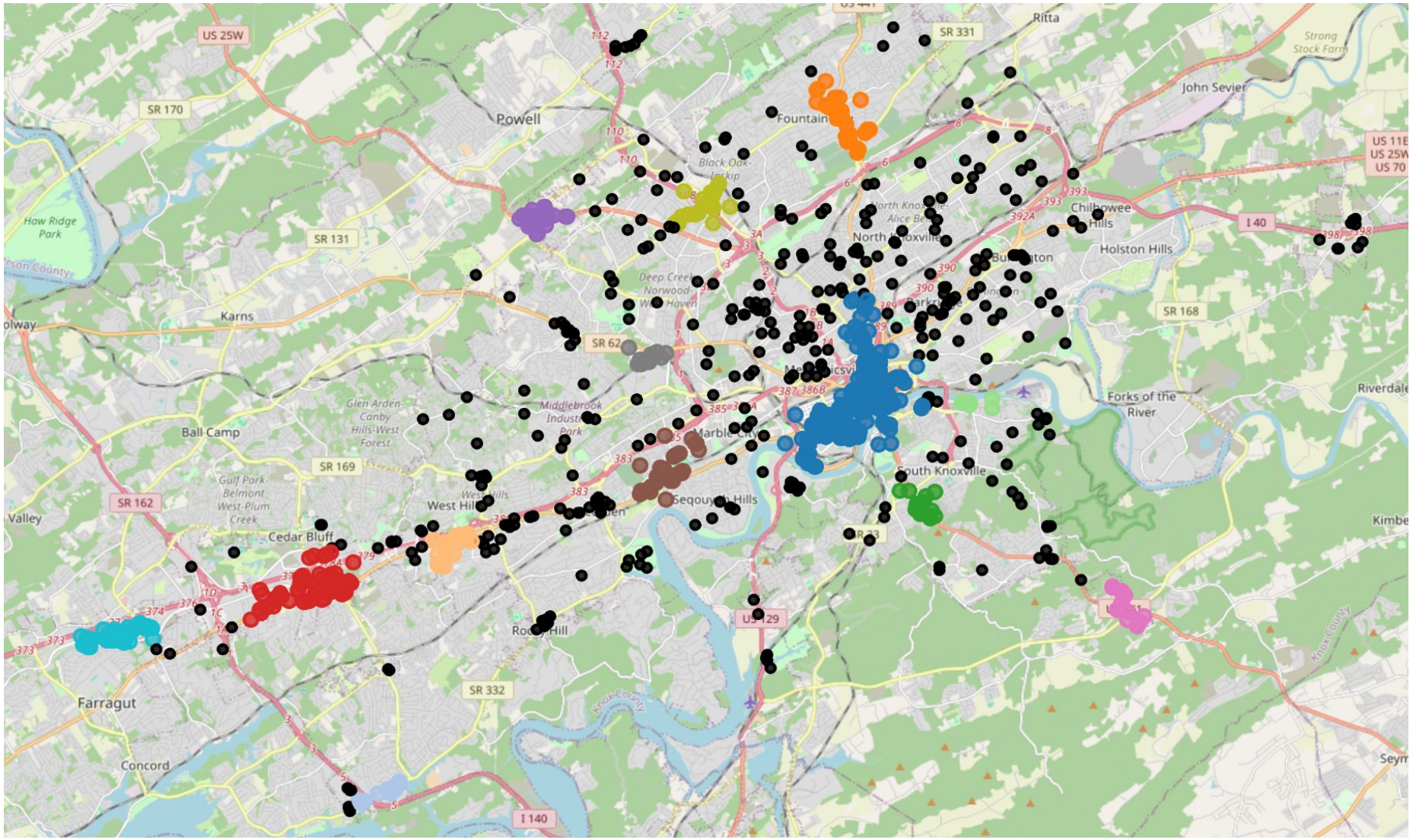
Method	What It Measures	Goal
Silhouette	How close points are to own cluster vs other clusters	Max (↑)
Davies-B	Average similarity/overlap between clusters	Min (↓)

