Large-Scale Data Mining: Models and Algorithms
ECE 232E Spring 2018
Prof. Vwani Roychowdhury
UCLA, Department of ECE

Project 5
Graph Algorithms

Due on Friday, June 15, 2018 by 11:59 PM

Introduction

In this project we will explore graph theory theorems and algorithms, by applying them on real data. In the first part of the project, we consider a particular graph which models correlations between stock price time series. In the second part, we analyse traffic data on a dataset provided by Uber.

1 Stock Market

In this part of the project, we study data from stock market. The data is available on this *Dropbox Link*. The goal of this part is to study correlation structures among fluctuation patterns of stock prices using tools from graph theory. The intuition is that investors will have similar strategies of investment for stocks that are effected by the same economic factors. For example, the stocks belonging the transportation sector may have different absolute prices, but if for example fuel prices change or are expected to change significantly in the near future, then you would expect the investors to buy or sell all stocks similarly and maximize their returns. Towards that goal, we construct different graphs based on similarities among the time series of returns on different stocks at different time scales (day vs a week). Then, we study properties of such graphs. The data is obtained from Yahoo Finance website for 3 years. You're provided with a number of csv tables, each containing several fields: Date, Open, High, Low, Close, Volume, and Adj Close price. The files are named according to *Ticker Symbol* of each stock. You may find the market sector for each company in `Name_sector.csv`.

1.1 Return correlation

In this part of the project, we will compute the correlation among log-normalized stock-return time series data. Before giving the expression for correlation, we introduce the following notation:

- $p_i(t)$ is the closing price of stock i at the t^{th} day
- $q_i(t)$ is the return of stock i over a period of $[t-1, t]$

$$q_i(t) = \frac{p_i(t) - p_i(t-1)}{p_i(t-1)}$$

- $r_i(t)$ is the log-normalized return stock i over a period of $[t-1, t]$

ri(t) = log pi(t) – log pi(t – τ)

$$r_i(t) = \log(1 + q_i(t))$$

it's the same as before.

Then with the above notation, we define the correlation between the log-normalized stock-return time series data of stocks i and j as

$$\rho_{ij} = \frac{\langle r_i(t) r_j(t)\rangle - \langle r_i(t)\rangle\langle r_j(t)\rangle}{\sqrt{(\langle r_i(t)^2\rangle - \langle r_i(t)\rangle^2)(\langle r_j(t)^2\rangle - \langle r_j(t)\rangle^2)}}$$

where $\langle\cdot\rangle$ is a temporal average on the investigated time regime (for our data set it is over 3 years).

Question 1: Provide an upper and lower bound on ρ_{ij}. Also, provide a justification for using log-normalized return ($r_i(t)$) instead of regular return ($q_i(t)$).

1.2 Constructing correlation graphs

In this part,we construct a correlation graph using the correlation coefficient computed in the previous section. The correlation graph has the stocks as the nodes and the edge weights are given by the following expression

$$w_{ij} = \sqrt{2(1 - \rho_{ij})}$$

Compute the edge weights using the above expression and construct the correlation graph.

Question 2: Plot the degree distribution of the correlation graph and a histogram showing the un-normalized distribution of edge weights.

1.3 Minimum spanning tree (MST)

In this part of the project, we will extract the MST of the correlation graph and interpret it.

Question 3: Extract the MST of the correlation graph. Each stock can be categorized into a sector, which can be found in `Name_sector.csv` file. Plot the MST and color-code the nodes based on sectors. Do you see any pattern in the MST? The structures that you find in MST are called Vine clusters. Provide a detailed explanation about the pattern you observe.

1.4 Sector clustering in MST's

In this part, we want to predict the market sector of an unknown stock. We will explore two methods for performing the task. In order to evaluate the performance of the methods we define the following metric

$$\alpha = \frac{1}{|V|} \sum_{v_i \in V} P(v_i \in S_i)$$

where S_i is the sector of node i. Define

$$P(v_i \in S_i) = \frac{|Q_i|}{|N_i|}$$

where Q_i is the set of neighbors of node i that belong to the same sector as node i and N_i is the set of neighbors of node i. Compare α with the case where

$$P(v_i \in S_i) = \frac{|S_i|}{|V|}$$

Question 4: Report the value of α for the above two cases and provide an interpretation for the difference.

1.5 Correlation graphs for weekly data

In the previous parts, we constructed the correlation graph based on daily data. In this part of the project, we will construct a correlation graph based on weekly data. To create the graph, sample the stock data weekly on Mondays and then calculate ρ_{ij} using the sampled data. If there is a holiday on a Monday, we ignore that week. Create the correlation graph based on weekly data.

Question 5: Extract the MST from the correlation graph based on weekly data. Compare the pattern of this MST with the pattern of the MST found in question 3.

2 Let's Help Santa!

Companies like Google and Uber have a vast amount of statistics about transportation dynamics. Santa has decided to use network theory to facilitate his gift delivery for the next christmas. When we learned about his decision, we designed this part of the project to help him. We will send him your results for this part!

2.1 Download the Data

Go to “Uber Movement” website and download data of **Monthly Aggregate (all days), 2017 Quarter 4**, for San Francisco area [1]. The dataset contains pairwise traveling time statistics between most pairs of points in San Francisco area. Points on the map are represented by unique IDs. To understand the correspondence between map IDs and areas, download **Geo Boundaries** file from the same website [2]. This file contains latitudes and longitudes of the corners of the polygons circumscribing each area. In addition, it contains one street address inside each area, referred to as `DISPLAY_NAME`. To be specific, if an area is represented by a polygon with 5 corners, then you have a 5×2 matrix of the latitudes and longitudes, each row of which represents latitude and longitude of one corner.

2.2 Build Your Graph

Read the dataset at hand, and build a graph in which nodes correspond to locations, and undirected weighted edges correspond to the mean traveling times between each pair of locations (**only December**). Add the following attributes to the vertices:

1. Display name: the street address
2. Location: mean of the coordinates of the polygon’s corners (a 2-D vector)

[1] If you download the dataset correctly, it should be named as `san_francisco-censustracts-2017-4-All-MonthlyAggregate.csv`

[2] The file should be named `SAN_FRANCISCO_CENSUSTRACTS.JSON`

The graph will contain some isolated nodes (extra nodes existing in the Geo Boundaries JSON file) and a few small connected components. Remove such nodes and just keep the giant connected component of the graph. In addition, merge duplicate edges by averaging their weights [3]. We will refer to this cleaned graph as G afterwards.

Question 6: Report the number of nodes and edges in G.

2.3 Traveling Salesman Problem

Question 7: Build a minimum spanning tree (MST) of graph G. Report the street addresses of the two endpoints of a few edges. Are the results intuitive?

Question 8: Determine what percentage of triangles in the graph (sets of 3 points on the map) satisfy the triangle inequality. You do not need to inspect all triangles, you can just estimate by random sampling of 1000 triangles.

Now, we want to find an approximation solution for the traveling salesman problem (TSP) on G. Apply the 2-approximate algorithm described in the class[4]. Inspect the sequence of street addresses visited on the map and see if the results are intuitive.

Question 9: Find the empirical performance of the approximate algorithm:

$$\rho = \frac{\text{Approximate TSP Cost}}{\text{Optimal TSP Cost}}$$

Question 10: Plot the trajectory that Santa has to travel!

[3] Duplicate edges may exist when the dataset provides you with the statistic of a road in both directions. We remove duplicate edges for the sake of simplicity.

[4] You can find the algorithm in: Papadimitriou and Steiglitz, *"Combinatorial optimization: algorithms and complexity"*, Chapter 17, page 414

3 Analysing the Traffic Flow

Next December, there is going to be a sport event between Stanford University and University of California, Santa Cruz (UCSC). A large number of students are enthusiastic about the event, which is going to be held in UCSC. Stanford fans want to drive from their campus to the rival's. We would like to analyse the maximum traffic that can flow from Stanford to UCSC.

3.1 Estimate the Roads

We want to estimate the map of roads without using actual road datasets. Educate yourself about *Delaunay triangulation* algorithm and then apply it to the nodes coordinates[5].

Question 11: Plot the road mesh that you obtain and explain the result. Create a subgraph G_Δ induced by the edges produced by triangulation.

3.2 Calculate Road Traffic Flows

Question 12: Using simple math, calculate the traffic flow for each road in terms of cars/hour.

Hint: Consider the following assumptions:

- Each degree of latitude and longitude $\approx$ 69 miles
- Car length $\approx$ 5 m = 0.003 mile
- Cars maintain a safety distance of 2 seconds to the next car
- Each road has 2 lanes in each direction

[5] You can use `scipy.spatial.Delaunay` in python

Assuming no traffic jam, consider the calculated traffic flow as the max capacity of each road.

3.3 Calculate the Max Flow

Consider the following addresses:

- Source address: 100 Campus Drive, Stanford
- Destination address: 700 Meder Street, Santa Cruz

Question 13: Calculate the maximum number of cars that can commute per hour from Stanford to UCSC. Also calculate the number of edge-disjoint paths between the two spots. Does the number of edge-disjoint paths match what you see on your road map?

3.4 Defoliate Your Graph

In G_Δ, there are a number of unreal roads that could be removed. For instance, there are many fake bridges crossing the bay. Apply a threshold on the travel time of the roads in G_Δ to remove the fake edges. Trim the fake edges and call the resulting graph $\tilde{G}_\Delta$.

Question 14: Plot $\tilde{G}_\Delta$ on real map coordinates. Are real bridges preserved?

Hint: You can consider the following coordinates:

- Golden Gate Bridge: [[-122.475, 37.806], [-122.479, 37.83]]
- Richmond, San Rafael Bridge: [[-122.501, 37.956], [-122.387, 37.93]]

- San Mateo Bridge: [[-122.273, 37.563], [-122.122, 37.627]]
- Dambarton Bridge: [[-122.142, 37.486], [-122.067, 37.54]]
- San Francisco - Oakland Bay Bridge: [[-122.388, 37.788], [-122.302, 37.825]]

Question 15: Now, repeat question 8 for $\tilde{G}_{\Delta}$ and report the results. Do you see any significant changes?