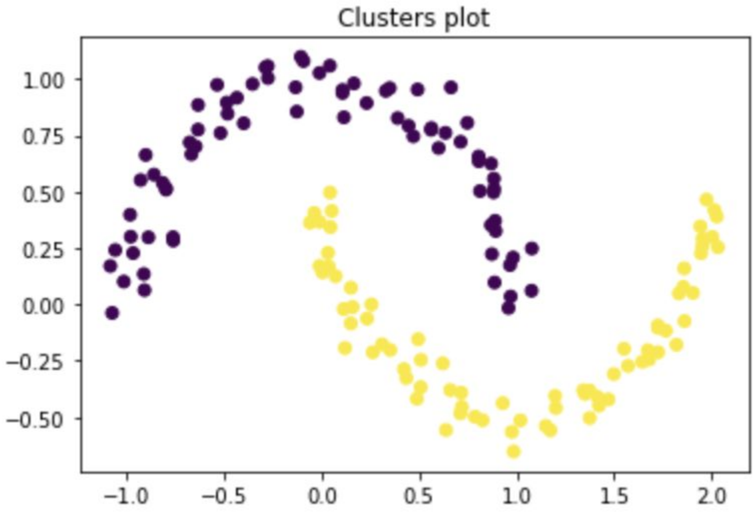


Clusters obtained from Spectral clustering

Graph Representations

Jay Urbain, PhD - 11/17/2022

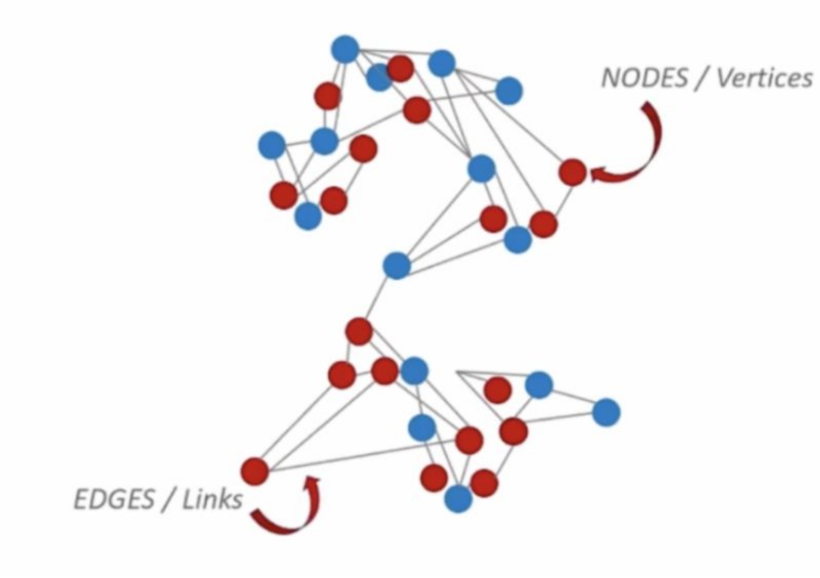
“Creativity is just connecting things. When you ask creative people how they did something, they feel a little guilty because they didn't really do it, they just saw something. It seemed obvious to them after a while. That's because they were able to ***connect experiences*** *they've had and synthesize new things.”*

-- Steve Jobs

Data Representation - Structured data

Structured data - in addition to defining the data points, define the relations between data points.

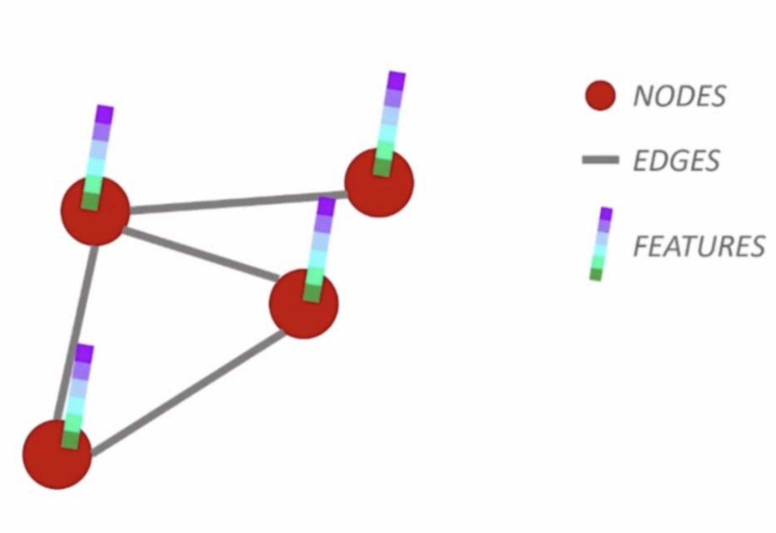
Edges connect nodes related to each other.

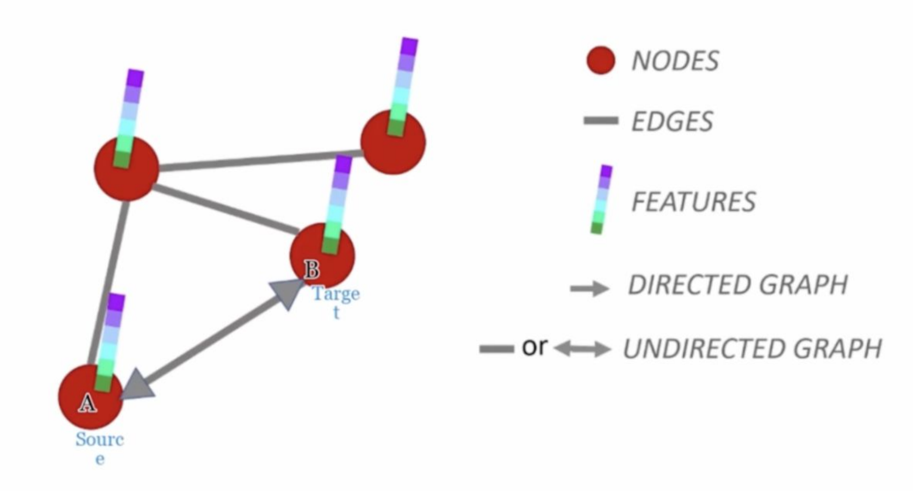


Features are additional node information

Graphs can be undirected, directed, or heterogeneous.

