Multifamily Ground and Rooftop Layers

1. Data Acquisition for Multifamily Layers

- **Impervious Raster:** National Land Cover Database (NLCD) Impervious Surface.
    - **Source:** https://www.mrlc.gov/data/type/impervious-surface
- **Multifamily Housing Data (CA):** (Accessed from Yohan. R data file. Zillow)
- **Microsoft Building Footprint (CA):** Open-source building footprint data.
    - **Source:** https://github.com/microsoft/GlobalMLBuildingFootprints
    - (Stopped using the US building footprints, since a lot of data hasn't been updated in 10+ years. Global data has been updated recently this year but much harder to clean/access. Not sorted by State, need to manually retrieve CA data.)
    - First Downloaded the entire US footprint from github, then

Once downloaded all US footprints, run this in terminal to organize the files in order for it to work when inputting all of the US footprints into the python script which filters only for CA footprints and converts that data to a geopackage.

Shell

```
# This loop finds each unique file group (e.g., part-00000, part-00001)

# and combines all its pieces into a single file in the new directory.

echo "Starting file combination..."

for prefix in $(ls part-*.csv.gz* | sed -E 's/(\.csv\.gz\..*|\.csv\.gz)//' |
sort -u); do

    echo "Combining files for: ${prefix}"

    cat "${prefix}".csv.gz* > "../combined_footprints/${prefix}.csv.gz"
```

Then I have to run code to filter these footprints to only CA. Need to download all state boundary file, easily available online from Census website:
Code below is titled: ***filter_USfootprints_toCA.py*** in github

Python

```python
import pandas as pd
import geopandas as gpd
import os
import warnings
import gzip

# Ignore warnings
warnings.filterwarnings('ignore', 'GeoSeries.notna', UserWarning)

# --- Configuration ---
# ⚠️ Update these paths to match your system
US_DATA_DIR =
'/Users/shahil/Documents/UCSB/2024-2025/Spatial/united_states_footprints'
ALL_STATES_BOUNDARY_FILE =
'/Users/shahil/Documents/UCSB/2024-2025/Spatial/tl_2024_us_state/tl_2024_us_sta
te.shp'
OUTPUT_FILE =
'/Users/shahil/Documents/UCSB/2024-2025/Spatial/california_building_footprints_
complete.gpkg'

# --- Script ---
print(f"Loading boundary for California from {ALL_STATES_BOUNDARY_FILE}...")
try:
    states = gpd.read_file(ALL_STATES_BOUNDARY_FILE)
    california_boundary = states[states.NAME == 'California']
    if california_boundary.empty:
        print("Error: Could not find 'California' in the states shapefile.")
        exit()
except Exception as e:
    print(f"Error: Could not read the shapefile. Details: {e}")
    exit()

# Set the CRS for the boundary file to ensure it's correct
california_boundary = california_boundary.to_crs("EPSG:4326")
print("California boundary loaded successfully.")

all_ca_buildings = []

try:
    files_to_process = [f for f in os.listdir(US_DATA_DIR) if
f.endswith('.csv.gz')]
    if not files_to_process:
        print(f"Error: No .csv.gz files found in '{US_DATA_DIR}'.")
```

```
        exit()
except FileNotFoundError:
    print(f"Error: The directory '{US_DATA_DIR}' does not exist.")
    exit()

print(f"Found {len(files_to_process)} US data files to process...")

for i, filename in enumerate(files_to_process):
    filepath = os.path.join(US_DATA_DIR, filename)

    try:
        with gzip.open(filepath, 'rt', encoding='utf-8') as f:
            gdf = gpd.read_file(f, driver='GeoJSONSeq')

        if gdf.empty:
            continue

        # ========================================================================
        # FIX: Assign the correct CRS to the building footprint data.
        # This ensures it matches the california_boundary CRS for the join.
        gdf.crs = "EPSG:4326"
        # ========================================================================

        # Use a spatial join to find buildings that intersect the boundary
        buildings_in_ca = gpd.sjoin(gdf, california_boundary, how="inner",
predicate="intersects")

        if not buildings_in_ca.empty:
            print(f"({i+1}/{len(files_to_process)}) Found
{len(buildings_in_ca)} buildings in {filename}")
            all_ca_buildings.append(buildings_in_ca)

    except Exception as e:
        print(f"Could not process {filename}. It may be empty or corrupt.
Error: {e}")

if all_ca_buildings:
    print("\nCombining all found buildings into a single file...")
    final_gdf = pd.concat(all_ca_buildings, ignore_index=True)

    # Keep only the essential 'geometry' column from the original data
    final_gdf = final_gdf[['geometry']]
    # Drop duplicates in case a building is in multiple source tiles
```

```
    final_gdf =
final_gdf.drop_duplicates(subset='geometry').reset_index(drop=True)

    print(f"Saving {len(final_gdf)} total building footprints to
{OUTPUT_FILE}...")
    final_gdf.to_file(OUTPUT_FILE, driver='GPKG')
    print("✅ Done! California building footprints should now be complete.")
else:
    print("⚠️ No buildings were found. The script finished but found no data in
the target area.")
```

