Final exam review

Community Analysis

Kai Shu

Spring 2022

Social Community

[real-world] community

A group of individuals with common *economic*, *social*, or *political* interests or characteristics, often living in *relative proximity*.

Social Media Communities

- **Formation:**
  - When like-minded users on social media form a link and start interacting with each other
- **More Formal Formation**:
  1. A set of at least two nodes sharing some interest, and
  2. Interactions with respect to that interest.
- Social Media Communities
  - **Explicit**: formed by user subscriptions
  - **Implicit**: implicitly formed by social interactions
    - **Example:** individuals calling Canada from the United States
    - Phone operator considers them one community for promotional offers
- Other community names: *group*, *cluster*, *cohesive subgroup*, or *module*

What is Community Analysis?

- **Community detection**
  - Discovering implicit communities
- **Community evolution**
  - Studying temporal evolution of communities
- **Community evaluation**
  - Evaluating detected communities

Community Detection

Community Detection vs. Clustering

Clustering

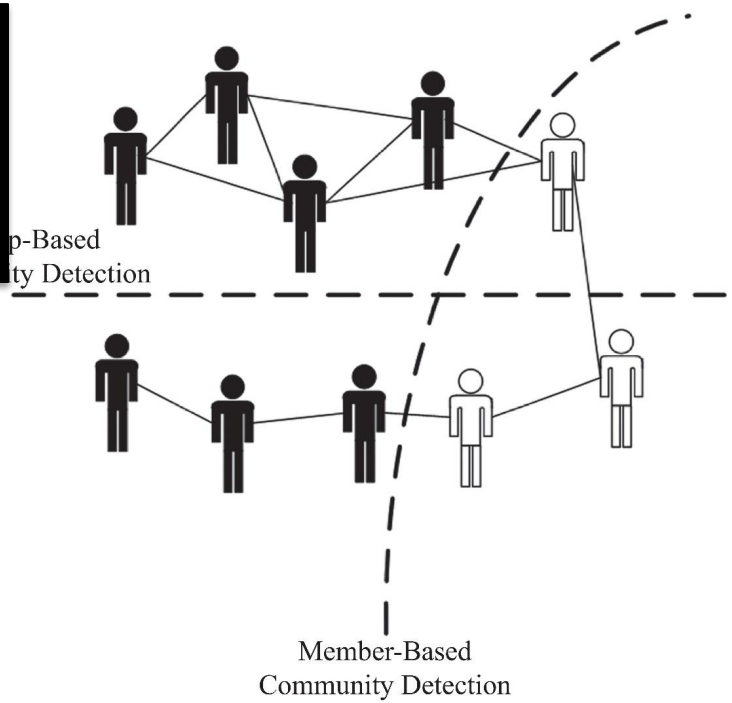
- Data is often non-linked (matrix rows)
- Clustering works on the distance or similarity matrix, e.g., k-means.
- If you use k-means with adjacency matrix rows, you are only considering the ego-centric network

Community detection

- Data is linked (a graph)
- Network data tends to be "discrete", leading to algorithms using the graph property directly
  - k-clique, quasi-clique, or edge-betweenness

Community Detection Algorithms

Group Users based on Group attributes



Group Users based on Member attributes

Member-Based Community Detection

Member-Based Community Detection

- Look at node characteristics; and
- Identify nodes with similar characteristics and consider them a community

Node Characteristics

A. Degree

- Nodes with same (or similar) degrees are in one community
- Example: cliques

B. Reachability

- Nodes that are close (small shortest paths) are in one community
- Example: k-cliques, k-clubs, and k-clans

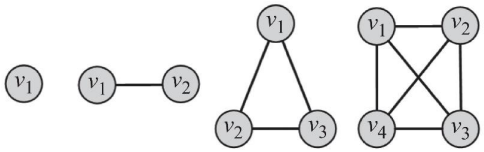
C. Similarity

- Similar nodes are in the same community

A. Node Degree

Most common subgraph searched for:

- **Clique**: a maximum complete subgraph in which all nodes inside the subgraph adjacent to each other



Find communities by searching for

1. **The maximum clique**: the one with the largest number of vertices, or
2. **All maximal cliques**: cliques that are not subgraphs of a larger clique; i.e., cannot be further expanded

To overcome this, we can

I. Brute Force
II. Relax cliques
III. Use cliques as the core for larger communities

Both problems are NP-hard

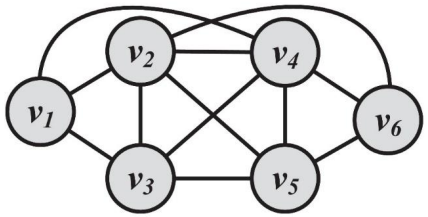
II. Relaxing Cliques

- ***k*-plex**: a set of vertices V in which we have

$$d_v \geq |V| - k, \forall v \in V$$

- d_v is the degree of v in the induced subgraph
  - Number of nodes from V that are connected to v

- Clique of size k is a 1-plex
- Finding the maximum k-plex: **NP-hard**
  - In practice, relatively easier due to smaller search space.