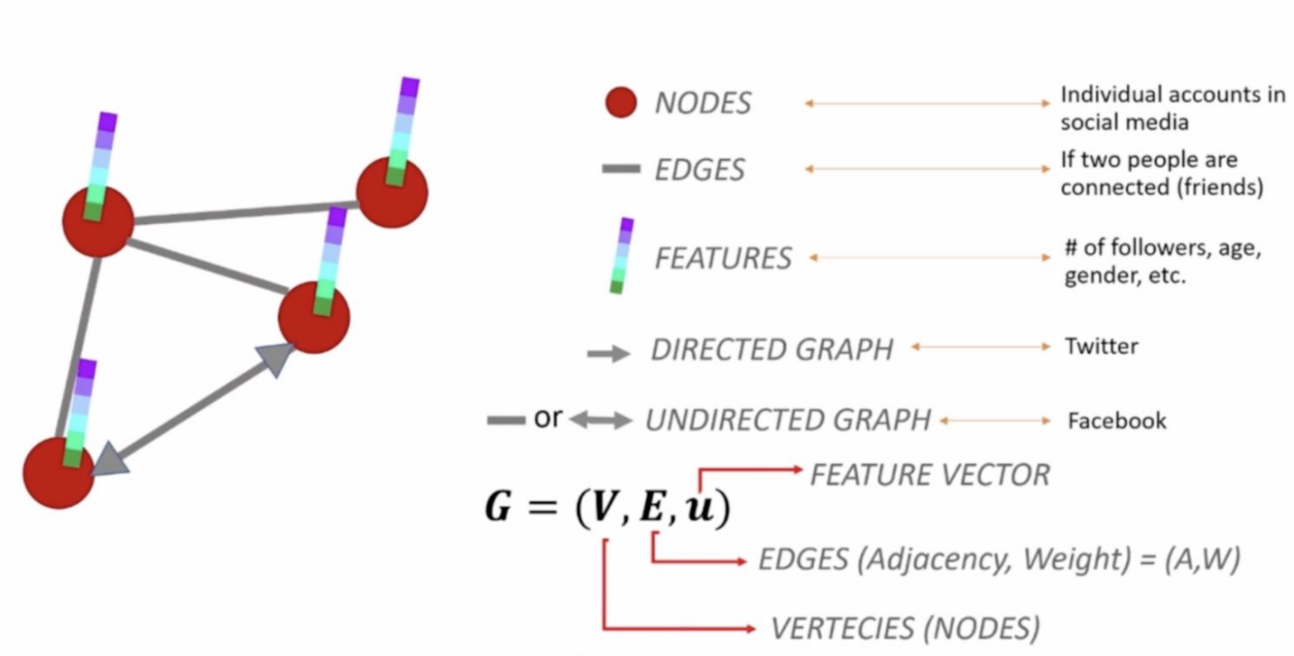
Examples of undirected, directed, and heterogeneous graphs?





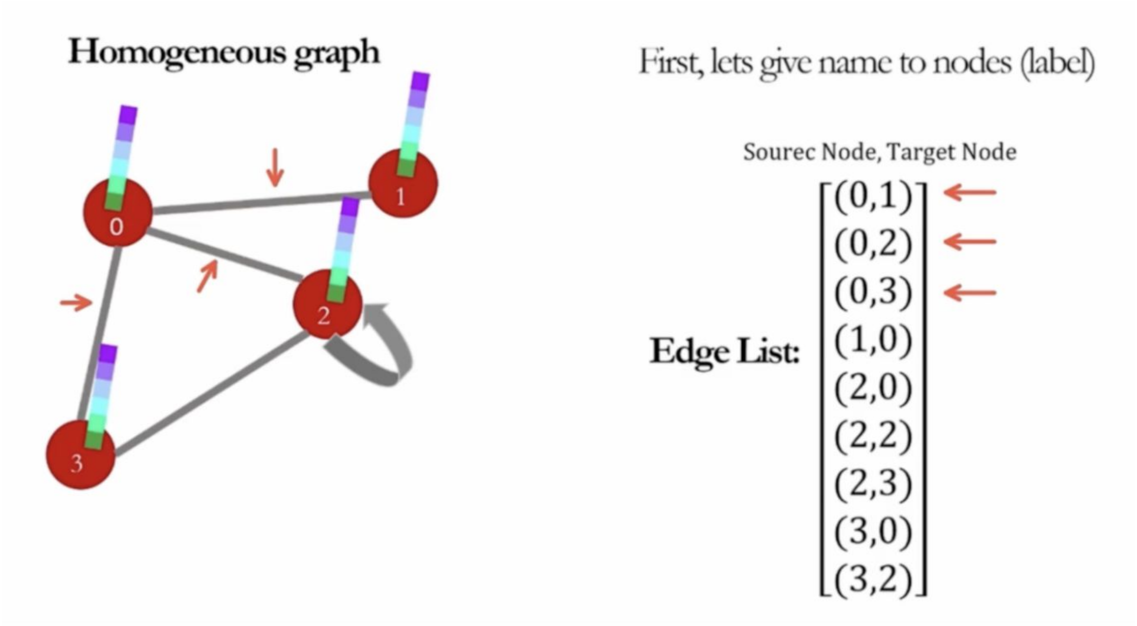
Features are additional node information

Graphs can be undirected, directed, or both.

