DS210 Project Writeup

Summary

My project revolves around heart disease data and analyzes the risk of exercise induced angina by modeling a graph network. Each of my nodes represents a different patient from my dataset. The edges indicate the similarity between patients based on select health metrics. My project had three different modules: main.rs, graph.rs, and adjacent.rs.

Dataset

I chose a heart disease dataset that had 1025 patients and 14 columns that consisted of different health metrics. These metrics were age, sex, cp (chest pain (0-4)), trestbps (resting blood pressure), chol (serum cholesterol in mg/dl), fbs (fasting blood sugar > 120 mg/dl), restecg (resting electrocardiographic results (values 0,1,2)), thalach (maximum heart rate achieved), exang (exercise induced angina), oldpeak (ST depression induced by exercise relative to rest), slope (the slope of the peak exercise ST segment), ca (number of major vessels (0-3) colored by fluoroscopy), thal (result of a thallium stress test (0-3)), and target (whether patient has heart disease). I added another column called ptID so that I could identify the patients by number. When coding my graph I only chose a select few columns to include. The columns I chose were cp, chol, thalach, exang, oldpeak, ca, and target. I chose these specific ones because when I analyzed the data five of these columns fell in the top 5 indicators and the others I knew from my previous classes as being important to heart health.

Module 1: graph.rs

I started this module by creating a public struct called Graph which represents a graph with four different fields - the number of nodes, a mapping of patient IDs to tuples containing six features and a value representing whether a patient has angina, a hashmap representing the neighbors for each node (adjacency list), and a matrix that represents whether two nodes are connected. When I am implementing Graph I have a variety of functions: new, undirected, high_risk_pts, predict_angina, distances, node_distance, analyze_neighborhoods, edge_density, average_path_length, shortest_paths, clustering_coefficient, components, and, find_components. The function, new, initializes a new graph. The function, undirected, ensures bidirectional connections between all the nodes which converts a directed graph into an undirected one. The function, high_risk_pts, identifies high-risk patients whose neighbors have a high proportion of angina cases by counting the total neighbors and the subset of neighbors with angina, calculating the angina rate, and then classifying them as at high risk if over a certain percent of their neighbors are positive for angina. The function, predict_angina, predicts whether a given patient is likely to have angina based on their neighbors by retrieving neighbors of the given patient, counting the number of neighbors with angina, computing the ratio, and comparing it to a threshold. The function, distances, computes and prints the distances from one node to all the connecting nodes. The function, node_distance, computes the shortest distances from a single

node to all other nodes using BFS. The function, analyze_neighborhoods, prints each node and its immediate neighbors for analysis. The function, edge_density, calculates the graph's edge density using the formula (number of edges) / (n * (n - 1) / 2). The function, average_path_length, calculates the average shortest path length across all pairs of nodes. This has a helper function, shortest_paths, that computes the shortest paths from a given node. The function, clustering_coefficient, calculates the clustering coefficient of the graph by counting the number of actual connections between its neighbors, dividing by the maximum possible connections to compute a local clustering coefficient, and then averaging the coefficients over all nodes. The function, components, identifies and prints all connected components in the graph. This has a helper function, find_components, that explores all nodes in a connected component by using BFS to traverse and mark all nodes in the same component and then counting the number of nodes. These functions are important because together, they enable me to explore the graph in great detail. I did consult ChatGPT for some of these functions, I prompted ChatGPT to give me ideas for what functions might help me explore my graph more and make sense in the context of healthcare data.

Module 2: adjacent.rs

I only have one function in this module, createadj. This function generates the adjacency list and adjacency matrix for a graph based on the nodes and a selected similarity threshold. An important part of this function is that I normalized all the values that I drew from the data. I normalized them by calculating the absolute difference for each attribute between two nodes and dividing that by the maximum value for that attribute. I then used a euclidean distance formula to combine them all into one distance which then represents the similarity between the two nodes. This is the function in my code that allows me to construct a graph where nodes represent patients, and edges represent similarity relationships in order to make comparisons and draw conclusions.

Module 3: main.rs

I started this module with the function read which prepared my raw data for graph construction and analysis. I created my graph and then performed the operations I defined in the graph.rs module. This module also contained my tests. I had four different tests in my code: test_read_function, test_graph_initialization, test_high_risk_pts, and test_predict_angina. The test, test_read_function, ensures the dataset is correctly read and contains the expected number of nodes. The test, test_graph_initialization, verifies that the graph initializes correctly with the expected number of nodes. The test, test_high_risk_pts, ensures the graph correctly identifies high-risk patients. The test, test_predict_angina, validates the angina prediction logic.

Output

I had quite a lot of output from my code. I saved it in my src file under the name ____ because there was too much output to fit in the terminal at once. I had a few different sections in my output. The first set listed a node and its immediate neighbors, as seen in Image 1.

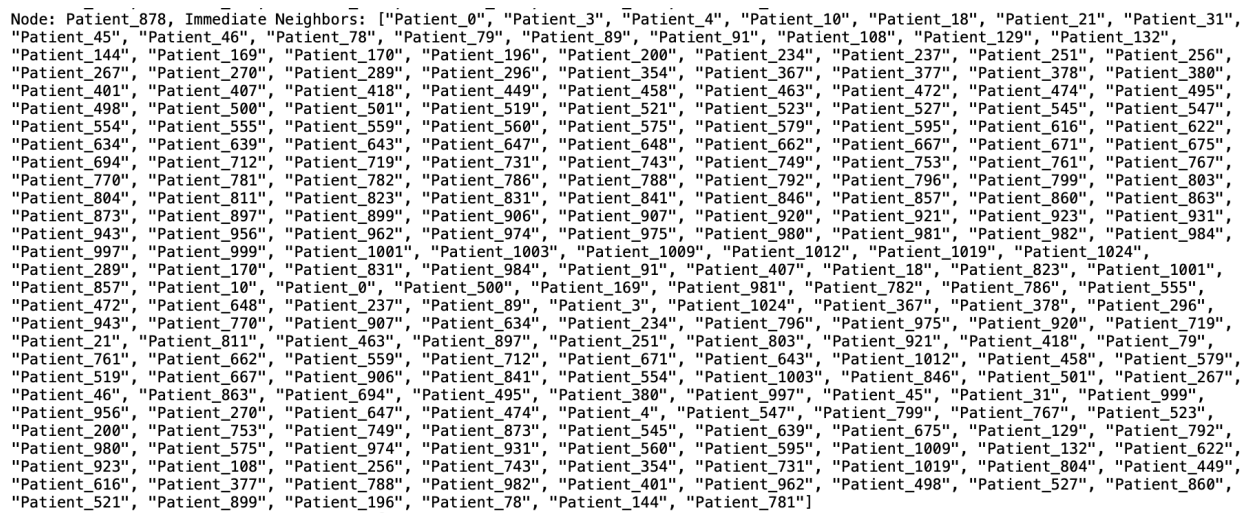

Image 1. *An example from the first set of output. Lists a node, Patient_878, and all of its immediate neighbors.*

The next set listed a node and the distance each of its neighbors was from it. This is seen in Image 2 which shows a snippet of what this output looked like.

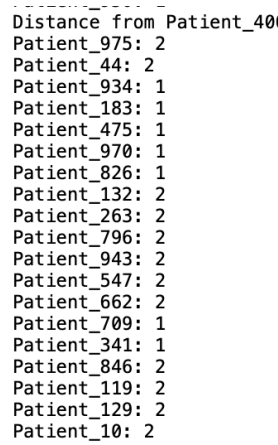

Image 2. *An example from the second set of output. Lists a node, Patient_400, and the distances of a few of its neighbors.*

The next set of output listed the number of components as well as how many nodes there were in each component. This is seen in Image 3.

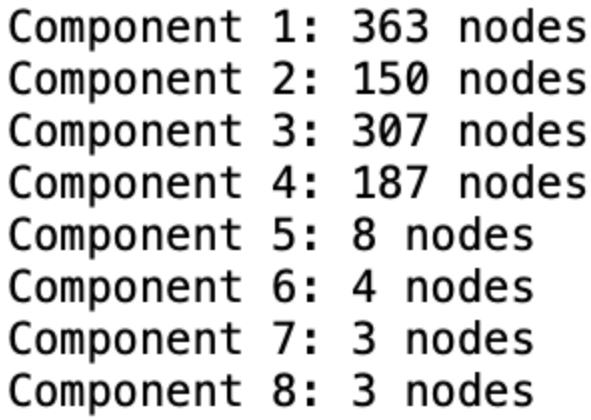

Image 3. *An example from the third set of output. This shows that there were 8 components as well as how many nodes were in each component.*

The next set of output listed the at risk patients based on neighbor angina rate. This is seen in Image 4.

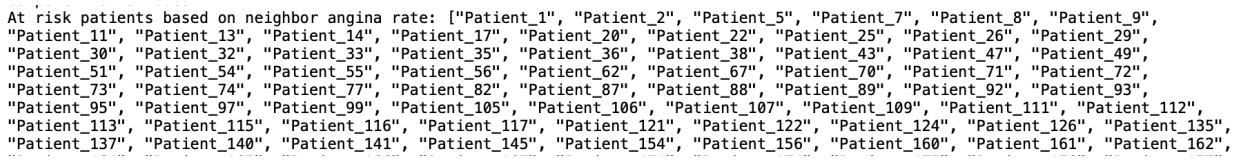

Image 4. *An example from the fourth set of output. This shows a select few of the patients that could be at risk for exercise induced angina based on their neighbors.*

The next set of output listed characteristics of the graph such as the edge density, average path length, and clustering coefficient. This is seen in Image 5.

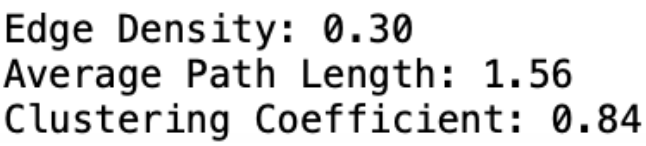

Image 5. *An example from the fifth set of output. This shows the values for three characteristics of the graph.*

The last set of output listed the results of whether one select patient was likely to have exercise induced angina. This is seen in Image 6.

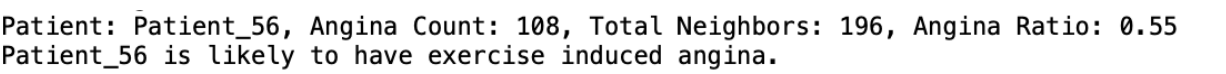

Image 6. *An example from the sixth set of output. This shows the information for Patient_56 and indicates that they are likely to have exercise induced angina.*

Analysis

Since my data was health attribute based, connections between the different nodes in my graph represent relationships between patients based on similarity in health attributes. By analyzing this graph, I can look a little bit into patient clusters, patterns of angina risk, and just the general structure of the patient network. The edge density and clustering coefficient numbers indicate a moderately dense graph that highlights a strong tendency for patients to form very tightly knit groups based on health similarities. High-risk patients can serve as focal points for identifying and addressing potential health complications. If something like this is implemented in the real world at a greater depth, it might be able to help healthcare providers let patients know in advance if they are at a greater risk of angina which might help them change their lifestyle habits and therefore maybe reduce their risk. By leveraging all of these different insights, healthcare providers can design effective strategies for patient monitoring, resource allocation, and targeted interventions, ultimately improving health outcomes across the network.