Week 10: Introduction to Network Analytics

Data Science Certificate Program

Ryerson University

Bora Caglayan

02 Apr, 2015

Announcements

- Homework 3 is up.
- Homework 2 questions?
- The final will be in 2 weeks.
    - We will do a brief review for the final next week.

Outline

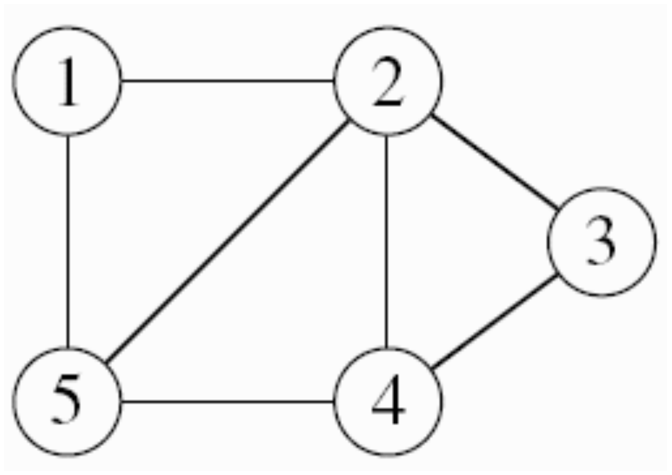
- Basic Network Notation
- Network Representation Methods
- Basic Network Measures
    - Distance
    - Centrality
    - Clustering Networks

Note: In math a similar notation is used for the graphs.

Basic Network Notation

Networks are constructed from nodes (N) and edges (E).

$$< N, E >$$



Nodes: {1,2,3,4,5}

Edges: {(1,5),(1,2),(2,5),
(2,3),(2,4), (3,4), (4,5)}

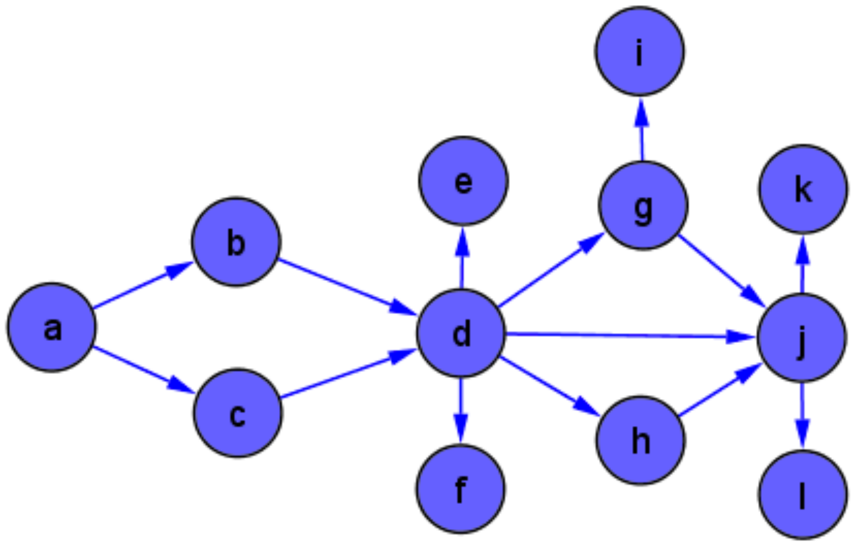
This is a network with 5 nodes and 7 edges.

Basic Network Notation

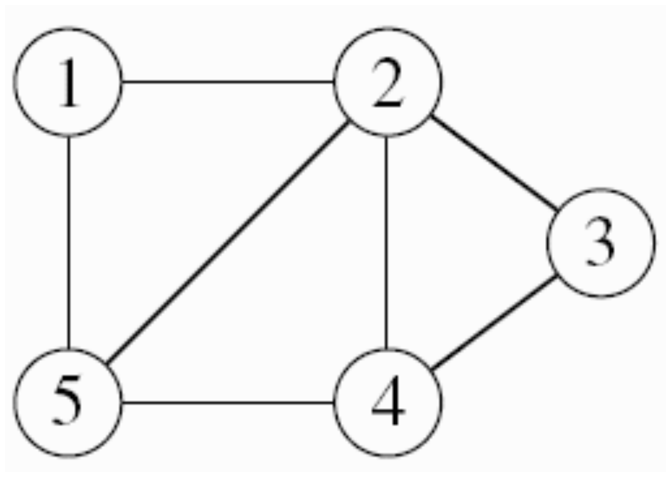
- **Directed network:** Edges have direction.
- **Undirected network:** Edges do not have direction.
- **Non-weighted network:** Edges do not have weight.
- **Weighted network:** Edges have weight.

Basic Network Notation

Directed Network Example

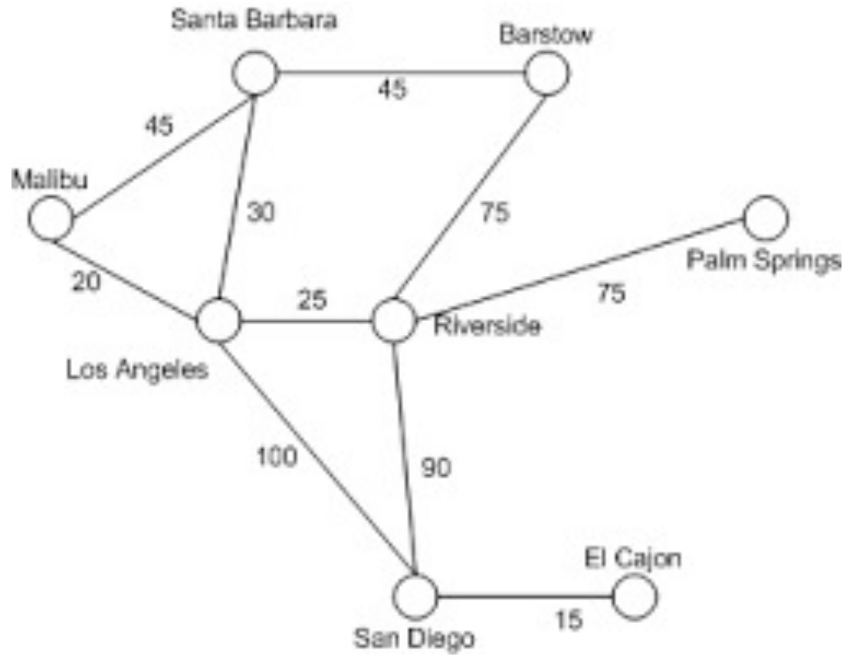


Undirected Network Example



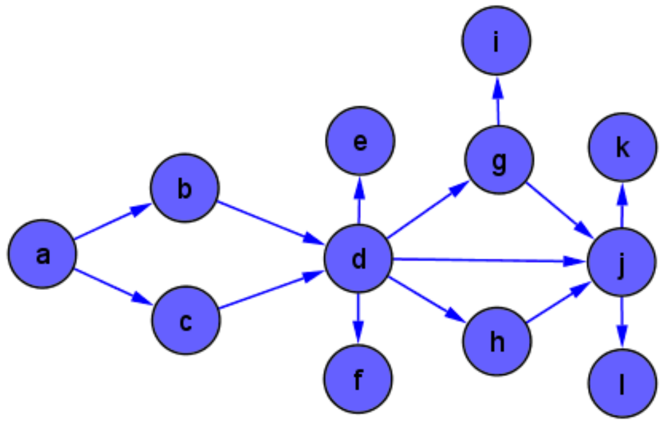
Basic Network Notation

A weighted network example:



Basic Network Notation

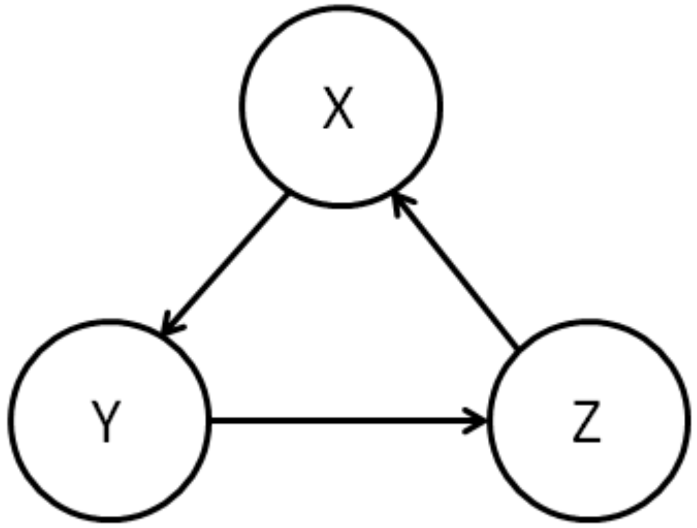
- **Path**
  - A connection between two nodes in finite number of steps.
- **Shortest Path**
  - Path with least cost between two nodes.
  - There may be 2+ shortest paths.