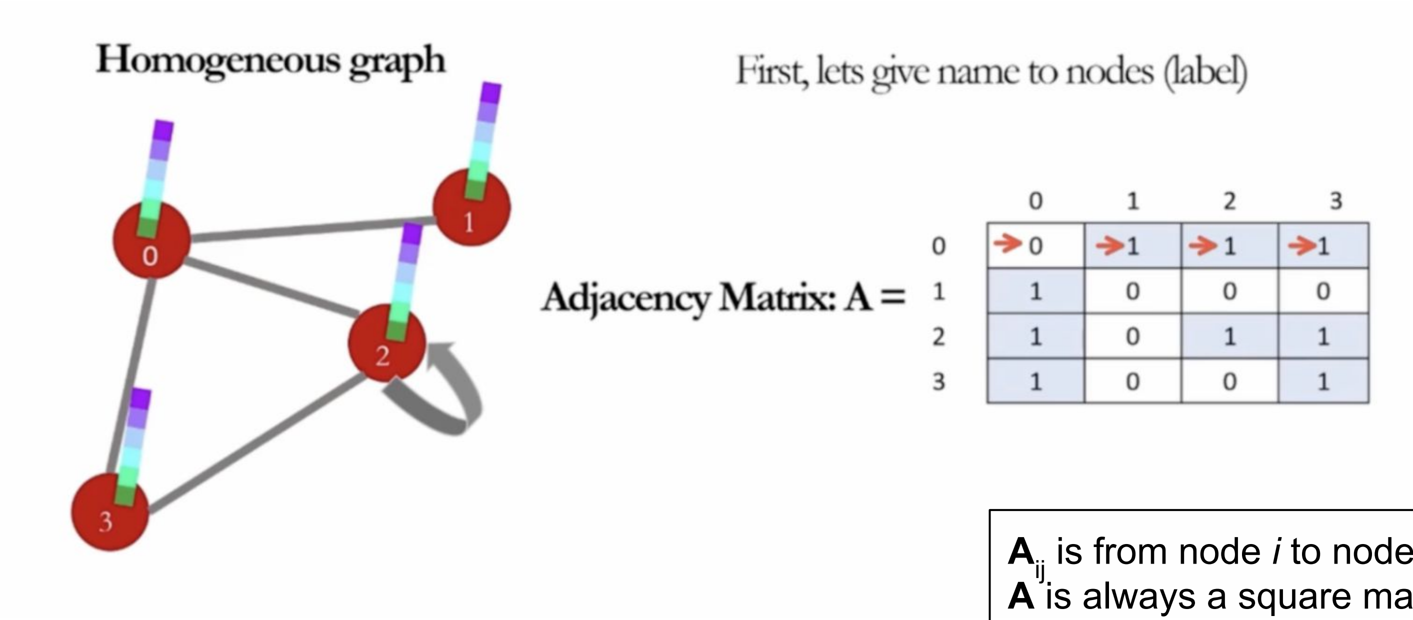


Edge list representation

Entry for each source/target node pair.



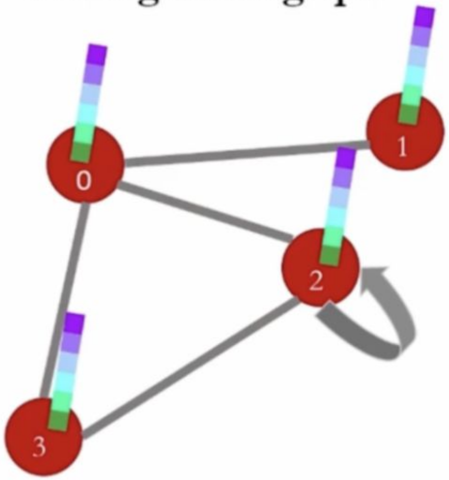
Adjacency matrix representation



A_{ij} is from node *i* to node *j*
A is always a square matrix
A is symmetric along diagonal in undirected graphs

Adjacency matrix representation

Homogeneous graph



First, lets give name to nodes (label)

Adjacency Matrix: =

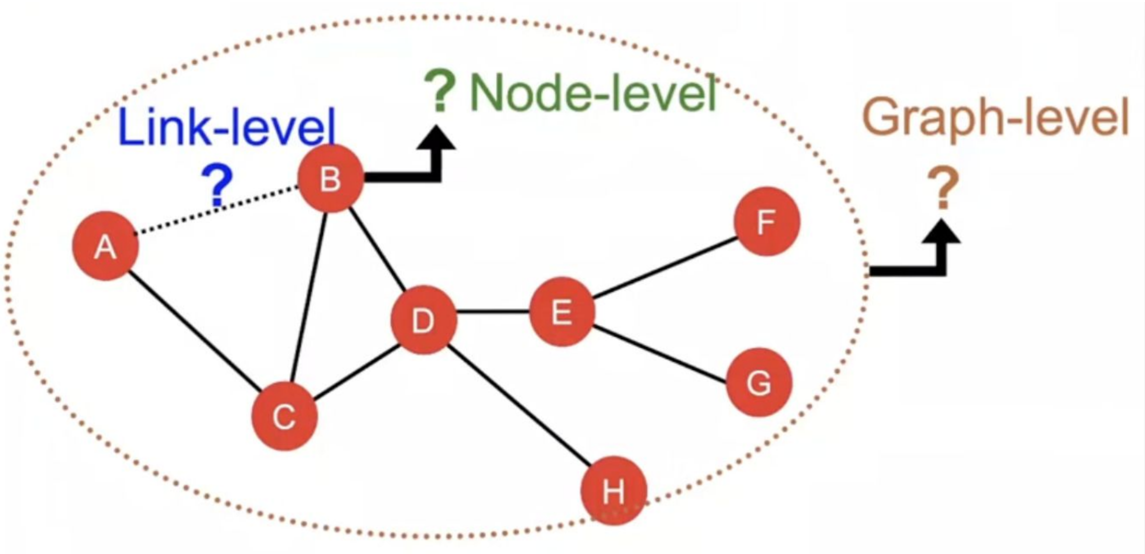
	0	1	2	3
0	0	2	1.5	4
1	5	0	0	0
2	1.5	0	1	1
3	12	0	0	1

Can use weights to define how powerful the connections are.

