جامعة الشارقة
UNIVERSITY OF SHARJAH

College of Computing and Informatics
Department of Computer Science
GIS Programming Fundamentals (150153)
Fall 2025 – 2026

Spatial Health Risk Assessment in Sharjah Using GIS & Automated Python-Based Weighted Overlay

AMIRA A ALNAQBI
MAITHA A ALDARMAKI
ANJITHA S NAIR

Table of Contents

Introduction

- Environmental pollution affects public health
- Industry, traffic, and population increase exposure
- GIS helps identify areas of potential health risk

Background & Context

- Rapid urban and industrial growth in Sharjah
- Industrial zones and major roads near residential areas
- High population density increases environmental exposure
- GIS supports spatial integration of risk factors

Problem Statement

1. Environmental risk factors exist as separate datasets
2. Combined health risk is not clearly mapped
3. Difficult to compare exposure between neighborhoods
4. An integrated spatial approach is required

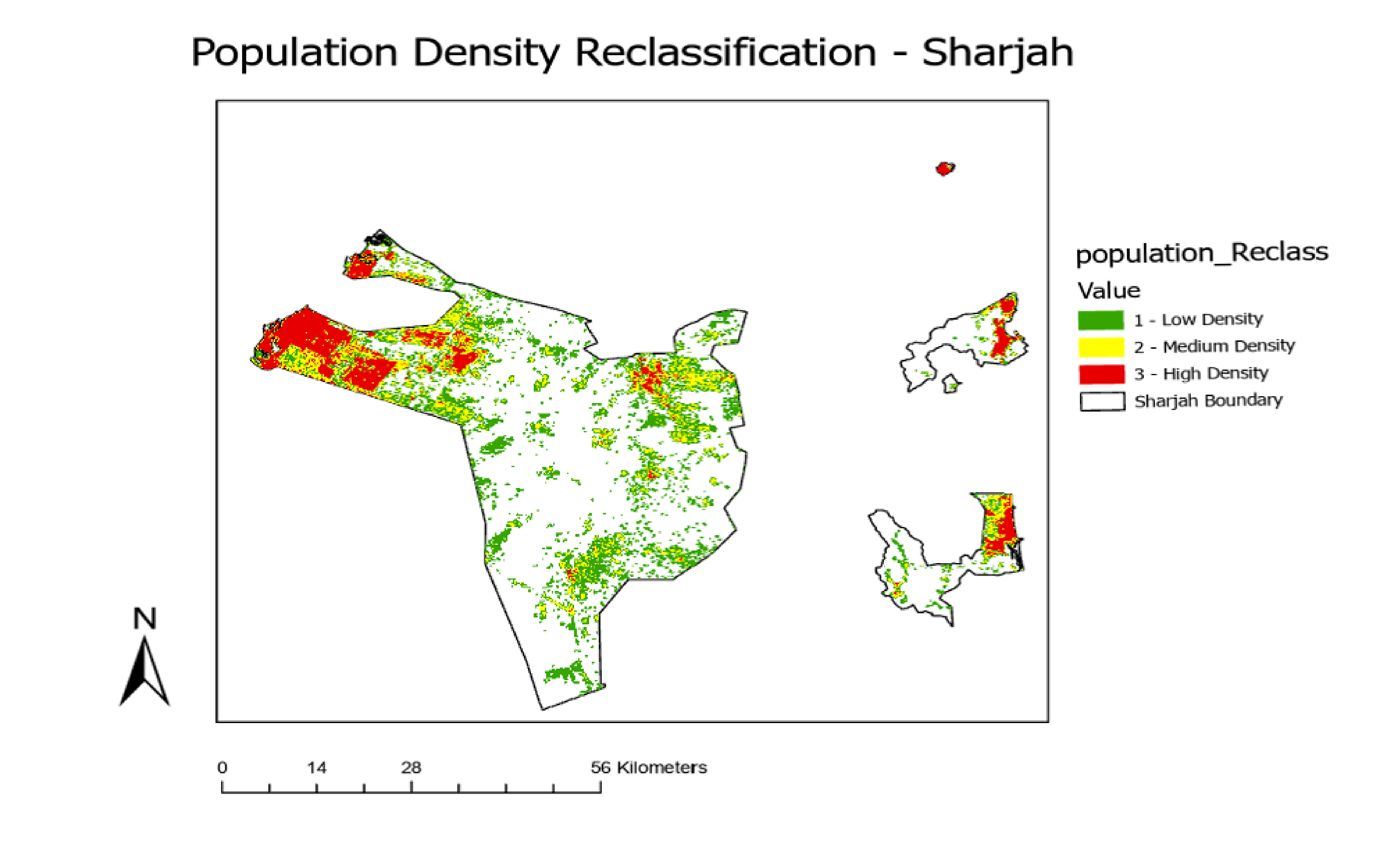
Objectives of the Study

1. Analyze population, industry, and road networks
2. Reclassify datasets into risk levels
3. Apply weighted overlay in GIS
4. Identify potential high-risk zones in Sharjah