1-plex : $\{v_2, v_3, v_4, v_5\}$

2-plex : $\{v_1, v_2, v_3, v_4, v_5\}, \{v_2, v_3, v_4, v_5, v_6\}$

3-plex : $\{v_1, v_2, v_3, v_4, v_5, v_6\}$

Maximal k-plexes

III. Using Cliques as a Seed of a Community

Clique Percolation Method (CPM)

- Uses cliques as seeds to find larger communities
- CPM finds overlapping communities

- **Input**
  - A parameter k, and a network
- Procedure
  - Find out all cliques of size k in the given network
  - Construct a clique graph
    - Two cliques are adjacent if they share $k - 1$ nodes
  - Each connected components in the clique graph form a community

B. Node Reachability

The two extremes

Nodes are assumed to be in the same community

1. If there is a path between them (regardless of the distance) or
2. They are so close as to be immediate neighbors

How? Find using BFS/DFS

Challenge: most nodes are in one community (giant component)

How? Finding Cliques

Challenge: Cliques are challenging to find and are rarely observed

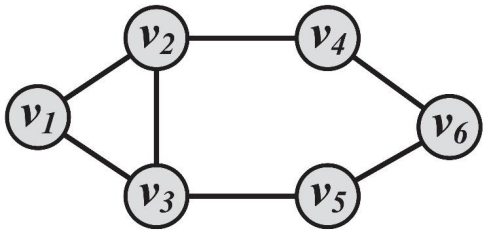
Solution: find communities that are in between cliques and connected components in terms of connectivity and have small shortest paths between their nodes

Special Subgraphs

1. ***k*-Clique**: a **maximal** subgraph in which the largest shortest path distance between any nodes is less than or equal to *k*

2. ***k*-Club**: follows the same definition as a *k*-clique
   - **Additional Constraint:** nodes on the shortest paths should be part of the subgraph (i.e., diameter)

3. ***k*-Clan:** a *k*-clique where for all shortest paths within the subgraph the distance is equal or less than *k*
   - All *k*-clans are *k*-cliques, but not vice versa



2-cliques : $\{v_1, v_2, v_3, v_4, v_5\}, \{v_2, v_3, v_4, v_5, v_6\}$

2-clubs : $\{v_2, v_3, v_4, v_5, v_6\}, \{v_1, v_2, v_3, v_4\}, \{v_1$

2-clans : $\{v_2, v_3, v_4, v_5, v_6\}$

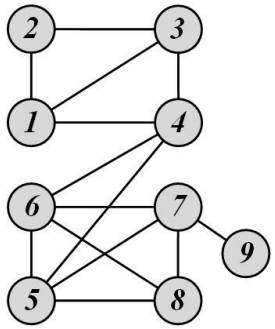
C. Node Similarity

- Similar (or most similar) nodes are assumed to be in the same community
  - A classical clustering algorithm (e.g., k-means) is applied to node similarities to find communities

- Node similarity can be defined
  - Using the similarity of node neighborhoods (**Structural Equivalence**) – Ch. 3
  - Similarity of social circles (**Regular Equivalence**) – Ch. 3

Structural equivalence: two nodes are structurally equivalent iff. they are connecting to the same set of actors

Nodes 1 and 3 are structurally equivalent, So are nodes 5 and 6.



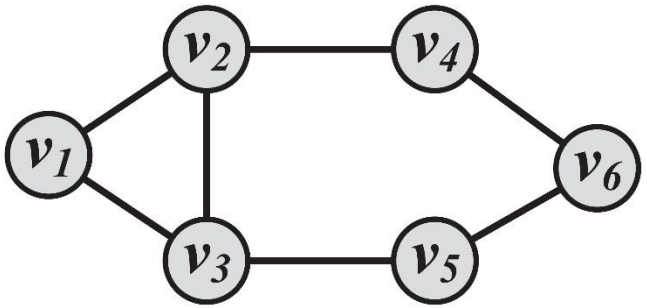
Node Similarity (Structural Equivalence)

Jaccard Similarity

$$\sigma_{\text{Jaccard}}(v_i, v_j) = \frac{|N(v_i) \cap N(v_j)|}{|N(v_i) \cup N(v_j)|}$$

