

Real Estate Analysis and Graph Clustering

In the ever-evolving field of data analysis and real estate market evaluation, the ability to accurately and efficiently analyze large datasets is crucial. Our project, titled "Real Estate Sales Analysis and Graph Clustering", aims to address this need by implementing a robust system for processing and analyzing extensive real estate sales data. The primary objective of this project is to develop a comprehensive analysis tool capable of handling real estate data spanning from 2001 to 2020, focusing particularly on key metrics such as assessed values, sales ratios, and residential types.

At its core, this project leverages the power of Rust, a language renowned for its performance and safety, to manipulate and analyze large volumes of data. By applying advanced data cleaning techniques, we ensure the integrity and reliability of our dataset.

Through this project, we aim to provide a nuanced understanding of real estate trends over two decades, allowing for a more informed perspective on property valuation and market dynamics. Our approach not only serves as a valuable tool for real estate professionals and analysts but also demonstrates the application of modern programming techniques and algorithms in transforming raw data into actionable insights.

The Dataset:

<https://www.kaggle.com/datasets/anoopjohny/real-estate-sales-2001-2020-state-of-connecticut>

Serial Number	List Year	Date Recorded	Town	Address	Assessed Value	Sale Amount	Sales Ratio	Property Type	Residential Type	Non Use Code	Assessor Remarks
900403	2009	08/23/2010	Farmington	3 INDIAN HILL RD	286550.00	355000.00	0.807183099	Single Family	Single Family		
60172	2006	04/04/2007	Windsor Locks	211 S CENTER ST	140900.00	235000.00	0.599574468	Single Family	Single Family	03 - Inter Corporation	
61317	2006	08/31/2007	Danbury	SHELTER ROCK RD	207300.00	444000.00	0.466891892	Single Family	Single Family		
170321	2017	03/14/2018	Southington	10 STRAWBERRY LN	279610.00	415000.00	0.6737	Single Family	Single Family		
181462	2018	04/09/2019	Waterbury	4 KENILWORTH ST	117370.00	78750.00	1.4904	Single Family	Single Family	14 - Foreclosure	
150621	2015	07/14/2016	Norwich	520 ASYLUM ST EXT	130400.00	149000.00	0.875167785	Single Family	Single Family		
90406	2009	05/17/2010	Middletown	66 ASPEN DR	154940.00	264000.00	0.586893939	Single Family	Single Family		
20300	2002	04/14/2003	Vernon	14 WILSON LN	80710.00	145000.00	0.55662069				
20094	2020	11/23/2020	Bethel	18 WEED ROAD	231630.00	375000.00	0.6176	Residential	Single Family		
60179	2006	07/10/2007	East Haddam	48 CRAGMERE RD	41950.00	123000.00	0.341056911	Single Family	Single Family		
150700	2015	06/15/2016	Stratford	194 LAUREL ST	275520.00	400000.00	0.6888	Single Family	Single Family		
201216	2020	05/24/2021	Griswold	154 GEER RD	96250.00	225000.00	0.427777778	Residential	Single Family		
60422	2006	09/17/2007	Cromwell	4 MARGO CT	81100.00	160900.00	0.504039776	Condo	Condo		

Our project utilizes an extensive dataset of real estate transactions spanning nearly two decades, containing close to 900,000 entries. Each record in this dataset encapsulates critical information about property sales, including the serial number, recording date, town, address, assessed value, sale amount, sales ratio, property type, residential type, and non-use codes, along with any assessor remarks. The vast amount of data, sourced from a comprehensive registry of real estate transactions, provides a rich foundation for in-depth analysis and trend identification within the real estate market.

Our Goal:

The main goal of this project is to organize properties into groups based on how much they're worth. We create these groups, or clusters, by placing properties into different categories, or bins, according to their assessed values. Think of each bin as a basket where we gather properties that have similar values. Once we've got these clusters set up, we calculate two important things for each one: the average sales ratio, which helps us understand how often properties are sold compared to their worth, and the most common type of residential property, like single-family homes or condos. This way, we can see which types of properties are most common in each value range and how their sale prices compare to their assessed values. It's like taking a big, varied collection of data and sorting it into neat, organized piles that tell us more about the real estate market.

Project Methodology

Sampling & Data Cleaning:

Given the sheer volume of the dataset, analyzing all 900,000 entries would be computationally intensive and potentially unnecessary for obtaining a representative overview of the market trends. To address this, we employed a sampling technique to select a subset of 10,000 entries, ensuring a manageable yet statistically significant sample size for analysis.