Path (a -> l): [a,b,d,g,j,l]

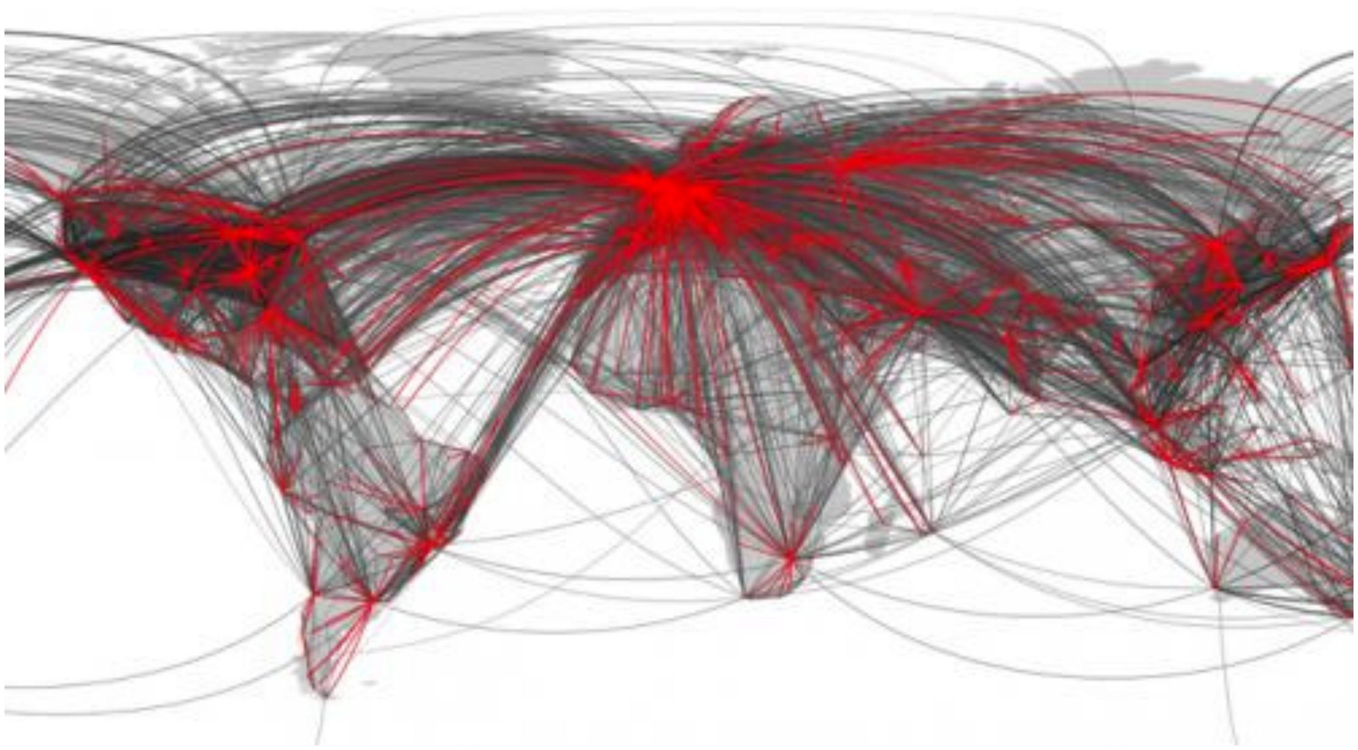
Shortest Path (a -> l): [a,b,d,j,l]

- **Loop**
  - A path that begins and terminates on the starting node.
  - Directed networks without a loop are called directed acyclic.



Some Complex Network Examples

- Network of Air Traffic



Some Complex Network Examples

- Network of Facebook Friendships

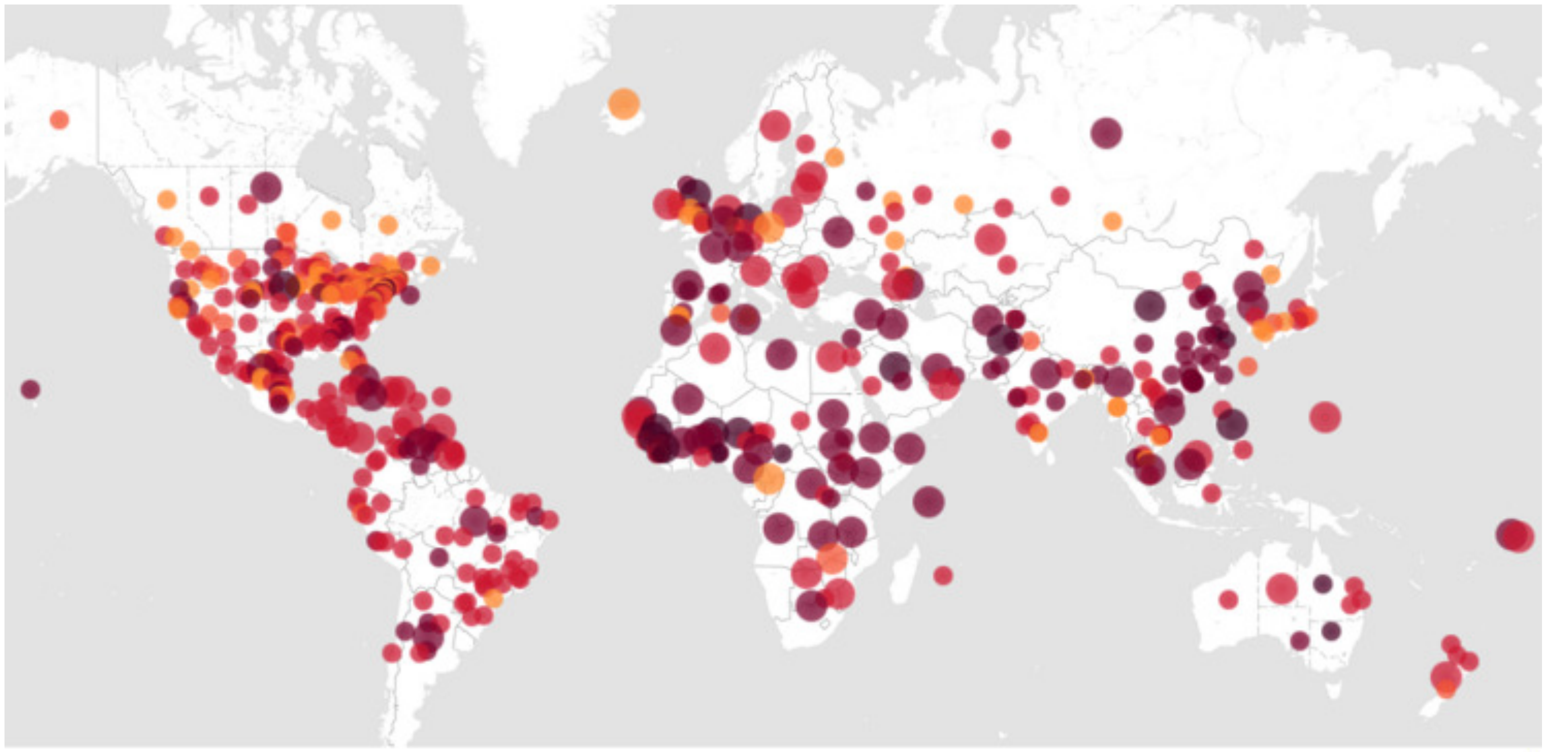
facebook
December 2010

Common Problems in Network Analytics

- Identify influential nodes (trendsetters, important web pages etc.)
- Recommendations of new edges (new friends)?
- Predict important events based on temporal trends? (viral news, epidemics etc.)
- Predict the future state of the network (network growth prediction)…

An Application Example

Recently, researchers used online communication to predict disease outbreaks.

