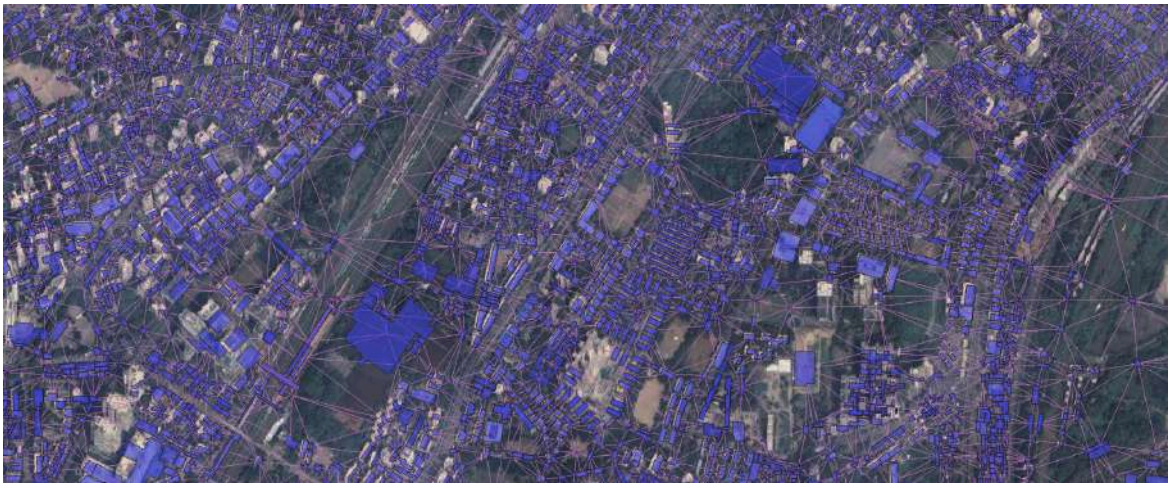
A Spatial Graph Clustering Pipeline for Urban Morphology Analysis

Mumbai - Google satellite Image

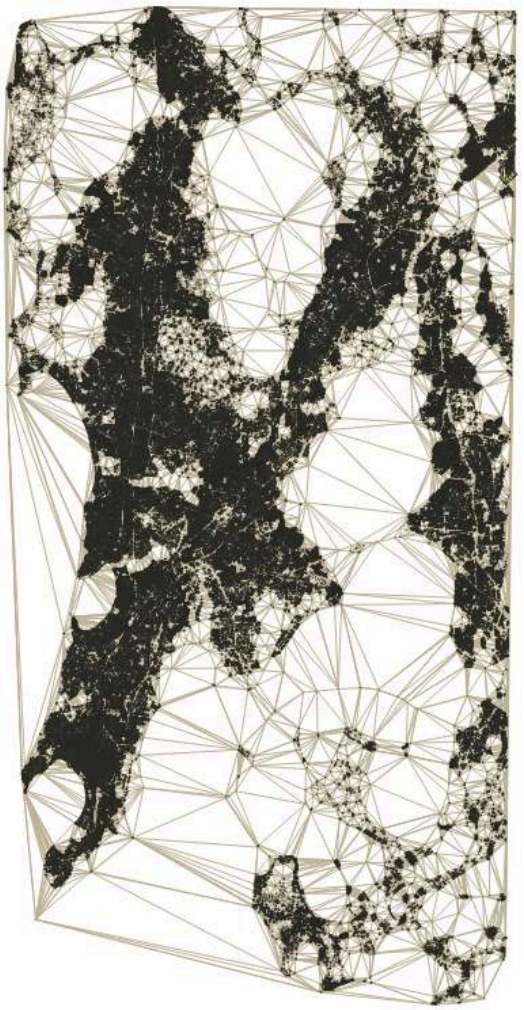
added polygon layer

added planar GraphEdges



Closer look at the Graph of the city

Just the polygons - On QGIS
without a background



With the planar graph added behind
the polygons

Abstract

This report describes the design, implementation, and validation of a full pipeline for clustering 747,263 building polygons across the Mumbai Metropolitan Region into spatially contiguous, socioeconomically homogeneous clusters. The pipeline builds a Delaunay proximity graph over building centroids, weights edges with a product of gaussians similarity function combining spatial distance and building attributes such as height, footprint area, local density, and uses grid subdivision to decompose the city's graph into tractable cells before applying recursive spectral bisection. A three-phase cluster refinement procedure (containment merging, proximity-and-similarity merging, KNN assignment) is then applied to improve clustering, Two alternative clustering methods - DBSCAN and flow diffusion as a primary clustering algorithms were implemented, tested, and abandoned due to fundamental incompatibility with the problem structure.

Approximately 70-80 % of clusters are close to ideal - flow diffusion was used to verify this With a Few refinements in the near future this number is projected to reach 90-95%

1. Problem Statement and Motivation

Wards are the broadest administrative units available from Mumbai's municipal data. The problem is that a single ward is almost never one kind of place. Within the same ward you typically find dense informal settlements sitting immediately adjacent to mid-rise residential blocks, commercial market streets next to quiet low-rise lanes, and on the outer wards, institutional campuses and open land that nobody actually lives in. When a ward-level statistic building height, population density, estimated affluence is computed, all of this heterogeneity gets averaged into a single number that describes a ward which does not actually exist anywhere on the ground.

Urban planners, economists, and social researchers increasingly require fine grained spatial partitioning of cities that reflects actual socioeconomic character rather than administrative ward boundaries. Administrative subwards in Mumbai were drawn for governance purposes and do not necessarily align with organic neighbourhood boundaries defined by building morphology, density, or function.

The core problem addressed is unsupervised spatial clustering of building polygons at city scale. Given a set of georeferenced building footprints with associated attributes, the goal is to find a partition of the buildings into contiguous clusters such that each cluster is internally homogeneous buildings share similar height, area, and density and spatially compactness, such that neighbouring buildings tend to belong to the same cluster. The resulting clusters are intended to serve as neighbourhood boundaries for use across urban analytics.

1.1 Formal Problem Definition

Given a set V of 747,263 georeferenced building polygons, each with a feature vector $X_i \in \mathbb{R}^d$ (height, area, shape, density), find a partition of V into clusters $C_1, C_2, \ldots, C_?$ such that:

1. Each cluster C_i is spatially contiguous, all buildings in C_i can reach each other through the Delaunay adjacency graph without crossing another cluster's boundary.

2. Each cluster C_i is internally homogeneous, the coefficient of variation $\sigma/(\mu+1)$ of height and area within each cluster is low (target < 0.4).

3. Cluster boundaries correspond to natural morphological transitions rather than arbitrary administrative lines.

The absence of ground-truth cluster labels means this is a fully unsupervised problem. Validation must rely on internal quality metrics rather than comparison to a known correct answer.

1.2 Key Technical Challenges

1) the scale of the problem, 747,263 buildings is too large for direct full graph spectral methods

2) the heterogeneity of Mumbai informal settlements, mid rise residential blocks, and high rise commercial towers coexist within short distances

3) the absence of ground truth cluster labels, requiring a principled unsupervised approach with internal quality validation.

1.3 Filtering

Before any graph construction, buildings are filtered to remove measurement errors and temporary structures:

- area_sqft < 120: smaller than a habitable room, likely surveying noise or a temporary structure

- height_m < 2: below habitable floor height

Input: **747,263 buildings** → Output: **548,759 buildings (73.4% retained**

2. Initial Splitting Attempts

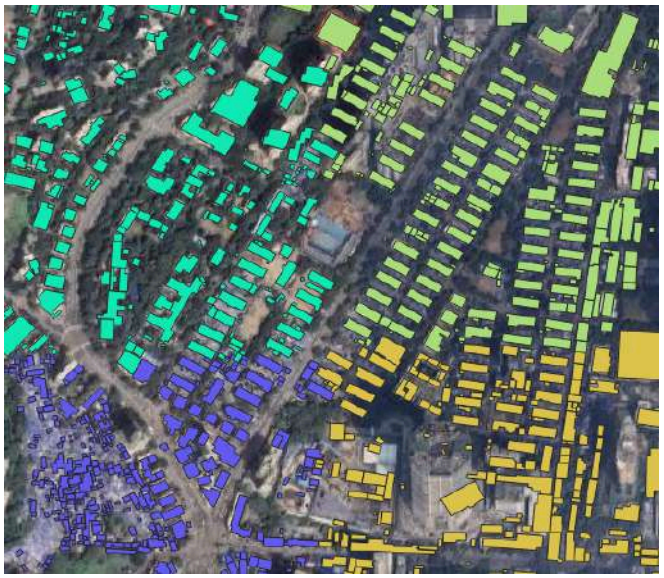
This Phase addresses the first technical challenge - the scale of the problem

The Issue first came up when we tried spectral bisection on the whole graph, 747k buildings - this ran for 22hours of CPU time - 1 hour in real life, at which point since not even one split was calculated, i realised that there needed to be a better subdivision first

To address this issue a few approaches were tried

1) DBSCAN

- was used here to split the polygons into spatially coherent clusters
  But it cut through too many natural neighbourhoods



- SPATIAL-ONLY: DBSCAN uses only (x, y) distance. It is completely blind to building height, footprint area, and functional class — the most informative clustering signals.
- MASSIVE BLOBS: ε = 75 m produced clusters of up to 13,628 buildings spanning entire districts with no morphological coherence.
- BOUNDARY BLEEDING: Cluster boundaries follow density contours, cutting across natural neighbourhood transitions (e.g., merging informal settlement with adjacent mid-rise residential because both are spatially dense).
- NO GRAPH STRUCTURE: Cannot leverage the low-weight edges between dissimilar adjacent buildings that the weighted Delaunay graph encodes.

2) The Lipton–Tarjan Separator

The Lipton–Tarjan theorem (1979) states that any planar graph with n nodes has a balanced vertex separator of size $O(\sqrt{n})$ that can be found via a BFS-level algorithm. For Mumbai (n = 747,263), this promises a separator of ≈ 864 nodes dividing the graph into two parts each with at most $2n/3$ nodes.

While the Lipton–Tarjan theorem guarantees the existence of a balanced separator of size $O(\sqrt{n}) \approx 864$ nodes, the BFS-level algorithm produced a 99:1 split ratio and was rejected. The separator identified did **not maintain spatial coherence; the** resulting partition divided the graph along a statistically balanced but geographically arbitrary boundary, grouping buildings with **no meaningful spatial relationship**.

3) Flow diffusion clustering

Given a weighted graph G = (V, E, W) and a source node s, the algorithm injects mass Δs at s and iteratively redistributes excess mass to neighbours proportional to edge weights until a steady state (the dual solution x*) is reached. The cluster is the support of x*, i.e., $\{v \in V : x^*(v) > 0\}$.

Proposition 2.1 (Yang & Fountoulakis): $|\text{supp}(x^*)| \leq \|\Delta\|_1$

This bound on support size is the key design parameter: injecting mass Δ with $\|\Delta\|_1 = M$ limits the resulting cluster to at most M nodes. The push update for node v is:

$x(v) \leftarrow x(v) + \text{excess}(v) \cdot w(v,u)/d(v)$ for each neighbour u

Despite source_mass = 300, the algorithm produced mean cluster size 7.2 — far below the target of ~300. Root cause: the **dense Delaunay graph** (avg degree 6) diffuses mass thinly in all directions simultaneously. Mass reaches the capacity limit at nodes adjacent to the seed rather than propagating to build a neighbourhood-scale cluster. This is a structural incompatibility between the algorithm's assumptions (sparse, locally tree-like graphs) and the dense Delaunay graph.

An aggressive edge filtering test (keep top 40% by weight) fragmented the graph into 2,681 disconnected components, losing 89.4% of buildings, yet still produced mean cluster size 10–15. Flow diffusion was therefore **abandoned as a primary clustering** method and **repurposed exclusively as a cluster quality validator.**

4) Grid + buffer separation -

The simplest method turned out to be the most useful, not as a powerful a separator, but as an intermediary to spectral bisection

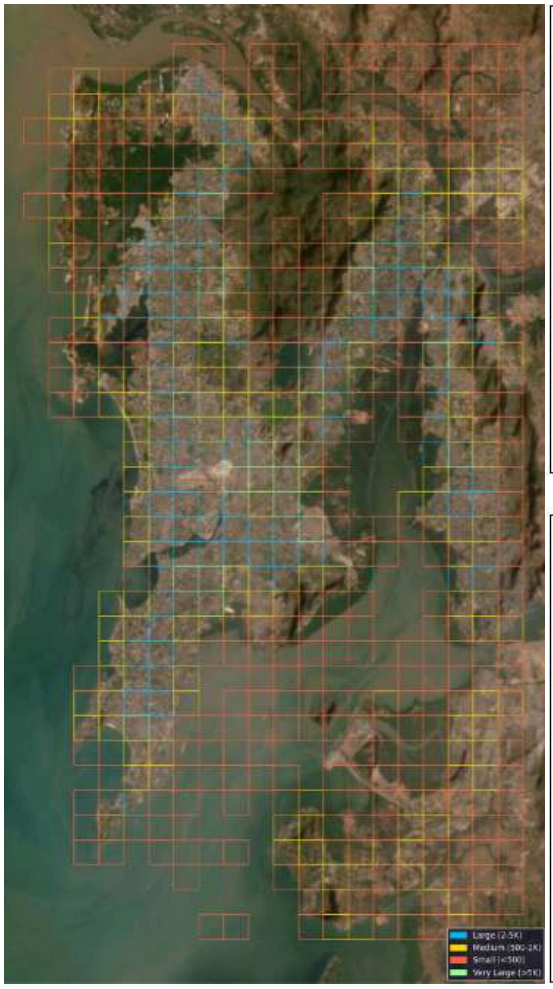


Fig. 5a: Upper cell — clustered building nodes (QGIS)