In tackling the vast dataset of nearly 900,000 real estate entries, our approach began with a crucial data cleansing step: we removed any entries that were missing information about the type of residential property, such as whether a property is a single-family home, condo, and so on. This ensured that our analysis would be based on complete and reliable data. Following this, we extracted a sample of 10,000 entries to work with, a sizable yet manageable portion that represents the broader dataset effectively. It's within this sample that we then further refined the data by removing the highest and lowest outliers—the top 150 and bottom 150 records based on sales ratio. These outliers can often distort the analysis due to their extreme nature, such as being unusually high or low sales. By excluding them, we aimed to achieve a more accurate reflection of typical property sales and values within the market. In line with our initial data cleansing, we further streamlined the dataset by removing columns that weren't necessary for our study. We chose to keep only the essential information: Serial Number, Assessed Value, Sales Ratio, and Residential Type. We also held on to the Sale Value because it helps us understand market trends more clearly. This careful selection of data ensured that our analysis was focused on the most relevant factors for an accurate market evaluation.

New Clean and Sampled Dataset:

Serial Number	Assessed Value	Sale Amount	Sales Ratio	Residential Type
16266	646660.0	955000.0	0.67713089	Condo
19763	46830.0	102100.0	0.4587	Single Family
190280	174860.0	249900.0	0.6997	Single Family
190958	669340.0	855000.0	0.7829	Single Family
13186	264900.0	392400.0	0.675076453	Condo
60766	0.0	195000.0	0.0	Condo
180871	73080.0	80000.0	0.9135	Single Family
120041	209000.0	280000.0	0.746428571	Single Family
180261	115710.0	200000.0	0.5785	Single Family
162001	269600.0	300000.0	0.898666667	Single Family
1900401	177170.0	122000.0	1.4522	Single Family

Module Organization

csv_parser.rs

This module is dedicated to handling CSV data. It includes functionalities for reading data from and writing data to CSV files. The core component is the `Record` struct, which maps to the columns of our dataset, including fields like `Serial Number`, `Assessed Value`, `Sales Ratio`, and `Residential Type`.

- **`read_csv`**: A function that takes a file path as input and reads data from a CSV file into a vector of `Record` structs.
- **`write_csv`**: This function writes a collection of `Record` structs back to a CSV file, enabling us to export our processed data.

data_cleaner.rs

The data cleaning module is responsible for preparing the dataset for analysis. It filters out incomplete or irrelevant records, ensuring that our analysis is based on accurate and complete data.

- **`clean_data`**: This function filters the dataset, removing records that lack essential information like the `Residential Type`.

sampler.rs

This module focuses on sampling the dataset to make it more manageable. It selects a representative subset of the data for detailed analysis.

- ***sample_data***: It handles the sampling process, filtering records based on Residential Type, sorting them by Sales Ratio, excluding outliers, and then randomly sampling a specified number of records.

graph.rs

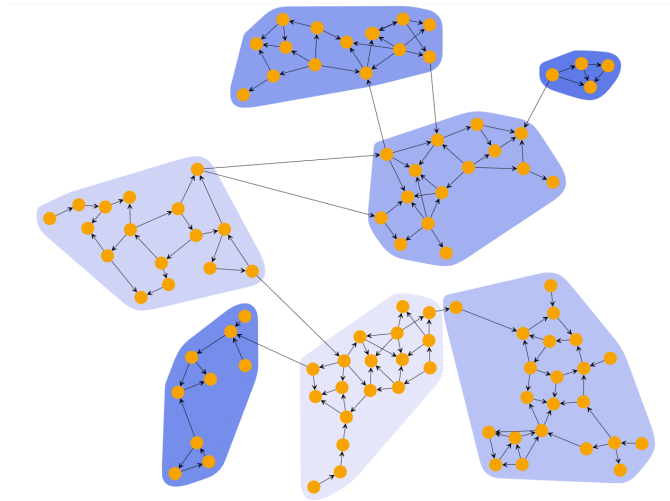
This module creates a graph-based representation of our data, where properties are organized into clusters or bins based on their assessed values.

- ***create_bins***: Assigns properties to clusters based on their assessed values.
- ***analyze_clusters***: Calculates the average sales ratio and identifies the most common residential type within each cluster.

Brief Code Description

- In `csv_parser.rs`, we utilize the *serde* crate for serialization and deserialization of data, allowing for efficient reading and writing of CSV files. The `Record` struct serves as the backbone, mirroring the structure of our dataset.
- The `data_cleaner.rs` module's `clean_data` function iterates through each record, retaining only those that meet our criteria for completeness. This step is crucial for the integrity of subsequent analyses.
- In `sampler.rs`, `sample_data` first filters and sorts the data, then removes the extreme outliers before randomly selecting a set number of records. This approach ensures both the manageability and representativeness of our dataset.
- Finally, `graph.rs` is where the data is transformed into a graph-like structure. The `create_bins` function organizes properties into clusters based on assessed value ranges, and `analyze_clusters` performs the crucial task of extracting meaningful insights from each cluster.