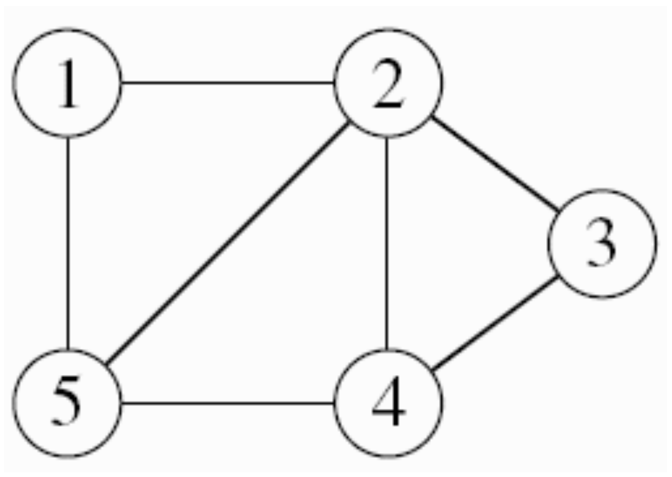


Network Representation

1- Edge list

Network is represented as a list of edges.

```
1,2
1,5
2,3
2,4
2,5
3,4
4,5
```

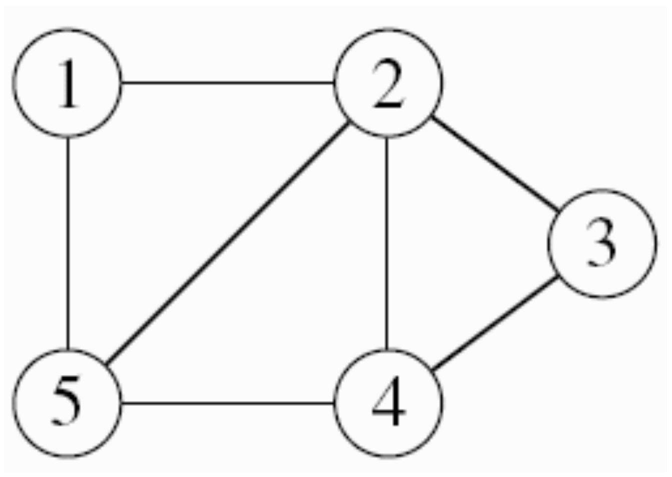


Network Representation

2- Adjacency matrix

Network is represented as an NxN binary matrix.

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

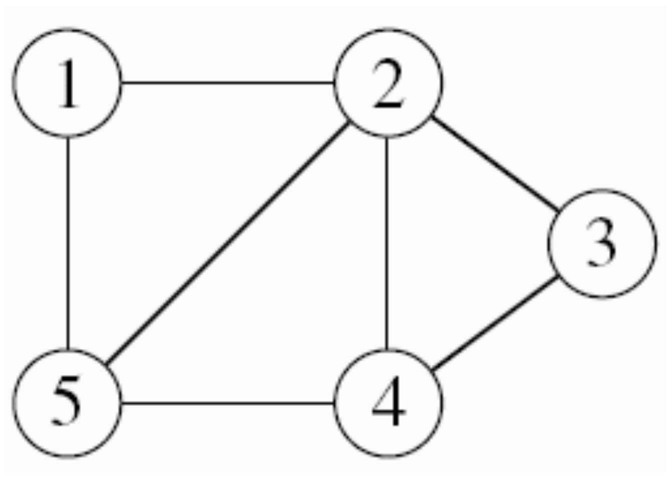


Network Representation

3- Adjacency list

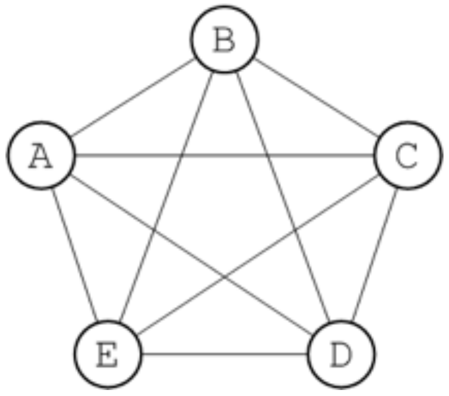
Network is represented by the nodes and their neighbours.

```
1 2,5
2 1,3,4,5
3 2,4
4 2,5
5 1,2,4
```



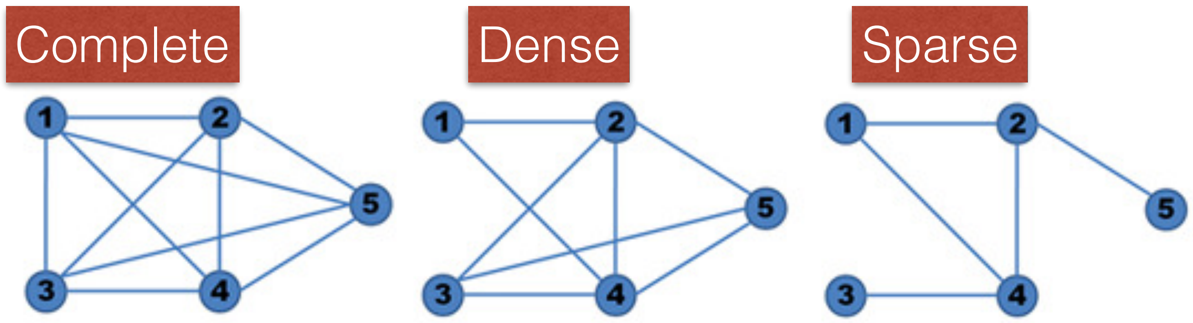
Interlude Question

- A fully connected network is a network in which each of the nodes is connected to each other.
- What is the number of edges in a complete undirected network with N nodes?



Network Representation

Dense network: The network has most of the possible edges
Sparse network: The network has only a small proportion of the possible edges. In this case the adjacency matrix would have a lot of 0s (waste of space).



Density estimation formula for an undirected network

$$\rho = \frac{E}{N*(N-1)/2} = \frac{2E}{N*(N-1)}$$

Network Representation

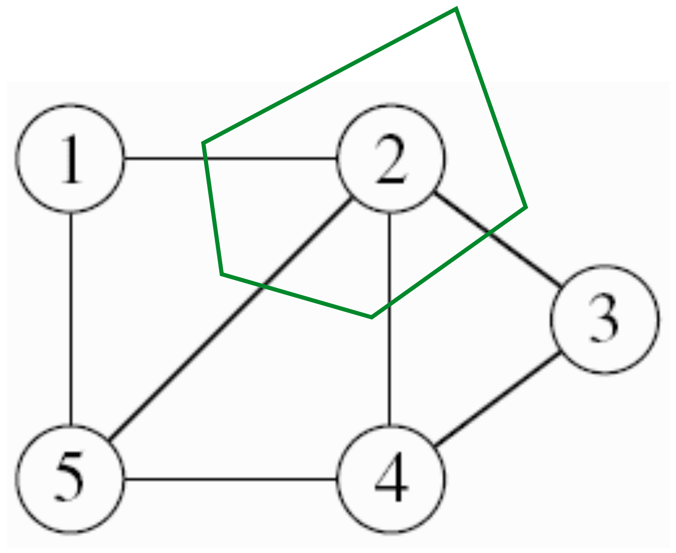
Comparison

Edge List	1. Finding the degree of a node is O(N) 2. Memory complexity is lower than adjacency matrix. 3. Usually used as an input and transformed by graph/network analytics libraries.
Adjecancy Matrix	1. Memory complexity is O(N^2). Not applicable for large networks. 2. nth order neighbours can be found by matrix multiplications. 3. Time complexity to access a node is low
Adjecancy List	1. Good for conserving memory for sparse networks. 2. Finding degrees of a node is O(1).