Once this is completed, you will get a geopackage of **All Buildings in CA. (code for this is in github)**

- **California Schools/Colleges (`cscd_2021`):** California School Campus Database.
  - **Source:** https://gis.data.ca.gov/datasets/CADeptEd::california-school-campus-database-cscd-2021/about
- **California Public Lands (`cced_2024b_release`, `cpad_2024b_release`):** California Protected Areas Database (CPAD) and California Conservation Easement Database (CCED).
  - **Source (CPAD):** https://www.calands.org/data/cpad/
  - **Source (CCED):** https://www.calands.org/data/cced/
- **National Resources Inventory (NRI) Data:** (`NRI_Shapefile_CensusTracts`, `NRI_Table_CensusTracts`)
  - **Source:** https://www.nrcs.usda.gov/data-and-reports/national-resources-inventory-nri
- **Protected Areas Database of the United States (PADUS):** (`PADUS4_1_State_CA_GDB_KMZ`)
  - **Source:** https://www.usgs.gov/core-science-systems/science-analytics-and-synthesis/gap/science/protected-areas

2. Multifamily Ground Layer Generation

This layer identifies potentially developable land near multifamily housing after a multi-stage filtering process.

- **Step 2.1: Multifamily Point Processing**
  - Loaded and filtered the housing dataset to multifamily properties.
  - Converted coordinates to **California Albers (EPSG:3310)**.
  - Created **250-meter** proximity buffers around each point and merged them into a unified Area of Interest (AOI).
- **Step 2.2: Impervious Surface Analysis**
  - Clipped the NLCD impervious raster to the AOI.
  - Extracted pixels with imperviousness values between **1-20%** (representing less developed areas) and converted them to vector polygons.
- **Step 2.3: Institutional Exclusion Processing**
  - Loaded and combined five school/university layers, CPAD, and CCED protected lands datasets.
  - Performed spatial difference operations using optimized `overlay` functions to remove these institutional and protected areas from the potential ground layer.
- **Step 2.4: Building Buffer Exclusion**
  - Created **5-meter** buffers around all building footprints.
  - Performed a final spatial difference to ensure ground areas do not overlap with existing buildings or their immediate surroundings.
- **Step 2.5: Impervious Surface Filtering (Secondary Check)**
  - To further refine the layer and ensure suitability, the intermediate ground layer was filtered against a secondary, high-resolution impervious surface dataset (`LC24_EVT_250_filtered_flat_points_OPTIMIZED.tif`).
  - Polygons that did not overlap with any valid impervious surface pixels in this secondary dataset were removed. This step eliminated **3,028 polygons** (a 15.4% reduction), enhancing the quality of the final layer.
- **Step 2.6: Final Quality Control and Optimization**
  - Removed small polygons with an area less than **100 square meters**.
  - Exploded multipolygons into single-part features to filter out small geometric slivers.
  - The final geometries were simplified with a **2.0-meter tolerance** to reduce file size and improve performance.
  - **Final Output**: An optimized GeoPackage (`final_multifamily_ground_layer_optimized.gpkg`) containing **16,691** clean, verified polygons suitable for development analysis.

---

3. Multifamily Rooftop Layer Generation

This layer provides a comprehensive dataset of building footprints associated with multifamily housing clusters.

- **Step 3.1: Building-Housing Spatial Association**
  - A **15-meter** buffer was applied to each multifamily housing point to account for geocoding inaccuracies.
  - An optimized spatial join (`sjoin_nearest`) was performed between the buffered points and the complete California building footprints dataset to associate buildings with multifamily locations.
  - Duplicate building entries were removed, resulting in **1,578,556** unique building footprints.
- **Step 3.2: Community Clustering Analysis**
  - The **DBSCAN** clustering algorithm was applied to the centroids of the building footprints to identify dense communities of multifamily housing.
  - **Parameters**: `eps=100` meters (max distance between buildings in a cluster) and `min_samples=5` (minimum buildings to form a cluster).
  - **Results**: **19,922** distinct housing clusters were identified, and **74,708** isolated buildings were filtered out as noise.
- **Step 3.3: Final Layer Assembly and Optimization**
  - The dataset was reduced to only essential columns: **`geometry`** and **`cluster`**.
  - Building footprint geometries were simplified with a **2.0-meter tolerance** to dramatically reduce file size.
  - The `cluster` ID was downcast to a more efficient `int32` data type.
  - **Final Output**: An optimized GeoPackage (`final_multifamily_rooftop_layer_clustered_simplified.gpkg`) containing the clustered and size-reduced rooftop footprints.