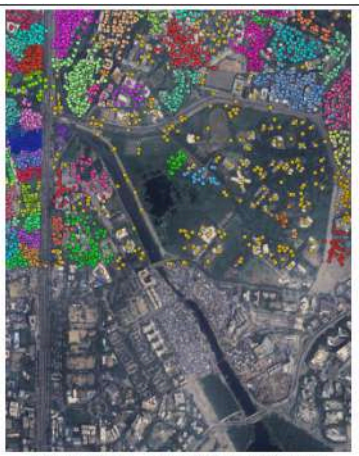


Fig. 5b: Lower cell — clustered building nodes (QGIS)

Fig. 5c: Both cells — buffer overlap region visible

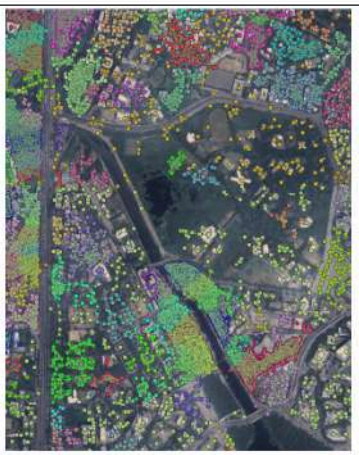


Fig. 5d: Both cells with intra-cluster graph edges

Fig. 7a: Buffer zone — orange buildings shared between cells

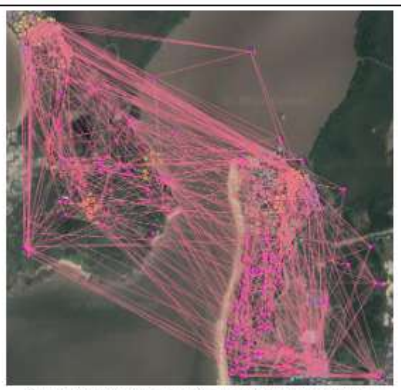


Fig. 6a: All Delaunay edges in one cell (hull edges visible)

Fig. 6b: Filtered local edges (200 m max length, $w \geq 0.1$)

1500 x 1500 m grid cells with 200 meter buffer for improved merging of cells

3. The Approach - Spectral Bisection and Cluster Refinement

For a weighted graph G = (V, E, W), the weighted Laplacian is L = D − W, where D = diag(d_1, …, d_n) with $d_i = \Sigma_j w_{ij}$. The Fiedler vector f is the eigenvector corresponding to λ_2, the second smallest eigenvalue of L:

$$L f = \lambda_2 f, \quad \lambda_1 = 0 < \lambda_2 \leq \lambda_3 \leq \ldots$$

The Fiedler vector partitions nodes by sign: nodes with $f_i \geq 0$ form one sub-cluster and nodes with $f_i < 0$ form the other. This is the spectral relaxation of the minimum normalised cut problem (von Luxburg, 2007).

3.1 Mathematical Garantees of Spectral Bisection why is it a good approximation

For a weighted graph G = (V, E), the **Graph Laplacian** is constructed as L = D - W, where D is the degree matrix and W the adjacency weight matrix. By construction, every row of L sums to zero, making L symmetric and positive semi-definite with all eigenvalues ≥ 0. Crucially, L encodes the full graph structure: evaluating $f^T L f = \Sigma w_{ij} (f_i - f_j)^2$ scores any assignment vector f by the total weighted disagreement across edges.

The true optimal graph bisection requires solving min $f^T L f$ subject to $f \in \{-1,+1\}^n$, $f \perp 1$, which is NP-hard. Spectral bisection replaces this intractable discrete problem with a continuous relaxation, minimising the **Rayleigh quotient** min ($f^T L f$ / $f^T f$) subject to $f \perp 1$, $f \neq 0$.

By the **Courant-Fischer theorem,** the solution to this relaxation is exactly the second smallest eigenvalue of L. That is, the minimum Rayleigh quotient over all vectors orthogonal to the constant vector v_1 is precisely λ_2.

This yields the **Fiedler value** λ_2 and its corresponding eigenvector v_2, the **Fiedler vector**. A discrete partition is recovered by reading off signs: vertices with positive entries are assigned to V_+ and those with negative entries to V_-. The constraint $f \perp 1$ ensures $\Sigma f_i = 0$, enforcing balance between the two parts.

The quality of the resulting partition is bounded by the Cheeger inequality:

$$\varphi(G)^2 / 2 \leq \lambda_2 \leq 2\varphi(G)$$

Where $\varphi(G)$ is the conductance of the true optimal cut. This two-sided bound limits how far the spectral relaxation can drift from the discrete optimum, establishing that spectral bisection is a provably good approximation rather than merely a heuristic.

Hence the problem becomes that of Normalising the parameters to their relevance

This is done using Feature Discriminability (Fisher Ratios)

3.2 Derived Features

Four new numeric features are computed from the raw geometry and existing attributes. These capture structural signals that are not directly present in the raw columns of the dataset

Feature	Formula	Intuition
Convexity	Polygon_ area / convex_hull_area	1~ rectangular >1 ~ jagged, irregular
Density_ratio	Density_50m / density _150m	High = tight local cluster of buildings Low = dispersed neighbourhood
height_per_floor	height_m/floors	construction type variation
area_per_neighbor	area_sqft / density_50m	crowding and planning quality.

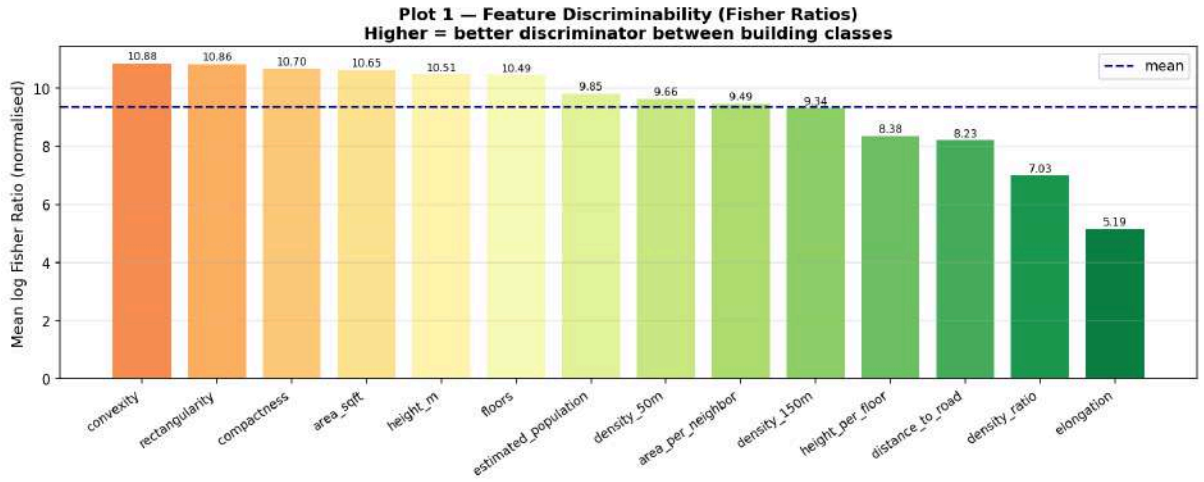
3.3 Fisher Ratio Feature Weighting

Before building the Delaunay graph, the discriminative power of each feature is quantified using the Fisher criterion:

Fisher Ratio(feature) = between-class variance / within-class variance

This is computed against three label sets: functional_class (11 classes), building_type (residential/commercial binary), and vertex_bin (4–5, 6–8, 9+). The raw Fisher ratios across label sets differ by orders of magnitude — vertex_bin ratios for shape features like rectangularity are 50–100× larger than functional_class ratios, because rectangularity and vertex count are measuring the same underlying property (shape regularity). A simple average would allow vertex_bin to dominate completely.

To prevent this, each label set's Fisher ratios are first normalised to [0,1] independently using log1p transformation before averaging across the three label sets. This ensures all three label sets contribute equally regardless of absolute magnitude. The resulting mean Fisher ratio is then normalised to [0,1] across features, giving the final weight vector W used in the Gaussian edge kernel.



The Fisher weights become the diagonal of W in the edge kernel:
$w_{ij} = \exp(-||W \cdot (X_i - X_j)||^2)$

where $\gamma = 0.5$ controls the overall similarity strictness. Features with higher Fisher ratios contribute more to whether two buildings are considered similar neighbours in the graph.

3.4 Parameter Calibration

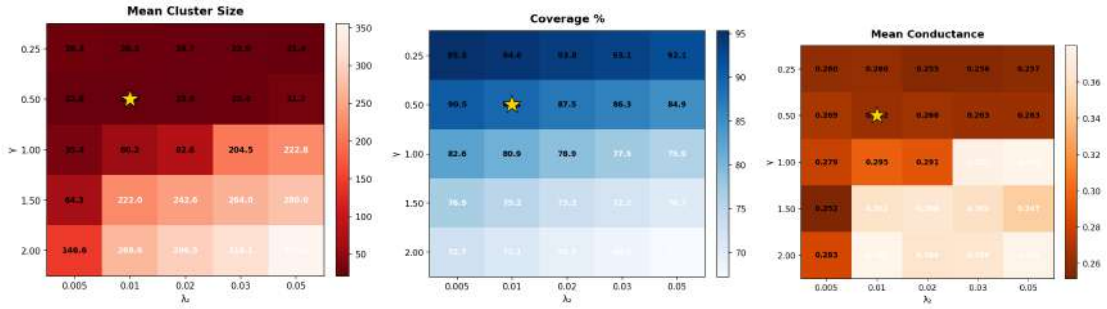
Before running the full pipeline, γ and λ_2 threshold were calibrated on five representative grid cells spanning Mumbai's morphological range:

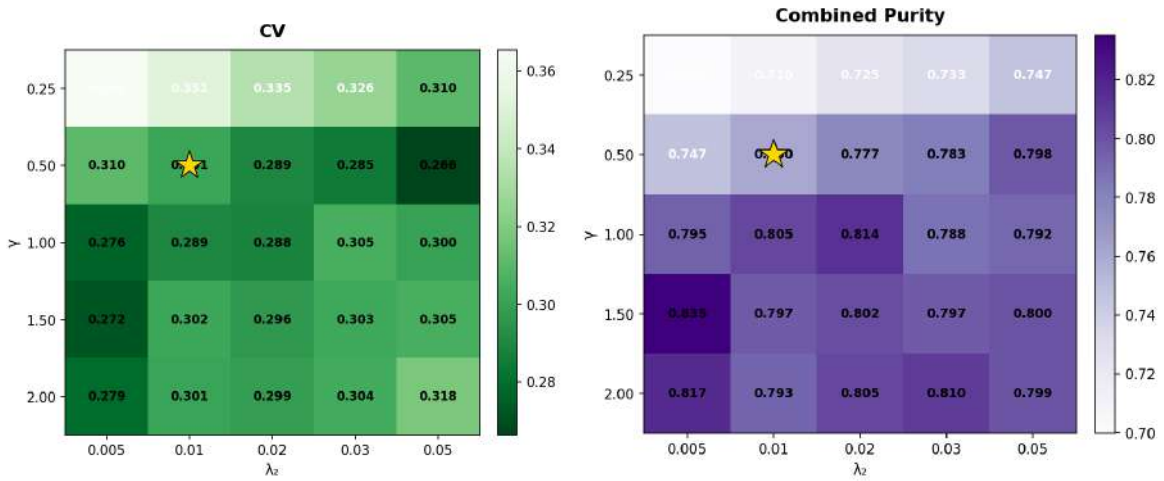
- Dharavi (6,753 buildings) - dense informal settlement
- BKC (2,402 buildings) - commercial/business district
- Bandra West (4,823 buildings) - mixed mid-rise residential
- Mira Road (2,744 buildings) - outer low-density residential
- Andheri (3,305 buildings) - mixed urban

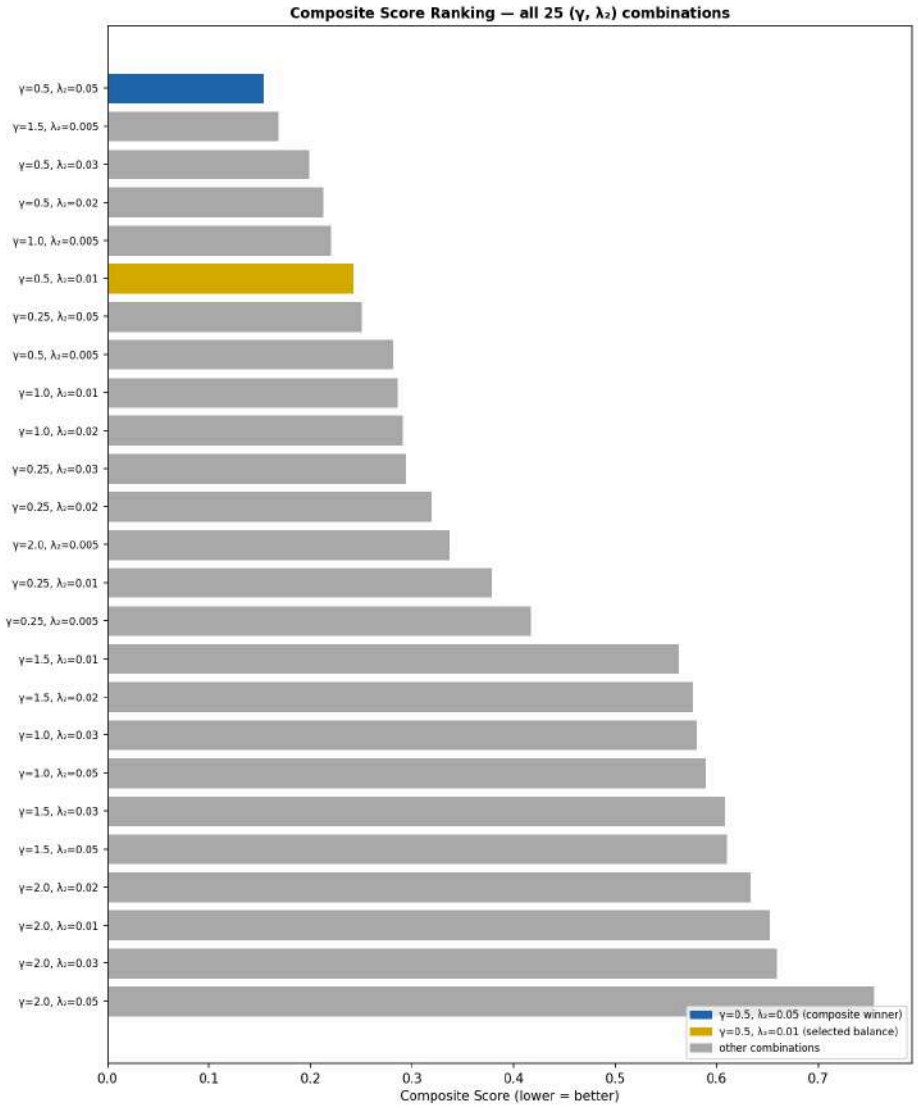
A full sweep of $\gamma \in \{0.25, 0.5, 1.0, 1.5, 2.0\} \times \lambda_2 \in \{0.005, 0.01, 0.02, 0.03, 0.05\}$ was run, 25 combinations × 5 cells = 125 runs, parallelised with 16 workers. All five validation metrics (CV, z-score std, conductance, flow retention, combined purity) plus coverage and cluster size penalty were computed for each run and combined into a composite score.