A_{ij} is from node *i* to node *j*
A is always a square matrix
We could have weights instead of 0/1 -> Weight matrix

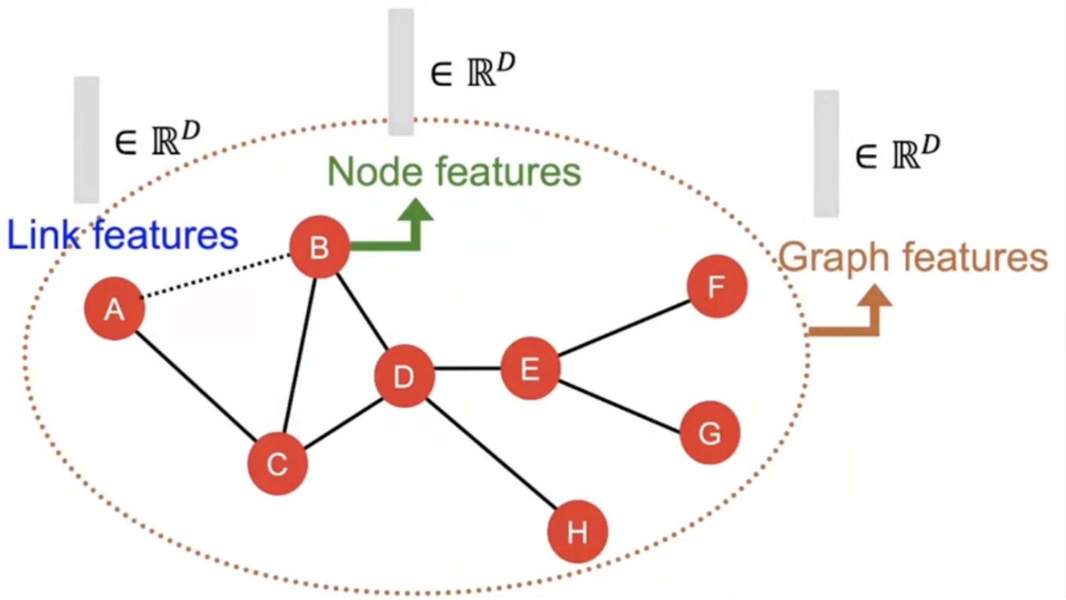
Machine Learning Tasks on Graphs

- Node-level prediction
- Link-level prediction
- Graph-level prediction



Traditional ML Pipeline

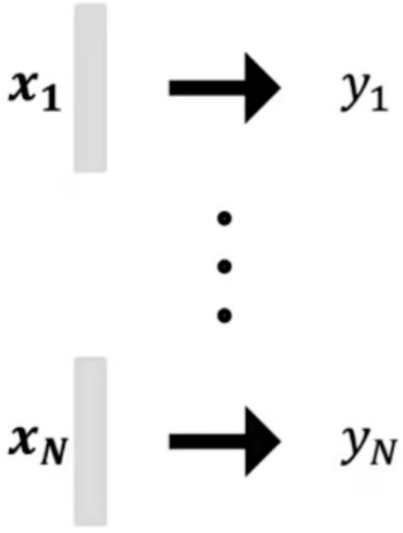
- Design features for nodes/links/graphs
- Obtain features for all training data



Traditional ML Pipeline

- Define features.
- Train an ML model. E.g., Random Forest, SVM, NN, etc.
- Apply the model. Given a node/link/graph, obtain it’s features and make a prediction.

Train



Apply

Questions you may want to ask about a network?

Some questions you may want to ask about a network

- Which parts are most influential?
- What is the best way to measure importance or influence in my particular context?
- How can I visualize my network or at least some simplification of it?
- How well does information flow through the network? How could it best be improved?
- Is there a center to my network, or a small number of centers? How should I define a center?
- My graph is huge. How do I randomly sample it, preserving structure?
- Can a simulation produce a network that usefully resembles my actual network?
- How is my network likely to evolve over time?
- What sort of communities or clusters are there in my network? How are they bridged?
- What would happen if a few connections were dropped at random? Or strategically removed by an adversary?

Feature Design

- Using effective features over graphs is the key to achieving good test performance.
- Traditional ML pipelines use *hand-designed features*.
- Review:
  - **Node-level prediction**
  - Link-level prediction
  - Graph-level prediction

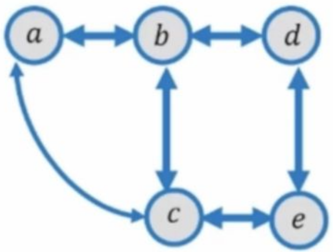
Node level features

Goal: characterize the structure and position of a node in a network.

- **Node degree**
- Node centrality
- Clustering coefficient
- Graphlets

Degree of graph

Degree of graph (**D**)

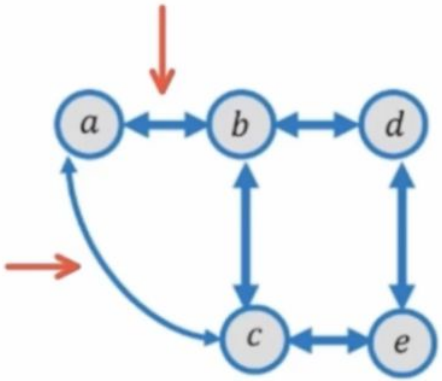


$$x = \begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix} \qquad A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

What is the degree of a node?

What is the degree of the graph?