Basic Network Measures

Node degree: Number of neighbours of a node in an *undirected network*.

$$degree(node(2))) = 4$$

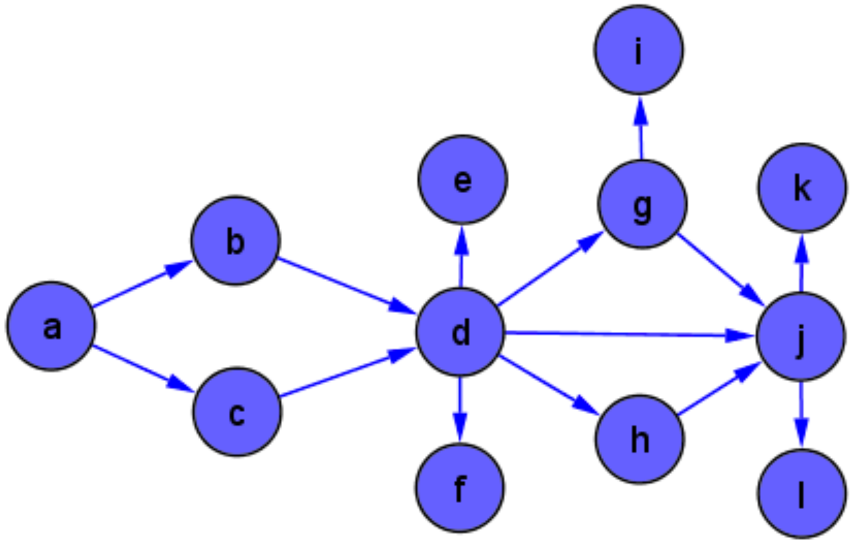


Basic Network Measures

Node in-degree: Number of incoming connections to a node (*for example a popular website*)

Node out-degree: Number of outgoing connections from a node in a *directed network.* (*for example a list of websites*)

$$indegree(node(d))) = 2$$
$$outdegree(node(d))) = 5$$

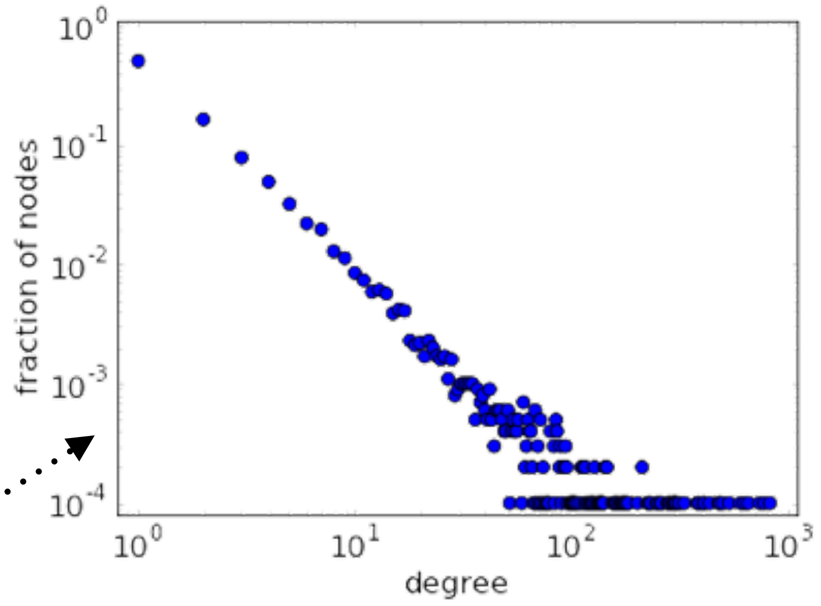


Basic Network Measures

Degree Distribution

Distribution of node degrees is an important network statistic.

In real networks usually the degree distribution obeys the power law (linear distribution on log-log scale).



Nodes with unusually high degree is sometimes called **hubs**.

Basic Network Measures

Centrality

- The importance of the nodes in a network.
- Centrality is estimated based on difference criteria.

1. Degree Centrality

- Based on the degree of a node for undirected network (or the in-degree in a directed network)

- Easy to calculate but may not be a good estimation for certain networks.

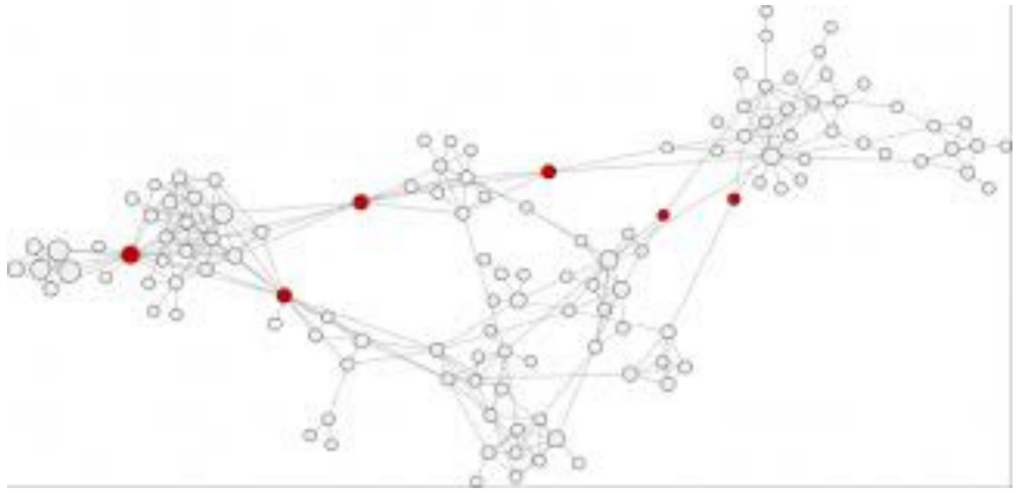
Basic Network Measures

2. Betweenness Centrality

- Shortest paths that include a given node is divided by the shortest paths between any nodes to find betweenness centrality.

- Nodes that act like bridges in the network have high betweenness centrality.
- Betweenness centrality is harder to compute but it is more robust.

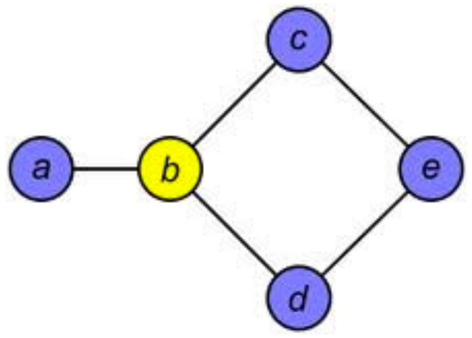
$$betw(node(x)) = \frac{\text{Number of shortest paths in the network that include node x}}{\text{Number of shortest paths in the network}}$$



Basic Network Measures

2. Betweenness Centrality

An Example



Simple Paths

{a,b}
{a,b,c}
{a,b,d}
{a,b,c,e}
{b,c}
{b,d}
{b,c,e}
{c,e}
{d,e}

Betweenness Centralities

$$betw(a) = \frac{4}{9}$$

$$betw(b) = \frac{9}{9}$$

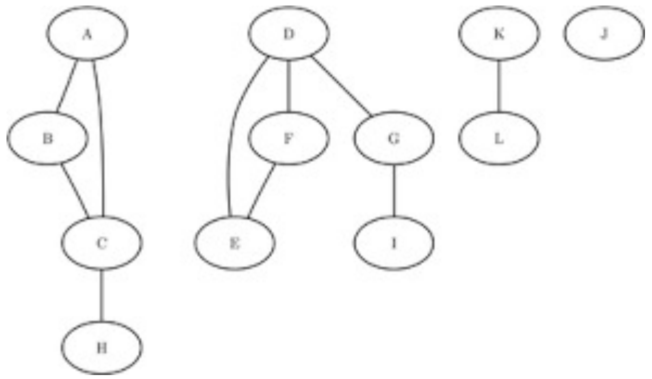
$$betw(c) = \frac{5}{9}$$

$$betw(d) = \frac{3}{9}$$

$$betw(e) = \frac{4}{9}$$

Basic Network Measures

Connected Components
Largest set of nodes where every node pair connected.



