AI & Geospatial Analytics in Tourism

Lab:Spatial Clustering : Algorithms, Parameters & Implementation

Course: AI and Tourism – MIT-AI @ Gandaki University

Bidur Devkota, PhD
GCES Pokhara

January 6, 2026

Spatial Clustering Overview

Objective: Group geographical points into meaningful clusters

Key Algorithms Implemented:

- K-Means - Partition-based clustering
- DBSCAN - Density-based clustering
- HDBSCAN - Hierarchical density-based clustering

Data: Restaurant locations in Lakeside, Nepal

- Latitude/Longitude coordinates
- Preprocessed for Nepal geographical bounds

K-Means Algorithm

Formula:

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

where:

- k = number of clusters
- C_i = set of points in cluster i
- μ_i = centroid of cluster i

Algorithm Steps:

1. Initialize k centroids randomly
2. Assign points to nearest centroid
3. Recalculate centroids as mean of assigned points
4. Repeat until convergence

Use: When number of clusters is known/specified

K-Means Implementation Details

Parameters:

- n_clusters = 5 (specified)
- random_state = 42
- n_init = 10

Parameter Determination:

- *Manual specification* or
- *Auto-selection* via elbow method

Elbow Method: (_find_optimal_k())

- Plot inertia vs. number of clusters
- Choose point where inertia decrease slows

Implementation Features:

- Auto-k selection using elbow + silhouette
- Silhouette score for validation
- Metrics calculation:
  - Silhouette score
  - Calinski-Harabasz
  - Davies-Bouldin

DBSCAN Algorithm

Density-Based Concepts:

- Core point: $\geq$ min_samples within ε radius
- Border point: Within ε of core point
- Noise point: Neither core nor border

Algorithm:

1. Find all core points
2. Form clusters from density-connected core points
3. Assign border points to nearest cluster
4. Label remaining as noise

Formula:

$$\text{Cluster} = \{p \mid \text{density\_reachable}(p, q) \text{ for some core point } q\}$$

Use: Arbitrary shapes, noise detection

DBSCAN Implementation Details

Parameters:

- eps = 0.02 (specified)
- min_samples = 5
- metric = 'euclidean'

Parameter Determination:

- *Auto-selection* via k-distance graph
- Uses KneeLocator for optimal ε (eps)

k-Distance Graph Method:

1. Compute distances to k-nearest neighbors
2. Sort distances for all points
3. Find "knee point" (elbow) in curve
4. Set ε at knee point

Default Calculation:

- min_samples = max(5, min(10, n_samples/20))
- ε = 75th percentile of k-distances

HDBSCAN Algorithm

Hierarchical DBSCAN:

- Extends DBSCAN to varying densities
- Builds cluster hierarchy
- Extracts flat clusters based on stability

Algorithm:

1. Build mutual reachability graph
2. Construct minimum spanning tree
3. Build cluster hierarchy
4. Condense tree based on cluster stability
5. Extract flat clusters

Key Features:

- No need for ε parameter
- Handles varying cluster densities
- Provides cluster stability scores

Use: Complex density patterns, hierarchical relationships

HDBSCAN Implementation Details

Parameters:

- min_cluster_size = 10
- min_samples = 5
- metric = 'euclidean'
- cluster_selection_method = 'eom'

Parameter Determination:

- Default: min_cluster_size = max(5, n_samples/50)
- min_samples = max(2, min_cluster_size/2)

EOM vs Leaf:

- EOM (Excess of Mass):
  - Prefers more stable clusters
  - Less sensitive to outliers
- Leaf:
  - Creates more smaller clusters
  - Captures fine-grained structure

Output Features:

- Cluster probabilities
- Noise detection
- Hierarchical relationships

Complete Parameter Summary

Algorithm	Parameter	Value & Determination
K-Means	n_clusters	5 (specified) or auto via elbow method
	random_state	42 (reproducibility)
DBSCAN	eps	0.02 (specified) or auto via k-distance
	min_samples	5 (specified) or 1% of data
HDBSCAN	min_cluster_size	10 (specified) or 2% of data
	min_samples	5 (specified) or half of min_cluster_size

Auto-selection Methods:

- K-Means: Elbow method + Silhouette score
- DBSCAN: k-distance graph with KneeLocator
- HDBSCAN: Data size-based heuristics

Clustering Evaluation Metrics

Silhouette Score:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

- $a(i)$ = average distance to points in same cluster
- $b(i)$ = average distance to points in nearest other cluster
- Higher is better (-1 to 1)

Interpretation Guidelines:

- > 0.70: Strong structure
- $0.51 - 0.70$: Reasonable structure
- $0.26 - 0.50$: Weak structure
- ≤ 0.25: No substantial structure