Cosine similarity

$$\sigma_{\text{Cosine}}(v_i, v_j) = \frac{|N(v_i) \cap N(v_j)|}{\sqrt{|N(v_i)||N(v_j)|}}$$



$$\sigma_{\text{Jaccard}}(v_2, v_5) = \frac{|\{v_1, v_3, v_4\} \cap \{v_3, v_6\}|}{|\{v_1, v_3, v_4, v_6\}|} = 0.25$$

$$\sigma_{\text{Cosine}}(v_2, v_5) = \frac{|\{v_1, v_3, v_4\} \cap \{v_3, v_6\}|}{\sqrt{|\{v_1, v_3, v_4\}||\{v_3, v_6\}|}} = 0.40$$

Group-Based Community Detection

Group-Based Community Detection

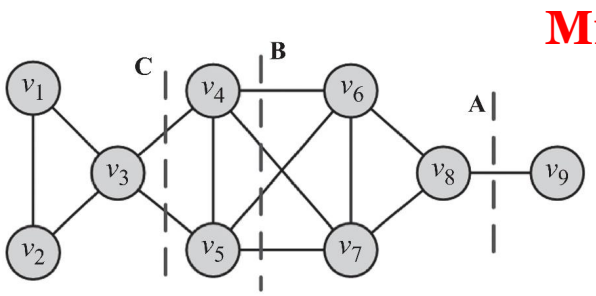
Group-based community detection: finding communities that have certain **group properties**

Group Properties:

I. **Balanced:** Spectral clustering
II. **Robust:** k-connected graphs
III. **Modular:** Modularity Maximization
IV. **Dense:** Quasi-cliques
V. **Hierarchical:** Hierarchical clustering

I. Balanced Communities

- Community detection can be thought of *graph clustering*
- **Graph clustering:** we cut the graph into several partitions and assume these partitions represent communities
- **Cut**: partitioning (*cut*) of the graph into two (or more) sets (*cutsets*)
  - **The size of the cut** is the number of edges that are being cut
- **Minimum cut (min-cut) problem**: find a graph partition such that the number of edges between the two sets is minimized



Min-cuts can be computed efficiently using the max-flow min-cut theorem

Min-cut often returns an imbalanced partition, with one set being a singleton

Ratio Cut and Normalized Cut

- To mitigate the min-cut problem we can change the objective function to consider community size

$$\text{Ratio Cut}(P) = \frac{1}{k}\sum_{i=1}^{k}\frac{\text{cut}(P_i,\bar{P}_i)}{|P_i|}$$

$$\text{Normalized Cut}(P) = \frac{1}{k}\sum_{i=1}^{k}\frac{\text{cut}(P_i,\bar{P}_i)}{\text{vol}(P_i)}$$

- $\bar{P}_i = V - P_i$ is the complement cut set
- $cut(P_i, \bar{P}_i)$ is the size of the cut
- $vol(P_i) = \sum_{v \in P_i} d_v$

II. Robust Communities

- The goal is find subgraphs robust enough such that removing some edges or vertices does not disconnect the subgraph

- ***k*-vertex connected (*k*-connected) graph:**
  - k is the minimum number of nodes that must be removed to disconnect the graph
- ***k*-edge connected:** at least k edges must be removed to disconnect the graph

- Examples:
  - Complete graph of size n: unique n-connected graph
  - A cycle: 2-connected graph

III. Modular Communities

Consider a graph $G(V, E)$, where the degrees are known beforehand however edges are not

- Consider two vertices *vi* and *vj* with degrees *di* and *dj*

What is an expected number of edges between v_i and v_j?

- For any edge going out of *vi* randomly the probability of this edge getting connected to vertex *vj* is

$$\frac{d_j}{\sum_i d_i} = \frac{d_j}{2m}$$

Modularity and Modularity Maximization

- Given a degree distribution, we know the expected number of edges between any pairs of vertices

- We assume that real-world networks should be far from random. Therefore, the more distant they are from this randomly generated network, the more structural they are

- Modularity defines this distance and modularity maximization tries to maximize this distance

Normalized Modularity

Consider a partitioning of the data $P = (P_1, P_2, P_3, \ldots, P_k)$

For partition P_x, this distance can be defined as

$$\sum_{i,j \in P_x} A_{ij} - \frac{d_i d_j}{2m}$$

This distance can be generalized for a partitioning P

$$\sum_{x=1}^{k} \sum_{i,j \in P_x} A_{ij} - \frac{d_i d_j}{2m}$$

The normalized version of this distance is defined as Modularity

$$Q = \frac{1}{2m} \sum_{x=1}^{k} \sum_{i,j \in P_x} A_{ij} - \frac{d_i d_j}{2m}$$