Degree graph (D)



$$x = \begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix}$$

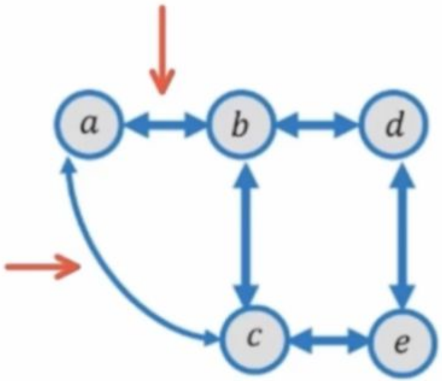
$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

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

Degree Matrix **D** is a diagonal matrix defining the number of connections per node.

Why is node degree important?

Degree graph (D)



$$x = \begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix}$$

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

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

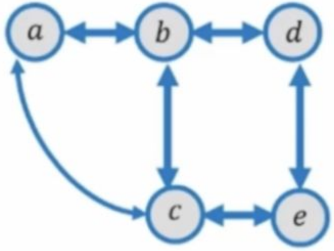
Degree Matrix **D** is a diagonal matrix defining the number of connections per node.

Why is node degree important? - Shows influence

Node degree

- Node degree counts the neighboring nodes without capturing their importance.
- Node centrality, $\boldsymbol{c_v}$ takes the node importance in a graph into account.
- Ways to measure importance:
  - **Eigenvector centrality**
  - Betweenness centrality
  - Closeness centrality
  - Many others

Graph Laplacian (L) - Eigenvector



$$x = \begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix} \quad A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \quad D = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

Laplacian matrix (**L**) is a $\mathrm{L} = \mathrm{D} - \mathrm{A}$ OR $\mathrm{L} = \mathrm{D} - \mathrm{W}$ in weighted matrix

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

Graph Laplacian

In Euclidean space, the Laplace operator is the ***divergence of the gradient of a function.***

$$\nabla f = div(grad(f))$$

The Laplacian has been shown to be very strong in characterizing important properties of a graph.

Captures gradient between connected nodes.

If smooth, well connected.

Can also show how influence can flow in a graph.

1. Graph functions (or) Graph signals

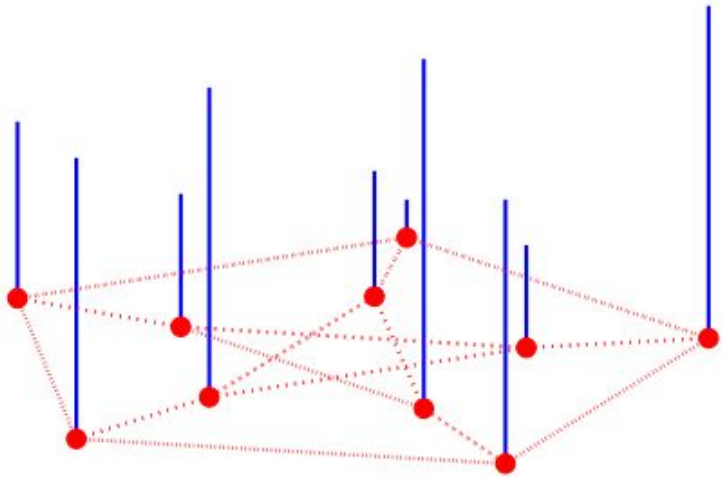
We can define a graph function to be a mapping from every vertex of a graph to a number.

The height of the blue lines indicates the magnitude of the function at that vertex.

For example, the Facebook graph with users as vertices and edges indicating friendships among them.

An integer valued graph function would be the number of friends each person has. Can call this the influence function on the graph.

The Emerging Field of Signal Processing on Graphs
https://arxiv.org/pdf/1211.0053.pdf



2. Graph functions (or) Graph signals

In Euclidean space, the gradient operator gives the **derivative** of the function along each **direction**.

In the discrete case, the analog for the derivative is the difference.

The analog for directions in a graph are edges.

Unlike in the regular Euclidean space, the discrete *'space'* of a graph allows for a different number of directions at every point (vertex).

Define the gradient of graph function to be an array of differences in the function value across each edge.

That is, gradient of the function along the edge $e=(u,v)$ is given by $grad(f)|e=f(u)-f(v)$.

The Emerging Field of Signal Processing on Graphs
https://arxiv.org/pdf/1211.0053.pdf

Measuring connectivity with graph Laplacian eigenvalues

Looking at the eigenvalues of the graph Laplacian (Spectral graph theory), can tell us how well a graph is connected, and also how well it's connected.

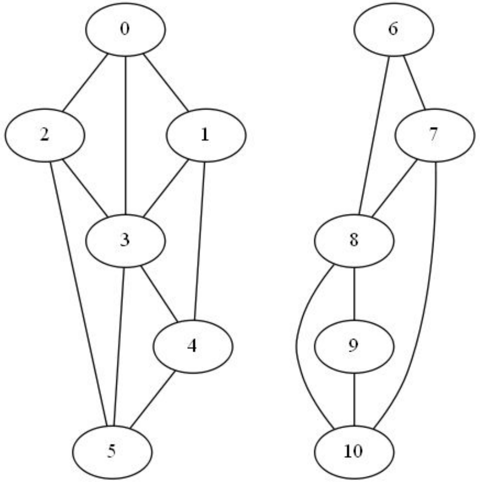
The smallest eigenvalue of *L*, λ1, is always 0.

The second smallest eigenvalue λ2 tells you about the connectivity of the graph. If the graph has two disconnected components, λ2 = 0. And if λ2 is **small**, this suggests the graph is **nearly** disconnected, that it has two components that are not very connected to each other.

In other words, the second eigenvalue gives you a sort of **continuous measure of how well a graph is connected**.