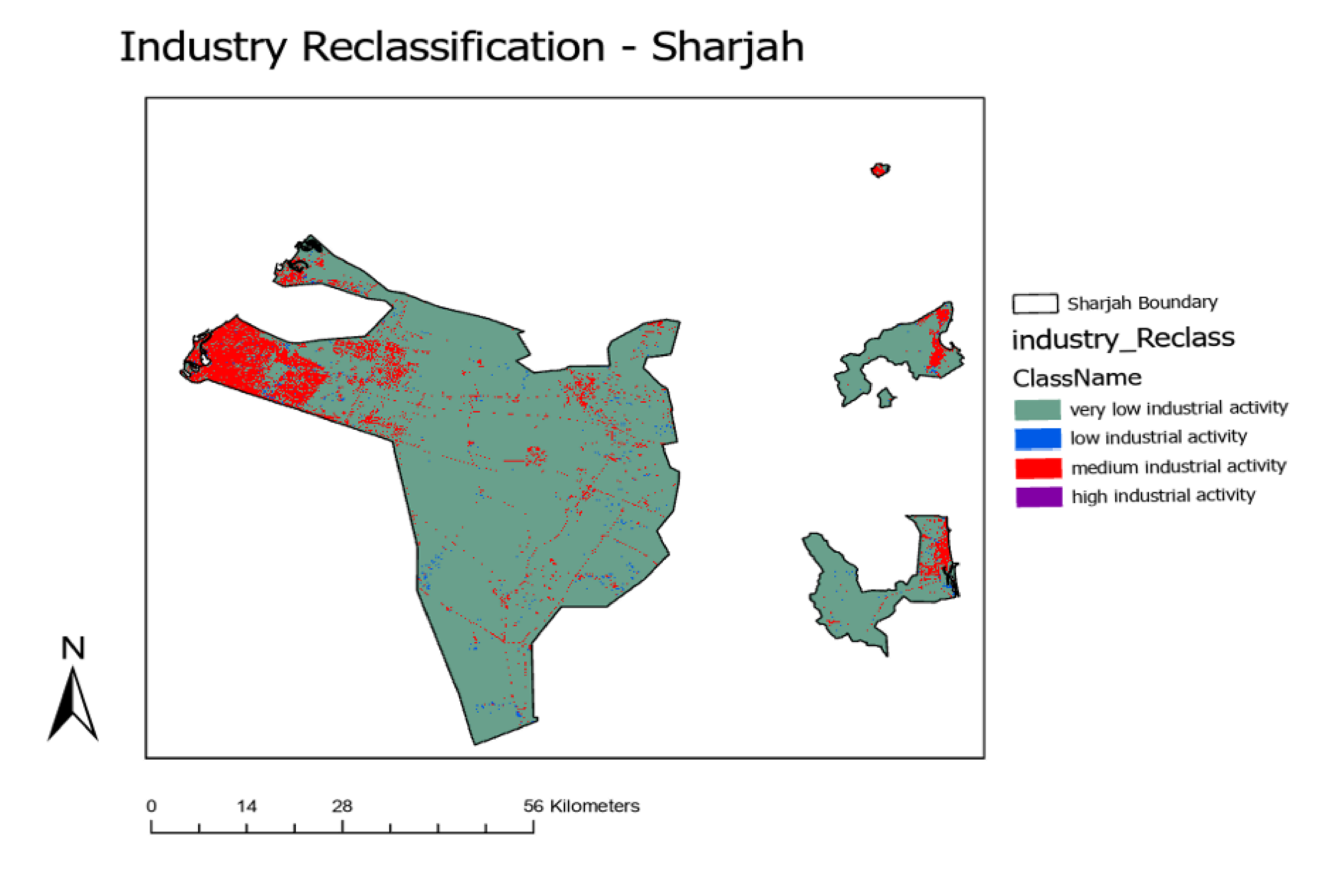
Study Area

- Emirate of Sharjah, UAE
- Urban, industrial, coastal, and inland regions
- Major transport corridors and industrial zones
- Unified study extent for all datasets