Additional Metrics:

- Calinski-Harabasz: Ratio of between to within cluster dispersion (higher better)
- Davies-Bouldin: Average similarity between clusters (lower better)

Code Output & Results

Main Outputs:

1. Visualizations:
   - Cluster plots for each algorithm
   - Parameter selection plots (elbow, k-distance)
   - Metrics comparison charts
2. Data Files:
   - 'restaurants_clustered.csv' with all cluster assignments
   - Columns: cluster_kmeans, cluster_dbscan, cluster_hdbscan
   - Noise indicators for density-based methods
3. Console Output:
   - Clean data statistics
   - Parameter selection details
   - Cluster counts and noise percentages
   - Performance metrics
   - Recommendations summary

Results Object Structure:

- results['results']: Raw clustering outputs
- results['dataframes']: Labeled dataframes per method
- results['combined_df']: Merged results dataframe

Complete Workflow Summary

1. Data Loading & Preprocessing
   - Read CSV with latitude/longitude
   - Filter for Nepal geographical bounds
   - Handle missing values
2. Parameter Selection
   - Manual specification OR
   - Automatic optimization via heuristics
3. Clustering Execution
   - Run all three algorithms
   - Calculate evaluation metrics
   - Identify noise points
4. Results Generation
   - Visualize clusters and metrics
   - Save labeled data
   - Print summary recommendations

Key Strengths:

- Comprehensive comparison of multiple algorithms
- Automatic parameter optimization
- Multiple evaluation perspectives
- Ready-to-use visualizations

Algorithm Selection Recommendations

Scenario	Recommended Algorithm	Reason
Known cluster count	K-Means	Direct control over k
Varying densities	DBSCAN/HDBSCAN	Density-based approach
Noise detection needed	DBSCAN	Explicit noise identification
Hierarchical structure	HDBSCAN	Builds cluster hierarchy
Spherical clusters	K-Means	Optimized for convex shapes
Arbitrary shapes	DBSCAN	Density-based connectivity

Parameter Tuning Tips:

- K-Means: Use elbow method for unknown k
- DBSCAN: Start with k-distance visualization
- HDBSCAN: Adjust min_cluster_size based on data scale

Final Output: Ready-to-use clustered dataset with comparative analysis

Conclusion

Summary of Spatial Clustering Implementation

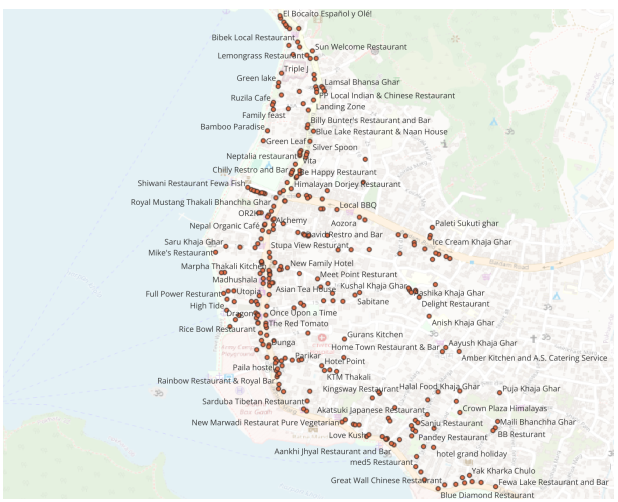
- Three complementary algorithms implemented
- Both manual and automatic parameter selection
- Comprehensive evaluation metrics
- Visual and numerical outputs

Key Takeaways:

1. Different algorithms suit different data characteristics
2. Parameter selection critical for performance
3. Multiple metrics provide complete picture
4. Visualization essential for spatial data

Output: Fully processed dataset with cluster labels for further analysis

Input



Outputs

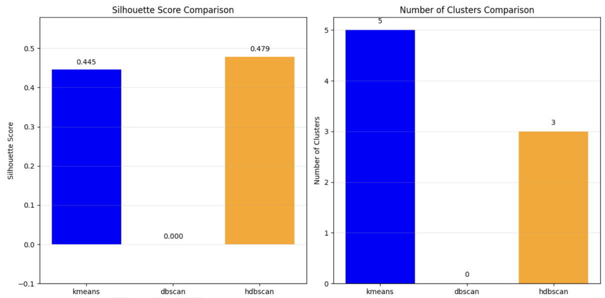
```
============================================================
SPATIAL CLUSTERING ANALYSIS
============================================================
Number of points: 293

1. KMEANS CLUSTERING
----------------------------------------
Performing KMeans clustering with 5 clusters...
   Clusters: 5
   Silhouette: 0.445

2. DBSCAN CLUSTERING
----------------------------------------
Performing DBSCAN clustering with eps=0.020, min_samples=5...
Found 0 clusters with 293 noise points (100.0%)
   Clusters: 0
   Noise points: 293 (100.0%)
   Silhouette: -1.000

3. HDBSCAN CLUSTERING
----------------------------------------
Performing HDBSCAN clustering with min_cluster_size=10, min_samples=5...
Found 3 clusters with 14 noise points (4.8%)
   Clusters: 3
   Noise points: 14 (4.8%)
   Silhouette: 0.479
```