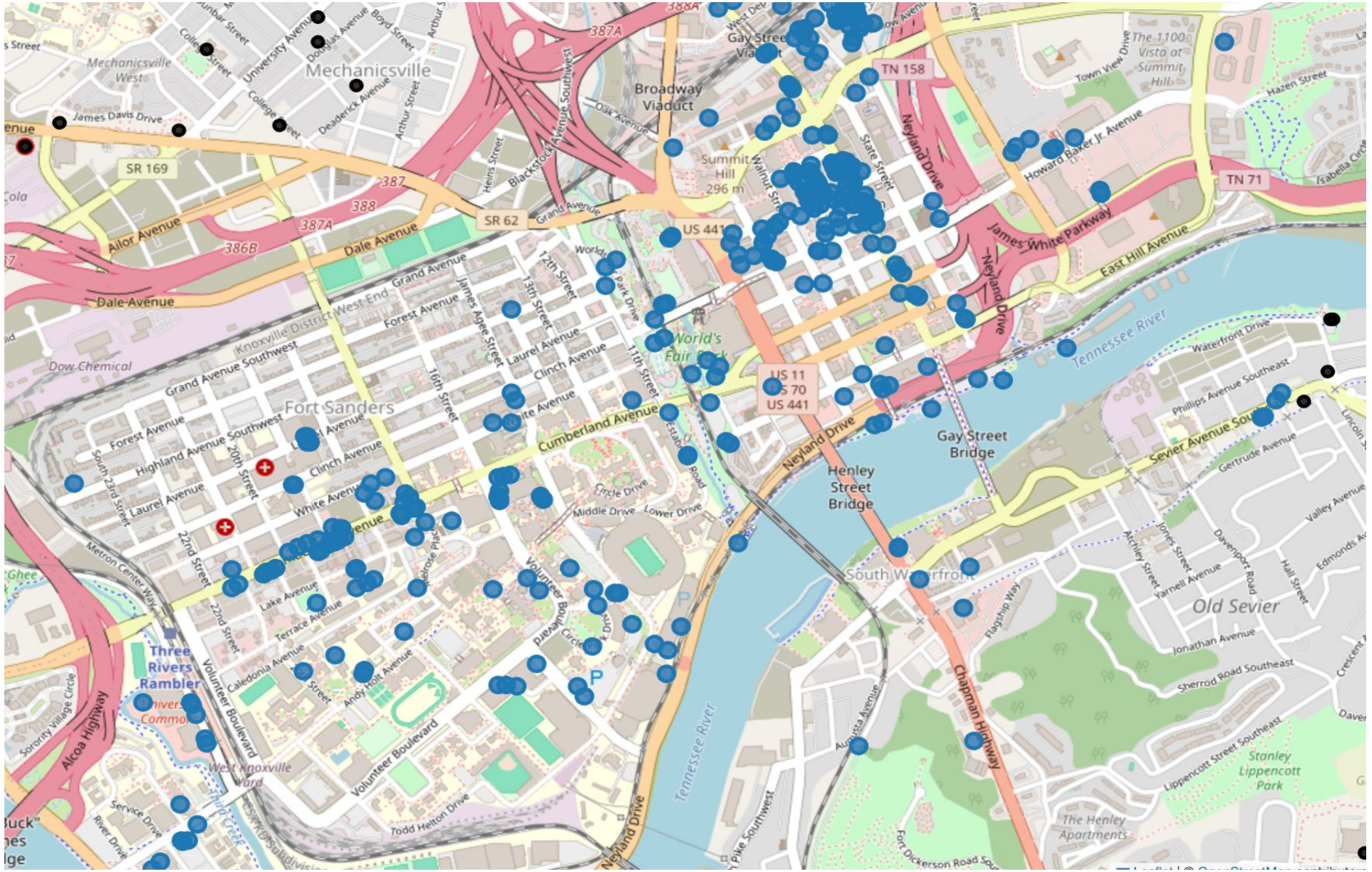
Our Results

Metric	K-Means	DBSCAN	Outperforms
Silhouette (↑)	0.4798	0.6885	DBSCAN
Davies-B (↓)	0.8332	0.2605	DBSCAN



DBSCAN Map


US 25W
SR 170
Powell
SR 331
Ritta
John Sevier
Fountain
Chilhowee Hills
Holston Hills
I 40
US 11E
US 25W
US 70
SR 131
Karns
SR 62
SR 168
Forks of the River
Riverdale
Ball Camp
SR 169
SR 162
Valley
Cedar Bluff
West Hills
Sequoyah Hills
South Knoxville
Kimbe
US 129
Farragut
Concord
SR 332
I 140
Knox County
Haw Ridge Park
Glen Arden Canby Hills-West Forest
Middlebrook Industrial Park
Gulf Park-Belmont West-Plum Creek


Mechanicsville
Broadway Viaduct
Summit Hill 296 m
Fort Sanders
Dale Avenue
Cumberland Avenue
Neyland Drive
Gay Street Bridge
Henley Street Bridge
Tennessee River
Old Sevier
Three Rivers Rambler
Volunteer Boulevard
Alcoa Highway
Chapman Highway
James White Parkway
East Hill Avenue
Howard Baker Jr. Avenue
Sevier Avenue
SR 169
SR 62
US 441
US 11
US 70
TN 158
TN 71
387
387A
388
386B
Dow Chemical
Stanley Lippencott Park
The Henley Apartments

Open Issues

- Choosing parameters remains difficult.