Graph Laplacian

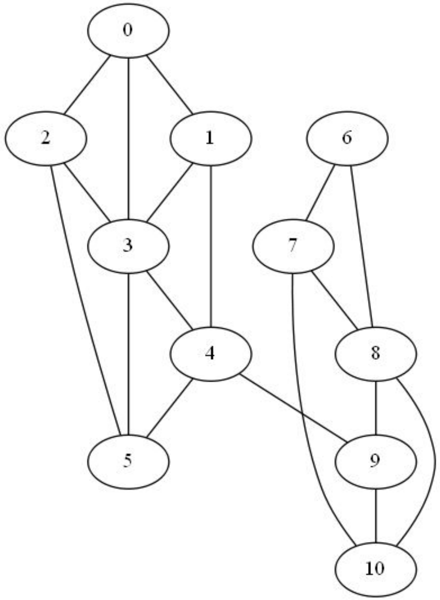
Start with a disconnected graph and see what happens to λ2 as we add edges connecting its components. The second smallest eigenvalue λ2 tells you about the connectivity of the graph. If the graph has two disconnected components, λ2 = 0.



Graph Laplacian

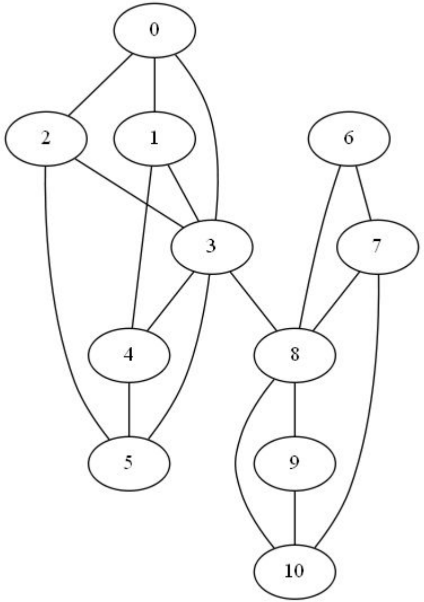
Next, we add an edge between nodes 4 and 9 to form a weak link between the two clusters.

In this graph the second eigenvalue λ2 jumps to 0.2144.



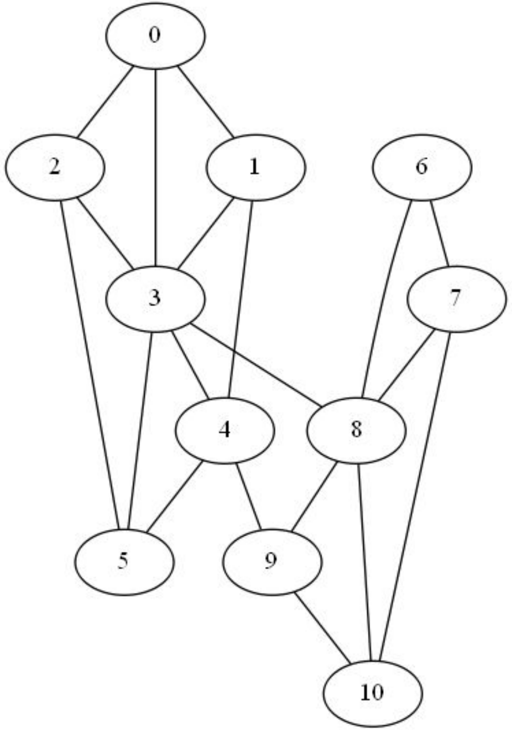
Graph Laplacian

If we connect nodes 3 and 8 instead of 4 and 8, we create a stronger link between the components since nodes 3 and 8 have more connections in their respective components. Now λ2 becomes 0.2788.



Graph Laplacian

Finally if we add both, connecting nodes 4 and 9 and nodes 3 and 8, λ2 increases to 0.4989.



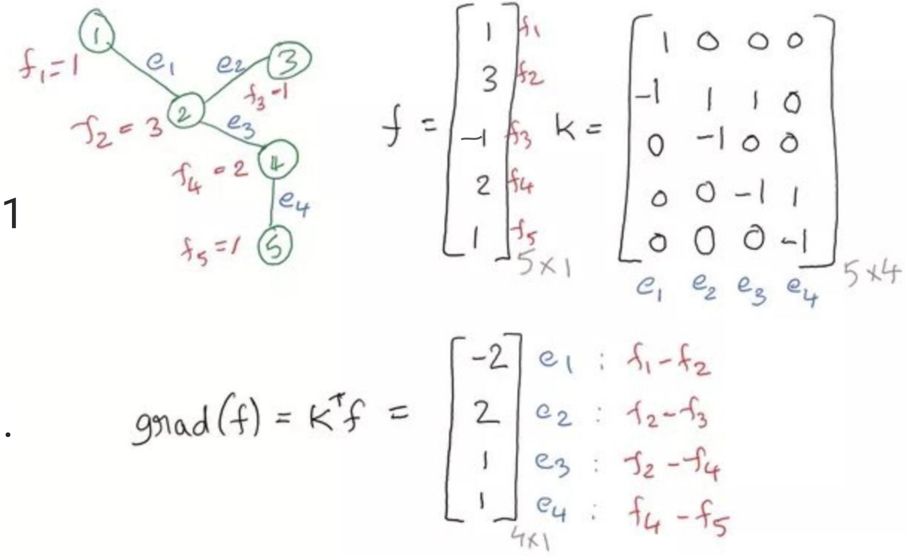
2. Graph functions (or) Graph signals - Incidence Matrix

A way to represent the gradient of a function along each edge is: $grad(f)=KTf$

K is an $V \times E$ matrix called the **incidence matrix**.

For every edge $e=(u,v)$ in the graph, assign $Ku,e=+1$ and $Kv,e=-1$.

Provides a way to keep track of the ‘incidence’ of various edges along various vertices in the graph.