Connection is more robust if the connections between nodes cannot be prevented by removing a few edges.

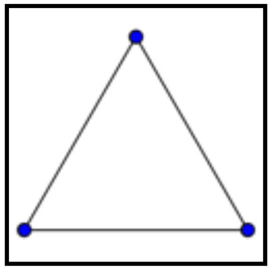
Example: Air traffic network. If you remove a random airport the system would remain connected.

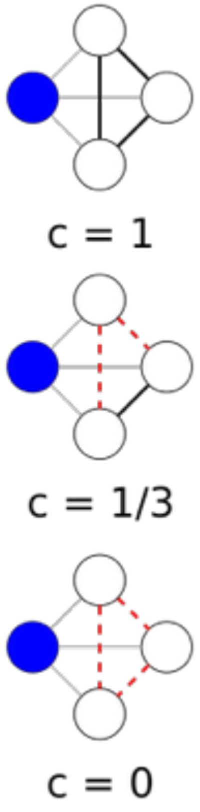
Example network has 3 components.

Basic Network Measures

Triangles

Triangles are three nodes that are connected to each other.





Clustering coefficient of a network shows the strength/robustness of the connections between the components.

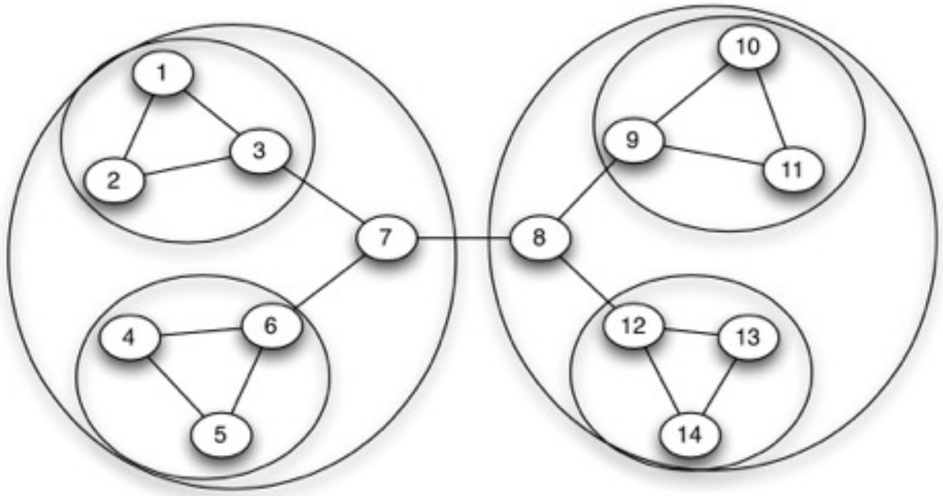
$$C = \frac{\text{number of closed triplets}}{\text{number of connected triplets of vertices}}.$$

Clustering Networks

Approach 1: Top-down

In this approach edges are removed iteratively until desired number of components remain.

We can start from the edges with highest betweenness values.

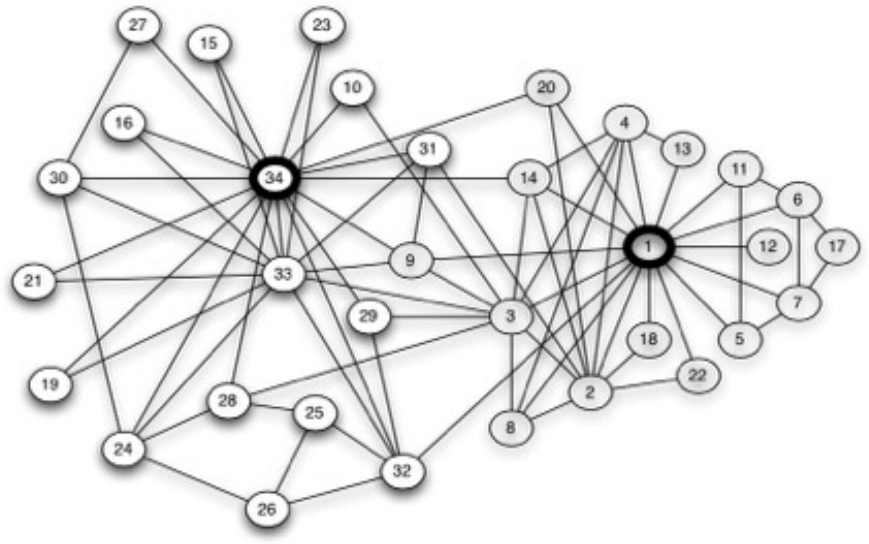


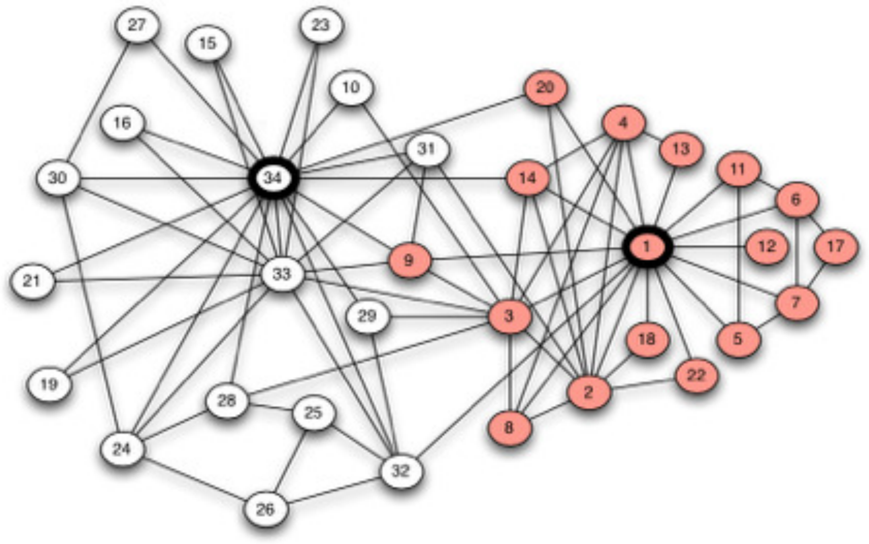
Clustering Networks

Approach 1: Top-down

Minimum cut is the minimum number of edge removals necessary to create 2 disconnected components.

We can find the minimum-cut that creates two components with similar sizes with the top-down approach.