Modularity Maximization

Modularity matrix $B = A - dd^T/2m$

$d \in \mathbb{R}^{n\times 1}$ is the degree vector for all nodes

Reformulation of the modularity $Q = \frac{1}{2m}\mathrm{Tr}(X^T B X)$

- $X \in \mathbb{R}^{n\times k}$ is the indicator (partition membership) function:
  - $X_{ij} = 1$ iff. $v_i \in P_j$
- Similar to Spectral clustering,
  - We relax X to be orthogonal, i.e., matrix $\hat{X}$
  - The optimal solution for $\hat{X}$ is the top k eigenvectors of B
  - To recover the original X, we can run k-means on $\hat{X}$

Hierarchical Community Detection

- **Goal**: build a hierarchical structure of communities based on network topology
- Allow the analysis of a network at different resolutions
- Representative approaches:
  - Divisive Hierarchical Clustering
  - Agglomerative Hierarchical clustering

Divisive Hierarchical Clustering

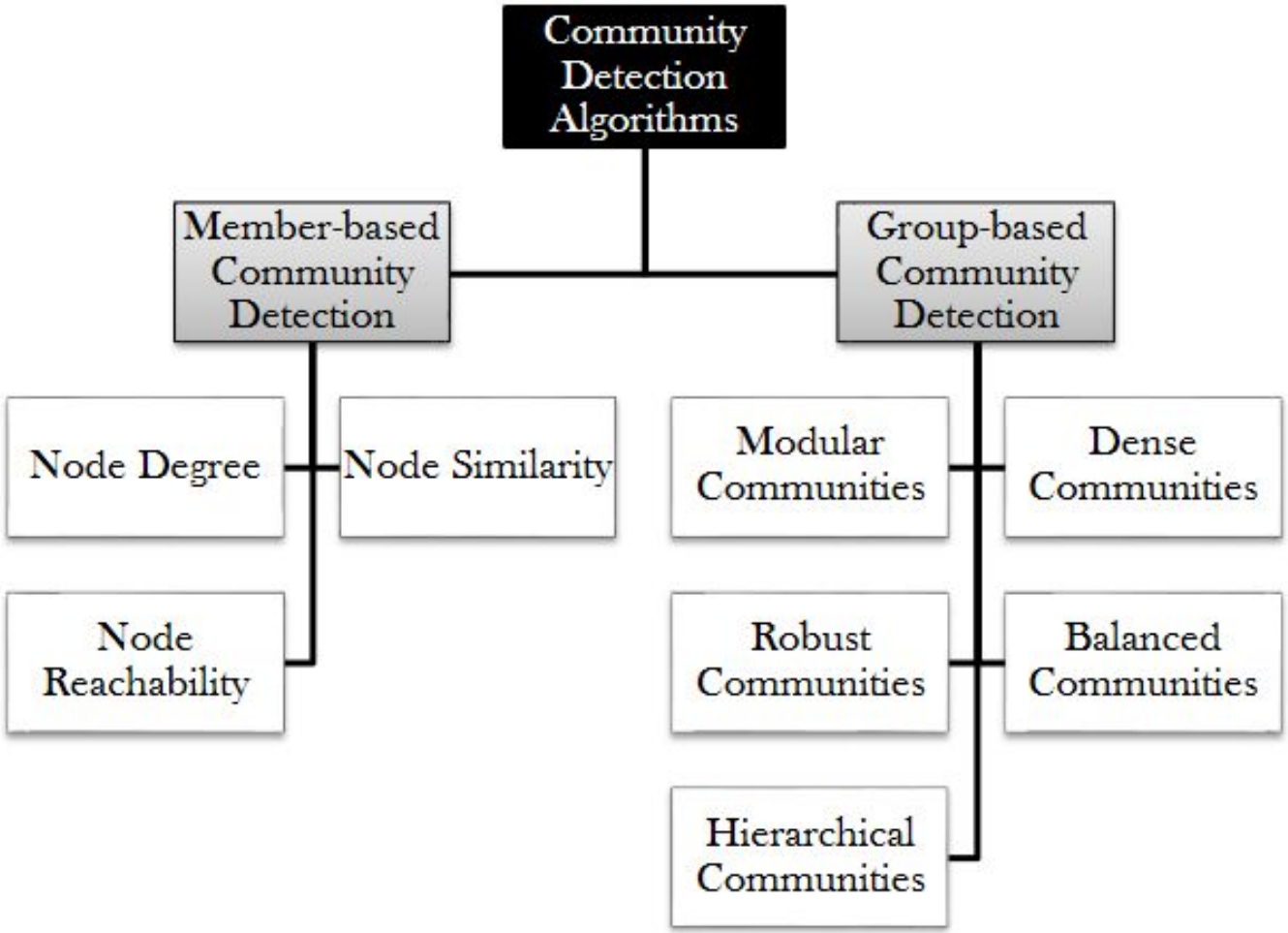
- Divisive clustering
  - Partition nodes into several sets
  - Each set is further divided into smaller ones
  - Network-centric partition can be applied for the partition
- One particular example:

Girvan-Newman Algorithm: recursively remove the "weakest" links within a "community"

- Find the edge with the weakest link
- Remove the edge and update the corresponding strength of each edge

- Recursively apply the above two steps until a network is discomposed into a desired number of connected components.
- Each component forms a community

Community Detection Algorithms



Community Evolution

Network Growth Patterns

1. Network Segmentation
2. Graph Densification
3. Diameter Shrinkage

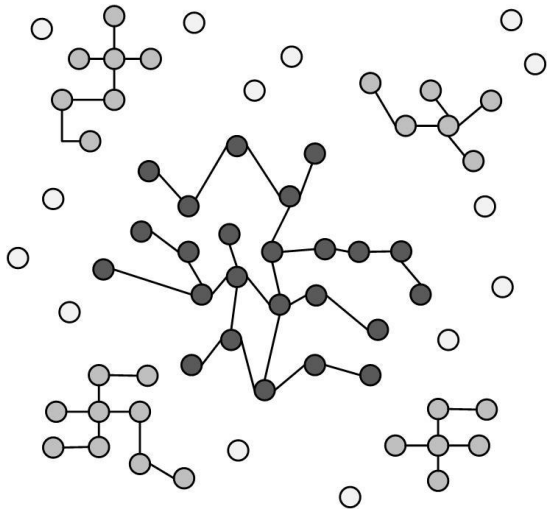
1. Network Segmentation

- Often, in evolving networks, segmentation takes place, where the large network is decomposed over time into three parts

1. **Giant Component**: As network connections stabilize, a giant component of nodes is formed, with a large proportion of network nodes and edges falling into this component.

2. **Stars**: These are isolated parts of the network that form star structures. A star is a tree with one internal node and n leaves.

3. **Singletons**: These are orphan nodes disconnected from all nodes in the network.



2. Graph Densification

- The density of the graph increases as the network grows
  - The number of edges increases faster than the number of nodes does

$$E(t) \propto V(t)^{\alpha}$$

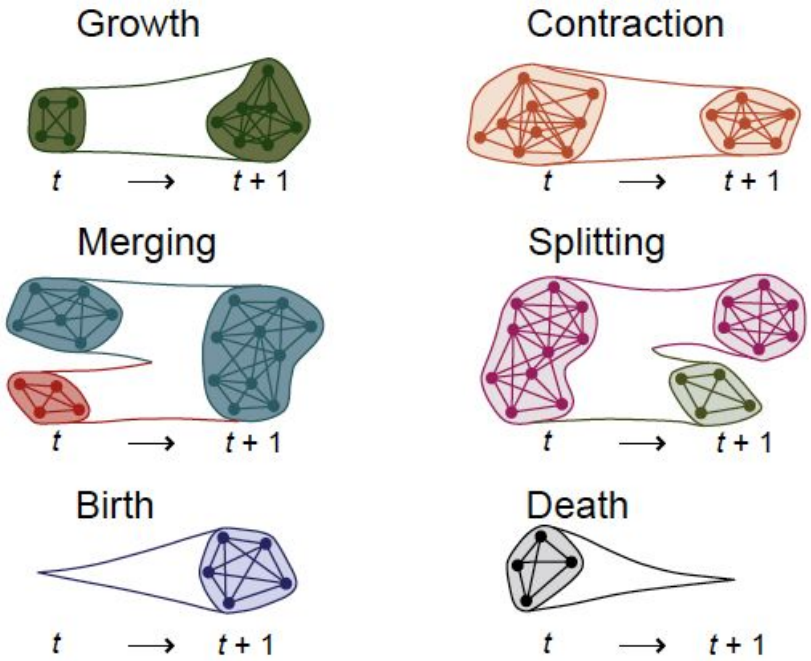
- Densification exponent: $1 \leq \alpha \leq 2$:
  - $\alpha = 1$: linear growth – constant out-degree
  - $\alpha = 2$: quadratic growth – clique

$E(t)$ and $V(t)$ are numbers of edges and nodes respectively at time t

How Communities Evolve?