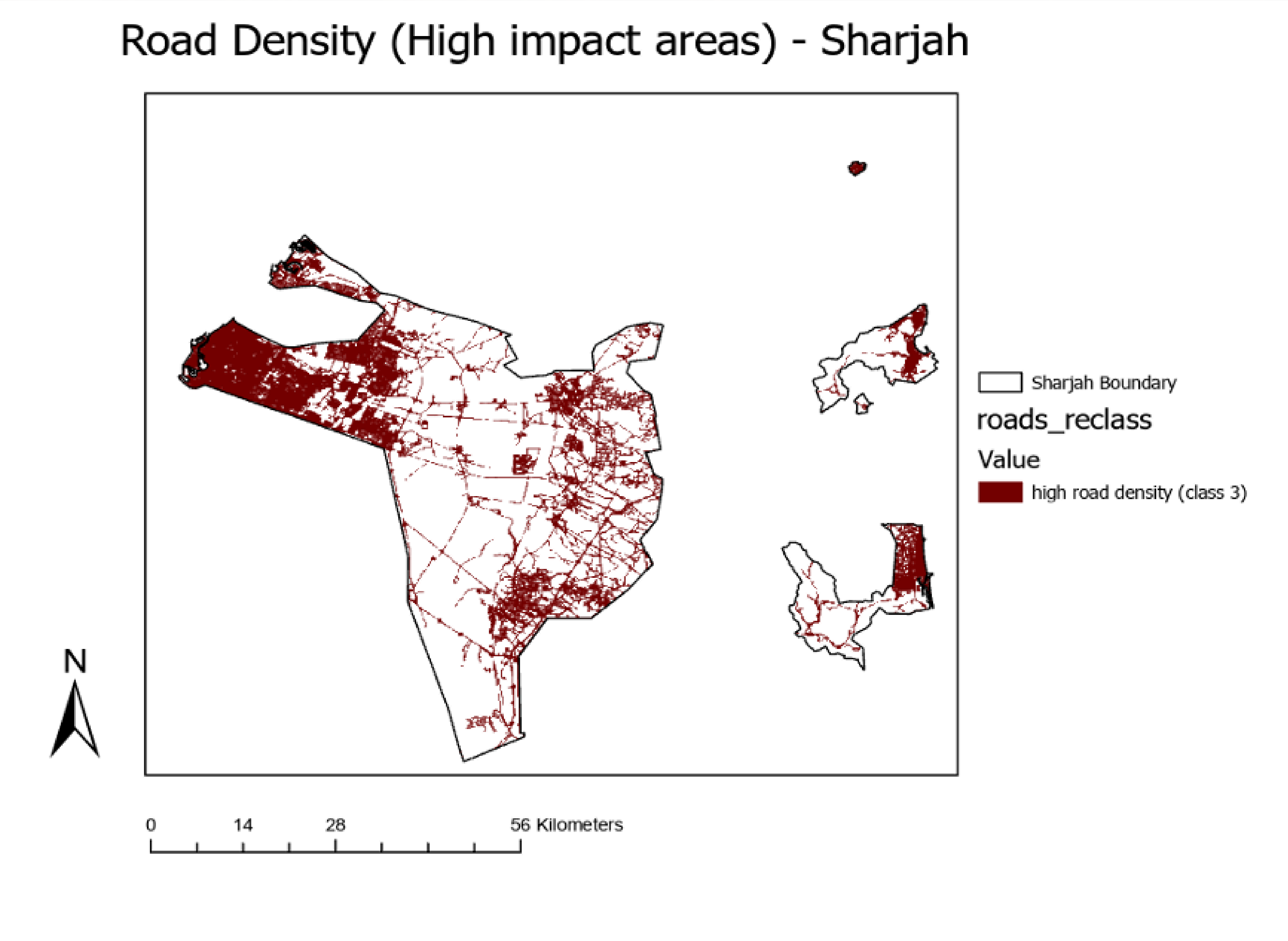
— Input Layers Used in the Analysis



— Input Layers Used in the Analysis

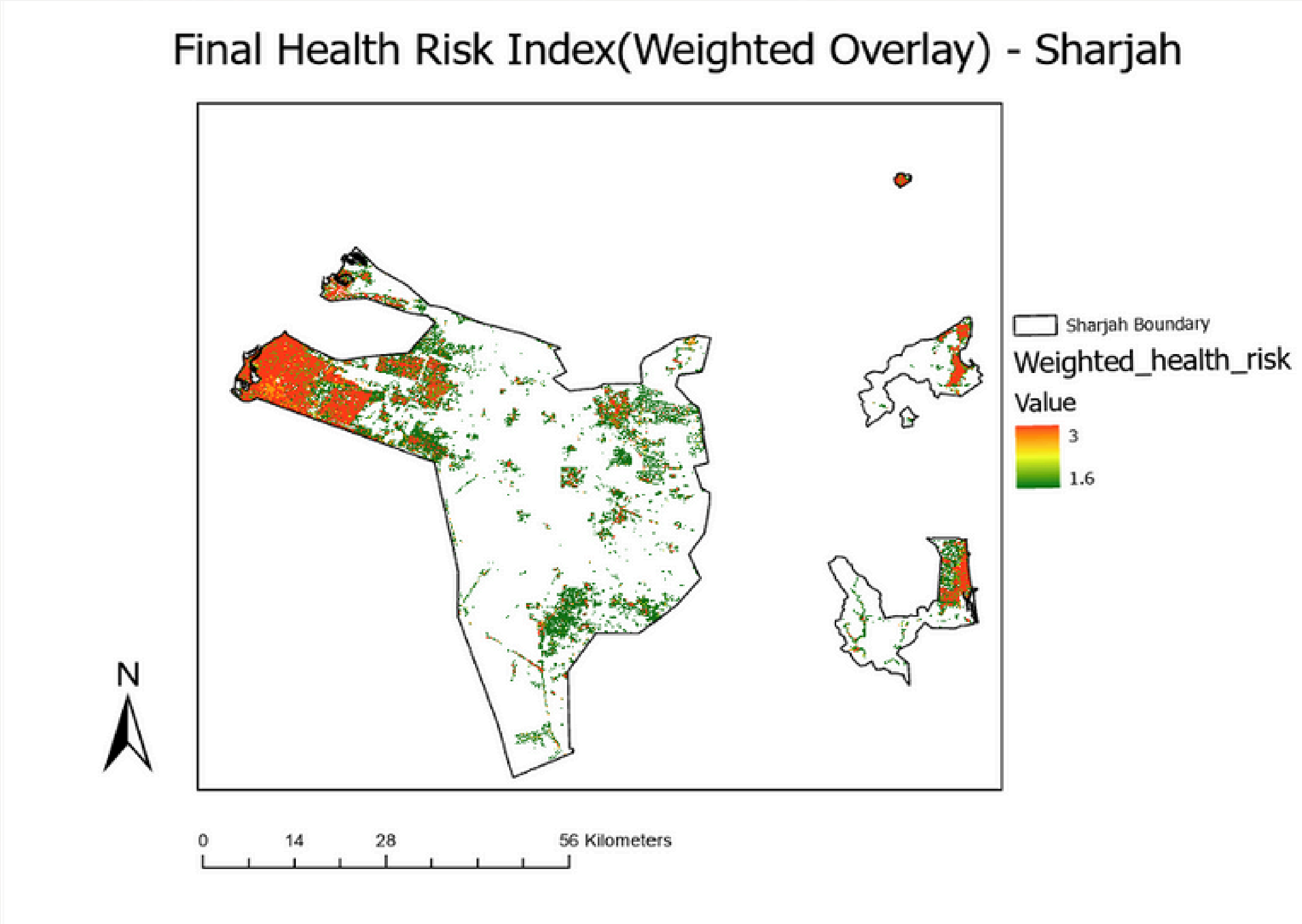


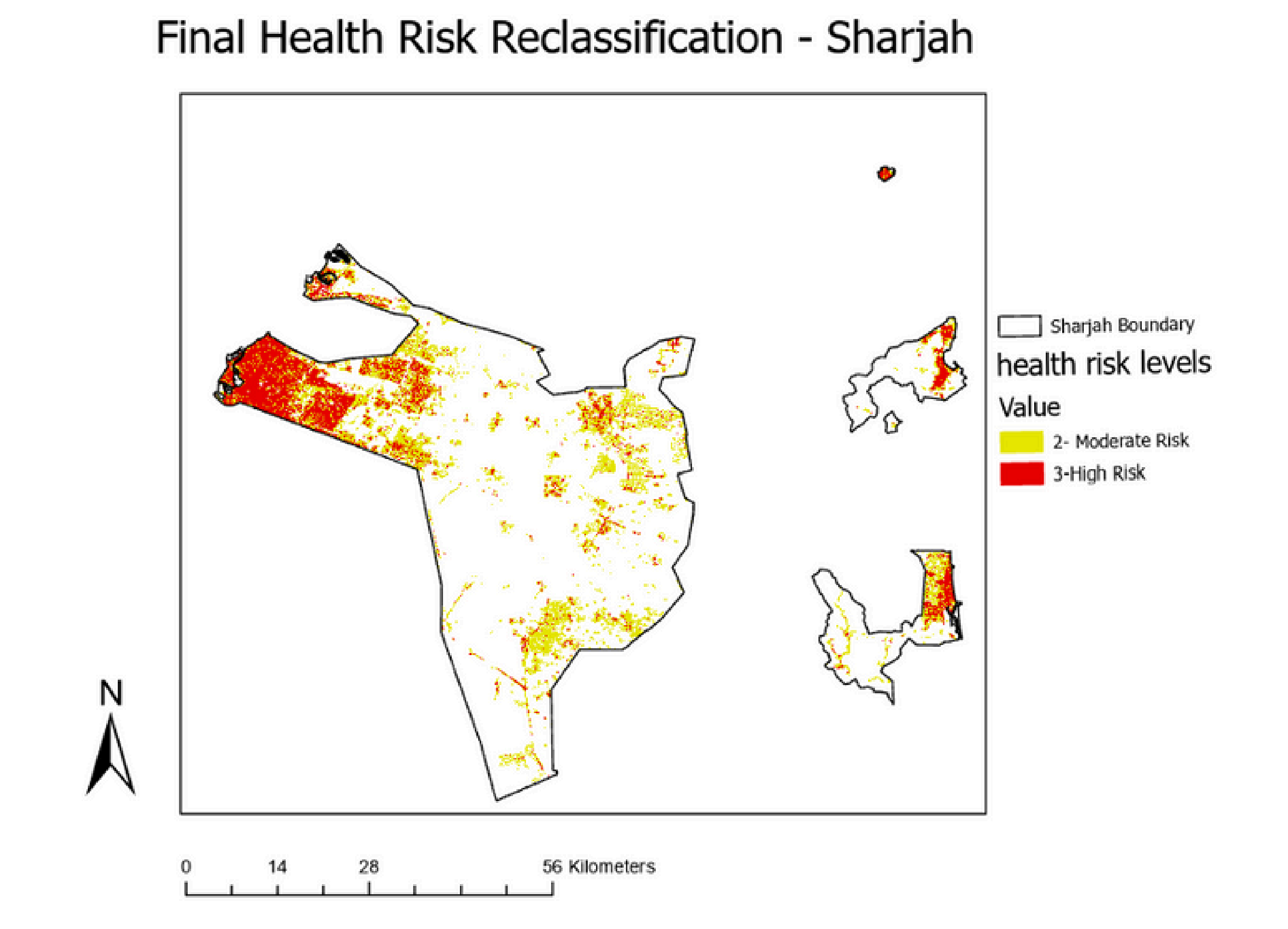
— Input Layers Used in the Analysis



Manual Risk Map

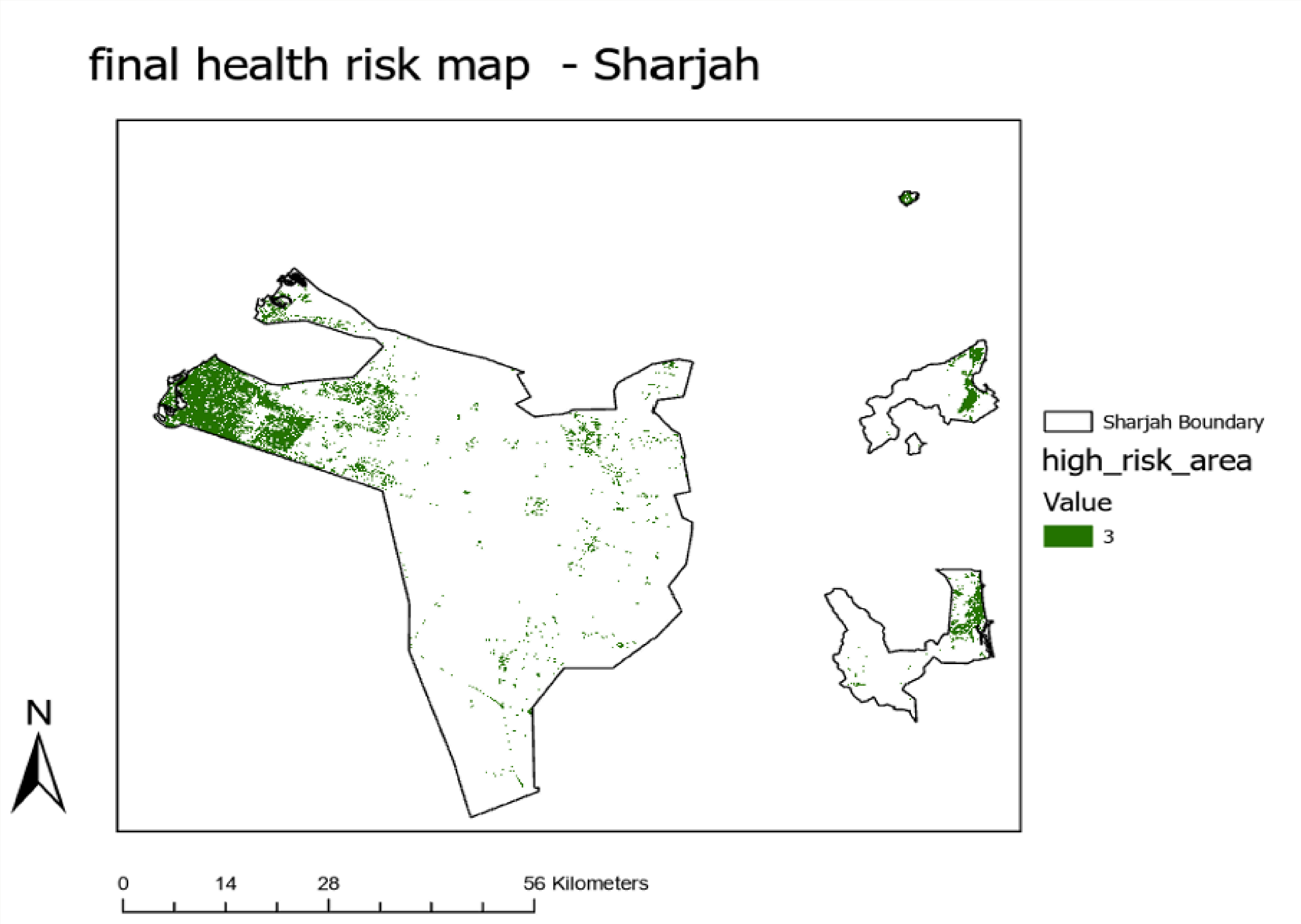
- Population, industry, and roads reclassified
- Layers combined using weighted overlay
- Analysis performed in ArcGIS Pro
- Output represents a health risk index





Interpretation of Results

- High-risk zones near industrial areas
- Dense road networks increase exposure
- Population density amplifies vulnerability
- Industry is the dominant risk factor



Methodology

- Collected geospatial layers (roads, industrial activity, population density, boundary).
- Prepared and standardized data inside a geodatabase.
- Applied manual weighted overlay to generate baseline results.
- Developed Python-based toolbox to automate the weighted overlay process.
- Classified outputs to identify health risk levels.