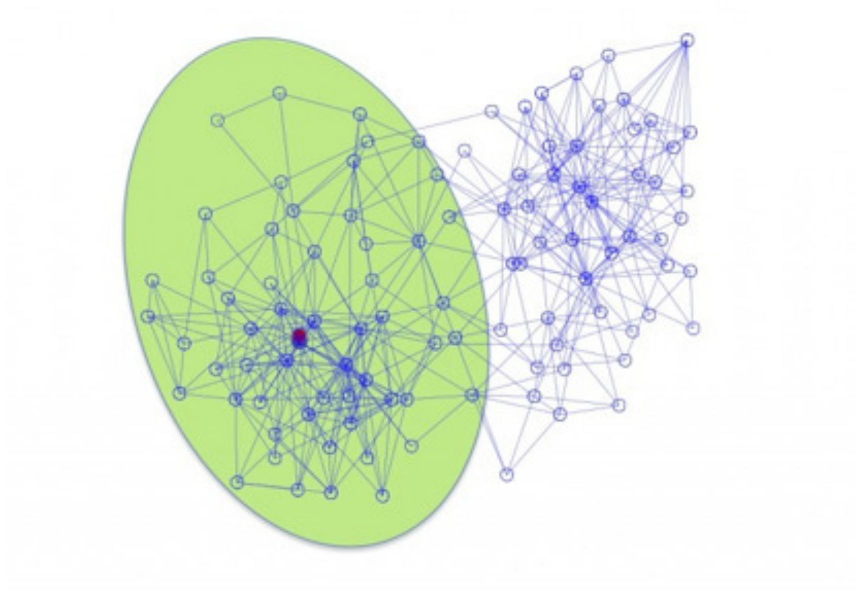




Clustering Networks

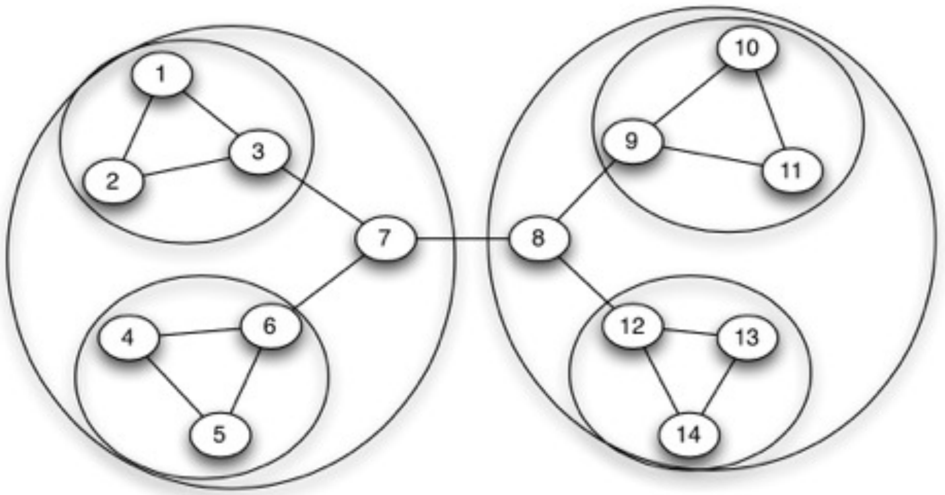
Approach 2: Bottom-up

Start from random nodes and randomly add new nodes to the cluster



Clustering Networks

Number of within-cluster edges divided should be much higher than the number of inter-cluster edges.

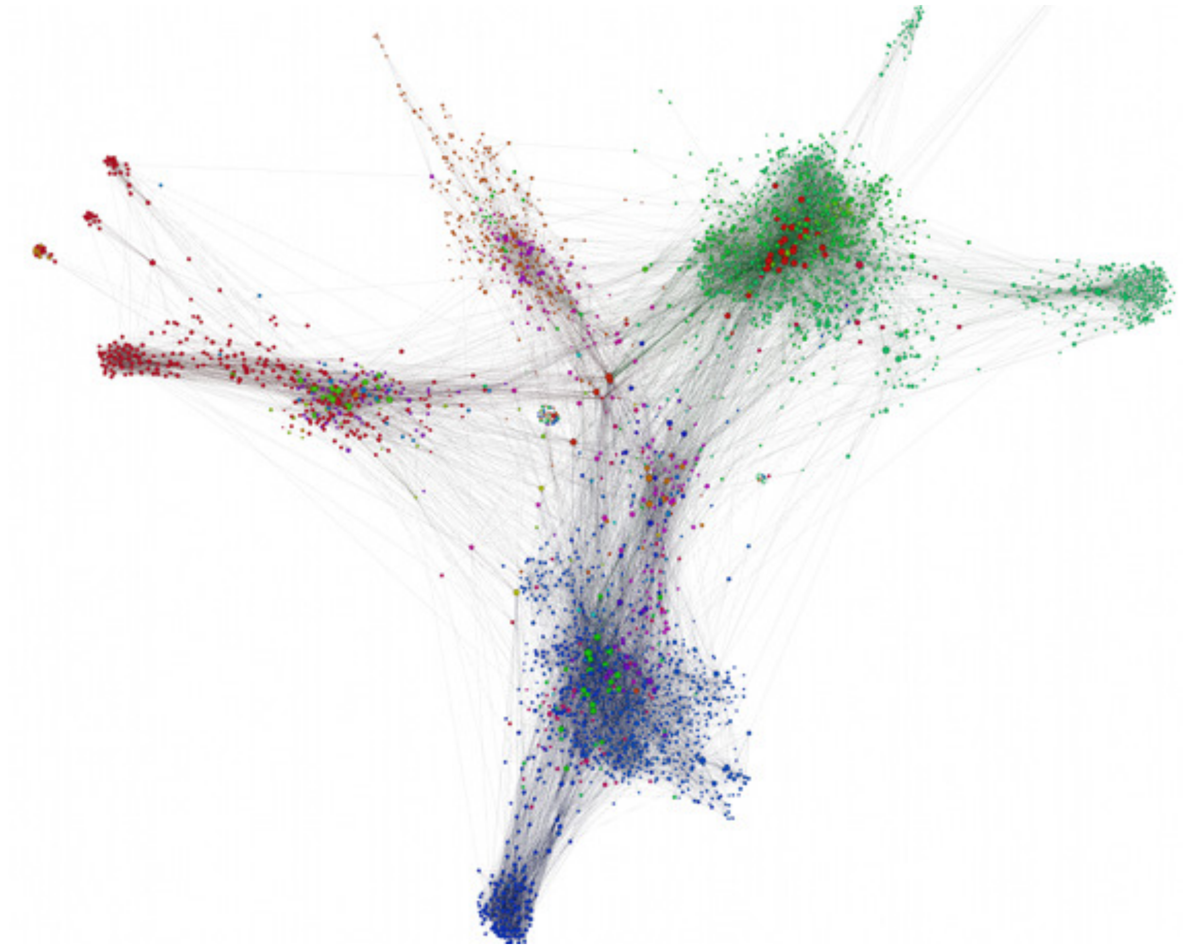


Visualization of Large Networks

Basic visualization can be done with R (we will see in the application part).

For large networks I recommend Gephi for visualization.