Community Evolution

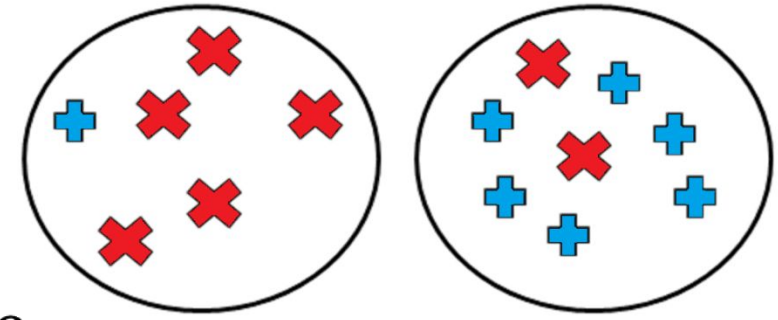
- Communities also expand, shrink, or dissolve in dynamic networks



Evaluating the Communities

We are given objects of two different kinds (+, ×)

- **The perfect community:** all objects inside the community are of the same type



- **Evaluation with ground truth**
- **Evaluation without ground truth**

Evaluation with Ground Truth

- When ground truth is available
  - We have partial knowledge of what communities should look like
  - We are given the correct community (clustering) assignments

- **Measures**
  - Precision and Recall, or F-Measure
  - Purity
  - Normalized Mutual Information (NMI)

Precision and Recall

$$Precision = \frac{Relevant\ and\ retrieved}{Retrieved}$$

$$Recall = \frac{Relevant\ and\ retrieved}{Relevant}$$

$$P = \frac{TP}{TP + FP}$$

$$R = \frac{TP}{TP + FN}$$

True Positive (TP) :

- When similar members are assigned to the same communities
- A **correct** decision.

False Negative (FN) :

- When similar members are assigned to different communities
- An **incorrect** decision

True Negative (TN) :

- When dissimilar members are assigned to different communities
- A **correct** decision

False Positive (FP) :

- When dissimilar members are assigned to the same communities
- An **incorrect** decision

Purity

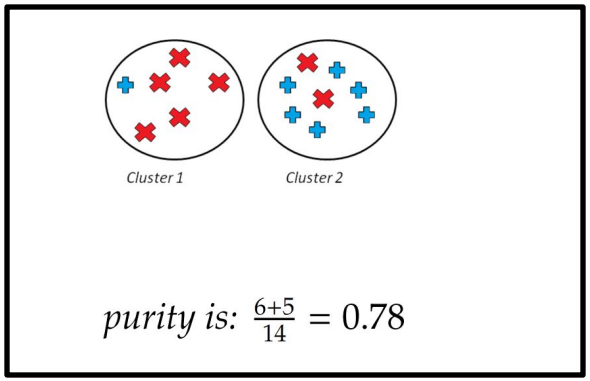
Purity can be easily tampered by

- Points being singleton communities (of size 1); or by
- Very large communities

member to evaluate the communities

Purity. The fraction of instances that have labels equal to the community's majority label

$$Purity = \frac{1}{N}\sum_{i=1}^{k}\max_{j}|C_i \cap L_j|$$



- k: the number of communities
- N: total number of nodes,
- L_j: the set of instances with label j in all communities
- C_i: the set of members in community i

Mutual Information

- **Mutual information (MI).** The amount of information that two random variables share.
  - By knowing one of the variables, it measures the amount of uncertainty reduced regarding the others

$$MI = I(H, L) = \sum_{h \in H} \sum_{l \in L} \frac{n_{h,l}}{n} \log \frac{n \cdot n_{h,l}}{n_h n_l}$$

- L and H are labels and found communities;
- n_h and n_l are the number of data points in community h and with label l, respectively;
- $n_{h,l}$ is the number of nodes in community h and with label l; and n is the number of nodes

Normalizing Mutual Information (NMI)

- Mutual information (MI) is unbounded
- To address this issue, we can normalize MI
- How? We know that

$$MI \leq min(H(L), H(H)),$$
$$(MI)^2 \leq H(H)H(L).$$
$$MI \leq \sqrt{H(H)}\sqrt{H(L)}.$$

- $H(.)$ is the entropy function

$$H(L) = -\sum_{l \in L} \frac{n_l}{n} \log \frac{n_l}{n}$$
$$H(H) = -\sum_{h \in H} \frac{n_h}{n} \log \frac{n_h}{n}.$$

Normalized Mutual Information

$$NMI = \frac{MI}{\sqrt{H(L)}\sqrt{H(H)}}.$$

$$NMI = \frac{\sum_{h \in H} \sum_{l \in L} n_{h,l} \log \frac{n \cdot n_{h,l}}{n_h n_l}}{\sqrt{(\sum_{h \in H} n_h \log \frac{n_h}{n})(\sum_{l \in L} n_l \log \frac{n_l}{n})}}.$$

We can also define it as

Note that $MI < 1/2(H(H) + H($

$$NMI = \frac{I(H;L)}{\frac{1}{2}(H(L) + H(H))}$$

Information Diffusion in Social Media

Kai Shu

Spring 2022

Information Diffusion

- Information diffusion: a process by which a piece of information (knowledge) is spread and reaches individuals through interactions
- Studied in a myriad of sciences
- We discuss methods from
  - Sociology, epidemiology, and ethnography
  - All are useful for social media mining
- How to model information diffusion