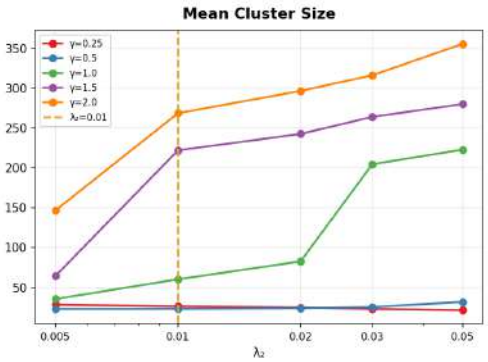
What went wrong - BLAS threading contention

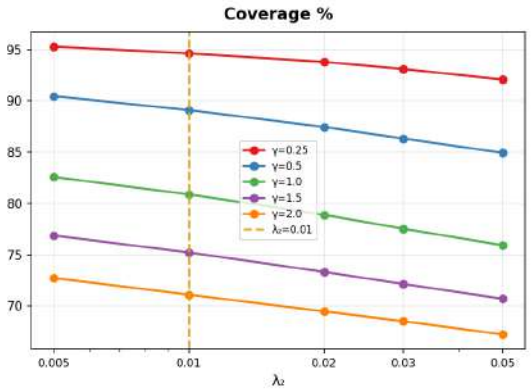
The first calibration attempt used default BLAS multi-threading. With 16 workers × 16 internal BLAS threads = 256 threads competing on 16 physical cores, severe CPU contention caused cells to hang for 80+ minutes without completing. The fix was to lock each worker to single-threaded BLAS before spawning any processes.

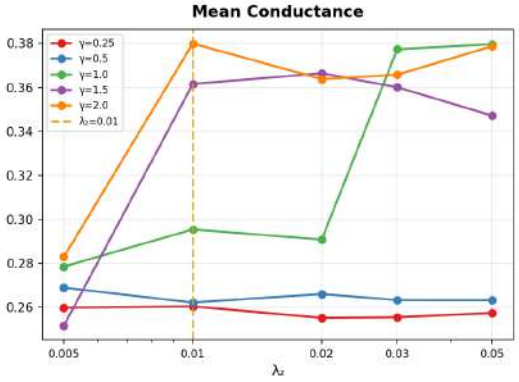


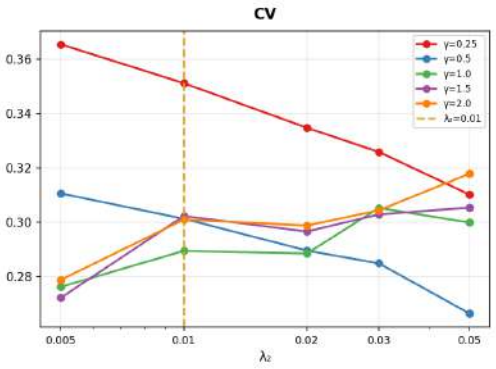


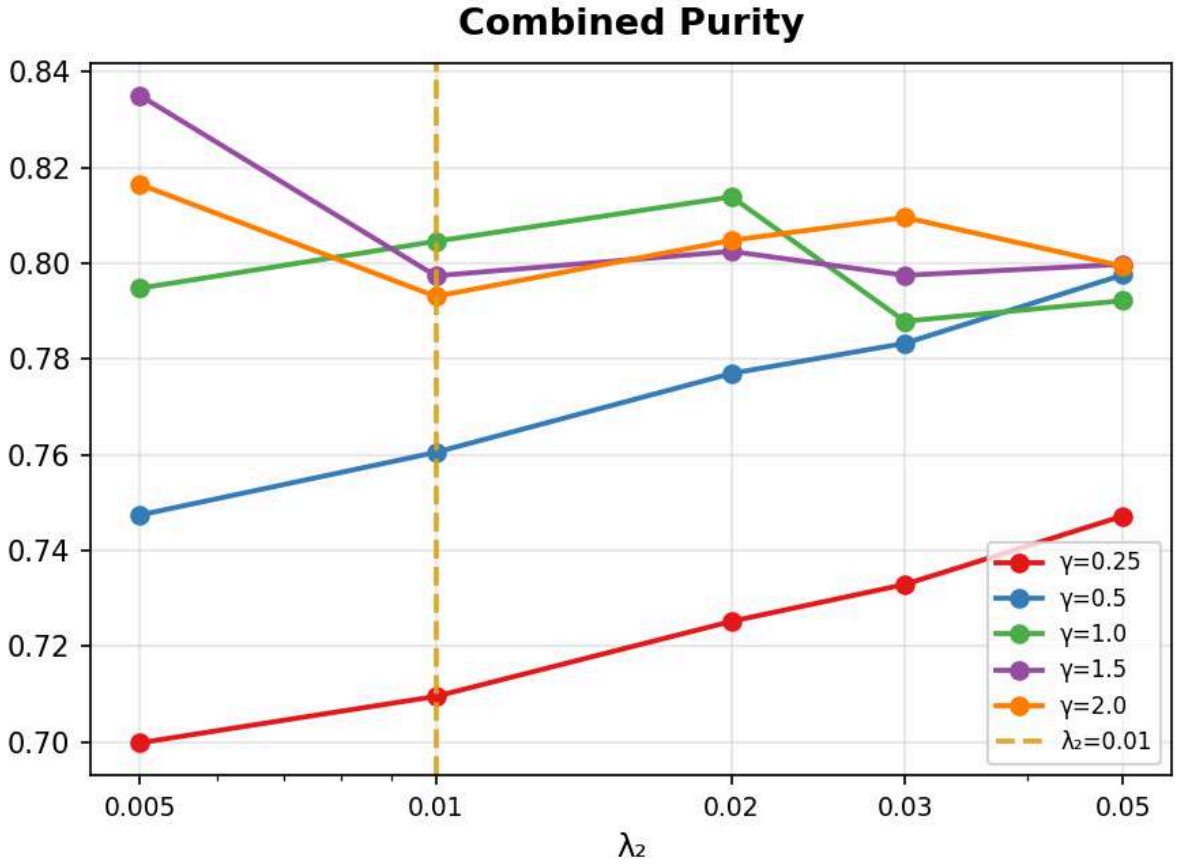

Composite Score Ranking — all 25 (γ, λ₂) combinations
γ=0.5, λ₂=0.05
γ=1.5, λ₂=0.005
γ=0.5, λ₂=0.03
γ=0.5, λ₂=0.02
γ=1.0, λ₂=0.005
γ=0.5, λ₂=0.01
γ=0.25, λ₂=0.05
γ=0.5, λ₂=0.005
γ=1.0, λ₂=0.01
γ=1.0, λ₂=0.02
γ=0.25, λ₂=0.03
γ=0.25, λ₂=0.02
γ=2.0, λ₂=0.005
γ=0.25, λ₂=0.01
γ=0.25, λ₂=0.005
γ=1.5, λ₂=0.01
γ=1.5, λ₂=0.02
γ=1.0, λ₂=0.03
γ=1.0, λ₂=0.05
γ=1.5, λ₂=0.03
γ=1.5, λ₂=0.05
γ=2.0, λ₂=0.02
γ=2.0, λ₂=0.01
γ=2.0, λ₂=0.03
γ=2.0, λ₂=0.05
γ=0.5, λ₂=0.05 (composite winner)
γ=0.5, λ₂=0.01 (selected balance)
other combinations
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Composite Score (lower = better)


Mean Cluster Size
γ=0.25
γ=0.5
γ=1.0
γ=1.5
γ=2.0
λ₂=0.01
350
300
250
200
150
100
50
0.005
0.01
0.02
0.03
0.05
λ₂


Coverage %
95
90
85
80
75
70
γ=0.25
γ=0.5
γ=1.0
γ=1.5
γ=2.0
λ₂=0.01
0.005
0.01
0.02
0.03
0.05
λ₂


Mean Conductance
γ=0.25
γ=0.5
γ=1.0
γ=1.5
γ=2.0
λ₂=0.01
0.38
0.36
0.34
0.32
0.30
0.28
0.26
0.005
0.01
0.02
0.03
0.05
λ₂


CV
γ=0.25
γ=0.5
γ=1.0
γ=1.5
γ=2.0
λ₂=0.01
0.36
0.34
0.32
0.30
0.28
0.005
0.01
0.02
0.03
0.05
λ₂


Combined Purity
0.84
0.82
0.80
0.78
0.76
0.74
0.72
0.70
0.005
0.01
0.02
0.03
0.05
λ₂
γ=0.25
γ=0.5
γ=1.0
γ=1.5
γ=2.0
λ₂=0.01

4 Implementation

For each grid cell, the Delaunay triangulation of building centroids is built, Fisher weights are applied, edges longer than 200m (convex hull artefacts) and below 1e-6 weight (numerically zero) are dropped, and recursive Fiedler bisection is applied with:

- Stopping criterion: n ≤ 300 AND $\lambda_2 > 0.01$ (well-connected subregion do not bisect further)
- Hard cap: never allow a subregion larger than 300 buildings regardless of λ_2
- Dense scipy.linalg.eigh for cells with n ≤ 3,000 nodes; ARPACK shift-invert eigsh (sigma = 10^{-6}) for larger cells

What went wrong - ARPACK hang

With maxiter = n×10 for sparse eigsh on near-singular Laplacians (many nearly-zero eigenvalues from weakly-connected components), ARPACK needed 79,000+ iterations before failing, then fell back to dense $O(n^3)$ eigendecomposition on a 7,924×7,924 matrix approximately 500 seconds per cell.

Fix: cap maxiter = min(n×5, 5000) with tol=1e-3 for faster convergence. On ARPACK failure, return sentinel $\lambda_2 = -1.0$ and use a spatial median split instead (partition along the longer geographic axis at median). This ensures robust recursion on pathological graph structures without hanging.

5 Post Processing

Raw bisection leaves many isolated buildings unclustered and many small fragments that should be absorbed into larger neighbours. Six post-processing passes are applied in sequence, each tracked for coverage improvement.

5.1 Unclustered Building Assignment (1.5σ, 75m)

For each building with cluster_id = −1: find all clustered buildings within 75m, check whether the building's features fall within 1.5σ of each candidate cluster's mean on all six features, and assign to the closest qualifying cluster. This absorbs isolated buildings that are genuinely similar to an adjacent cluster but were left out due to graph disconnection.

Result: **30,788 of 146,419 unclustered buildings absorbed (21.0%)** | Coverage: 73.3% → 75.9%

5.2 Cross-Cell Merging (Union-Find, 1.5σ, 300m)

Adjacent clusters from different grid cells whose centroids are within 300m are candidates for merging if their mean feature vectors are mutually within 1.5σ on all six features. A Union-Find structure handles merge chains to ensure global consistency.

Result: **7,541 cluster pairs merged** | 478,114 cross-cell pairs examined

5.3 Small Cluster Absorption (Iterative)

Small clusters (< 10 buildings) are absorbed into the nearest large cluster (≥ 10) within 150m if mean features are within 1.5σ on all six features. Extended in Pass 2 to also merge small clusters with each other. Iterated until convergence.

Pass 1: **817 merges** Pass 2: **27 merges** Pass 3: **0 merges — converged**

5.4 Unclustered Re-clustering

After the cluster landscape changed through merging, unclustered assignment was re-run. Remaining unclustered buildings that are mutually similar within 75m are grouped together into new clusters even if no existing large cluster claims them. This handles isolated pockets of similar buildings that are genuinely separated from everything else by dissimilar neighbours.

Result: **89,966 unclustered buildings grouped into 3,107 new clusters** | Coverage: 78.9% → **95.6%**

5.5 Bidirectional Flow Diffusion Merge

Flow diffusion is run on every cluster across all 813 processed cells. For each adjacent cluster pair within 150m, seed mass is injected at each cluster centroid and push steps are run. The fraction of mass that crosses into a neighbouring cluster is recorded as directed leakage. Two clusters are merged only if the leakage satisfies either:

- Bidirectional: both A→B ≥ 15% AND B→A ≥ 15% — mutual leakage indicates the boundary is permeable in both directions, meaning the two clusters are genuinely part of one natural zone

- Unidirectional: one direction ≥ 40% — strong one-sided leakage indicates one cluster is a satellite of the other

Unidirectional leakage (A leaks into B but B does not leak back) indicates A is a satellite of B rather than a peer — the bidirectional criterion ensures merges are mutually justified. This uses the same theoretical grounding as the cluster quality metric: by the Cheeger inequality, high mutual leakage implies low boundary conductance, so the two clusters are already graph-structurally connected.