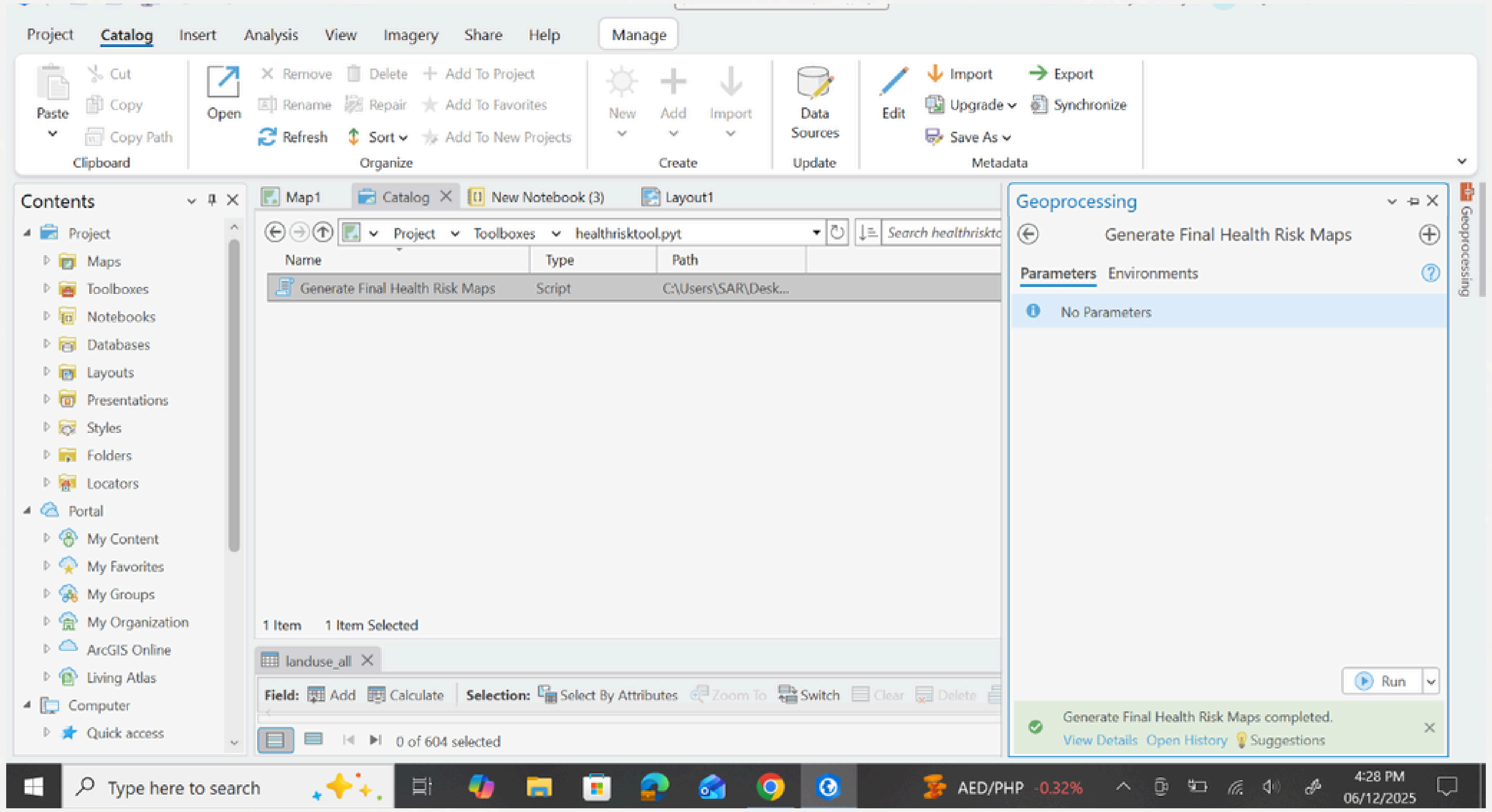
Automated Workflow

Automated Weighted Overlay Process

- Python toolbox replicates the manual workflow automatically.
- Reads and converts required layers.
- Applies assigned weights and creates final risk map.
- Reduces human error, speeds processing, and supports repeatability.

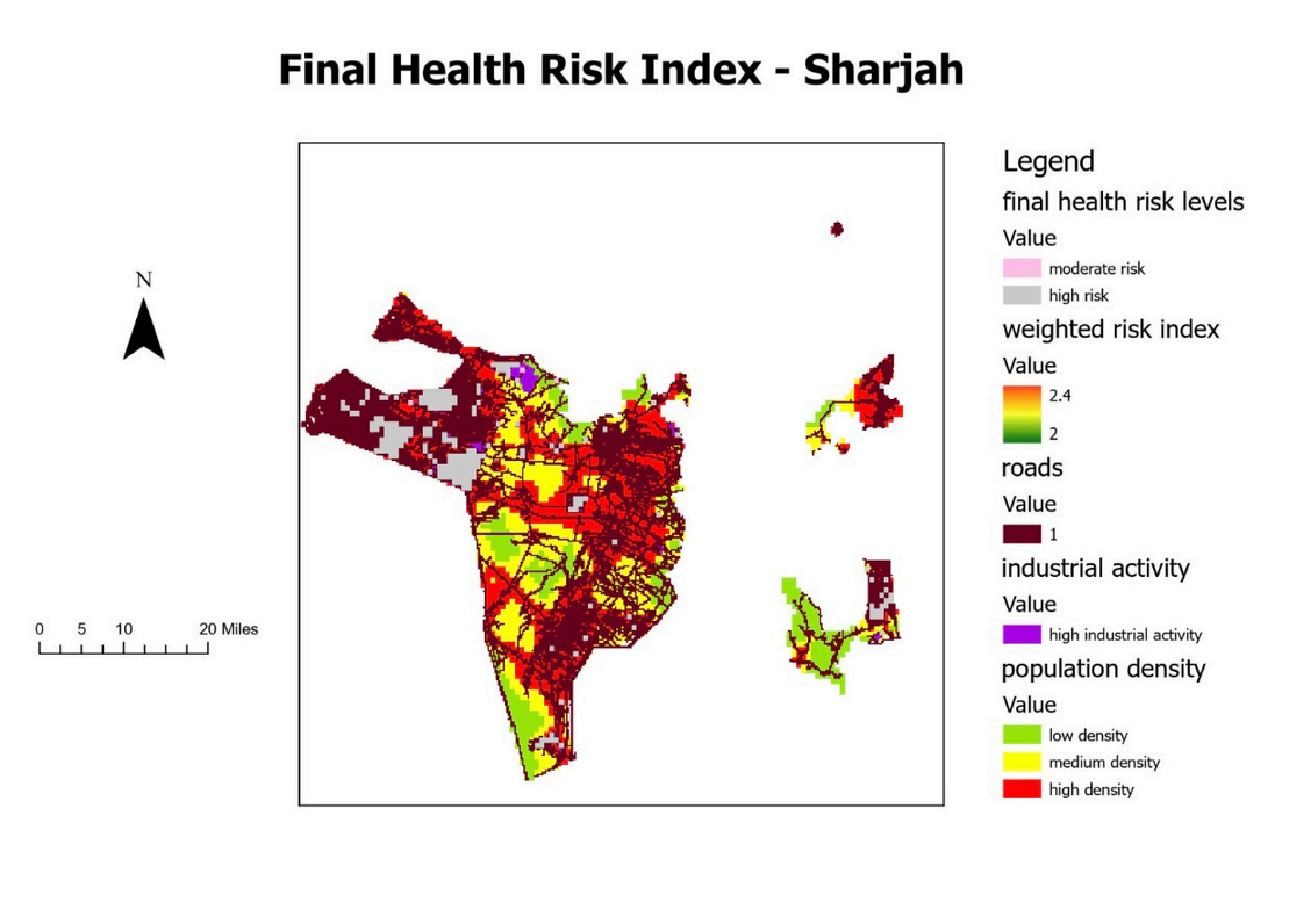
Automated Workflow Execution (Python Toolbox)

- Custom toolbox developed in ArcGIS Pro
- Runs full weighted overlay steps automatically
- Converts layers, applies weights, and generates output
- Reduces manual effort and human error



Final Automated Health Risk Map

- Output generated directly from Python script
- Combined risk surface of all factors
- High-risk zones align with dense urban + industrial locations
- Supports planning by visualizing priority zones



Script Logic Behind the Automation

```python
import arcpy
from arcpy.sa import *

class Toolbox(object):
    def __init__(self):
        self.label = "Health Risk Mapping Toolbox"
        self.alias = "healthrisk"
        self.tools = [HealthRiskTool]

class HealthRiskTool(object):
    def __init__(self):
        self.label = "Generate Final Health Risk Maps"
        self.description = "Reclassifies population, industry and roads and

    def getParameterInfo(self):
        return []

    def isLicensed(self):
        return True

    def updateParameters(self, parameters):
        return

    def updateMessages(self, parameters):
```

```python
                                    [300, 99999, 3]]))
        pop.save("population_density")

        industry = Reclassify(industry_in, "Value",
                              RemapRange([[1, 1, 1],
                                          [2, 2, 2],
                                          [3, 3, 3]]))
        industry.save("industrial_activity")

        roads = Reclassify(roads_in, "Value",
                           RemapRange([[0, 0.0001, 1],
                                       [0.0001, 0.001, 2],
                                       [0.001, 9999, 3]]))
        roads.save("roads_high_impact")

        weighted = (industry * 0.5) + (roads * 0.3) + (pop * 0.2)
        weighted.save("weighted_risk_index")

        final = Reclassify(weighted, "Value",
                           RemapRange([[0, 2, 1],
                                       [2, 5, 2]]))
        final.save("final_health_risk_levels")

        arcpy.AddMessage("Health risk layers created successfully.")
```

```python
    def execute(self, parameters, messages):

        gdb = r"C:\Users\SAR\AppData\Local\Microsoft\Olk\Attachments\ooa-5
        arcpy.env.workspace = gdb
        arcpy.env.overwriteOutput = True
        arcpy.CheckOutExtension("Spatial")

        pop_raster_name = "popdensity"
        industry_raster_name = "landuse_ras"
        roads_raster_name = "roads_ras"

        pop_in = Raster(pop_raster_name)
        industry_in = Raster(industry_raster_name)
        roads_in = Raster(roads_raster_name)

        pop = Reclassify(pop_in, "Value",
                         RemapRange([[0, 100, 1],
                                     [100, 300, 2],
                                     [300, 99999, 3]]))
        pop.save("population_density")

        industry = Reclassify(industry_in, "Value",
                              RemapRange([[1, 1, 1],
                                          [2, 2, 2],
                                          [3, 3, 3]]))
```

The Python script mirrors the manual weighted overlay steps by converting layers to raster, normalizing them, applying weights, and generating a combined output. It also automatically classifies the results into readable risk levels, making the process faster, more consistent, and reusable for future studies without rebuilding the workflow.

Comparison & Validation of GIS Health Risk Models

Manual vs. Python

Validate model outputs against known census data and physical infrastructure patterns.

Confirms a highly uneven distribution, with 91% of the population residing in Sharjah city, leaving the interior vast and sparsely populated.

- Manual Map (Figure 1): Closely matches census data, showing low density throughout the interior and high-density clusters only in major urban centres (Sharjah city, East Coast enclaves).
- Python Map (Figure 2): Exaggerates population spread, displaying large, continuous blocks of moderate and high density across much of the territory.
- Conclusion: Figure 1 accurately depicts the concentration and absence of people. Figure 2 overstates and dilutes the true distribution, likely due to coarse processing or inappropriate parameters.

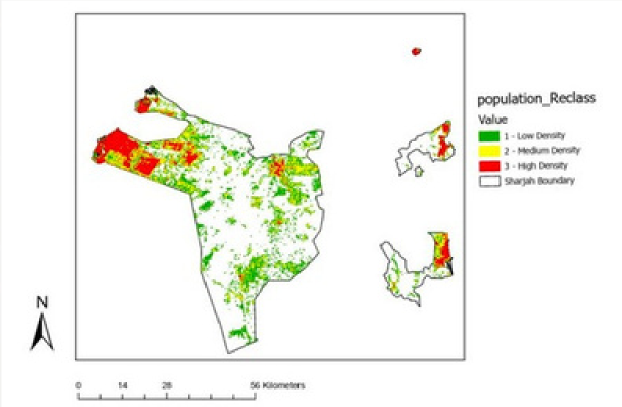


Figure 1

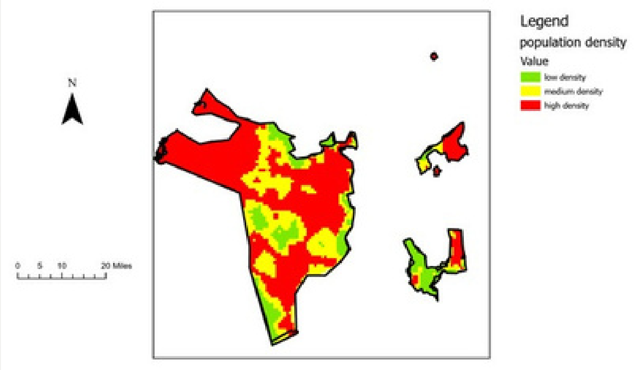


Figure 2

Alignment with Infrastructure & Spatial Coherence

The main health risks (industry, roads, high population) are physically grouped along the main city and coastal corridor. Any accurate map must reflect this clustered pattern.

Manual Map (Figure 4 & 7):

- Figure 4 clearly shows a high-risk pattern matching the concentrated economic activity and vehicle emissions.
- Figure 7 shows a clear continuous high-risk zone with a broad value range (1.6-3)

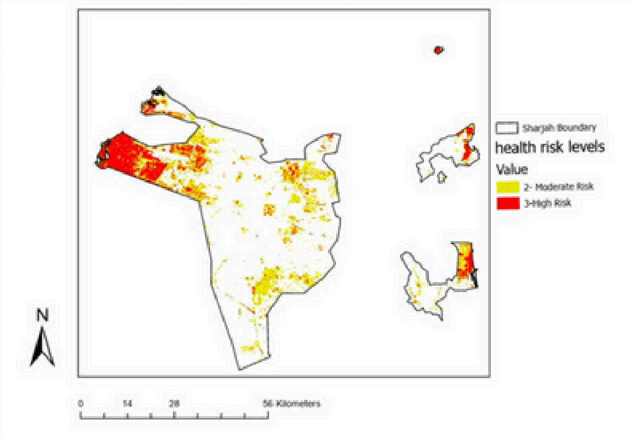


Figure 4

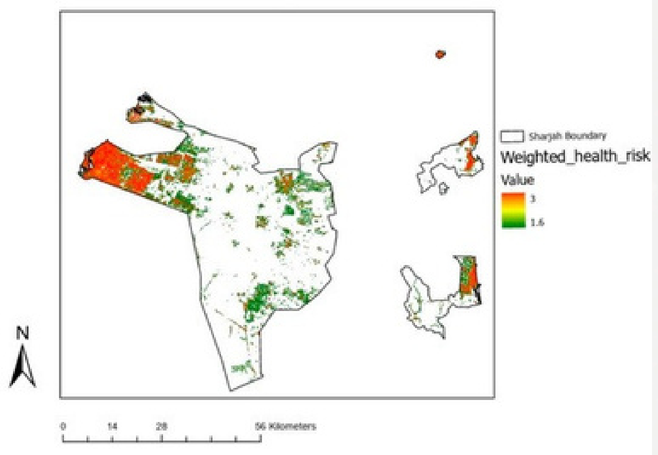


Figure 7

Alignment with Infrastructure & Spatial Coherence

The main health risks (industry, roads, high population) are physically grouped along the main city and coastal corridor. Any accurate map must reflect this clustered pattern.

Python Map (Figure 3 & 8):

- Figure 3 shows risk as scattered isolated spots that fail to match the real clustered distribution.

- Figure 8 has a fragmented spatial pattern and a narrow value range (2-2.4), under-representing true cumulative risk.

Figure 4 and 7 offer a more accurate and logical depiction of health risk by reflecting the true spatial reality.

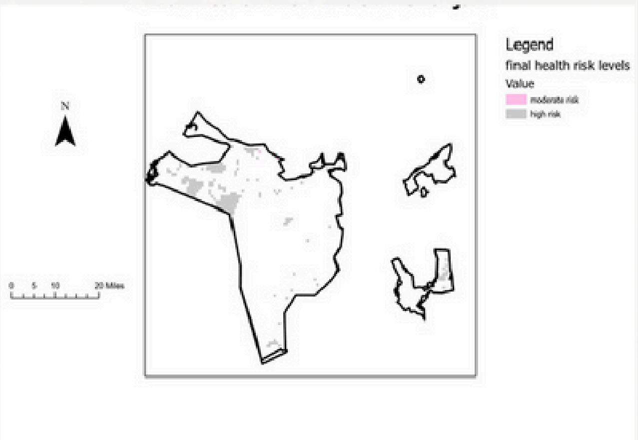


Figure 3

Figure 8

Discussion

Accuracy & Reliability Discussion

- The Manual Weighted Overlay outputs consistently demonstrated higher spatial accuracy and alignment with ground-truth data compared to the automated Python outputs.
- The manual maps provided the expected continuous high-risk zones that planners find actionable.
- The Python maps resulted in fragmented patterns and isolated risk patches, which miss the regional and cumulative nature of environmental risk.

While both methods shared the same logic, the Manual Process proved superior in capturing the critical spatial nuances required for an effective health risk assessment model.

Technical Disparity

Why Did Python Fail to Match Accuracy?

The manual reclassification steps allowed for a precise, visually guided selection of class breaks. The automated script likely used a generic or default classification scheme (e.g., Equal Interval), leading to the exaggeration of population spread (Figure 2).

The lack of spatial continuity and fragmented high-risk zones (Figure 8) may be due to problems with the automated weighting, classification, or spatial aggregation methods within the script.

Small errors may have accumulated during automated raster conversion or resample steps, introducing noise or shifting boundaries that were not present in the carefully controlled manual process.

Efficiency of automation does not guarantee accuracy. Future work must focus on embedding the exact, validated manual parameters into the Python script.

Conclusion

- The study successfully created and validated an integrated GIS-based health risk assessment for Sharjah, confirming risk concentration along the main urban corridor.
- The Manual Weighted Overlay Workflow yielded the most reliable, realistic, and actionable spatial depiction of cumulative health risk.
- The automated Python script, while demonstrating efficiency and consistency in execution, produced maps that were spatially inaccurate and failed to capture key demographic and infrastructural patterns.
- The results underscore the necessity of rigorous spatial validation, ensuring that automated GIS workflows accurately reflect the complex environmental realities they are designed to model.

Recommendations for Future Work

1. Improve Automation Accuracy:
    - Directly embed the precise reclassification and weighting parameters from the validated Manual process into the Python script.
    - Achieve the efficiency of automation without compromising the spatial accuracy of the final index (Figure 7).
2. Enhance Model Input:
    - Integrate additional environmental stressors, such as actual air quality monitoring data (PM2.5/NO_2) and detailed land-use data.
    - Increase the model's predictive power and provide a more comprehensive risk assessment.
3. Planning & Policy:
    - Utilize the high-accuracy Manual Map (Figure 7) to prioritize environmental monitoring, public health resource allocation, and targeted pollution mitigation efforts in the validated high-risk zones.

The End

THANK YOU FOR LISTENING