Graph Construction



As we constructed the graph for our project, we conceptualized properties as 'nodes' and the relationships between them as 'edges.' Each node in this graph represents a property, characterized by its Serial Number, Assessed Value, Sales Ratio, and Residential Type. The edges in our graph are conceptualized differently from a traditional graph: they are defined by the assessed value ranges, effectively grouping properties into bins or clusters.

When a property's assessed value falls within a specific range, it is placed in the corresponding bin. This placement is akin to creating an edge between the node (property) and the cluster (bin). For instance, all properties valued between \$100,000 and \$150,000 would be grouped together, forming a cluster of nodes based on their similar assessed values.

Within each cluster, we conducted a detailed analysis to determine two key metrics: the average sales ratio and the most common residential type. The average sales ratio was calculated by summing the sales ratios of all the properties (nodes) in a cluster and then dividing this total by the number of nodes in that cluster. This calculation gave us an average figure that reflects the typical market performance of properties within the same value range.

This average is an essential indicator, providing insights into the general sales trends for properties in each assessed value bracket. Additionally, identifying the most frequent type of residence in each cluster helped us understand the dominant property profiles within each value range.

Through this graph-based approach, we were able to map out and analyze the complex relationships and similarities among properties in our dataset. This method allowed us to gain a deeper understanding of the structure and dynamics of the real estate market, revealing patterns that are crucial for informed market analysis and decision-making.

Testing

Creating tests for our project involved designing specific test cases for each module to ensure the correctness and reliability of the individual components. Here's a brief overview of how we approached testing in each module:

csv_parser.rs

Tests for Reading and Writing CSV: We created mock data and wrote it to a temporary CSV file, then used our `read_csv` function to read this data back and verify its correctness. Similarly, we tested `write_csv` by creating a `Record` struct, writing it to a file, and then reading the file to ensure the data was correctly written.

data_cleaner.rs

Test for Data Cleaning: We created a set of `Record` structs with various combinations of complete and incomplete data. The test involved passing these records to the `clean_data` function and verifying that only complete records were retained.

sampler.rs

Tests for Sampling Methodology: We constructed a series of tests to ensure our sampling method worked as intended. This included tests to check if records without a `residential_type` were filtered out, if the data was correctly sorted by `sales_ratio`, if outliers were properly excluded, and if the final sample size matched the specified number.

graph.rs

Tests for Graph Construction and Analysis: We created tests to verify the correct formation of clusters or bins based on assessed values. We also tested the `analyze_clusters` function by checking if it accurately calculated the average sales ratio and identified the most common residential type within each cluster.

General Approach to Test Creation:

- **Mock Data:** In all tests, we used mock data that mimicked the structure and type of our real dataset. This approach allowed us to control the test conditions precisely.
- **Asserting Conditions:** Each test included assertions to check whether the output of the functions matched our expectations based on the input data.
- **Edge Cases:** We paid special attention to edge cases, such as empty data sets or sets with non-standard values, to ensure robustness.
- **Isolation:** Each test was designed to be independent, ensuring that the result of one test did not affect another. This isolation is critical for reliable testing.

Through these tests, we aimed to cover a wide range of scenarios and ensure that each part of our codebase functioned correctly under various conditions. This rigorous testing approach was crucial in building confidence in our application's reliability and accuracy.

```
running 6 tests
test graph::tests::test_assign_to_cluster ... ok
test graph::tests::test_create_bins ... ok
test graph::tests::test_analyze_clusters ... ok
test csv_parser::tests::test_read_csv ... ok
test csv_parser::tests::test_write_csv ... ok
test sampler::tests::test_sample_data_sorts_and_excludes_outliers ... ok
test result: ok. 6 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```


Analysis & Conclusions