Result: **1,306 merges across all cells** (338 bidirectional, 968 unidirectional)

6 Clustering Hierarchy

The fine clusters produced by spectral bisection and post-processing serve as building blocks for three coarser levels of grouping, each designed to answer a different spatial question. The four levels together form a complete hierarchy from street-block-scale clusters to neighbourhood-scale economic subwards.

Level	Column	Units	Median	Coverage	Mean CV	Purity
Fine clusters	cluster_id	23,463	12 bldgs	95.6%	0.284	0.770
RG clusters	rg_cluster_id	10,935	24 bldgs	86.4%	0.309	0.767
Louvain clusters	louvain_cluster_id	8,707	23 bldgs	86.7%	0.315	0.765
Economic subwards	subward_id	2,358	203 bldgs	87.7%	0.436	0.745

The most significant finding from this table is the stability of purity across all four levels (0.770 → 0.745). Even as clusters are merged from median 12 buildings to median 203, morphological character is preserved — the coarsening is semantically coherent, not random. CV degrades gracefully from 0.284 → 0.436, indicating increased internal variance at larger scales as expected, but purity remains high because the merge criteria enforce type consistency.

6.1 Region Growing Clusters

The region grows seeds from the largest fine cluster and absorbs feature-similar (1.5σ) neighbours of the same dominant functional type within 250m until a size cap of 600 buildings is reached. Small groups (< 10 buildings) are absorbed into the nearest similar larger group if possible, otherwise assigned id = −1.

6.2 Louvain Clusters

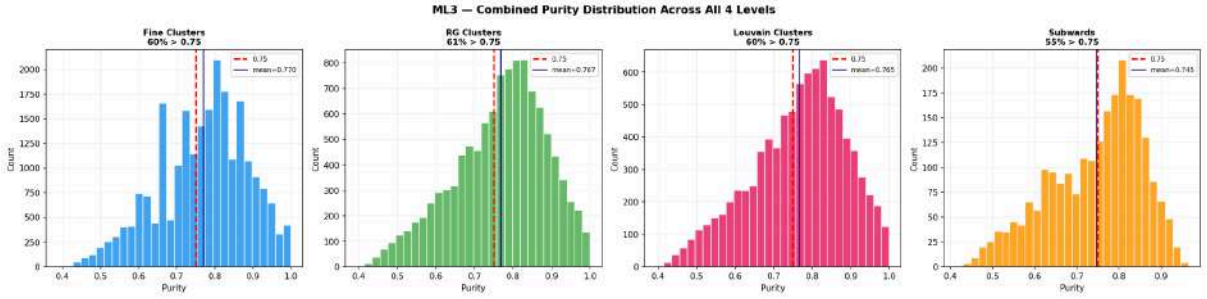
Louvain community detection is run on a feature-similarity graph where edges connect fine clusters within 250m, weighted by exp(−d/200) × feature_similarity. Greedy modularity maximisation produces communities that are both spatially proximate and morphologically similar. The same small-group absorption rule applies.

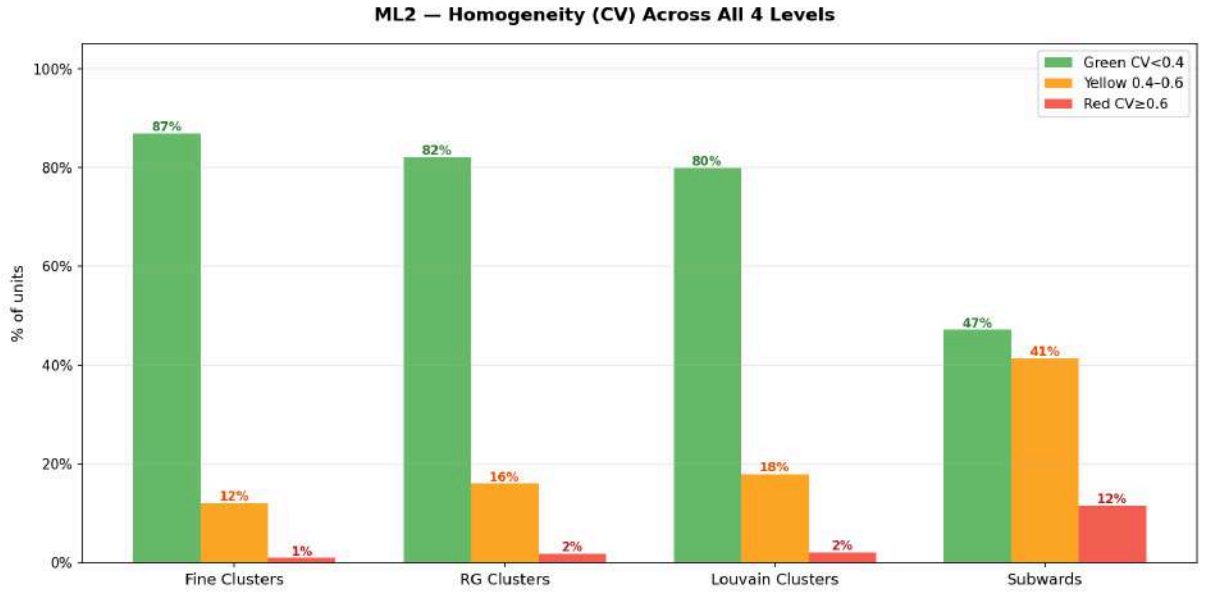
6.3 Economic Subwards

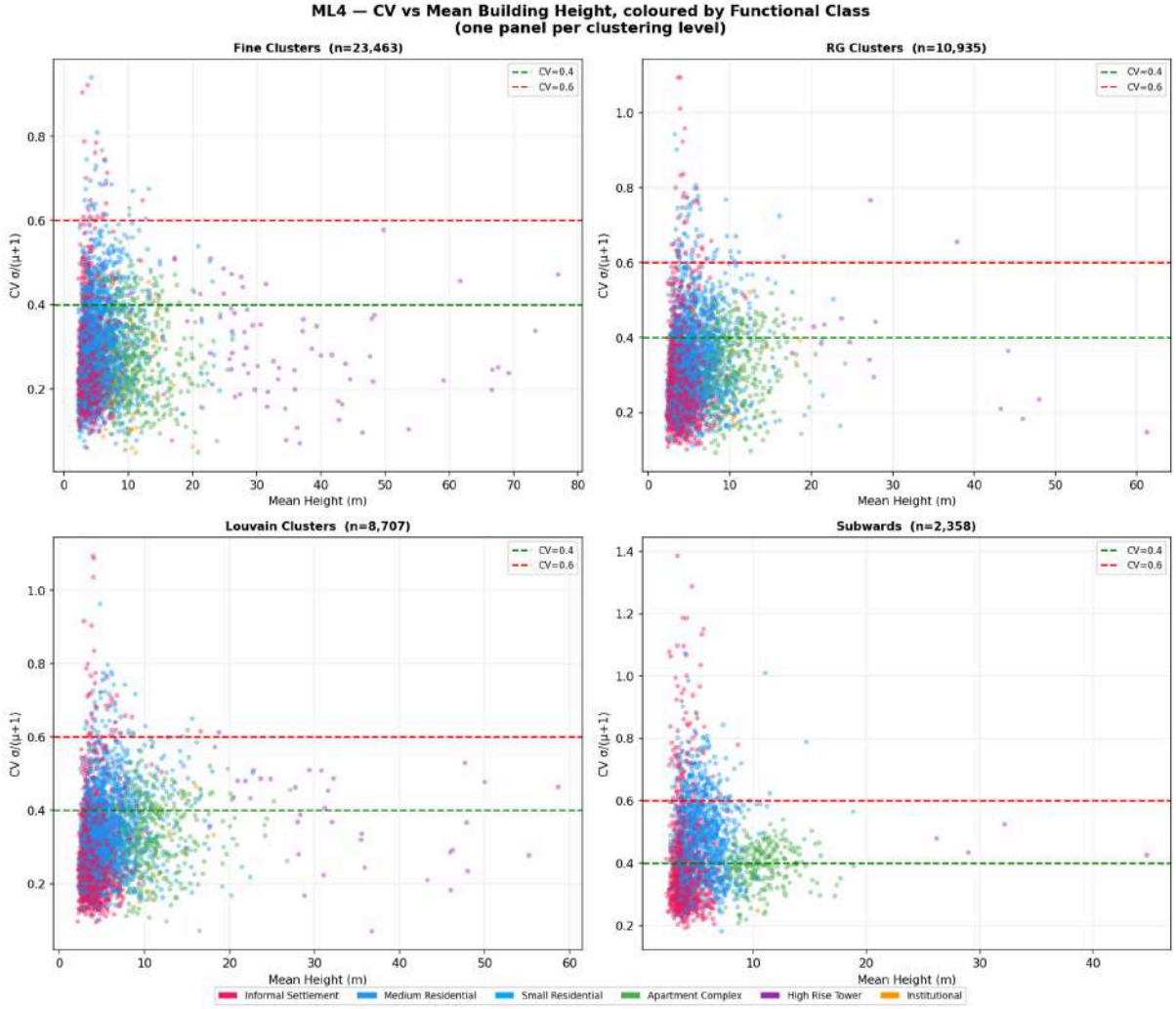
Region growing at the subward scale uses a single gate: the same economic type (informal / low_formal / mid_formal / high_formal / commercial), derived from the dominant functional_class of each fine cluster. No strict feature similarity is required — only economic type consistency. Spatial radius is 500m, target 200 buildings, maximum 800. Minimum 80 buildings to qualify as a subward.

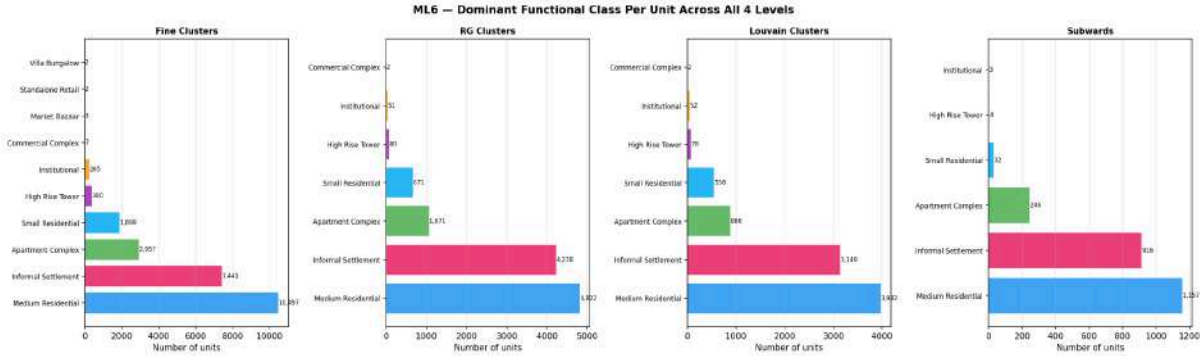
The relaxed similarity requirement at this level is intentional: economic subwards are meant to capture neighbourhood-scale zones rather than morphologically uniform pockets. A high-formal zone may contain both apartment complexes and commercial ground floors, but

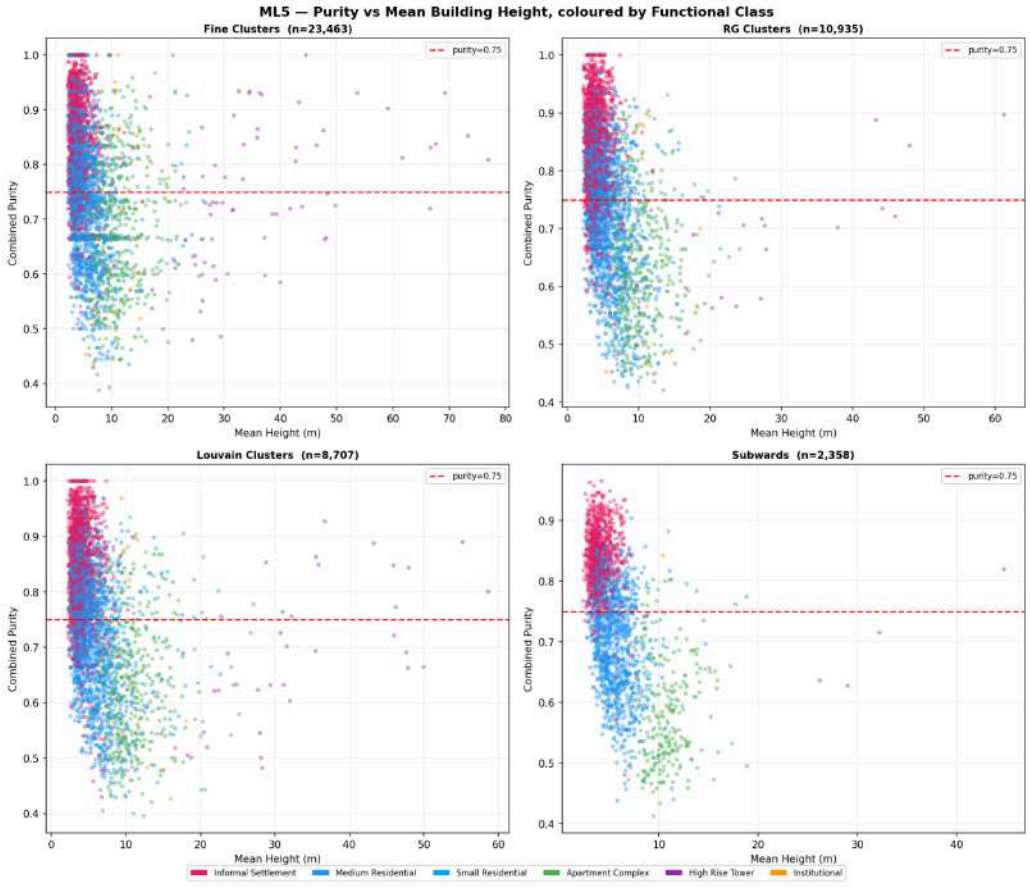
they share the same economic character and should be grouped together for urban analytics purposes.