  Possible solution: Use methods such as the Elbow Method, Silhouette Score, or k-distance graph to guide parameter selection.

- Handling complex cluster structures is still challenging.

  Possible solution: Use improved variants such as Kernel K-means, OPTICS, or HDBSCAN to better handle irregular shapes and varying densities.

- Scaling to high-dimensional and large datasets remains hard.

  Possible solution: Apply dimensionality reduction like PCA or t-SNE before clustering, and use distributed or approximate clustering methods for large-scale data.

References

Boeing, G. (2025). Modeling and analyzing urban networks and amenities with OSMnx. Geographical Analysis, 57(4), 567–577. https://doi.org/10.1111/gean.70009

Lloyd, S. (1982). Least squares quantization in PCM. IEEE Transactions on Information Theory, 28(2), 129–137.

Forgy, E. W. (1965). Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. Biometrics.

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96) (pp. 226–231).

Sapovadia, D. (n.d.). K-means clustering. Medium. https://darrsheni-sapovadia26.medium.com/k-means-clustering-96711652a0e9

Clear and visual explanation of the K-means algorithm applied to image compression. (n.d.). Medium. https://medium.com/data-science/clear-and-visual-explanation-of-the-k-means-algorithm-applied-to-image-compression-b7fdc547e410

Clustering like a pro: A beginner's guide to DBSCAN. (n.d.). Medium. https://medium.com/@sachinsoni600517/clustering-like-a-pro-a-beginners-guide-to-dbscan-6c8274c362c4

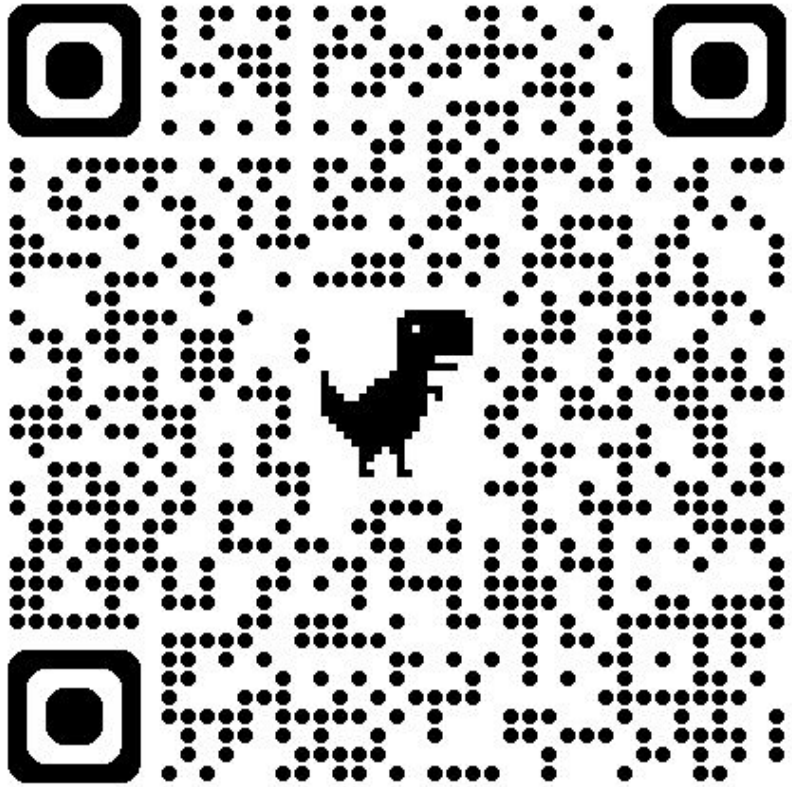
DBSCAN explained: Density-based clustering. (n.d.). Built In. https://builtin.com/articles/dbscan

Automatic delineation of malignancy in histopathological head and neck slides. (n.d.). ResearchGate. https://www.researchgate.net/publication/5799677_Automatic_Delineation_of_Malignancy_in_Histopathological_Head_and_Neck_Slides

Discussion

Test Questions

1. Who introduced the term K-Means?
2. What is the time complexity of K-Means?
3. What are the hyperparameters for DBSCAN?



https://forms.gle/JqkVaA1xeXbaXREu8