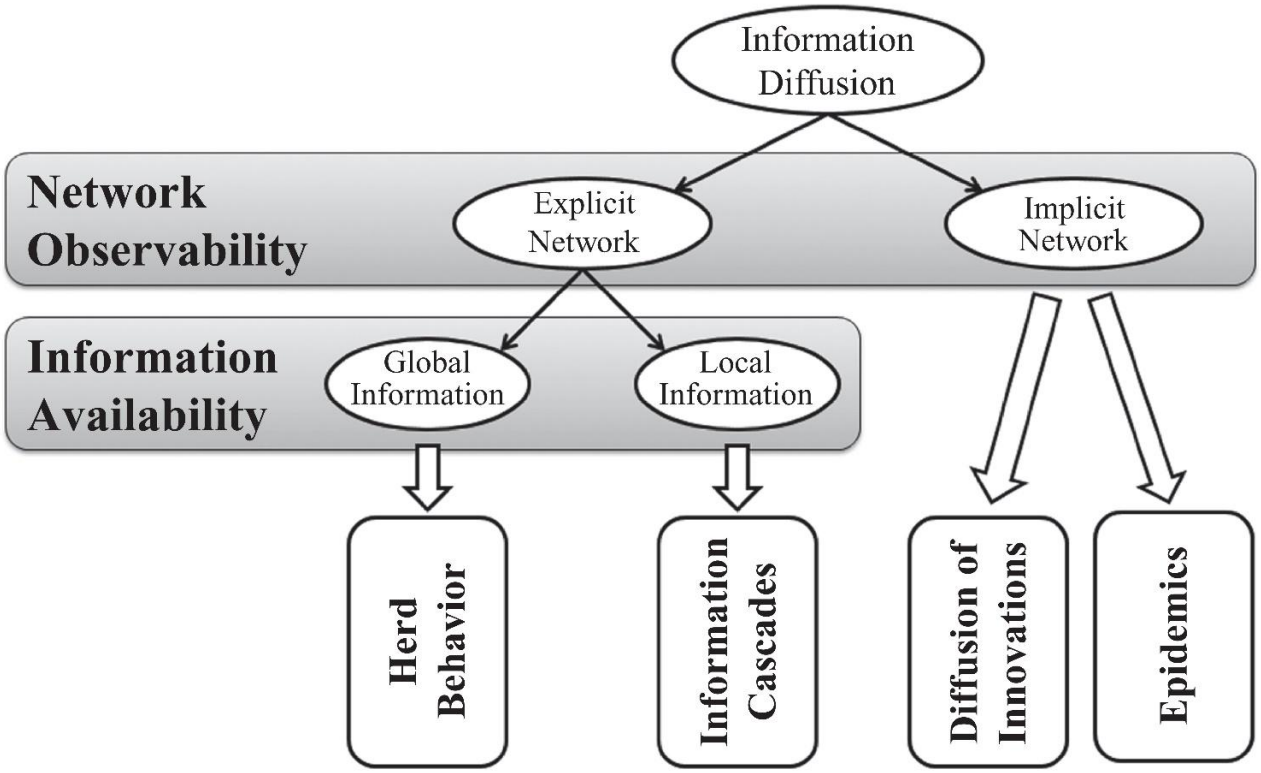
Information Diffusion

- **Sender(s).** A sender or a small set of senders that initiate the information diffusion process;

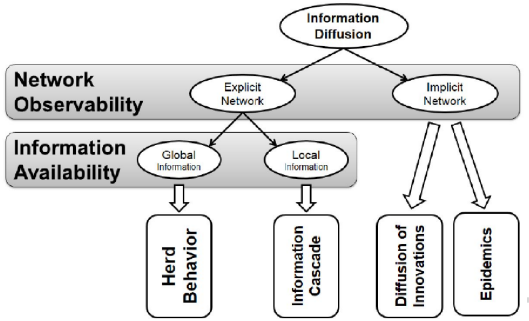
- **Receiver(s).** A receiver or a set of receivers that receive diffused information. Commonly, the set of receivers is much larger than the set of senders and can overlap with the set of senders; and

- **Medium.** This is the medium through which the diffusion takes place. For example, when a rumor is spreading, the medium can be the personal communication between individuals.