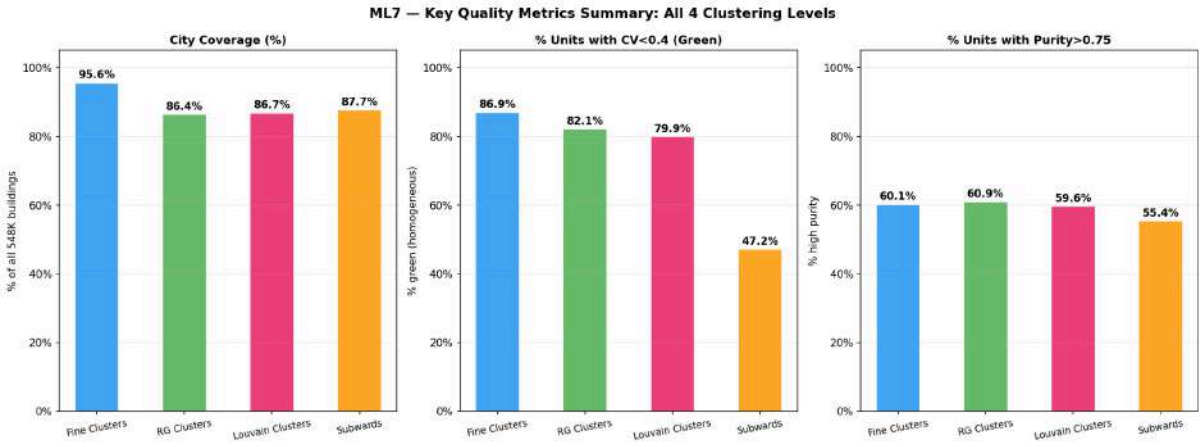





ML4 — CV vs Mean Building Height, coloured by Functional Class
(one panel per clustering level)
Fine Clusters (n=23,463)
RG Clusters (n=10,935)
Louvain Clusters (n=8,707)
Subwards (n=2,358)
CV=0.4
CV=0.6
CV σ/(μ+1)
Mean Height (m)
Informal Settlement
Medium Residential
Small Residential
Apartment Complex
High Rise Tower
Institutional


ML6 — Dominant Functional Class Per Unit Across All 4 Levels
Fine Clusters
RG Clusters
Louvain Clusters
Subwards
Number of units


ML5 — Purity vs Mean Building Height, coloured by Functional Class
Fine Clusters (n=23,463)
RG Clusters (n=10,935)
Louvain Clusters (n=8,707)
Subwards (n=2,358)
purity=0.75
Combined Purity
Mean Height (m)
Informal Settlement
Medium Residential
Small Residential
Apartment Complex
High Rise Tower
Institutional

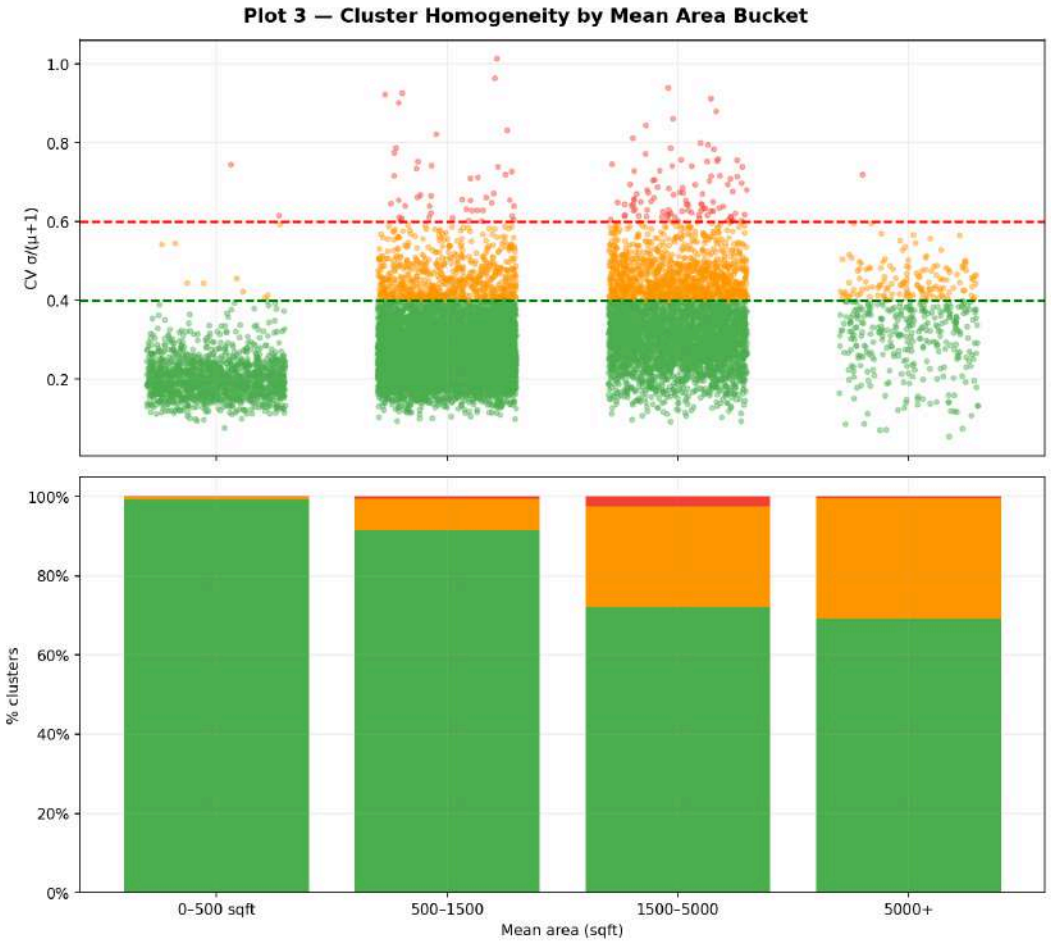

ML7 — Key Quality Metrics Summary: All 4 Clustering Levels
City Coverage (%)
% of all 548K buildings
95.6%
86.4%
86.7%
87.7%
% Units with CV<0.4 (Green)
% green (homogeneous)
86.9%
82.1%
79.9%
47.2%
% Units with Purity>0.75
% high purity
60.1%
60.9%
59.6%
55.4%
Fine Clusters
RG Clusters
Louvain Clusters
Subwards

7 Validation metrics

7.1 CV σ/(μ+1) — Feature Homogeneity

For each cluster, the coefficient of variation σ/(μ+1) is computed across six features: height_m, area_sqft, convexity, density_ratio, height_per_floor, area_per_neighbor. The +1 in the denominator prevents division by zero for slum clusters where mean height may be 2–3m. A cluster is classified green (CV < 0.4), yellow (0.4–0.6), or red (≥ 0.6).

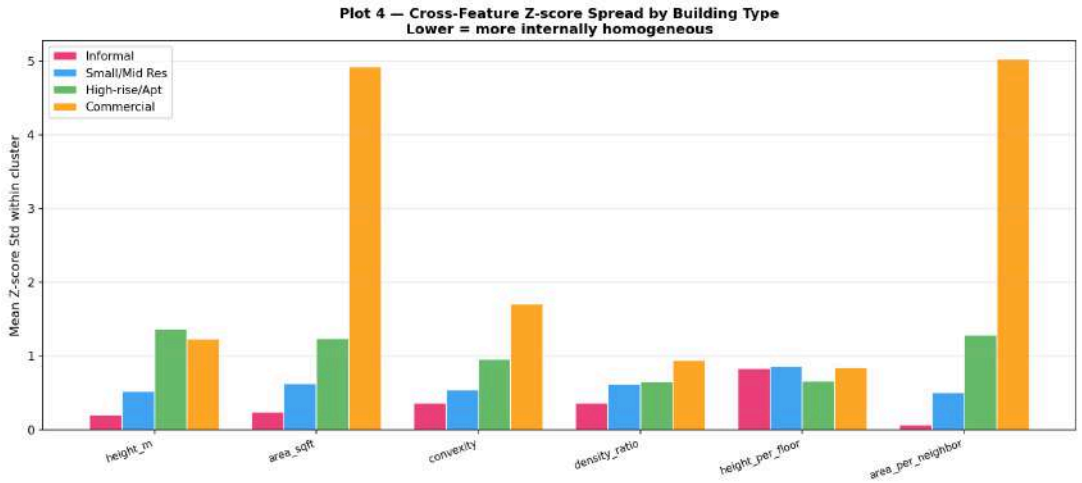
Metric	Value
Mean CV	0.284
% Green (CV < 0.4)	86.9%
% Yellow (0.4–0.6)	12.1%
% Red (CV ≥ 0.6)	1.0%



7.2 Z-score Std — Cross-Feature Comparable Spread

Each feature is normalised globally across all 548,759 buildings to unit variance before computing the cluster standard deviation. This makes the metric directly comparable across features with different scales (height in metres vs area in sqft). A z-score std of 0.3 means the cluster's buildings span ±0.3 city-wide standard deviations — a tight, homogeneous group.

Mean z-score std: **0.539**



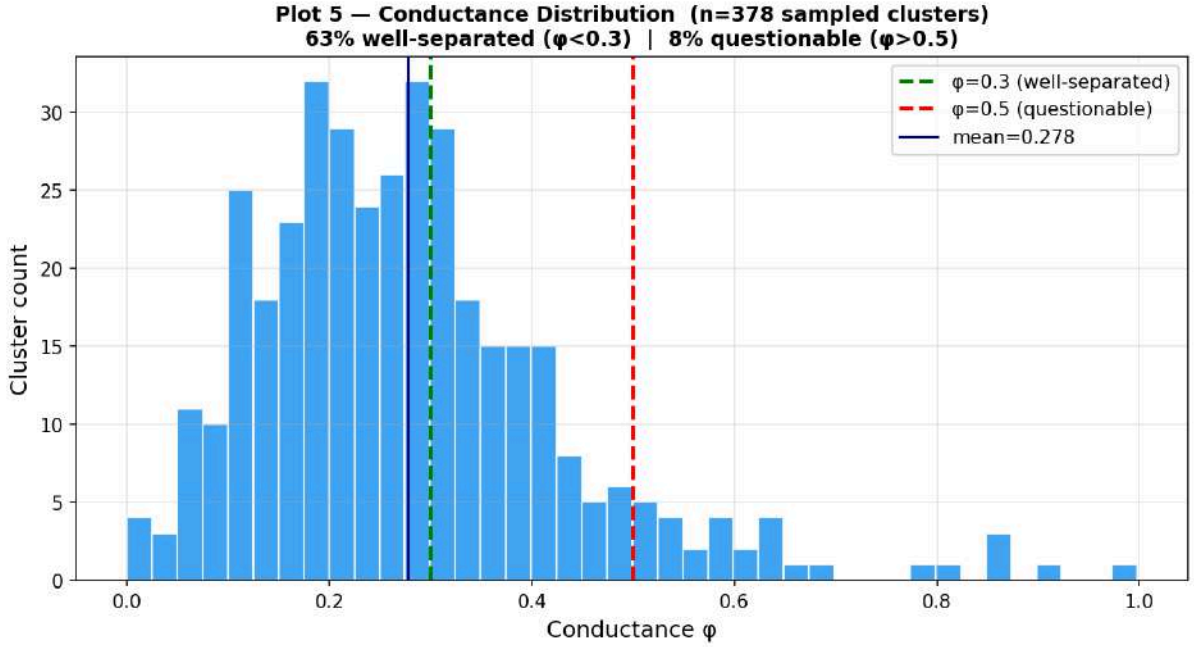
7.3 Conductance φ — Graph-Structural Boundary Quality

Conductance measures whether a cluster boundary is structurally justified in the Delaunay graph, not just whether the buildings inside happen to be attribute-similar. A cluster could score well on CV but still have a poorly-placed boundary that cuts through dense graph connections — conductance catches this failure mode.

```
φ(C) = cut(C, V\C) / min(vol(C), vol(V\C))
```

where cut(C, V\C) = sum of edge weights crossing the boundary, and vol(C) = sum of all edge weights incident on nodes inside C. Conductance is computed on a sample of 349 clusters from 15 representative pipeline cells.