Outputs



Outputs

```
5. SUMMARY
----------------------------------------

CLUSTERING ANALYSIS SUMMARY
==================================================
 Method  Clusters Noise % Silhouette  Points
 KMEANS         5    0.0%      0.445     293
 DBSCAN         0  100.0%     -1.000     293
HDBSCAN         3    4.8%      0.479     293
```


Outputs clusters

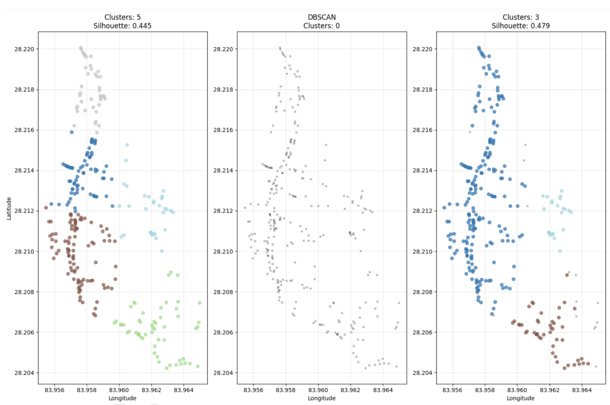
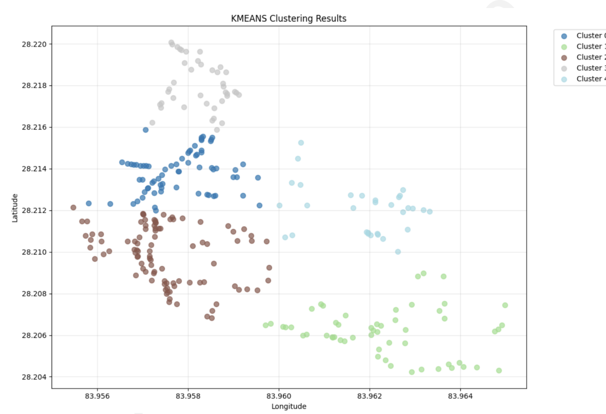
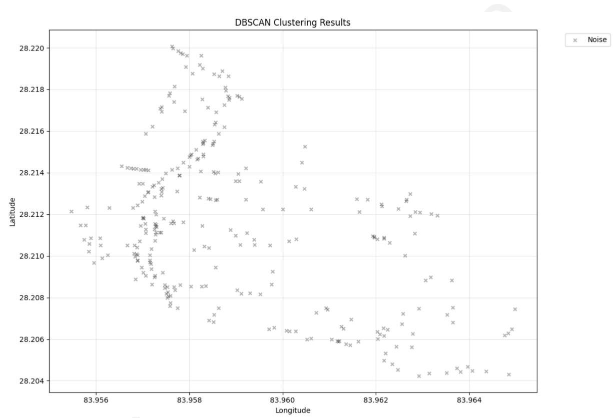


Figure: clusters formed with KMEANS,DBSCAN,HDBSCAN

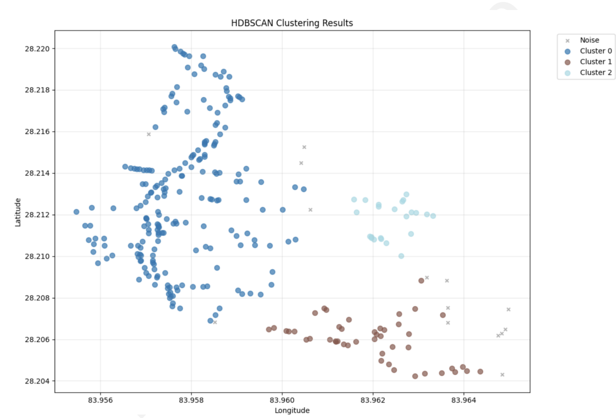
KMEANS Outputs (5 clusters)



DBSCAN Outputs (no clusters)



HDBSCAN Outputs (3 clusters)



HDBSCAN Outputs (3 clusters) in MAP