SI 206 | Final Project Report - Tackling America's Housing Crisis
Yusuf Haydar, Omar Aggour, Ali Allam
GitHub Repo: https://github.com/Oaggour/Tackling-Americas-Housing-Crisis-.git

Analysis of Housing, Demographic, and COVID-19 Data

Section 1 - Initial Goals

Initially, the project aimed to analyze the relationships between housing costs (specifically home values), emergency room visits, and income levels across US counties to inform policy interventions which could mitigate both the national housing crisis and the healthcare crisis. These three data points were planned on being extracted from three datasets/APIs: (1) the Zillow Home Value Index (ZHVI), (2) the National Syndromic Surveillance Program Emergency Department Visits API, and (3) the Census American Community Survey API.

Section 2 - Goals Achieved

The project successfully utilized APIs from the Center for Disease Control and Prevention (CDC), the Census Bureau, and the Department of Housing and Urban Development (HUD) to gather and analyze data on public health, demographics, and housing.

Apart from the Census API, two of the final data sources differed from those initially planned. This is because upon further exploration, the Zillow dataset was found to be only available as a CSV, not an API, which led to its replacement with the HUD API. Additionally, the NSSP API was found to be too complex and beyond the scope of this analysis, so it was replaced with the CDC API.

The CDC API provided two primary public health statistics which were of interest to this analysis: COVID-19 hospital admissions per 100k in population, and COVID-19 community level (a taxonomy of either Low, Medium, or High). The Census API provided basic demographic information such as population and median income, as well as number of owner-occupied households, number of renter-occupied households, and average commute time. The HUD API provided information about average rent rates for units ranging from one to four bedrooms. By integrating these datasets, the project enabled comprehensive analysis and visualization of relationships across these areas.

Section 3 - Problems Faced

The most challenging aspect of the project was choosing three suitable APIs. Several other APIs were considered or tried throughout the process, but were found to be inaccessible to the general public, not disaggregated by county, did not contain FIPS codes, or were beyond the scope of this project. Another challenge was learning the specific systems of parameters and outputs of each API. For example, the Census ACS API contains thousands of variables which can be called, each with a unique identifier code listed in a large page within the documentation.

Section 4 - Calculations From Data in Database

calculated_data.txt

```
Metrics by COVID-19 Community Level:
COVID Level | Average Rent | Average Income | Average Hospital Admissions
-----------------------------------------------------------------
Low         | $1621.30     | $86008.02      | 7.21 per 100k
Medium      | $1394.09     | $75472.36      | 12.97 per 100k
High        | $1320.65     | $69028.89      | 19.80 per 100k
```

Figure 1: The txt file shows an exported analysis of counties' average 2 bedroom rental cost, average income, and average COVID hospital admissions per 100k, by COVID-19 Community Level.