```
Bin 0: Range 75468 - 181688
Bin 1: Range 101710 - 119980
Bin 2: Range 119910 - 138250
Bin 3: Range 138380 - 159320
Bin 4: Range 159390 - 184870
Bin 5: Range 184890 - 218730
Bin 6: Range 218750 - 277130
Bin 7: Range 277420 - 416810
Bin 8: Range 416670 - 37913540
Bin 9: Range 0 - 75400
Bin 0: 972 records
Bin 1: 971 records
Bin 2: 970 records
Bin 3: 970 records
Bin 4: 970 records
Bin 5: 970 records
Bin 6: 970 records
Bin 7: 970 records
Bin 8: 970 records
Bin 9: 967 records
Cluster 7: Avg Assessed Value: 333010.10, Avg Sales Ratio: 0.85, Most Common Residential Type: Some(Some("Single Family"))
Cluster 5: Avg Assessed Value: 208093.22, Avg Sales Ratio: 0.89, Most Common Residential Type: Some(Some("Single Family"))
Cluster 4: Avg Assessed Value: 172011.69, Avg Sales Ratio: 1.00, Most Common Residential Type: Some(Some("Single Family"))
Cluster 0: Avg Assessed Value: 89695.17, Avg Sales Ratio: 0.86, Most Common Residential Type: Some(Some("Single Family"))
Cluster 8: Avg Assessed Value: 947221.96, Avg Sales Ratio: 1.12, Most Common Residential Type: Some(Some("Single Family"))
Cluster 1: Avg Assessed Value: 111118.34, Avg Sales Ratio: 0.90, Most Common Residential Type: Some(Some("Single Family"))
Cluster 6: Avg Assessed Value: 245731.63, Avg Sales Ratio: 0.91, Most Common Residential Type: Some(Some("Single Family"))
Cluster 2: Avg Assessed Value: 129114.87, Avg Sales Ratio: 0.83, Most Common Residential Type: Some(Some("Single Family"))
Cluster 9: Avg Assessed Value: 47888.40, Avg Sales Ratio: 0.75, Most Common Residential Type: Some(Some("Condo"))
Cluster 3: Avg Assessed Value: 148453.94, Avg Sales Ratio: 1.02, Most Common Residential Type: Some(Some("Single Family"))
```

```
Cluster 7: Avg Assessed Value: 333010.10, Avg Sales Ratio: 0.85,
Cluster 5: Avg Assessed Value: 208093.22, Avg Sales Ratio: 0.89,
Cluster 4: Avg Assessed Value: 172011.69, Avg Sales Ratio: 1.00,
Cluster 0: Avg Assessed Value: 89695.17, Avg Sales Ratio: 0.86,
Cluster 8: Avg Assessed Value: 947221.96, Avg Sales Ratio: 1.12,
Cluster 1: Avg Assessed Value: 111118.34, Avg Sales Ratio: 0.90,
Cluster 6: Avg Assessed Value: 245731.63, Avg Sales Ratio: 0.91,
Cluster 2: Avg Assessed Value: 129114.87, Avg Sales Ratio: 0.83,
Cluster 9: Avg Assessed Value: 47888.40, Avg Sales Ratio: 0.75,
Cluster 3: Avg Assessed Value: 148453.94, Avg Sales Ratio: 1.02,
```

Upon concluding our analysis, the findings have provided intriguing insights into the real estate market's functioning. Most notably, Cluster 8, which encompasses properties with the highest assessed values (\$416,670 to \$37,913,540), also exhibited the highest average sales ratio, suggesting that purchasers of more expensive houses may be obtaining better value for their money. This is an interesting trend, as it hints at a potential pattern where higher-value properties could offer more favorable sales terms.

Conversely, Cluster 9, at the opposite end of the spectrum with the lowest assessed values (ranging from \$0 to \$75,400), had the lowest average sales ratio. This observation suggests that buyers of less expensive properties might not be getting as good a deal compared to those purchasing higher-valued properties.

A consistent theme across all clusters, with the exception of Cluster 9, was the predominance of single-family homes as the most common residential type. This finding aligns with the general market composition, where single-family homes constitute the majority of residential property offerings. Interestingly, the most common residential type within the least expensive cluster (Cluster 9) was condos, diverging from the single-family trend observed in other clusters. This could reflect a market characteristic where lower-valued properties are more likely to be condominiums, potentially due to their smaller size or other factors affecting their valuation.

In summary, our analysis has highlighted a correlation between the assessed value of properties and the sales ratio, revealing a possible trend where the value proposition improves with the property price. Additionally, the predominance of single-family homes as the most common residential type further emphasizes their ubiquity in the property market, while the prevalence of condos in the lowest value cluster presents an interesting point for further investigation

Resources

Utilization of Chat GPT to help me with my function to exclude outliers:

I have attached the code screenshots for the `main.rs`, `graph.rs`, `data_cleaner.rs`, `csv_parser.rs`, `sampler.rs`. I also attached a screenshot of the output.

I have a problem however. Some clusters have very odd avg sales ratio because of outliers. I have already sampled my large dataset that contains approximately 900,000 into a sample of 10,000 entries, including removing the missing values that were in residential type column. Now, however, i want to get rid of the top 200 and the bottom 200 values because they are crazy outliers that have data entry errors. how can i adjust my code to create a new csv file with the samples but without the top 150 and bottom 150

ChatGPT

To exclude the top and bottom 150 outlier values from your dataset based on the sales ratio, you will need to sort your data by the sales ratio and then remove those entries. You can do this by adjusting your `sampler.rs` file, where you define the `sample_data` function.

Here's an approach you can follow:

1. Sort the vector of records by the sales ratio within the `sample_data` function.
2. Skip the first 150 records and exclude the last 150 records from the sorted vector.
3. Sample the data from this new range.

Below is a code snippet that you can incorporate into your `sampler.rs` to perform these steps: