Clustering Similar Nodes in Groups with Eigenvectors in Linear Algebra

by Sta. Maria, Lucas

1 Introduction

Eigenvectors and eigenvalues have a multitude of real-world applications that allow for efficient handling of data. Companies employ mathematical calculations to ensure that data is organized and sent as best as possible, refining these algorithms to maximise profits gleaned from the information they've collected. For many prominent services, linear algebra plays a massive role in the efficient organization of information. Specialized eigenvectors and eigenvalues allow corporations to achieve this.

Linear algebra plays an important part in the numeric representation of graphs using matrices. *Graphs* are structures used to display mathematical relationships. *Nodes* (otherwise known as vertices) are individual points, represented with a circle. *Edges* are set between two nodes to signify a relationship between them. Graphs can be *directed*; that is, if an edge exists between two nodes, it may go in one or two directions. Graphs can also be *weighted*; that is, the edges between the nodes can have a certain weight representing the significance of the relationship between the two nodes. Initially, I will cover *unweighted* graphs, where nodes either have an edge or don't. For our initial explorations, we shall only be covering graphs that have a maximum of a singular edge between nodes.

If we have a given set of nodes with edges between them, we can represent an edge between nodes in an unweighted, undirected graph. First, we assign each node a unique number from 1 to n, where n is equal to the number of nodes. Then we create an $n \times n$ matrix A where the entry $a_{i,j}$ holds a number 1 or 0. This is called an *adjacency matrix*. If the ith node has an edge between it with the jth vector, then the entry $a_{i,j}$ stores the value 1, else it stores the value 0 to represent no edge. If we have a graph with three nodes identified by values 1, 2, and 3, and Node 1 is connected to Node 2 and Node 3, we can represent it with the following matrix:

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

To provide an interesting observation, the transpose of matrix A representing an undirected graph is equivalent to matrix A. Mathematically, $A = A^T$.

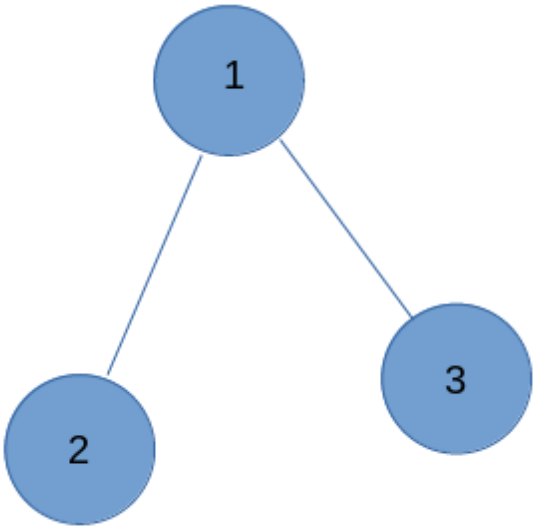

Figure 1. An unweighted, undirected graph with three nodes

There are some transformation matrices that stretch or shrink a vector when multiplied by it. The equation $\mathrm{Ax} = \lambda x$ represents a transformation matrix A stretching or shrinking the vector x by the scalar multiple λ. The scalar multiple λ is called an *eigenvalue*, and x is an *eigenvector* of A that corresponds with the eigenvalue λ.

The important application of eigenvectors and eigenvalues I shall be analyzing and extending is social media and how we can organize people into groups, making it easier to form new connections and make assumptions about individuals. This paper will describe how eigenvectors and eigenvalues enable us to do so.

2 Clustering

2.1 Method

In unweighted graphs with more nodes, it may be beneficial to organize nodes into "clusters", groups where nodes share similar edges to other nodes. When organized into clusters, it becomes easier to view the relationships between nodes and make generalizations about the groups. An example would be analyzing an individual's network in social media. With clustering, it becomes easier to identify groups such as family, friends, and co-workers.

Suppose we have an adjacency matrix A representing an unweighted graph. The first step of clustering is forming the diagonal matrix D from A, where the entry $d_{i,i}$ is the sum of the entries in the ith row. Next, we can form the *Laplacian matrix* L by subtracting A from D: $L = D - A$, which is used to represent a graph in an effective manner. By retrieving the eigenvalues of the Laplacian matrix, then finding the eigenvector corresponding to the second-smallest eigenvector, we get the *Fiedler vector*.

The Fiedler vector and its significance was discovered by Miroslav Fiedler. Its use dictates how nodes are clustered based on its entries. If we discretize the values to positive and negative, we can sort the nodes into two groups. The ith node is the first group if its entry is positive, else it will be in the second group since its value is negative.

This process allows us to split our group of nodes into two clusters. If we want to split it into more clusters, the process is marginally simpler. Let our first Fiedler vector be denoted with F_1. Our second Fiedler vector F_2 will be the eigenvector corresponding to the next smallest eigenvalue. We can then make a matrix F whose columns will store the Fiedler vectors. Therefore, $F = [\ F_1 \ \ F_2\ ]$. If we again discretize the entries in the Fiedler vectors to positive and negative values, each row

in F will store a pair where each element in the pair is either positive or negative. With the pairs, ordering matters; $[+\ -]$ shall be treated differently from $[-\ +]$, but $[+\ +]$ is the same as $[+\ +]$. The ith row in F, and thus each pair, corresponds with the ith node. With this second-level of clustering, we now have a maximum of four different clusters of nodes, where nodes are grouped with other nodes that have alike sign patterns.

Additional levels of clustering can be added by repeating the process: finding the eigenvector corresponding to the next smallest eigenvalue, adding another column to F, then grouping nodes with the same sign patterns together. Further clustering, however, can produce a *singleton*, a cluster of only one node.

2.2 Example

Suppose we have the following network of friends, shown in Figure 2, represented with an unweighted, undirected graph.

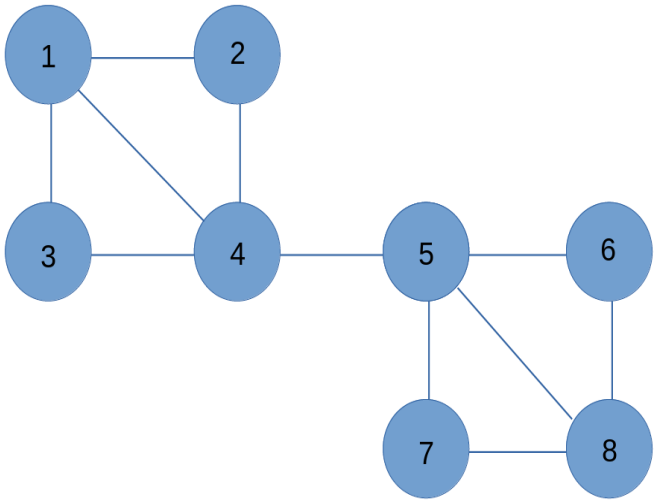

Figure 2. An unweighted, undirected graph with 8 nodes representing a social media network.

From visual inspection, we can roughly identify two clusters of nodes, where the first has nodes 1 through 4, and the second has nodes 5 through 8. If we represent their relationships in an adjacency matrix A, we get the following:

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

When we make the diagonal matrix D by summing up the rows of A, we get

$$D = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix}$$

Our Laplacian matrix, made from $L = D - A$ thus is

$$L = \begin{bmatrix} 3 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 2 & 0 & -1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 2 & -1 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & 4 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 4 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & -1 & 2 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 2 & -1 \\ 0 & 0 & 0 & 0 & -1 & -1 & -1 & 3 \end{bmatrix}$$

When we solve for the eigenvalues of L, we get $\lambda = 5.64575$, 4, 2.0441×10^{-15}, 0.35425, 2, 4, 4, 2. Our second smallest eigenvalue is 0.35425, and the eigenvector, or Fiedler vector, that corresponds to it is $[\ -0.38253 \quad -0.38253 \quad -0.38253 \quad -0.24702 \quad 0.24702 \quad 0.38253 \quad 0.38253 \quad 0.38253\ ]^T$. When we discretize the values of the Fiedler matrix, we get

$$\begin{bmatrix} - \\ - \\ - \\ - \\ + \\ + \\ + \\ + \end{bmatrix}$$

Our findings with the discretized Fiedler vector suggest that the first cluster should consist of nodes 1 through 4, and the second cluster should consist of nodes 5 through 8. See Figure 3 for a visual representation. This aligns with the initial visual inspection we performed at the start of the example. What if we wanted to partition the graph further into four clusters? Our next smallest eigenvalue is 2, which has two associated eigenvectors.

Our second Fiedler vector F_2 with its value discretized is

$$\begin{bmatrix} - \\ - \\ + \\ + \\ + \\ + \\ - \\ - \end{bmatrix}$$

When we form our matrix F from the two Fiedler vectors ($F = [\ F_1 \quad F_2\ ]$) and discretize the values, we get

$$\begin{bmatrix} - & - \\ - & - \\ - & + \\ - & + \\ + & - \\ + & - \\ + & + \\ + & + \end{bmatrix}$$

From the matrix F, we can partition our graph into four clusters where the first has nodes 1 and 2, the second 3 and 4, the third 5 and 6, and the last 7 and 8. See Figure 4 for a visual representation.

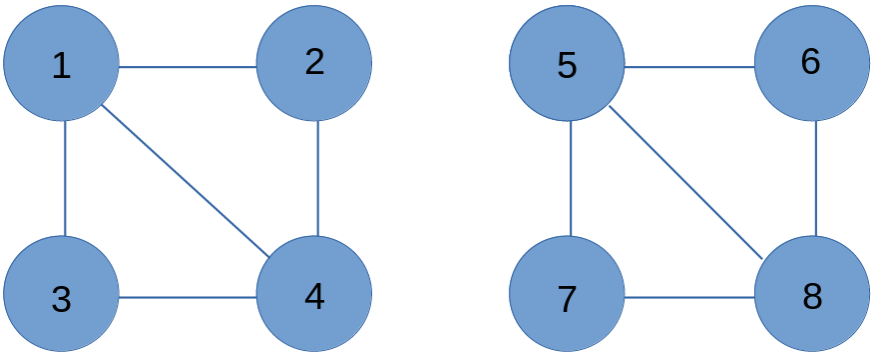

Figure 3. The first partition of the graph through the clustering method.

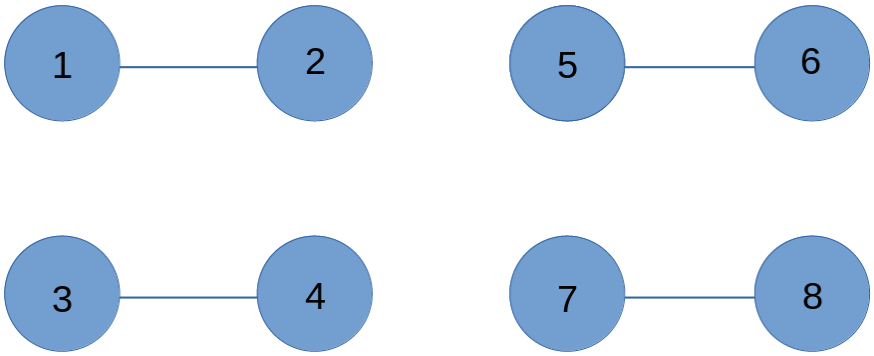

Figure 4. An additional partition of the graph through the extended clustering method.

2.3 Application

From our first partition into clusters, we identified two groups: the first comprised of nodes 1 through 4, and the second of nodes 5 through 8. This method of clustering is useful for social media platforms such as Snapchat, where being friends with another user is mutual. Let's take a look at the first cluster, consisting of nodes 1 through 4. Their relationships are represented as the following matrix:

$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

This group in social media could represent a friend group, a family, or colleagues in a company. In any case, this suggests that the unconnected persons 2 and 3 have a high chance of knowing each other. Snapchat can then suggest person 3 as a friend to person 2, and vice versa. Since there's a high chance of knowing each other, they're more likely to accept the friend request. Snapchat, however, is less likely to recommend someone from the second cluster as a friend, since there is a lower chance of forming a connection there.

Clustering is an elegant way of finding friends through grouping. An alternative would be to suggest second-layer friends, or friends of friends. This, however, is less accurate and would acquire the target user less friends than clustering. The reason is that clustering pools together friends with similar friends, so potential friends are ones the user is likely knows. Recommending second-layer

friends, on the other hand, may recommend a potential friend based on a friend who the user is not too close with, or not in their friend group. In other words, it wouldn't be effective unless it sorted potential friends based on how many friends they share, which is less elegant than clustering.

An additional problem associated with second-layer friend recommendations is its algorithmic nature. For each individual, an algorithm needs to traverse through their entire friends list, and for each friend of the friend, collect them into a set. It then needs to compare the set of second-layer friends with the individual's current friends then eliminate duplicates. Only then can it begin recommending friends, but yet again, the list may not be ordered accurately. Clustering, on the other hand, is more efficient because of how it partitions a group into two with each additional level.

Snapchat's goal is essentially to fill all non-diagonal entries in the cluster's matrix. Any non-diagonal entries that are 0 will cause Snapchat to recommend user i and user j to each other as friends. The deeper the levels of clustering, the more accurate friend recommendations become. If we start the level of clustering before a target node, or user, becomes a singleton, and use its cluster to recommend friends, we'll have the highest accuracy for friend recommendations. The level of clustering before that will hold lower-priority friend recommendations, and so on until there are no more clusters.

3 Extension

3.1 Going Further

Disclaimer: I don't know if this has been done before. This is all through my own investigation. I used supplementary materials to clarify my existing knowledge.

We've taken a look at the method of clustering and its applications to social media platforms such as Snapchat. Snapchat, however, is a platform that requires mutual friendship to form a connection. That is, if user i is friends with user j, then the opposite is likewise true. In platforms like Instagram that use the follow feature, this is not necessarily the case. If user i follows user j, that doesn't mean that user j follows user i. In this section, we will be looking at further extensions to clustering to accomodate platforms such as Instagram.

3.2 Clustering with Directed Graphs

In directed graphs, edges between two nodes can have either one or two directions. Directed graphs are more suitable for representing relationships in platforms such as Instagram, since a relationship between two users doesn't imply mutuality.

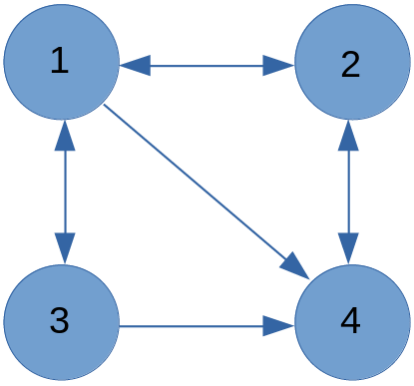

Figure 5. An unweighted, directed graph with 4 nodes representing a social media network.

In Figure 5 we have a directed graph representing a social media network where relationships are not mutual. The four nodes are interconnected, with user 1 and user 3 both following user 4, but user 4 not following them back. If we represent the relationships with a matrix A, we get

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Unlike before, the transpose of matrix A does not equal matrix A because it is a directed graph. Expressed mathematically, $A \neq A^T$. Let's produce the diagonal matrix by summing the rows of matrix A:

$$D = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

We form the Laplacian matrix L through $L = D - A$.

$$L = \begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 2 & 0 & -1 \\ -1 & 0 & 2 & -1 \\ 0 & -1 & 0 & 1 \end{bmatrix}$$

Solving for the eigenvalues of L, we get $\lambda = 4,\ 2,\ -2.10711 \times 10^{-15},\ 2$. Our second smallest eigenvalue is 2, with a multiplicity of 2. The eigenvectors associated with it, discretized, are

$$\begin{bmatrix} + \\ + \\ + \\ - \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} - \\ - \\ - \\ + \end{bmatrix}$$

The result is functionally equivalent. This method makes node 4 a singleton, grouping the other nodes together. To pair it with the Instagram example, this suggests that node 4 is more along the lines of a public figure. When a person has a significant difference between their followers count and the amount of people they follow, they are unlikely to follow people back, and are thus singled out from clusters.

3.3 Clustering with Weighted Graphs

Weighted graphs are another type of graph. Instead of edges being a boolean value (has a connection, does not have a connection), weighted graphs place higher significance on certain edges by assigning each edge a "weight". Social media platforms may want to place more value in a relationship if there is more interaction between the two users, where interaction could be defined by the chats sent between them, the length of their relationship, or the amount of group chats they share. Whatever the reason, using weighted edges to represent the strength of the relationship has useful applications for social media platforms.

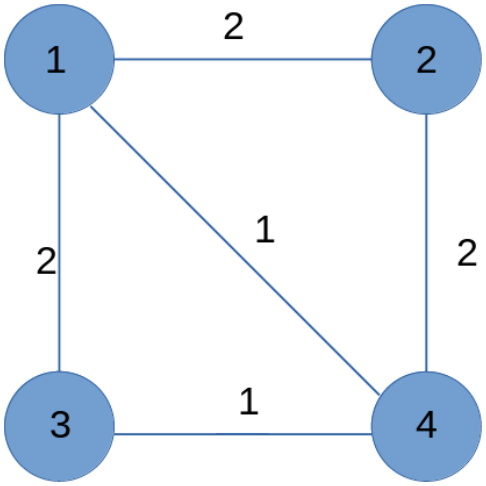

Figure 6. A weighted, undirected graph with 4 nodes representing a social media network.

In Figure 6, we have a very similar graph to Figure 5, but with weighted edges rather than directed ones. If we represent the relationships with matrix A, we get

$$A = \begin{bmatrix} 0 & 2 & 2 & 1 \\ 2 & 0 & 0 & 2 \\ 2 & 0 & 0 & 1 \\ 1 & 2 & 1 & 0 \end{bmatrix}$$

Our method of producing the diagonal matrix D for weighted graphs from A needs to be clarified. The diagonal entries should sum the number of non-zero entries in the row, rather than summing the entries themselves. Our D matrix shall look like this

$$D = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix}$$

We still form the Laplacian matrix L from $L = D - A$:

$$L = \begin{bmatrix} 3 & -2 & -2 & 1 \\ -2 & 2 & 0 & -2 \\ -2 & 0 & 2 & -1 \\ -1 & -2 & -1 & 3 \end{bmatrix}$$

Solving for the eigenvalues of L, we get $\lambda = 5.89869, \ -0.98784, 3.34455, 1.74461$. The second smallest eigenvalue is 1.74461 and its associated eigenvector, with values discretized, is

$$\begin{bmatrix} + \\ - \\ + \\ - \end{bmatrix}$$

At first, that doesn't quite make sense. Shouldn't the distribution be the same as the directed graph shown in Figure 5? After careful observation of Figure 6, however, the groups are consistent. Nodes 1 and 3 both have an edge of weight 1 to node 4, and have an edge of weight 2 between each other. Nodes 2 and 3 do not have an edge between each other, so they should not be in the same

cluster. Nodes 2 and 4 have an edge with weight 2 between each other. Thus, since nodes 1 and 3 have a "strong" edge between each other, both have a "weak" edge to node 4, and one doesn't have an edge to node 2, it makes sense to cluster them together. The remaining nodes, 2 and 4, cluster together because they have a strong edge.

With weighted graphs, social media platforms are able to cluster users together more accurately. As mentioned previously, weights are based on metrics that the platform deems an important factor in measuring a relationship. One such metric might be the amount of texts sent between two users, as it indicates that the two users enjoy conversation with each other. Another metric might be the amount of recent IP Addresses they share, which could indicate that they are schoolmates, work colleagues, or housemates. A combination of useful metrics allows social media platforms to effectively group together users.

3.4 Clustering with Weighted, Directed Graphs

By combining the two, we can represent a complicated network of users in a social media platform. There are several motivations behind this: first, directed graphs are better suited for platforms such as Instagram, where relationships aren't always mutual; second, weighted graphs allow us to place emphasis on the strength of certain relationships over others.

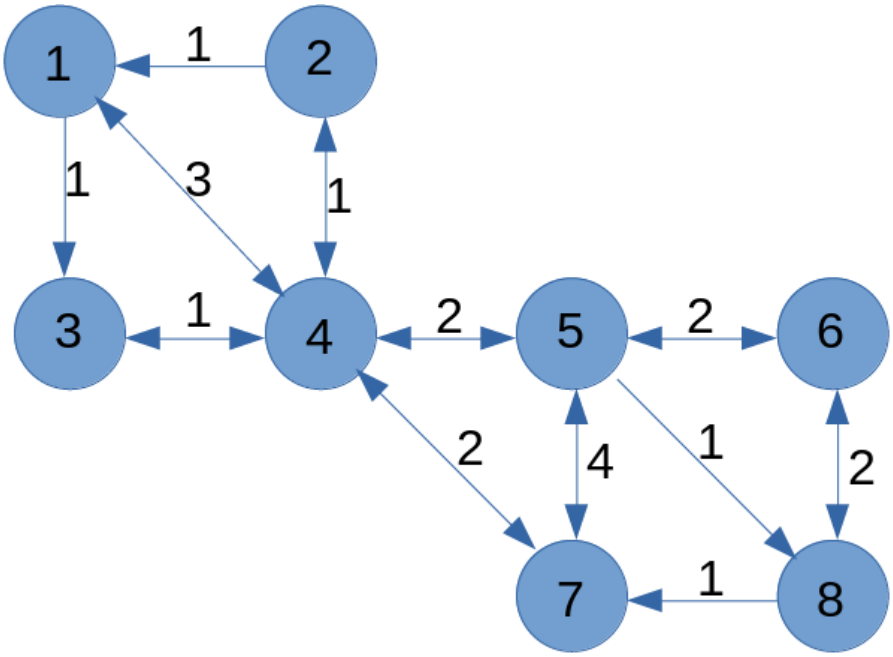

Figure 7. A weighted, directed graph representing a social media network.

Figure 7 shows a complicated weighted, directed graph showing the relationships between users. For non-mutual edges, or edges with a single direction, I took creative liberty by capping their weight at 1. My belief is that, in social media, there is much less significance in a relationship if it is one-sided. With matrix A, let's represent the relationships in this graph:

$$A = \begin{bmatrix} 0 & 0 & 1 & 3 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 3 & 1 & 1 & 0 & 2 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 & 0 & 2 & 4 & 1 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 2 \\ 0 & 0 & 0 & 2 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 1 & 0 \end{bmatrix}$$

Let's form the diagonal matrix D by summing the number of entries in each row.

$$D = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

Let's produce the Laplacian matrix L by performing $L = D - A$.

$$L = \begin{bmatrix} 2 & 0 & -1 & -3 & 0 & 0 & 0 & 0 \\ -1 & 2 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ -3 & -1 & -1 & 5 & -2 & 0 & -2 & 0 \\ 0 & 0 & 0 & -2 & 4 & -2 & -4 & -1 \\ 0 & 0 & 0 & 0 & -2 & 2 & 0 & -2 \\ 0 & 0 & 0 & -2 & -4 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 & -1 & 2 \end{bmatrix}$$

Solving for the eigenvalues of matrix L, we get $\lambda = -3.16155$, 7.97218, 7.21912, -0.38899, 3.77067, 2.24006, 1.48123, 0.86728. Our first Fiedler vector, discretized, produces

$$\begin{bmatrix} - \\ - \\ - \\ - \\ + \\ + \\ + \\ + \end{bmatrix}$$

This is the same result as our first example, shown in Figure 2. The clusters are shown in Figure 8. The partitions make sense, because there are more connections within these clusters than between them. When we use the extended clustering method to produce a matrix F where the columns are the discretized Fiedler vectors, we get

$$F = \begin{bmatrix} - & - \\ - & - \\ - & - \\ - & - \\ + & + \\ + & - \\ + & + \\ + & - \end{bmatrix}$$

F partitions our second cluster in two more, where the first contains nodes 5 and 7, and the second contains nodes 6 and 8. Again, this checks out. If we were to use visual inspection to split the second cluster, we would keep 5 and 7 together because of the edge between them having a weight of 4. The remainder would then be nodes 6 and 8.

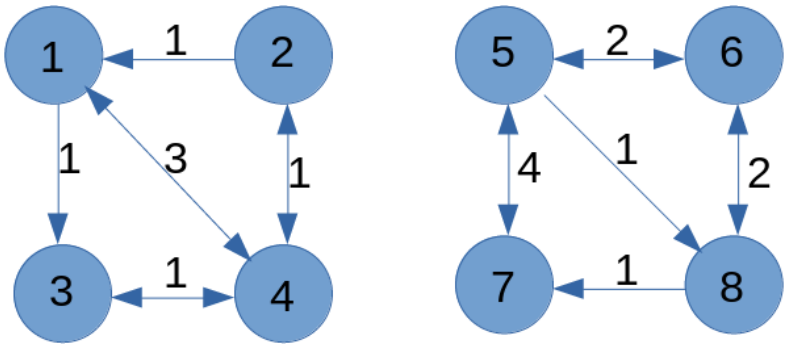

Figure 8. The first partition of the weighted, directed graph.

4 Conclusion

Clustering has the important application of using math to analyse relationships in larger networks. Today, social media platforms store the largest networks of people. In order for these platforms to provide services such as recommending friends, they use clustering to automate the process of creating arbitrary groups. This mathematical process is interesting to me because of how seemingly natural it is integrated into graph theory. The significance of the second smallest to largest eigenvalues and their corresponding eigenvectors have dramatically allowed the efficient handling and analysis of data.

The application of eigenvalues and eigenvectors in clustering is pretty cool. From the second smallest eigenvalue onwards, each of their corresponding eigenvector allows the graph to be partitioned into smaller groups. Clustering is centered around the discretized values of the eigenvectors, called Fiedler vectors. The entries in a Fiedler vector represent how connected a node is to the rest of the graph.

When I first began this project, the only applications of Linear Algebra in graph theory that I was aware of were Markov chains. In a sense, Laplacian matrices and stochastic matrices do have similarities, namely modeling the pathing between nodes. When I read through *When Life is Linear*, I became interested in clustering simply because it was amazing to see how math could organize data by itself.

My extension sprouted from my meager knowledge in graph theory (which stemmed from Computer Science). I was aware of directed and weighted graphs, and became interested in whether the clustering method still functioned with them. When I tried it out, I was surprised that it worked (thankfully, or else I wouldn't have an extension).

This project was incredibly interesting, so I wrote a lot. I remember last year, I was genuinely excited to take Linear Algebra because of its applications in Computer Science, and having this project over the test validates that excitement. I'll likely be using weighted graphs in my final project and further extending clustering there.

Basically, this is a pretty cool cookie.

5 Bibliography

Chartier, Timothy P. *When Life Is Linear*. The Mathematical Association of America., 2015.

Diestel, Reinhard. *Graph Theory*. http://diestel-graph-theory.com/basic.html?, 2016.

EMS Press. *Graph Theory*. https://encyclopediaofmath.org/index.php?title=Graph_theory, 2001.