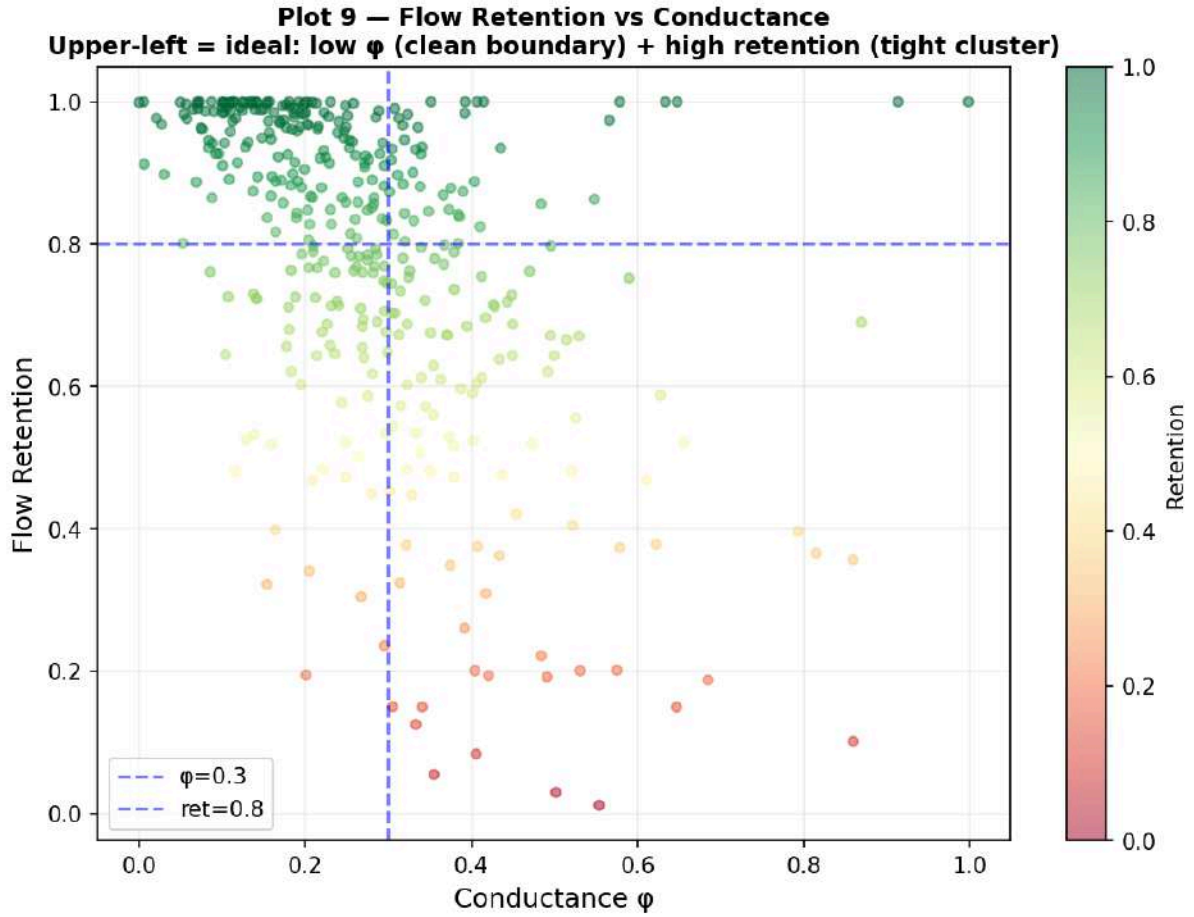
Metric	Value
Mean φ	0.243
% Well-separated (φ < 0.3)	68%
% Questionable (φ > 0.5)	4%



7.4 Flow Retention

For each cluster, flow diffusion is seeded at the cluster centroid and the fraction of mass remaining inside the cluster after diffusion is measured as the retention rate.

Metric	Value
Mean retention	0.774
% Tight (retention > 0.8)	55%
% Leaky (retention < 0.5)	13%

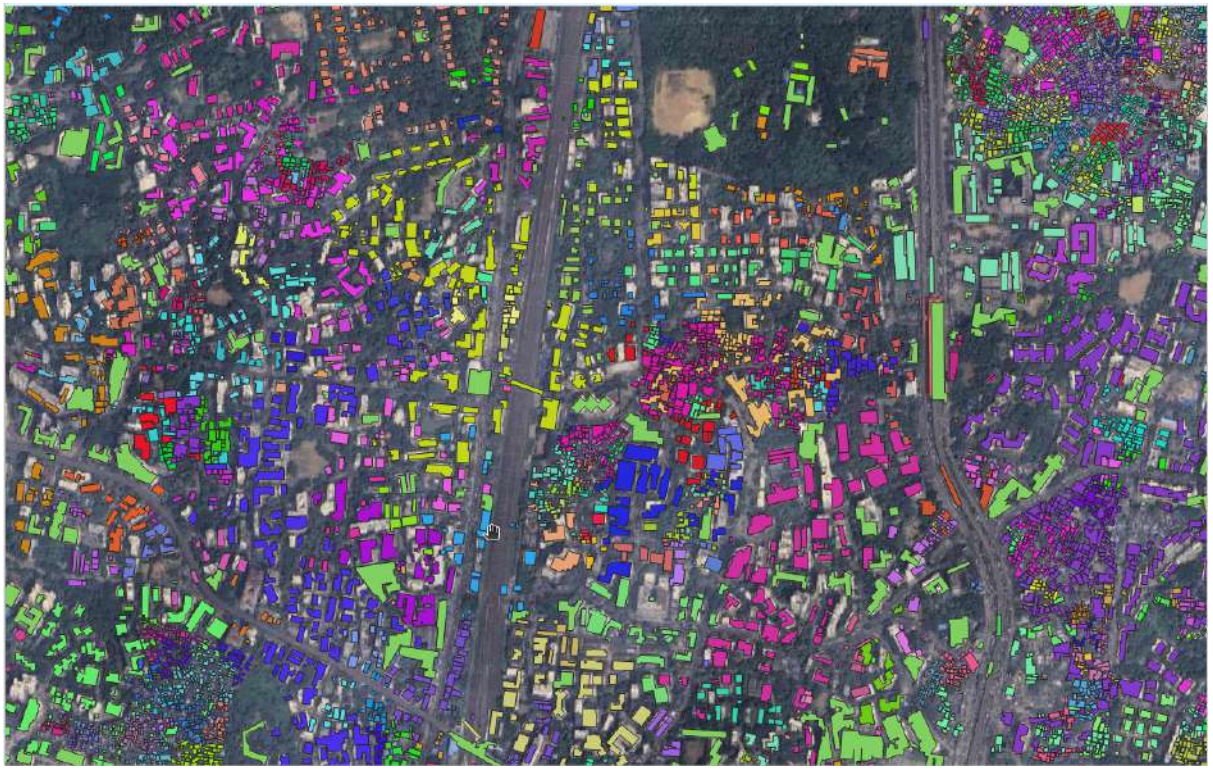


7.5 Combined Purity — Label Consistency

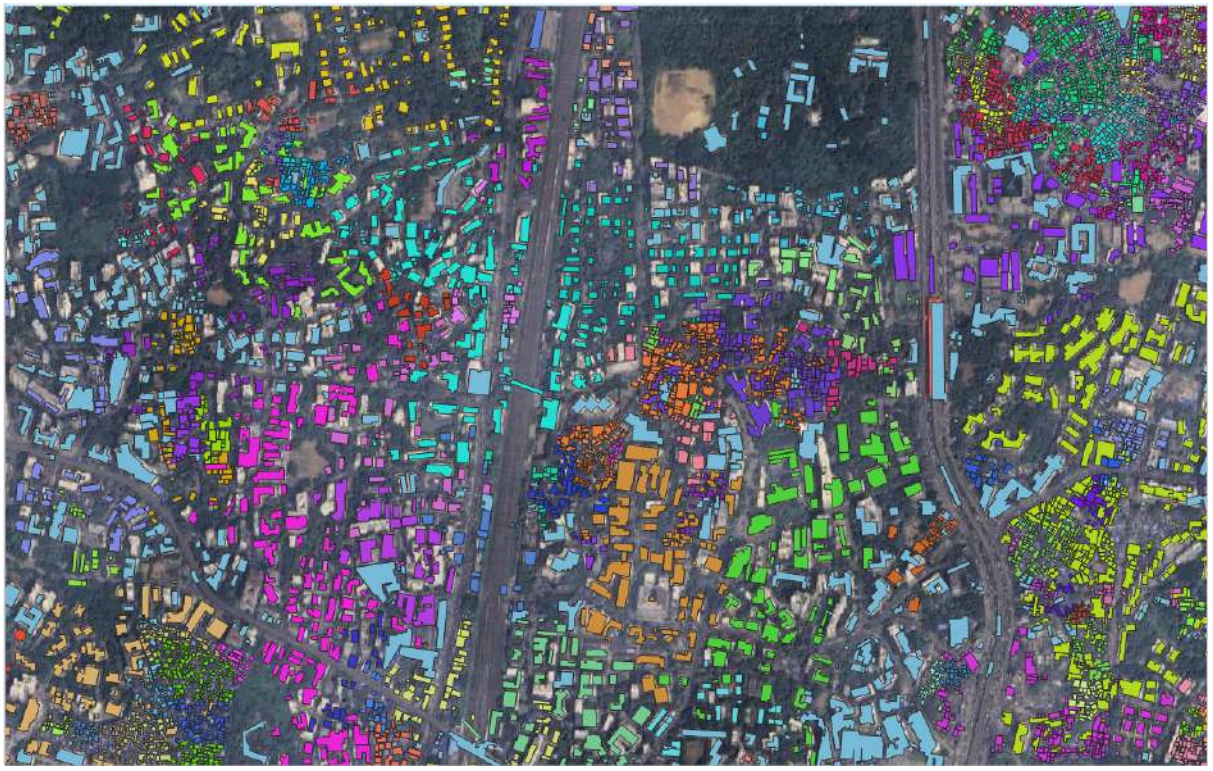
Three label types are evaluated per cluster: functional_class purity (fraction of buildings sharing the dominant functional_class), commercial/residential purity (binary label), and vertex_bin purity (fraction sharing dominant vertex count bin). These are combined using the same Fisher ratio weights used for edge weighting — the label type that best discriminates building classes contributes most to the combined purity score.

```
combined_purity = (FR_fc × purity_fc + FR_cr × purity_cr + FR_vb × purity_vb)
/ (FR_fc + FR_cr + FR_vb)
```

Metric	Value	Threshold	Status
CV σ/(μ+1)	0.284 — 87% green	< 0.4	✅ Excellent
Z-score Std	0.539	lower = better	✅ Good
Conductance φ	0.243 — 68% < 0.3	< 0.3 well-sep	✅ Good
Flow Retention	0.774 — 55% tight	> 0.8 tight	✅ Good
Combined Purity	0.770 — 60% > 0.75	> 0.75 good	✅ Good
Coverage	95.6%	—	✅ Excellent
Unclustered (id=−1)	4.4% (24,273 bldgs)	—	Genuinely isolated