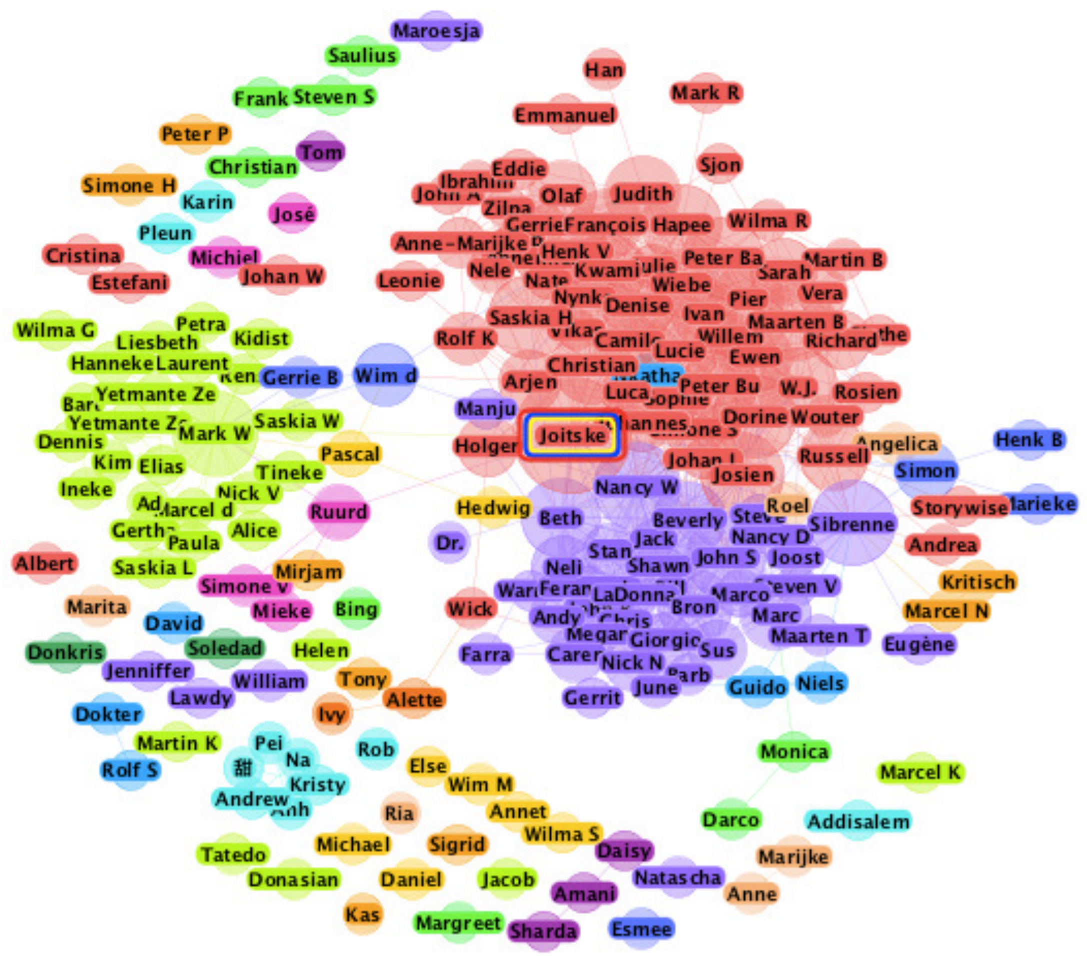
You can play with graph/ network visualization algorithms for better results.



Processing Requirements

- Some network algorithms are NP complete. (Examples: travelling salesman problem, map colouring problem…)
- If you have such a problem a *greedy algorithm* might be useful.
- Usually network datasets are sparse and edge list and adjacency lists are more efficient for network storage compared to adjacency matrices.
- Hadoop may not be good for implementing some network algorithms. There are distributed processing frameworks optimized for network analytics *(see graphlab and giraph)*.

Converting A Network to a Dataset For Machine Learning Problems



For the classification problem each node may be converted to an instance.

Each instance may have features based on its characteristics and based on its place in the network. We can use the network as an input for machine learning models by merging these features.

For example, *Pascal might have his own features (such as age, height etc.). In addition we can use his node's features in the network (such as degree, betweenness) for the classification problem.*

References

- http://gephi.github.io/ : Graph visualization software.
- https://publichealthwatch.wordpress.com/2014/08/10/how-a-computer-algorithm-predicted-west-africas-ebola-outbreak-before-it-was-announced/ : Graph analytics for epidemics prediction.
- https://www.cs.cornell.edu/home/kleinber/networks-book/ Great introduction book. Freely available online.
- http://www.barabasilab.com/ : Some influential recent research in this area.
- http://igraph.org/ : Most popular graph analytics library in R.

Week 10 Application Part: Graph Analytics with R

Preparation

```r
library("igraph") # graph analytics library in R
```

Creating Graphs

From an edge list

```
g <- graph( c(1,2, 1,5, 2,3, 2,5, 2,4,
              3,4, 4,5), n=5,
            directed=F)
```

Creating Graphs

Create complete or null graphs

```r
full_g <- graph.full(5)
empty_g <- graph.empty()
```

Show Nodes and Edges

```r
V(g)
```

```
Vertex sequence:
[1] 1 2 3 4 5
```

```r
V(full_g)
```

```
Vertex sequence:
[1] 1 2 3 4 5
```

```r
V(empty_g)
```

```
Vertex sequence:
numeric(0)
```

Show Nodes and Edges

```r
E(g)
```

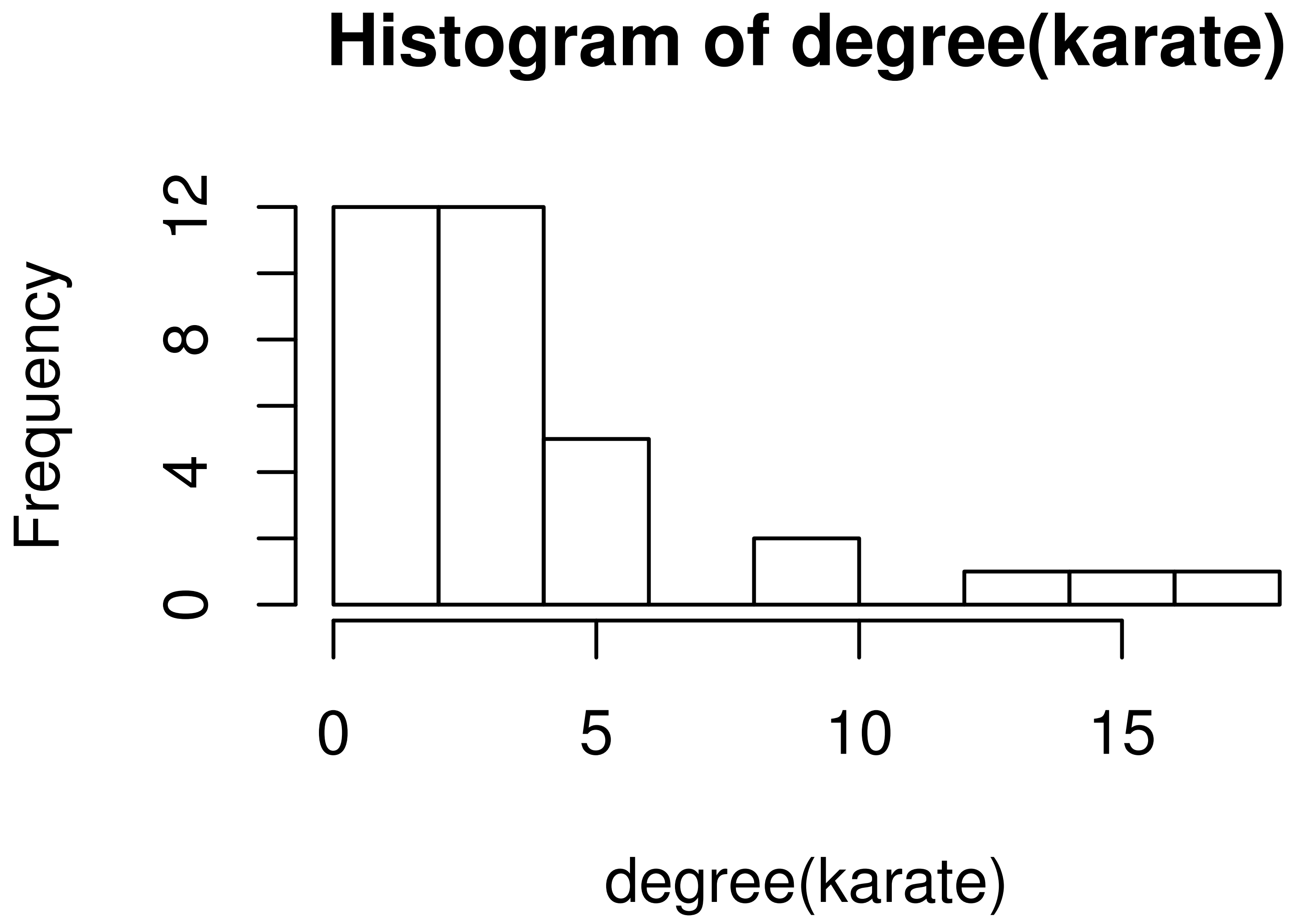
```
Edge sequence:

[1] 2 -- 1
[2] 5 -- 1
[3] 3 -- 2
[4] 5 -- 2
[5] 4 -- 2
[6] 4 -- 3
[7] 5 -- 4
```