Section 5 - Visualizations

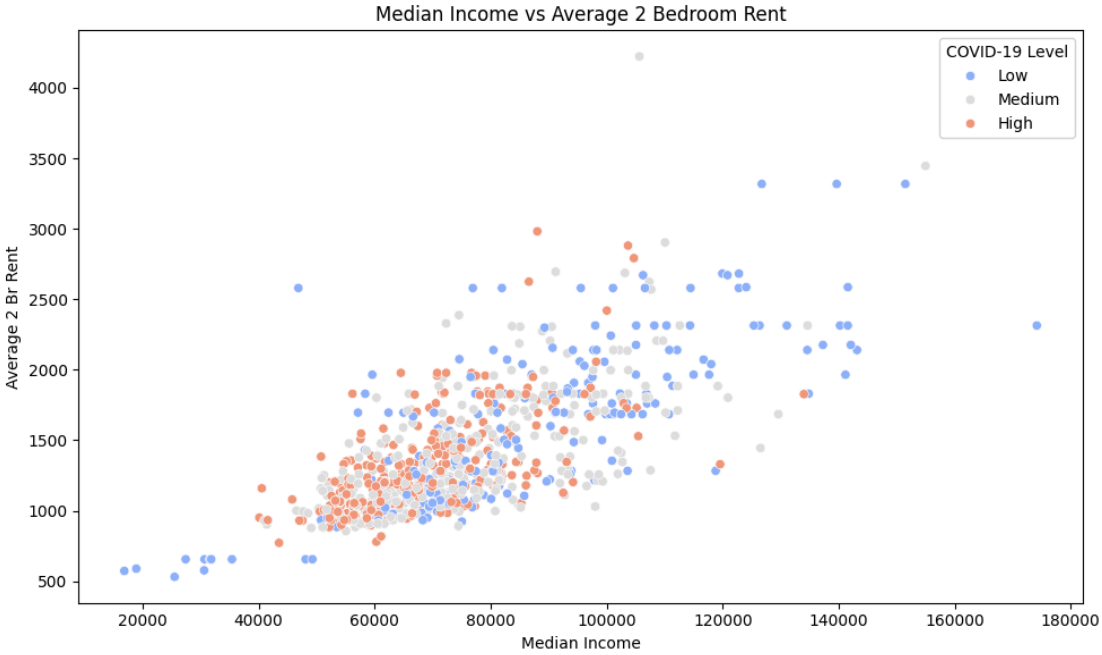

Figure 2: The scatterplot shows a positive correlation between Median Income and Average 2 Bedroom Rent across US Counties.

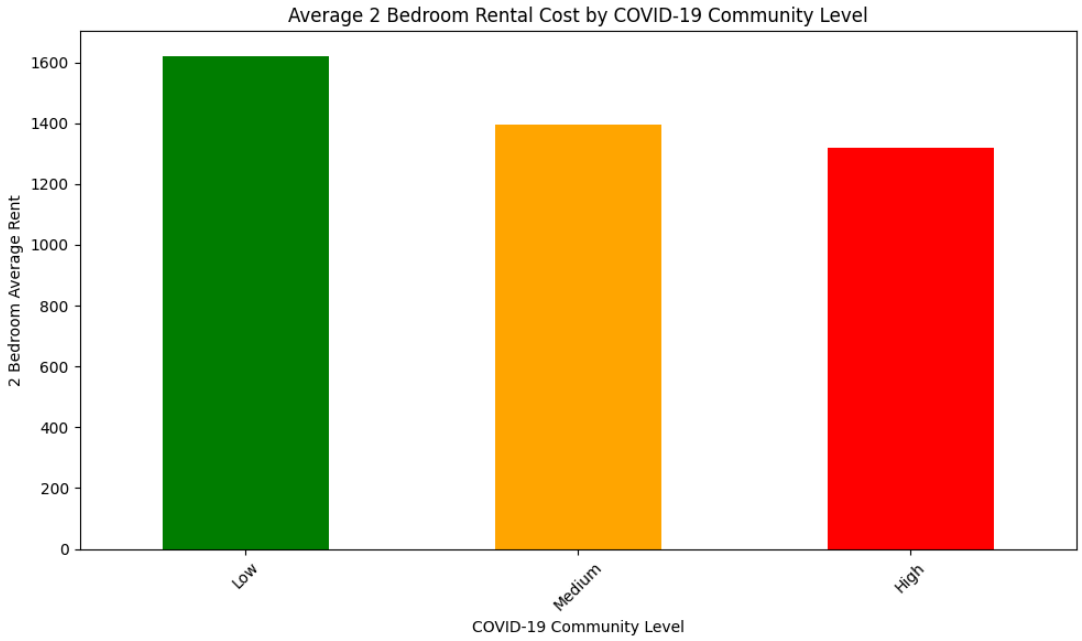

Figure 3: The bar-chart shows average 2 Bedroom Rental Cost by COVID-19 Community Level, with low-risk counties having the least affordable rent levels.