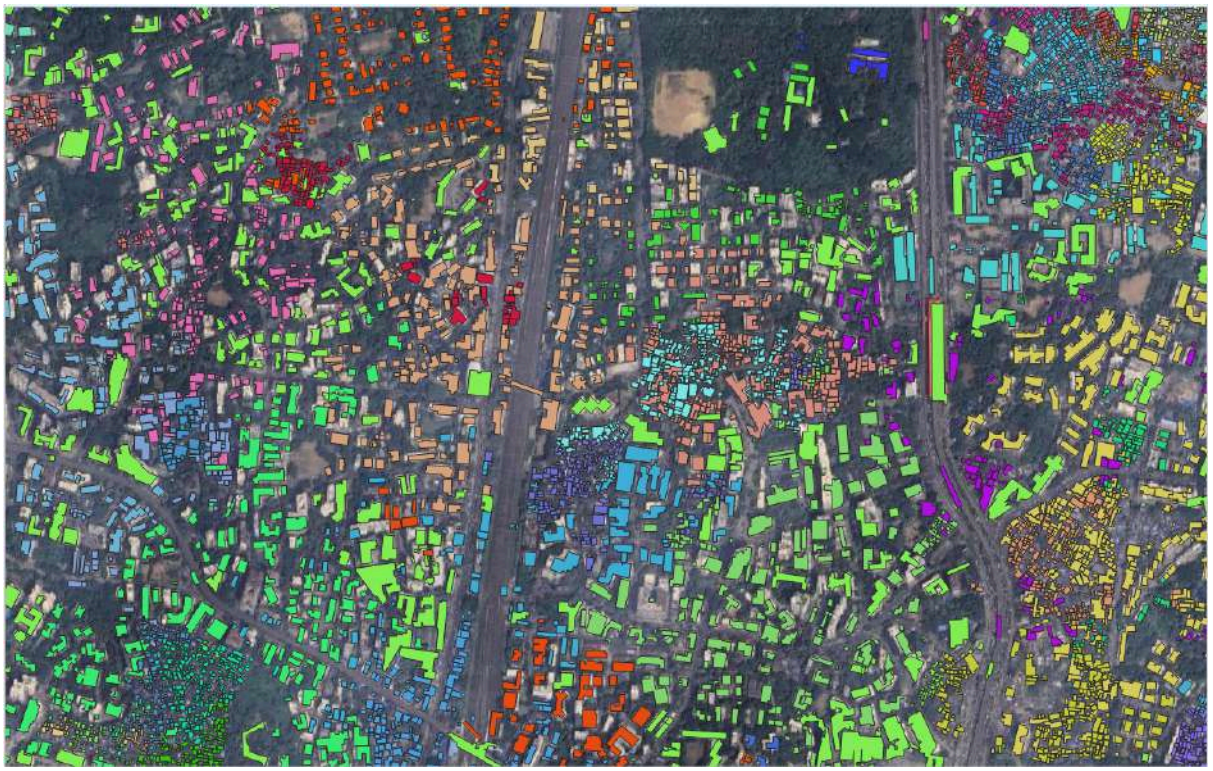


Fine Grained



Region Growing Clusters

Louvain Clusters



Subwards

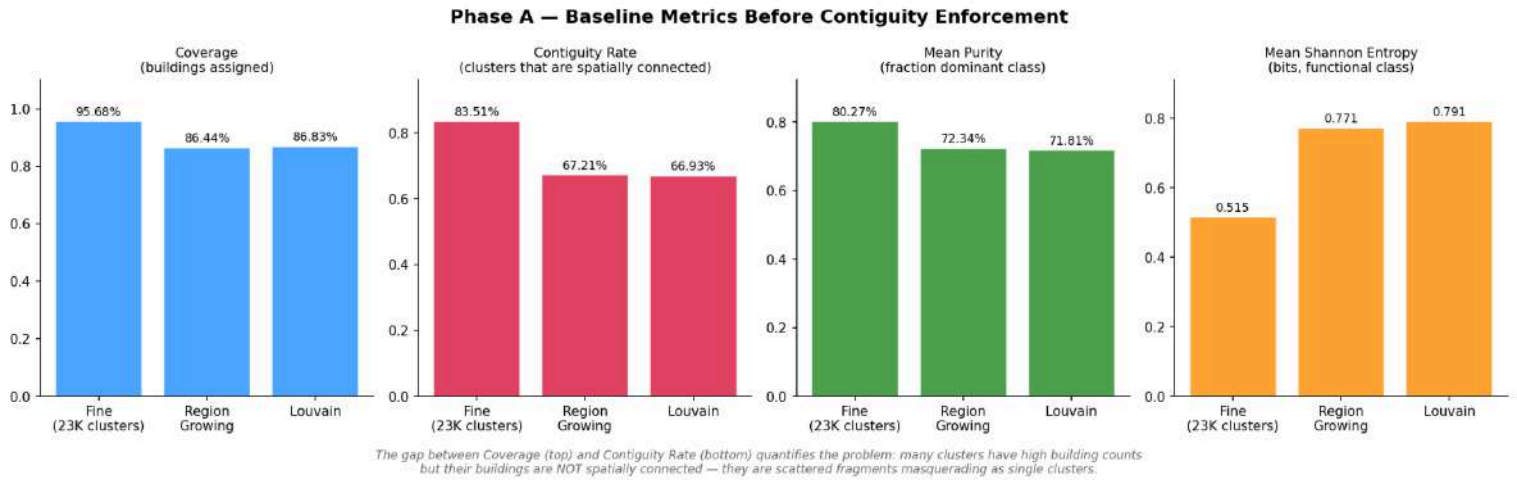
8 Geographic Contiguity

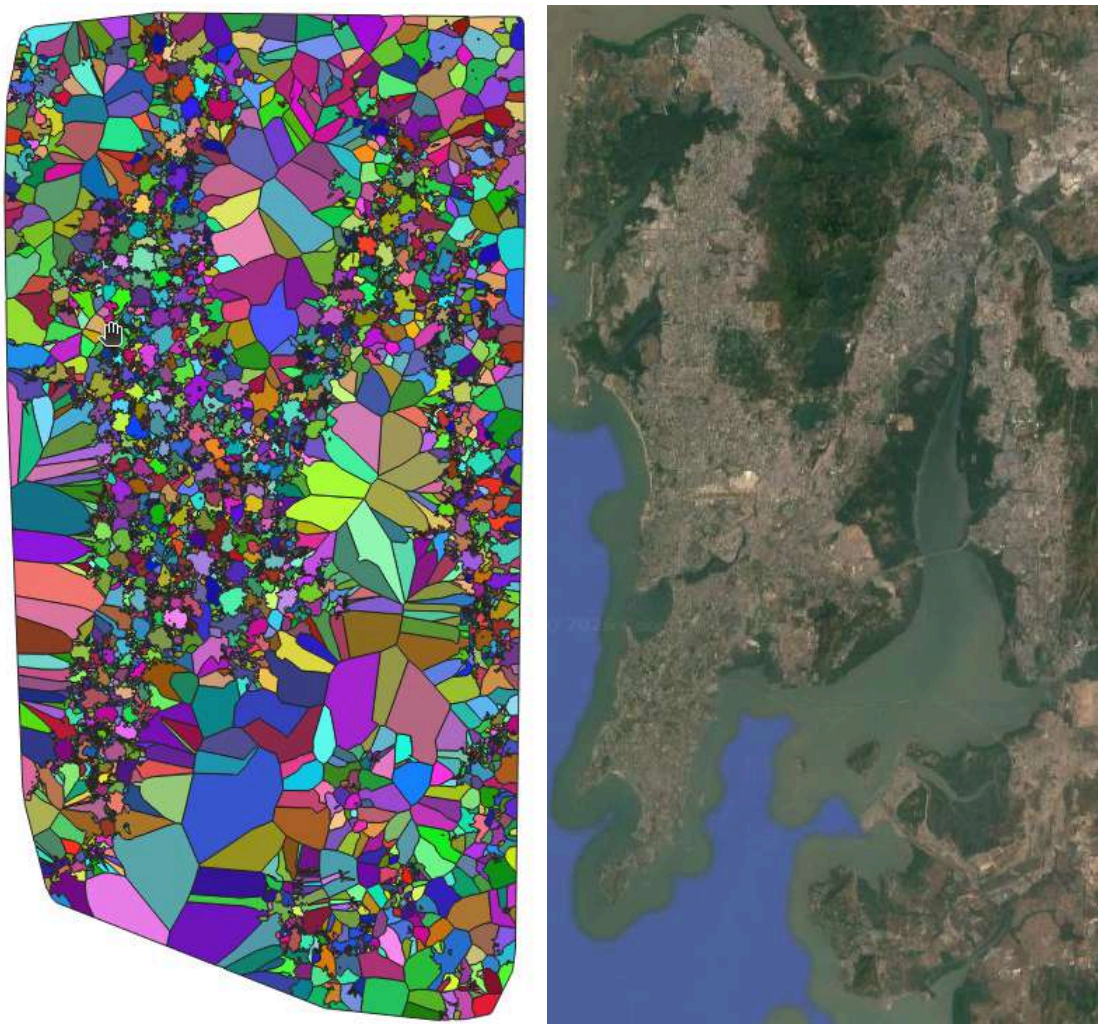
The four clustering methods developed in this project, fine grained spectral clustering, region growing, Louvain community detection, and economic subwards, each optimise for attribute homogeneity and coverage, but none enforce geographic contiguity. A cluster produced by any of these methods can contain buildings scattered across physically disconnected parts of the city while still achieving strong quality metrics, because the quality metrics measure feature similarity, not spatial connectedness.

As noted in the project review, this means one can obtain clusters with excellent scores that have no meaningful geographic coherence. This section addresses that gap. The geographic contiguity pipeline is applied independently to all four cluster levels, transforming each into a spatially coherent partition of the city.

Deliverable	Base clustering	Target coverage
`fine_cluster_geo`	Fine-grained spectral (23,463 clusters)	100%
`rg_cluster_geo`	Region growing (10,935 clusters)	100%
`louvain_cluster_geo`	Louvain community detection (8,707 clusters)	100%

Each deliverable is validated against validation metrics CV, silhouette score, conductance, purity, Shannon entropy, and the new spatial metrics (footprint coverage, volumetric density, perimeter ratio)





Side By Side view of the city's subwards

8.1 Alpha shapes (Concave hulls)

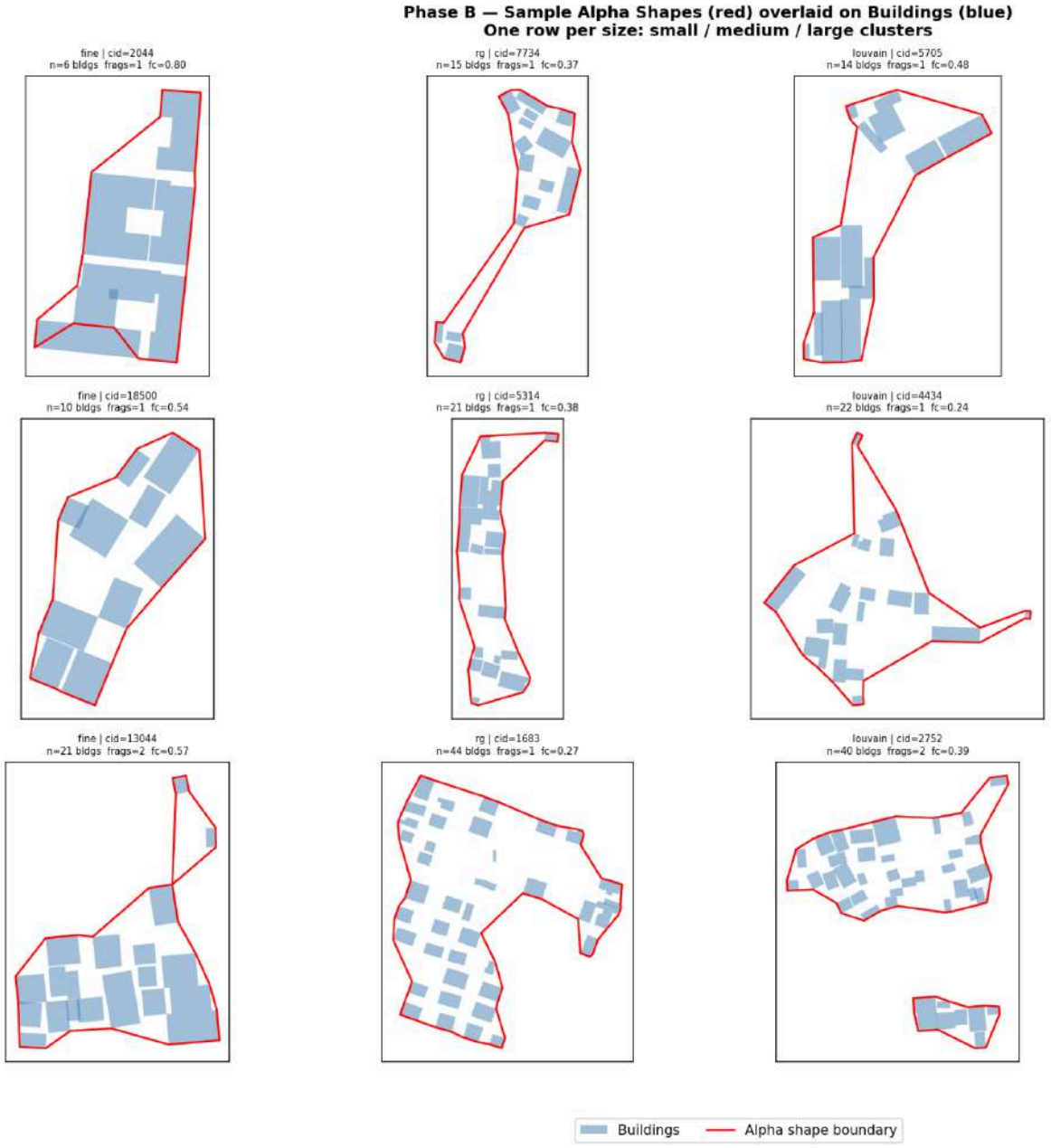
The first step in enforcing geographic contiguity is to construct a spatial boundary for each cluster.

This project uses the **concave hull**, also known as the alpha shape. The alpha shape is parameterised by a value α which controls how tightly the boundary fits around the cluster's buildings. At α = 0, the alpha shape degenerates to the convex hull. As α increases, the boundary follows the actual shape of the cluster more closely, hugging concavities

an edge between two boundary points is included if a circle of radius 1/α can be placed passing through both points without enclosing any other building in the cluster.

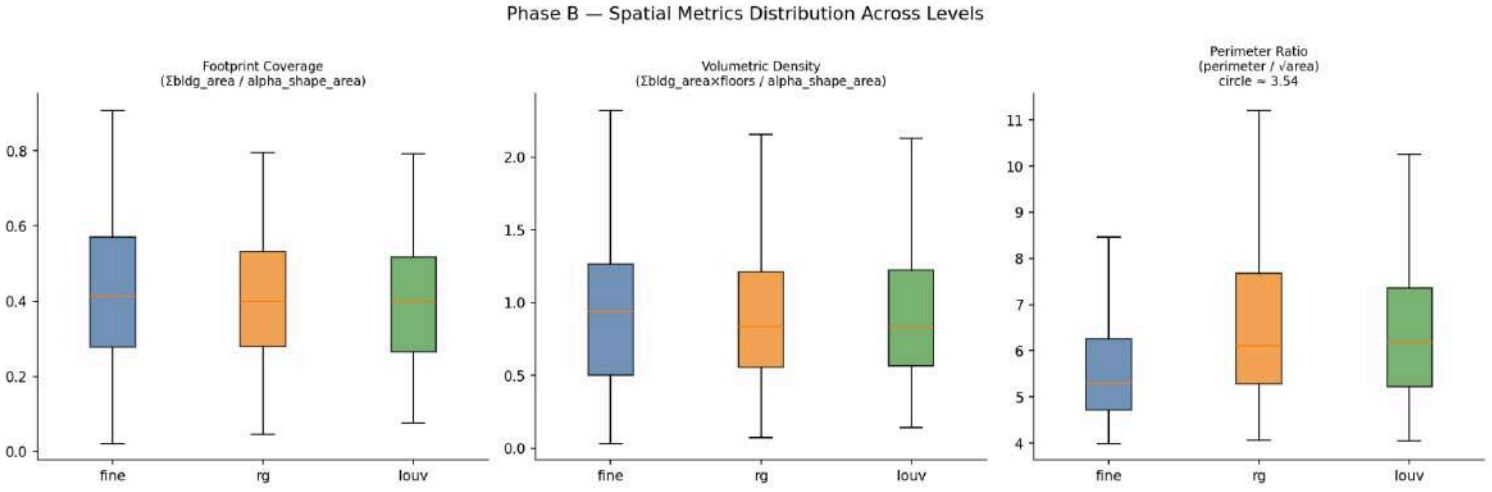
The alpha shape is computed using the actual polygon vertices of each building not the building centroids. This ensures the boundary encloses the entire footprint of every building in the cluster, not just its centre point. A building on the edge of a cluster is fully contained within the cluster's alpha shape even if its centroid is close to the boundary.

They are used to detect contested buildings
The alpha shape of each final cluster defines its geographic boundary on the map, and is stored as a polygon in the output regions table.



