Final Project Write-Up – DS210

Name: Molly Lehrer
Project Title: NYC Airbnb Neighborhood Graph
Course: DS210, Spring 2025

A. Project Overview

Goal:

This project explores the relationship between Airbnb listing prices and neighborhood structure in New York City. By building a graph where nodes represent neighborhoods and edges connect those with similar average prices, we examine which neighborhoods are most centrally connected. We apply graph algorithms such as Breadth-First Search (BFS) and closeness centrality to identify price-similar clusters and central hubs.

Dataset:

- Source: NYC Airbnb Open Data (Kaggle)
- Size: ~48,000 listings
- Preprocessing: Listings with prices over $1000 were considered outliers and removed using Python. The cleaned dataset (cleaned_airbnb.csv) was then used in Rust.

B. Data Processing

After the initial data cleaning in Python to filter out extreme price outliers, the dataset was then analyzed in Rust as follows:

- CSV parsed using the csv and serde crates
- Each row was deserialized into a Listing struct
- Listings were grouped by neighborhood
- For each neighborhood, the average listing price was calculated

- A graph was constructed by connecting neighborhoods with similar average prices (within $10)

This structure allowed us to model economic similarity between different areas of NYC.

C. Code Structure

Modules:

- main.rs: Manages high-level execution, including loading data, building the graph, and running algorithms.
- data.rs: Contains the Listing struct and CSV-loading logic.
- graph.rs: Defines graph-related structs (Node, Edge) and implements BFS and closeness centrality.

Key Types:

- Listing: Represents an individual Airbnb listing with fields like neighborhood, price, etc.
- Node: Represents a neighborhood and its average price.
- Edge: Represents a price-similarity connection between two neighborhoods.

Main Workflow:

1. Load the CSV into a vector of Listing
2. Group listings by neighborhood and compute average prices
3. Create nodes for each neighborhood and connect those within a $10 price range
4. Run BFS from a given neighborhood
5. Calculate closeness centrality for each node

D. Key Functions Explained

load_listings()
Loads and parses the CSV file using the csv crate. Each row is converted into a Listing object. Invalid entries are skipped. Returns a vector of listings.

group_by_neighborhood()
Groups listings by neighborhood and gathers their prices. Returns a map from neighborhood name to a list of prices.

build_graph()
Creates a list of Nodes and Edges. Each neighborhood becomes a node. An edge connects two nodes if their average prices differ by no more than $10.

bfs()
Performs a Breadth-First Search from a given node index. Returns a vector of visited node indices.

calculate_closeness_centrality()
For each node, runs BFS and calculates the sum of distances to all other nodes. Closeness centrality is the inverse of that sum. Returns a sorted list of neighborhoods by centrality score.

E. Tests

Tests were written to validate graph construction logic. For example, test_graph_construction() checks that the correct number of nodes and edges are produced from sample input.

Run with:

cargo test

Sample output:

```
   Compiling project210 v0.1.0
(/Users/mollylehrer/Desktop/DS210/project210)
warning: fields `neighbourhood_group` and `room_type` are never
read
  --> src/data.rs:13:9
   |
12 | pub struct Listing {
   |            ------- fields in this struct
13 |     pub neighbourhood_group: String, /...
   |         ^^^^^^^^^^^^^^^^^^
...
16 |     pub room_type: String, //Rooming type
```

```
   |         ^^^^^^^^^
   |
   = note: `Listing` has a derived impl for the trait `Debug`,
but this is intentionally ignored during dead code analysis
   = note: `#[warn(dead_code)]` on by default

warning: `project210` (bin "project210" test) generated 1
warning
    Finished `test` profile [unoptimized + debuginfo] target(s)
in 0.95s
     Running unittests src/main.rs
(target/debug/deps/project210-95299faf5c18bdcc)

running 1 test
test tests::test_graph_construction ... ok

test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0
filtered out; finished in 0.00s
```

F. Results

After loading ~48,000 listings and filtering out extreme prices, the graph contained:

- 221 neighborhoods as nodes
- ~7400 edges connecting price-similar neighborhoods

```
Top 5 neighborhoods by closeness centrality:
Bay Terrace, Staten Island -> 0.0019157088
Prospect-Lefferts Gardens -> 0.0019047619
Sunset Park -> 0.0019047619
St. Albans -> 0.0019047619
Astoria -> 0.0019047619
```

Interpretation: These neighborhoods are highly "central" in the price similarity network. This may reflect economic diversity or widespread appeal, and could be of interest to travelers looking for affordable access to multiple neighborhoods.

G. Usage Instructions

1. Clone or download the repository.
2. Place cleaned_airbnb.csv in the root directory.
3. Run the following commands:

cargo run # runs the program

Sample output:

```
warning: fields `neighbourhood_group` and `room_type` are never
read
  --> src/data.rs:13:9
   |
12 | pub struct Listing {
   |            ------- fields in this struct
13 |     pub neighbourhood_group: String, /...
   |         ^^^^^^^^^^^^^^^^^^
...
16 |     pub room_type: String, //Rooming type
   |         ^^^^^^^^^
   |
   = note: `Listing` has a derived impl for the trait `Debug`,
but this is intentionally ignored during dead code analysis
   = note: `#[warn(dead_code)]` on by default

warning: `project210` (bin "project210") generated 1 warning
    Finished `dev` profile [unoptimized + debuginfo] target(s)
in 0.07s
     Running `target/debug/project210`
Successfully loaded 48656 listings.
Example listing: Listing { neighbourhood_group: "Brooklyn",
neighbourhood: "Kensington", price: 149, room_type: "Private
room" }
Graph built with 221 nodes and 7457 edges!
Node { name: "Sea Gate", avg_price: 123.0 }
Node { name: "Neponsit", avg_price: 274.66666 }
Node { name: "Bergen Beach", avg_price: 106.7 }
Node { name: "Hunts Point", avg_price: 50.5 }
Node { name: "Two Bridges", avg_price: 127.06944 }
Edge { source: 0, target: 2 }
Edge { source: 0, target: 4 }
```

```
Edge { source: 0, target: 7 }
Edge { source: 0, target: 9 }
Edge { source: 0, target: 13 }
Starting BFS from: Sea Gate
Visited: Sea Gate
Visited: Bergen Beach
Visited: Two Bridges
Visited: Crown Heights
Visited: Greenpoint
Visited: Columbia St
Visited: Prospect-Lefferts Gardens
Visited: Morningside Heights
Visited: Holliswood
Visited: Flatlands
Visited: Prince's Bay
Visited: Van Nest
Visited: East Harlem
Visited: Coney Island
Visited: Sunset Park
Visited: Windsor Terrace
Visited: Williamsburg
Visited: Rosebank
Visited: Stuyvesant Town
Visited: Huguenot
Visited: Kew Gardens Hills
Visited: Pelham Bay
Visited: Clason Point
Visited: Forest Hills
Visited: Eastchester
Visited: Briarwood
Visited: Arrochar
Visited: Harlem
Visited: Canarsie
Visited: Rockaway Beach
Visited: West Farms
Visited: Howard Beach
Visited: Westchester Square
Visited: Eltingville
Visited: Jamaica Hills
Visited: Manhattan Beach
Visited: Unionport
```

Visited: Bay Terrace
Visited: Bay Ridge
Visited: Castleton Corners
Visited: Long Island City
Visited: Middle Village
Visited: Whitestone
Visited: St. George
Visited: Bay Terrace, Staten Island
Visited: Kew Gardens
Visited: Dyker Heights
Visited: Fresh Meadows
Visited: Springfield Gardens
Visited: New Brighton
Visited: Brighton Beach
Visited: Throgs Neck
Visited: Howland Hook
Visited: Sheepshead Bay
Visited: Ditmars Steinway
Visited: Washington Heights
Visited: Allerton
Visited: Kensington
Visited: Laurelton
Visited: Roosevelt Island
Visited: Flushing
Visited: St. Albans
Visited: Jamaica
Visited: Bellerose
Visited: Pelham Gardens
Visited: Edgemere
Visited: Glendale
Visited: Astoria
Visited: Flatbush
Visited: Bayside
Visited: Longwood
Visited: East Flatbush
Visited: Bedford-Stuyvesant
Visited: Richmond Hill
Visited: Williamsbridge
Visited: Midland Beach
Visited: Mott Haven
Visited: College Point

Visited: Bayswater
Visited: Inwood
Visited: Mariners Harbor
Visited: Stapleton
Visited: Port Richmond
Visited: Claremont Village
Visited: Marble Hill
Visited: Hollis
Visited: Fort Hamilton
Visited: Cypress Hills
Visited: Great Kills
Visited: South Beach
Visited: Prospect Heights
Visited: South Slope
Visited: Red Hook
Visited: Tottenville
Visited: Clinton Hill
Visited: Arverne
Visited: Gowanus
Visited: Navy Yard
Visited: Spuyten Duyvil
Visited: Shore Acres
Visited: Downtown Brooklyn
Visited: Fort Greene
Visited: Lighthouse Hill
Visited: Grymes Hill
Visited: Chinatown
Visited: Ozone Park
Visited: East New York
Visited: Sunnyside
Visited: Queens Village
Visited: East Morrisania
Visited: Clifton
Visited: Wakefield
Visited: Morrisania
Visited: Concourse
Visited: Woodside
Visited: Maspeth
Visited: Rego Park
Visited: Bushwick
Visited: Lower East Side

Visited: Melrose
Visited: Douglaston
Visited: North Riverdale
Visited: Co-op City
Visited: Oakwood
Visited: Randall Manor
Visited: Baychester
Visited: Rossville
Visited: Parkchester
Visited: Edenwald
Visited: Tompkinsville
Visited: Belmont
Visited: Norwood
Visited: Gravesend
Visited: Morris Heights
Visited: Concourse Village
Visited: Port Morris
Visited: Kingsbridge
Visited: Jackson Heights
Visited: Silver Lake
Visited: Rosedale
Visited: Fieldston
Visited: Ridgewood
Visited: Highbridge
Visited: New Springville
Visited: Little Neck
Visited: Cambria Heights
Visited: Richmondtown
Visited: Midwood
Visited: East Elmhurst
Visited: Morris Park
Visited: Westerleigh
Visited: West Brighton
Visited: Mount Hope
Visited: Schuylerville
Visited: University Heights
Visited: Bath Beach
Visited: Bensonhurst
Visited: Fordham
Visited: Graniteville
Visited: Brownsville

Visited: South Ozone Park
Visited: Elmhurst
Visited: Dongan Hills
Visited: Emerson Hill
Visited: Arden Heights
Visited: Woodhaven
Visited: Far Rockaway
Visited: Todt Hill
Visited: Belle Harbor
Visited: Carroll Gardens
Visited: Park Slope
Visited: Upper East Side
Visited: City Island
Visited: East Village
Visited: Boerum Hill
Visited: Upper West Side
Visited: Olinville
Visited: Little Italy
Visited: Mill Basin
Visited: Borough Park
Visited: Castle Hill
Visited: Woodlawn
Visited: Corona
Visited: Mount Eden
Visited: Concord
Visited: Grant City
Visited: New Dorp
Visited: New Dorp Beach
Visited: Bronxdale
Visited: Hunts Point
Visited: Tremont
Visited: Soundview
Visited: Bull's Head
Visited: Jamaica Estates
Visited: Vinegar Hill
Visited: Kips Bay
Visited: Civic Center
Visited: Brooklyn Heights
Visited: Gramercy
Visited: DUMBO
Visited: Nolita

```
Visited: Hell's Kitchen
Visited: Cobble Hill
Visited: Battery Park City
Visited: Breezy Point
Visited: Murray Hill
Visited: Financial District
Visited: Theater District
Visited: Greenwich Village
Visited: Chelsea
Visited: Riverdale
Visited: West Village
Visited: SoHo
Visited: Willowbrook
Visited: Midtown
Visited: Neponsit
Visited: Flatiron District
Visited: NoHo
Top 5 neighborhoods by closeness centrality:
Bay Terrace, Staten Island -> 0.0019157088
Prospect-Lefferts Gardens -> 0.0019047619
Sunset Park -> 0.0019047619
St. Albans -> 0.0019047619
Astoria -> 0.0019047619
```

cargo test # runs unit tests, see point E for sample output and more details.

Runtime is under 1 second.

H. Conclusion

This project demonstrates how basic graph algorithms can be applied to real-world economic data. By modeling neighborhoods as nodes connected through price similarity, we uncovered clusters of affordability and potential economic hubs in NYC's Airbnb market. With more time, we could expand this work by including room type or availability filters, or experimenting with different similarity metrics.