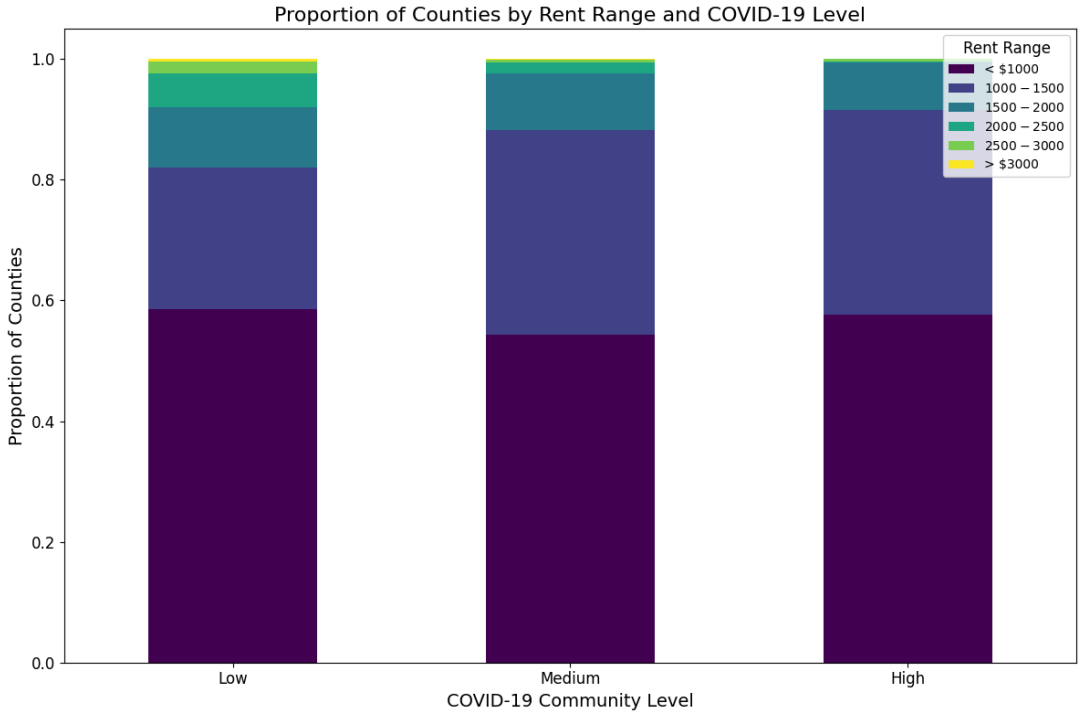

Figure 4: The stacked bar-chart shows Proportion of Counties by Rent Range and COVID-19 Community Level, with little to no counties with average rent above $2,500 at high risk.

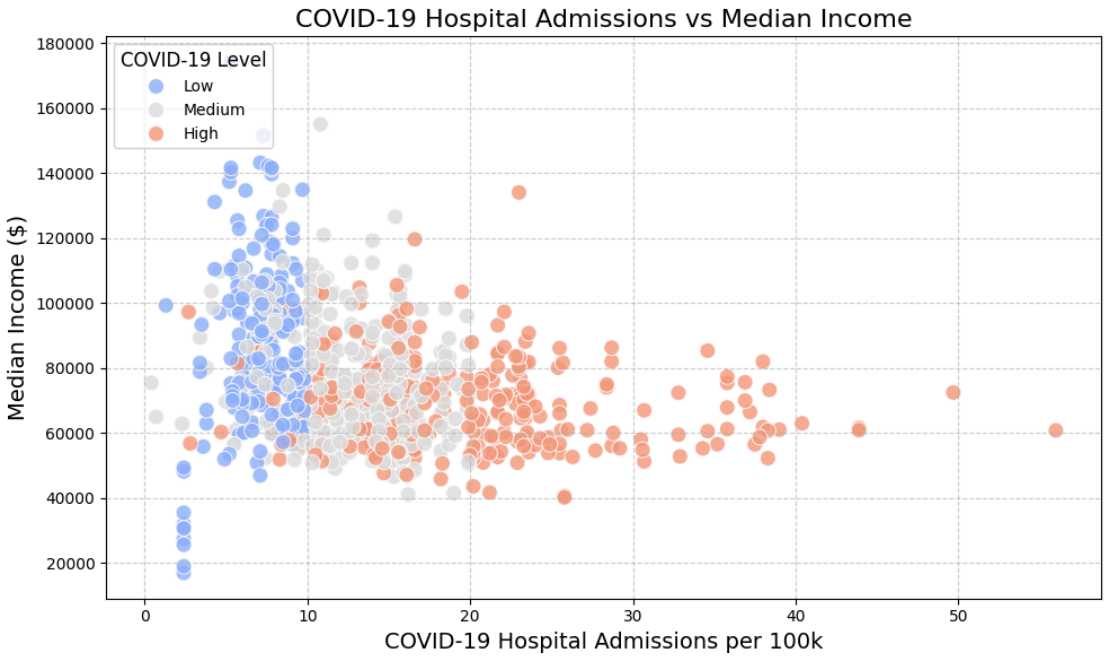

Figure 5: The scatterplot shows a moderate negative relationship between COVID Hospital Admissions per 100k and Median Income, indicating that residents of lower-income counties are more at risk for severe COVID symptoms.