New spatial metrics derived from mega polygon geometry

Metric	Formula	Interpretation
footprint_coverage	Σ(area_sqft) / alpha_shape_area	1.0 = fully packed, 0.1 = very sparse
volumetric_density	Σ(area_sqft × floors) / alpha_shape_area	FSI analog — separates informal from high-rise
vertical_sparsity	alpha_shape_area / Σ(area_sqft × height_m)	Sky per unit of built volume
perimeter_ratio	perimeter / √(alpha_shape_area)	Compactness — circle ≈ 3.54, L-shapes higher
fragmentation	Count of sub-polygons	1 = solid territory, >1 = islands



Level	k	CV	CV-height	Purity	Entropy	Mean size	Coverage
Fine	26,502	0.193	0.244	0.663	1.046	19.8	95.6%
RG	10,935	0.227	0.278	0.620	1.241	43.3	86.4%
Louvain	8,707	0.233	0.288	0.615	1.260	54.7	86.7%

8.2 Automatic Zoning Procedure (AZP)

Once contested buildings are identified, the Automatic Zoning Procedure (AZP) is used to redistribute them. AZP is a local search optimization originally proposed by Openshaw (1977) as a method for finding the best grouping of spatial units into contiguous regions. It operates by repeatedly evaluating candidate swaps at cluster boundaries and accepting moves that improve the overall quality of the partition, subject to the hard constraint that every cluster must remain geographically contiguous after each move.

The procedure begins from a feasible starting solution — one where every cluster is already a connected component in the building adjacency graph. Since the existing cluster assignments do not satisfy contiguity, a pre-step is first applied: any building that belongs to

a cluster but has no graph path to the rest of that cluster without crossing another cluster's territory is a disconnected fragment. These fragments are stripped from their current assignment and reassigned to whichever adjacent cluster they actually touch. This produces the feasible starting point that AZP requires. In practice, this pre-step is expected to affect very few buildings as the original spectral clustering was run on a Delaunay triangulation, a planar graph, which means the graph edges themselves encode spatial adjacency.

A cluster is selected at random from the full list. Every building on the boundary of that cluster, adjacent in the Delaunay graph to at least one building outside the cluster, is evaluated as a candidate for reassignment to a neighbouring cluster. A move is accepted only if two conditions are met:

1) The cluster the building is leaving must remain contiguous after its removal,
2) The move must produce a net improvement in the combined quality score across both affected clusters. The quality score incorporates CV, silhouette score, conductance, Shannon entropy, and the new spatial metrics, with additional weight given to improvements in whichever metric is currently weakest in the affected cluster.

To avoid being trapped in a local optimum, Simulated Annealing AZP is used rather than the plain greedy variant. This allows the algorithm to occasionally accept a move that slightly worsens the objective, with the probability of accepting such moves decreasing over time.

Level	Moves accepted	Clusters (before → after)
Fine	871,000	26,498 → 14,798
RG	469,000	10,935 → 10,362
Louvain	452,000	8,707 → 8,295

8.2.1 Additional Metrics

Metric	Score	Interpretation
`height_gradient`	1.345	Strongest new metric; height variability differs substantially across cluster types
`volumetric_density`	0.615	Effectively separates high-rise and informal morphologies
`cv_height`	0.384	Captures intra-cluster vertical variation
`road_exposure`	0.338	Reflects differing accessibility and street integration
`footprint_coverage`	0.183	Moderate sensitivity to built-up intensity
`perimeter_ratio`	0.151	Weak geometric boundary descriptor
`shannon_entropy`	0.144	Limited but nonzero diversity signal

`cv_area`	0.127	Weakest new metric; area variability contributes little to discrimination

8.2.2 Prev Existing Metrics

Feature	Score	Interpretation
`distance_to_road`	**389.0**	Strongest discriminator by a large margin; clusters are heavily structured by road proximity
`density_ratio`	72.6	Strong separation across urban density patterns
`height_m`	46.2	Building height contributes significantly to cluster differentiation
`area_per_neighbor`	43.7	Captures local spatial packing characteristics
`floors`	42.5	Useful proxy for vertical development intensity
`area_sqft`	27.3	Moderate contribution from footprint size
`convexity`	24.9	Shape regularity provides some separation
`compactness`	18.0	Weak-to-moderate morphological discriminator
`height_per_floor`	14.0	Limited additional structural information
`elongation`	4.9	Weakest existing feature; minimal discriminative utility

Level	k	CV	CV-height	Purity	Entropy	Mean size	Coverage
Fine	26,498	0.183	0.225	0.709	0.907	19.8	95.6%
RG	10,935	0.216	0.259	0.669	1.088	43.3	86.4%
Louvain	8,707	0.223	0.269	0.664	1.108	54.7	86.7%

8.3 Fisher Ratio Merge

AZP settles boundaries and enforces contiguity, but can leave behind small residual clusters, fragments because they were technically connected, but are too small or too similar to a neighbour to justify existing independently. These are cleaned up before region growing begins. This also helps clean up bad fragments that survived previous merge steps as there is better information available.

The merge criterion is the Fisher ratio, it measures the separation between clusters relative to their internal variance, a high Fisher ratio means clusters are tight internally and well-separated from each other. For every pair of spatially adjacent clusters where at least one is below a minimum size threshold, the pipeline evaluates what the Fisher ratio would be

if the two were merged. If the ratio does not drop beyond a set tolerance, and the merged cluster's CV and Shannon entropy remain within acceptable bounds, the merge is accepted. Clusters are processed smallest-first, and the global Fisher ratio is recomputed after each merge before the next candidate pair is evaluated.

The merge tolerance is loosened iteratively until the mean and median cluster size reaches approximately 100 buildings, matching the scale of the existing economic subwards and the general economic subwards of Mumbai but with full geographic contiguity enforced.

Level	k	CV	CV-height	Purity	Entropy	Mean size	Coverage
Fine	14,798	0.166	0.182	0.824	0.580	35.4	95.6%
RG	10,362	0.185	0.187	0.842	0.573	45.7	86.4%
Louvain	8,295	0.194	0.196	0.833	0.603	57.4	86.7%

Level	k (after AZP)	k (after merge)	Mean size	Median size
Fine	14,798	5,147	101.9	33
RG	10,362	2,262	209.5	135
Louvain	8,295	3,750	126.9	57

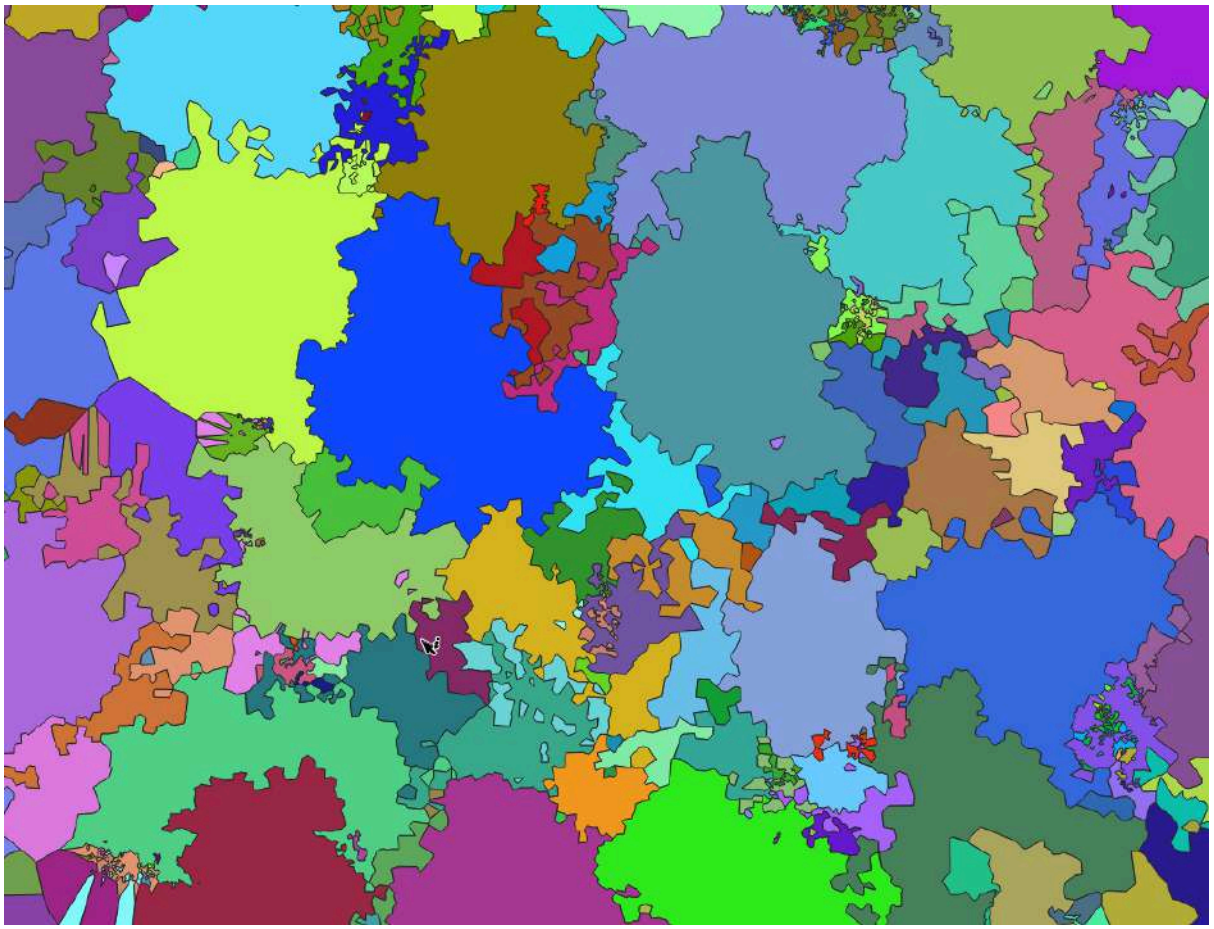
8.4 Organic Growth (Attempted Approach)

After AZP and merging, clusters had clean contiguous boundaries but gaps remained, buildings unassigned by the original clustering. The initial approach to closing these gaps was a Dijkstra-style priority queue expansion: every cluster would expand simultaneously, absorbing unassigned frontier buildings ranked by attraction score (feature similarity divided by distance to cluster edge). Two hard ceilings were set, cluster area and cluster radius could each grow by at most 15% in total, over a maximum of 20 passes.

This produced the result shown below. Clusters at the edge of the city, along the coastline and at the northern edge, had no competing clusters to block their expansion. They absorbed all unassigned buildings in their direction and then continued expanding via Voronoi - like growth into open land, sea, and empty space, producing regions orders of magnitude large. than the intended ~100-building scale. The 15% geometric cap was insufficient because. peripheral clusters started from small absolute radii, meaning 15% growth still allowed large. absolute territorial gains into unclaimed space.

8.5 Voronoi Tessellation (FinalConstruction)

The region polygon construction was built using Voronoi tessellation of building centroids (classified after the Fisher ratio merge). Each building centroid acts as a generator point and owns the space closest to it, its Voronoi cell. All cells belonging to the same cluster are dissolved into a single region polygon.This guarantees by construction that regions are non-overlapping and gap-free. Every point in space is closest to exactly one building centroid and therefore belongs to exactly one region. There is no growth process, no cap to tune, and no edge effect. The geometry is determined entirely by the spatial distribution of buildings that have already been assigned by the preceding pipeline steps. Four corner sentinel points placed at the city diameter outside the extent ensure all Voronoi cells are finite. This could potentially be changed to a concave hull around the city's polygons to avoid large subwards due to extremely sparse regions (including the arabian sea)



City Scape by subwards

Level	k	CV	CV-height	Purity	Entropy	Mean size	Coverage
Fine	5,147	0.280	0.331	0.709	0.990	106.6	100%
RG	2,262	0.411	0.498	0.585	1.468	242.6	100%
Louvain	3,750	0.348	0.415	0.670	1.204	146.3	100%

References

1. Yang, D. and Fountoulakis, K. (2023). Weighted flow diffusion for local graph clustering with node attributes: an algorithm and statistical guarantees. *arXiv preprint* arXiv:2301.13187v1.
2. von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17(4), pp. 395–416.
3. Lipton, R.J. and Tarjan, R.E. (1979). A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2), pp. 177–189.
4. Ester, M., Kriegel, H.-P., Sander, J. and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *KDD-96*, pp. 226–231.
5. Shewchuk, J.R. (1996). Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. *Applied Computational Geometry*, Lecture Notes in Computer Science, Vol. 1148, Springer.
6. Pedregosa, F. et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, pp. 2825–2830.
7. Hartigan, J.A. and Hartigan, P.M. (1985). The dip test of unimodality. *The Annals of Statistics*, 13(1), pp. 70–84.
8. Blondel, V.D. et al. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10), P10008.
9. Openshaw, S. (1977). A geographical solution to scale and aggregation problems in region-building, partitioning and spatial modelling. *Transactions of the Institute of British Geographers*, 2(4), pp. 459–472. *(Original AZP paper)*
10. Openshaw, S. and Rao, L. (1995). Algorithms for reengineering 1991 Census geography. *Environment and Planning A*, 27(3), pp. 425–446. *(AZP with simulated annealing and tabu search)*
11. Caliński, T. and Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics*, 3(1), pp. 1–27. *(Fisher ratio / variance ratio criterion used in merge step)*
12. Edelsbrunner, H., Kirkpatrick, D. and Seidel, R. (1983). On the shape of a set of points in the plane. *IEEE Transactions on Information Theory*, 29(4), pp. 551–559. *(Alpha shapes / concave hulls)*
13. Assunção, R.M. et al. (2006). Efficient regionalization techniques for socio-economic geographical units using minimum spanning trees. *International Journal of Geographical Information Science*, 20(7), pp. 797–811. *(SKATER — used as reference for contiguity-constrained clustering)*