RSC Team 2 Project Documentation

Methods, Codebase, and Data Processing Guide

Objective

This document summarizes the methodology, data sources, code organization, and implementation details behind Team 2's Capstone project for Rise South City. It is intended for technical collaborators and future student teams who wish to understand, reproduce, or build upon this work.

Data Sources

The data/ directory includes a README.md file describing the contents and formats of each data file. This file is recommended reading before running any data processing scripts.

Air Quality Data:

- Clarity Sensor Data (clean_clarity.csv): Oct 30, 2024 – Mar 31, 2025
- PurpleAir API Data (clean_api_purpleair.csv): Mar 30, 2024 – May 3, 2025
- Historical PurpleAir Data (clean_purpleair.csv): Dec 27, 2018 – Apr 9, 2025
- All datasets were cleaned, merged, and normalized using the EPA's AQI formula and scaled to a 0–1 range.
- Final PM2.5 AQI scores were aggregated by census tract and stored in:
    - tracts_with_combined_aqi.csv
    - tracts_with_combined_aqi.geojson

Health Risk Data:

- Normalized Health Risk Index: health_risk_index.csv
- Raw Indicators:
    - all_indicators.csv: Combines CalEnviroScreen 4.0 and San Mateo indicators
    - smc_indicators/: Source files from the All Together Better initiative
    - Health Equity Index (requires manual download)
- Health data were processed and normalized to create a tract-level Health Risk Index (HRI).

Codebase Overview

All code resides in the code/ directory, grouped by functionality.

Preprocessing Scripts:

- clean_clarity.py: Parses timestamps, renames columns, removes invalid entries
- clean_api_purpleair.ipynb: Processes PurpleAir API data and calculates daily averages
- clean_purpleair.py: Merges API and historical data
- purpleair_wrapper.py: Automates PurpleAir API retrieval
- combine_air_quality_data.py: Filters data to the analysis period, joins with census tracts, aggregates daily PM2.5
- health_preproc.ipynb: Cleans and standardizes raw indicators

AQI Weighting:

- calculate_sensor_weights.py: Computes source weights based on overlapping readings

    - Clarity: 0.76
    - PurpleAir: 0.24
- combine_air_quality_data.py: Applies weights to produce final AQI scores

Health Index Processing:

- health.ipynb: Computes the Health Risk Index using geometric means

Composite Risk Score

Implemented in streamlit_app.py, the risk score is calculated as:

$$\text{Risk} = \alpha \times \text{HRI} + (100 - \alpha) \times \text{AQI}$$

Users adjust α via a Streamlit slider. Census tract scores are updated dynamically in the dashboard using Folium maps.

Sensor Reliability Models

predictability.ipynb:

- Consistency Score: Predicts a sensor's next-day PM2.5 using the past 7 days
- Predictability Score: Predicts current-day PM2.5 using readings from 5 nearest monitors

Both metrics assess sensor trustworthiness and spatial redundancy. Predictability scores are visualized in the dashboard using color-coded monitor markers.

Dashboard Implementation

To run streamlit_app.py locally:

```
python -m streamlit run code/streamlit_app.py
```

Main features:

- Interactive composite risk map by tract
- Weighting slider for HRI vs. AQI
- Address-based risk and predictability lookup
- Sensor confidence visualized by predictability index
- Interpretive annotations for user guidance

The app is organized into two main sections: Risk Analysis and Additional Information, accessible through Streamlit tabs.

Suggested Workflow for Future Teams

Suggested script/notebook execution order:

- clean_clarity.py
- clean_api_purpleair.ipynb
- combine_air_quality_data.py
- calculate_sensor_weights.py

- health_preproc.ipynb
- health.ipynb
- predictability.ipynb
- streamlit_app.py

Refer to the README.md and Code Book in code/ for additional dependencies, file structure, and configuration notes.

Notes for Maintenance

- Data refresh: To incorporate updated air or health data, rerun the relevant preprocessing scripts.
- Predictability index: Can be recalculated if sensor locations or temporal ranges change.
- Deployment: We recommend using Streamlit Community Cloud for continued hosting.