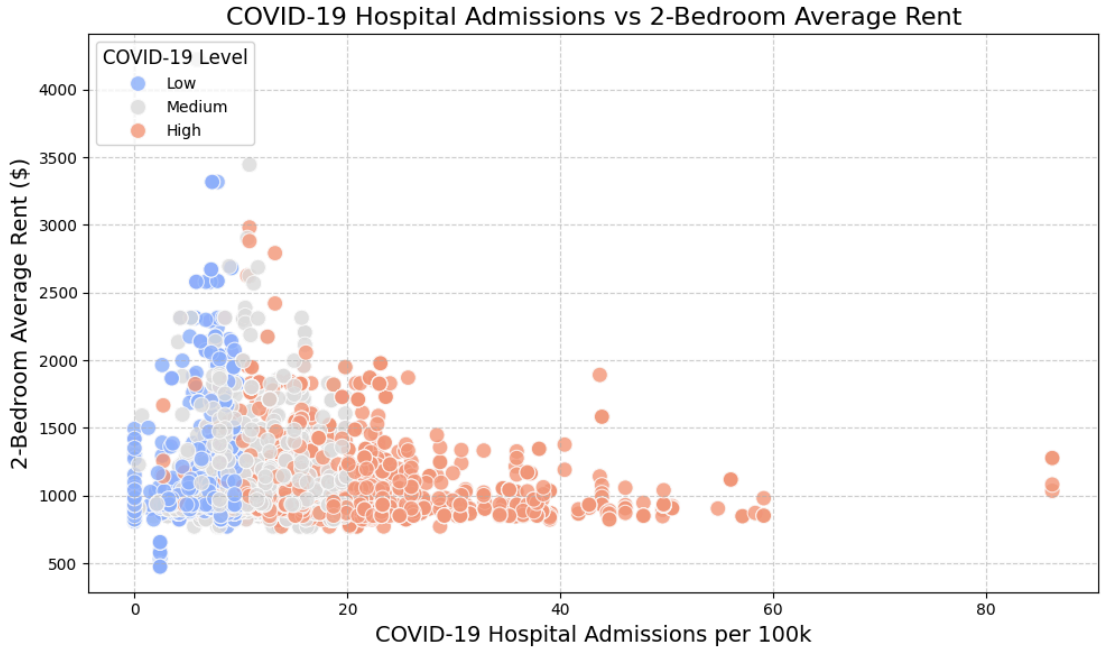

Figure 5: The scatterplot shows a moderate negative relationship between COVID Hospital Admissions per 100k and Average 2-Bedroom Rent, indicating that residents of higher-rent counties are at lower risk for severe COVID symptoms.

Instructions for Running Code

1) Run cdc.py eight times to insert all 3,220 records into the database.
2) Run census.py six times to insert all 854 records into the database.
3) Run hud.py eight times to insert all 3,161 records into the database.
4) Run data_processing2.py to export calculated_data.txt.
5) Run visualization.py to export the five charts.

Function Documentation

cdc.py

- Fetch_data
    - Description: Fetches COVID-19 community-level data from the CDC JSON API.
    - Inputs: None
    - Outputs: Returns a list of tuples, each containing (county_fips, covid_hospital_admissions_per_100k, covid_19_community_level_id).
- Create_tables
    - Description: Creates the combined_data and covid_community_level tables in the SQLite database if they do not already exist.
    - Inputs: conn (SQLite connection object)
    - Outputs: None
- Get_last_index
    - Description: Gets the count of rows where covid_hospital_admissions_per_100k is not NULL in the combined_data table.
    - Inputs: conn (SQLite connection object)
    - Outputs: Returns the count of rows as an integer.
- Insert_data
    - Description: Inserts rows of data into the combined_data table in batches, updating existing entries based on fips_code.
    - Inputs: conn (SQLite connection object), rows (list of tuples), start_index (integer)
    - Outputs: None
- Progressively_load_data
    - Description: Fetches data, creates tables if needed, gets the last inserted index, and inserts new data progressively.
    - Inputs: conn (SQLite connection object)
    - Outputs: None