Read Graph Files

```
karate <- read.graph("http://cneurocvs.rmki.kfki.hu/igraph/
                                  format="pajek")
```

Show Degree Distribution

```r
hist(degree(karate))
```



Check Betweenness Values

```r
betweenness(g)
```

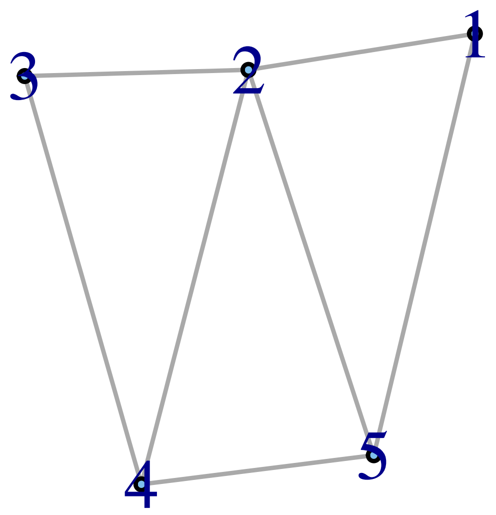
```
[1] 0.0 2.0 0.0 0.5 0.5
```

```r
edge.betweenness(g)
```

```
[1] 2.5 1.5 2.5 1.5 1.5 1.5 2.0
```

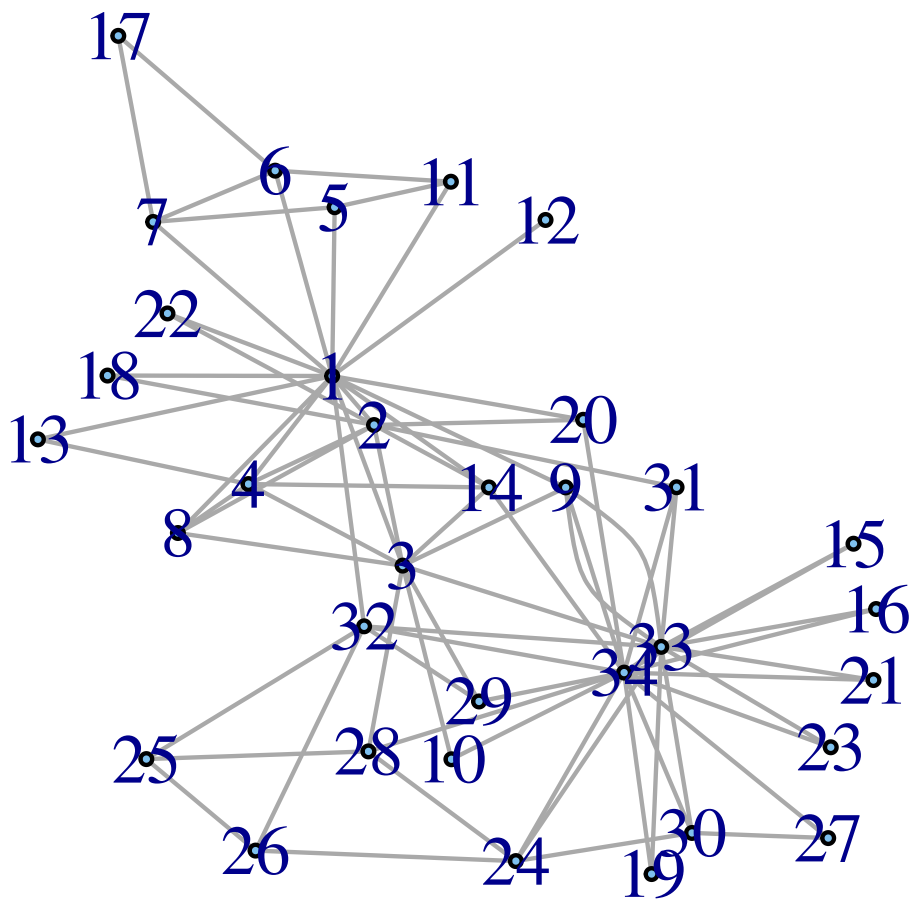
Visualization of Graphs

```
plot(g, vertex.label=V(g), vertex.size=0)
```



Visualization of Graphs

```r
plot(karate, vertex.label=V(karate), vertex.size=0)
```



Cluster a Graph

Check the connected components

```
is.connected(karate)
```

```
[1] TRUE
```

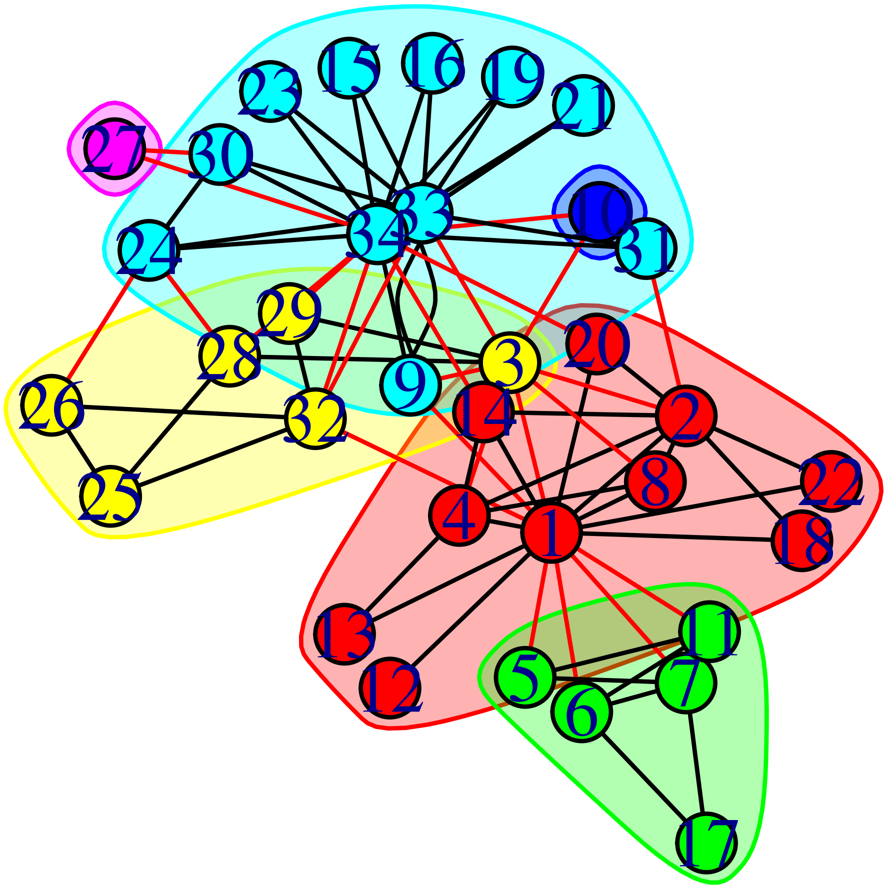
```
clusters(g)
```

```
$membership
[1] 1 1 1 1 1

$csize
[1] 5

$no
[1] 1
```

Cluster a Graph

Cluster karate club network

```
wc <- edge.betweenness.community(karate)
plot(wc, karate)
```



Lab Preparation

```
sample_graph <- barabasi.game(100, directed=F)
```

Lab Problems:

- Create a trivial graph with 3 nodes A,B and C. There is an edge between A and B. There is an edge between B and C. Plot this graph in the end.
- Show the degree distribution and plot the histogram of the degrees.
- Show the correlation between betweenness and degree for each node.
- Partition the sample graph and show its components.