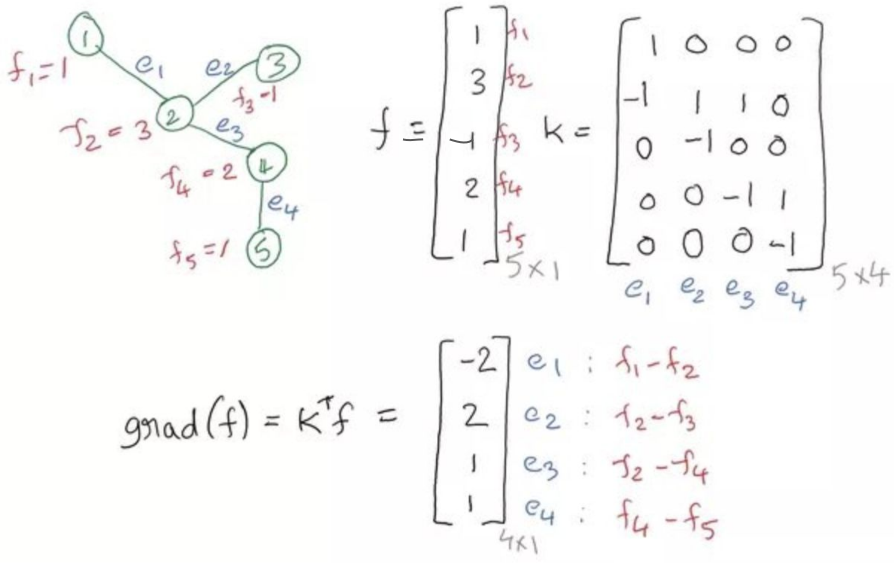


2. Graph functions (or) Graph signals - Incidence Matrix

Think of the gradient as a different function over the edges of the graph instead of the vertices.

For our example of the influence function over Facebook graph:

- gradient of the influence function gives an array of differences in influences among the users.



3. The final result: Graph Laplacian

Define the Laplacian of a graph function f as: $\Delta f = div(grad(f)) = KKTf$.

KKT is the Laplacian matrix.

The Laplacian matrix for any graph can be factored as $L = KKT$.

$$KK^T = \underset{5\times 4}{\begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & -1 \end{bmatrix}} \underset{4\times 5}{\begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{bmatrix}} = \underset{5\times 5}{\begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 3 & -1 & -1 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 2 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}} = L$$

The green circled elements along the diagonals, the are the degree of the matrix.

$L = D - W$ where D is the degree matrix and W is the weight matrix.

What is the Laplacian telling you?

For continuous spaces, the Laplacian is the second derivative of the function.

So it measures how “smooth” the function is over its domain.

For graph functions the Laplacian of a graph function determines how “smooth” the graph function is.

The first eigenvector of the graph Laplacian is the smoothest function you can find on the graph.

The second eigenvector is the next smoothest function of all graph functions that are orthogonal to the first one and so on.

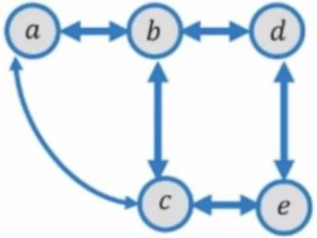
The next eigenvectors are just the higher ***frequency*** **modes** of the signal.

Normalized Graph

The bar above matrix means normalized with respect to the number of connections in the graph.

Can use Laplacian or other methods.

Normalized Graph



We can decide to show the relation between of the nodes, with any of the following matrices:

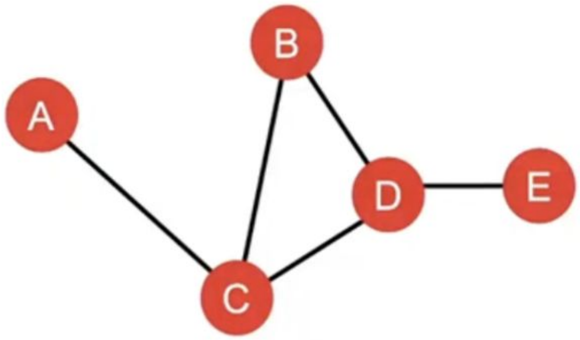
$$\mathbf{A}\ ,\boldsymbol{L}\ ,\bar{\boldsymbol{A}}\ ,\bar{\boldsymbol{L}}$$

Betweenness centrality

A node is important if it lies on many shortest paths between other nodes.

$$c_v = \sum_{s \neq v \neq t} \frac{\#(\text{shortest paths betwen } s \text{ and } t \text{ that contain } v)}{\#(\text{shortest paths between } s \text{ and } t)}$$

Example:



$$c_A = c_B = c_E = 0$$

$$c_C = 3$$

(A-$\underline{\text{C}}$-B, A-$\underline{\text{C}}$-D, A-$\underline{\text{C}}$-D-E)

$$c_D = 3$$

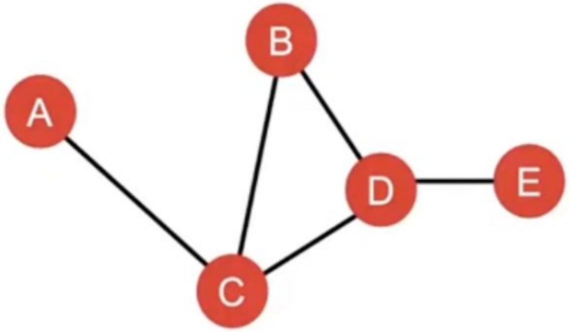
(A-C-$\underline{\text{D}}$-E, B-$\underline{\text{D}}$-E, C-$\underline{\text{D}}$-E)

Closeness centrality

A node is important if it has small shortest path lengths to all other nodes.

$$c_v = \frac{1}{\sum_{u \neq v} \text{shortest path length between } u \text{ and } v}$$

- **Example:**



$c_A = 1/(2+1+2+3) = 1/8$
(A-C-B, A-C, A-C-D, A-C-D-E)

$c_D = 1/(2+1+1+1) = 1/5$
(D-C-A, D-B, D-C, D-E)

Clustering Coefficient

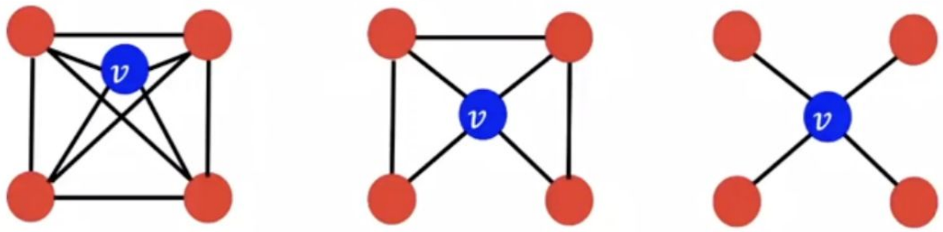
Measure's how connected ***v***'s neighboring nodes are. Count triangles.