census.py

- Api_url

    - Description: Constructs the API URL for fetching Census data for a given year and variables.
    - Inputs: year (string), variables (string)
    - Outputs: Returns a constructed URL string for the API request.
- Fetch_data
    - Description: Fetches Census data for the specified year and variables and extracts relevant columns.
    - Inputs: year (string)
    - Outputs: Returns a list of tuples containing fips and median_income.
- Create_table
    - Description: Ensures the combined_data table is created in the SQLite database.
    - Inputs: cur (cursor object), conn (SQLite connection object)
    - Outputs: None
- Get_last_index
    - Description: Gets the count of rows where median_income is not NULL in the combined_data table.
    - Inputs: conn (SQLite connection object)
    - Outputs: Returns the count of rows as an integer.
- Insert_data
    - Description: Inserts rows of data into the combined_data table in batches, updating existing entries based on fips_code.
    - Inputs: cur (cursor object), conn (SQLite connection object), data (list of tuples), start_index (integer)
    - Outputs: None
- Process_api_data
    - Description: Processes API data for the specified year.
    - Inputs: year (string)
    - Outputs: Returns a list of tuples containing fips and median_income.

hud.py

- Fetch_data
    - Description: Fetches HUD housing data for the specified indexes and extracts relevant columns.
    - Inputs: last_index (integer)
    - Outputs: Returns a list of tuples containing fips_code and Two-Bedroom.
- Create_table
    - Description: Ensures the combined_data table is created in the SQLite database.
    - Inputs: conn (SQLite connection object)
    - Outputs: None
- Get_last_index

    - Description: Gets the count of rows where two_bedroom is not NULL in the combined_data table.
    - Inputs: conn (SQLite connection object)
    - Outputs: Returns the count of rows as an integer.
- Insert_data
    - Description: Inserts rows of data into the combined_data table in batches, updating existing entries based on fips_code.
    - Inputs: conn (SQLite connection object), rows (list of tuples), start_index (integer)
    - Outputs: None
- Progressively_load_data
    - Description: Fetches data, creates tables if needed, gets the last inserted index, and inserts new data progressively.
    - Inputs: conn (SQLite connection object)
    - Outputs: None

data_processing.py

- Main
    - Description: Executes SQL queries on the combined dataset and outputs the results to a text file.
    - Inputs: None
    - Outputs: Processed results written to calculated_data.txt.

visualization.py

- Main
    - Description: Loads data, processes it, and generates various visualizations.
    - Inputs: None
    - Outputs: Plots and saves visualizations as image files (.png).

Resources Documentation

Date	**Issue Description**	**Location of Resource**	**Result**
11/26/24	Needed help understanding duplicate string project requirements	Office Hours	Understood how to resolve duplicate strings and structure our database

11/27/24	Could not understand how the NSSP API works	ChatGPT - Copy and pasted the documentation and asked it to explain it to me	I got a better understanding but realized the API is too complex for this project and does not output the data in a suitable format.
11/27/24	Could not figure out how to set up batch insert system	ChatGPT - gave it the project requirements pertaining to uploading data in batches and asked it to explain what functions we would need without providing code	Helped our understanding and led to creating get_last_index functions
11/28/24	Too many variables available for Census API to sift through manually	ChatGPT - Gave it list of all variables available and asked it to sift out ones related to housing	Outputted variables of interest
12/5/24	Wanted ideas for what kind of visualizations to create	ChatGPT - Gave it a summary of all the data we have	Outputted 10 ideas for visualizations we can create
12/5/24	Getting error message when running code with Matplotlib	ChatGPT - Asked it what the error means and how it can be fixed	Still did not resolve on my machine (Just Ali)
12/5/24	Wanted to learn how to color code points on a scatter plot to add more information	Matplotlib documentation	Learned how to use hue parameter
12/15/24	Covid_community_level table was not being created	ChatGPT - asked it to help me debug the code	Told me I forgot commas between the values in the SQL INSERT statement and I didn't conn.execute it.
12/16/24	Needed to handle last index while consolidating data into a single table	ChatGPT - asked it how we can use a single table while still preserving proper indexing for each API without providing any code	Suggested selecting count of records in the table where the respective columns are not null.