The local clustering coefficient of a vertex (node) in a graph quantifies how close its neighbours are to being a clique (complete graph).

$$e_v = \frac{\#(\text{edges among neighboring nodes})}{\boxed{\binom{k_v}{2}}} \in [0,1]$$

#(node pairs among k_v neighboring nodes)

- **Examples:**



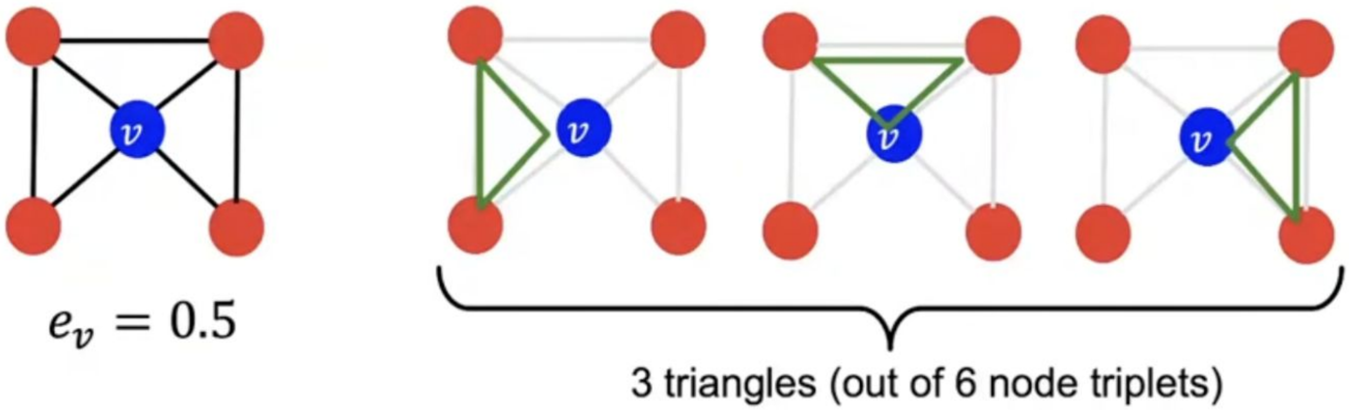
$e_v = 1$ $e_v = 0.5$ 3/6 $e_v = 0$

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Graphlets

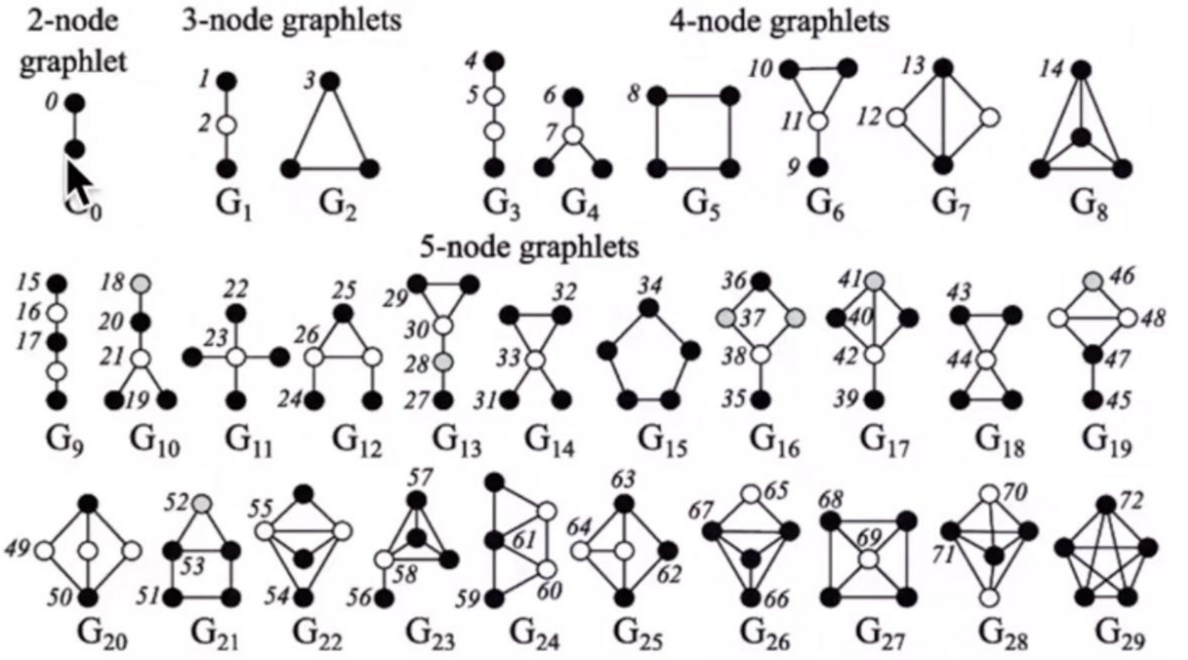
Clustering coefficient counts the number of triangles

Generalize by counting pre-specified subgraphs



Graphlets

Non-isomorphic - different number of nodes or edges.



If you cannot explain something in simple terms, you don't understand it.
– Richard Feynman