Types of Information Diffusion



Intervention is the process of interfering with information diffusion by expediting, delaying, or even stopping diffusion



Herd Behavior

- Network is observable
- Only public information is available

Herd Behavior

Herd behavior describes when a group of individuals performs actions that are highly correlated without any plan

Main Components of Herd Behavior

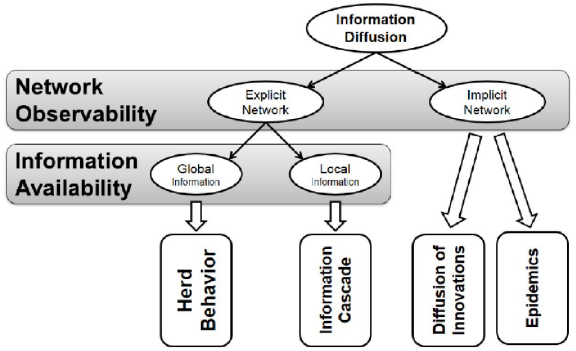
- A method to transfer behavior among individuals or to observe their behavior
- A connection between individuals

Examples of Herd Behavior

- Flocks, herds of animals, and humans during sporting events, demonstrations, and gatherings

Herding Intervention

To intervene the herding effect, we need one person to *tell the herd that there is nothing in the sky* or *simply ask what they're looking at*, and the first person started looking up was probably to stop his nose bleeding ☺️

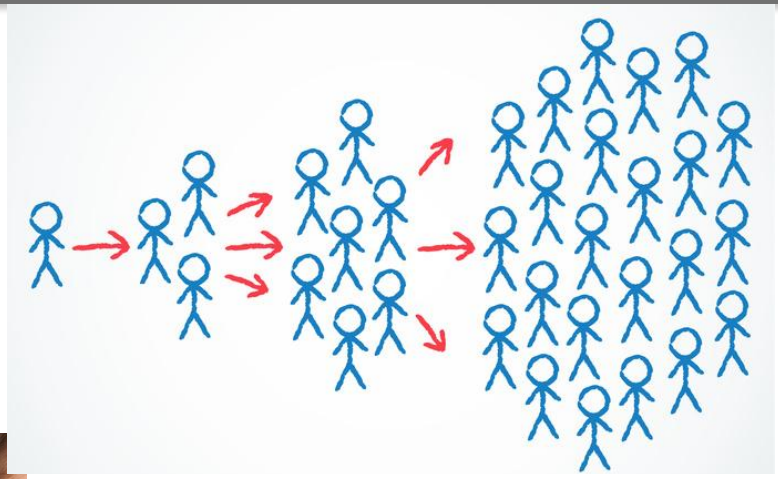


Information Cascade

- In the presence of a network
- Only local information is available

Information Cascade

- Users often repost content posted by others in the network.
  - Content is often received via immediate neighbors (friends).



- Information propagates through friends

An information cascade: a piece of information/decision cascaded among some users, where

individuals are connected by a network and

individuals are only observing decisions of their immediate neighbors

Cascade users have less information available

- Herding users have almost all information about decisions

Independent Cascade Model (ICM)

- **Independent Cascade Model (ICM)**
  - A *sender-centric* model of cascade
  - Each node has **one chance** to activate its neighbors

- In ICM, nodes that are active are senders and nodes that are being activated as receivers
  - The *linear threshold model* concentrates on the receiver (to be discussed later in Chapter 8)

ICM Algorithm

- Node activated at time t, has one chance, at time step $t + 1$, to activate its neighbors

- If v is activated at time t
  - For any neighbor w of v, there's a probability p_{vw} that node w gets activated at time $t + 1$.

- Node v activated at time t has a single chance of activating its neighbors
  - The activation can only happen at $t + 1$

ICM Algorithm

Algorithm 1 Independent Cascade Model (ICM)

Require: Diffusion graph $G(V, E)$, set of initial activated nodes A_0, activation probabilities $p_{v,w}$

return Final set of activated nodes A_∞
i = 0;
while $A_i \neq \{\}$ **do**

  $i = i + 1$;
  $A_i = \{\}$;
  for all $v \in A_{i-1}$ **do**
    for all w neighbor of $v, w \notin \cup_{j=0}^{i} A_j$ **do**
      rand = generate a random number in [0,1];
      if rand $< p_{v,w}$ **then**
        activate w;
        $A_i = A_i \cup \{w\}$;
      end if
    end for
  end for
end while
$A_\infty = \cup_{j=0}^{i} A_j$;
Return A_∞;

Maximizing the Spread of Cascades

Problem Setting

- **Given**
  - A limited budget **B** for initial advertising
    - Example: give away free samples of product
  - Connections between individuals
- **Goal**
  - To trigger a large spread
    - i.e., further adoptions of a product
- **Question**
  - Which set of individuals should be targeted at the very beginning?

Cascade Maximization: A Greedy Algorithm

The Algorithm

- Start with $B = \emptyset$
- *Evaluate f(v) for each node, and pick the node with maximum σ as the first node v1 to form B = {v1}*
- Select a node which will increase *f(B) most if the node is included in B.*

- *Essentially, we greedily* find a node $v \in V \setminus B$ *such that*

$$v = \arg\max_{v \in V \setminus B} f(B \cup \{v\})$$

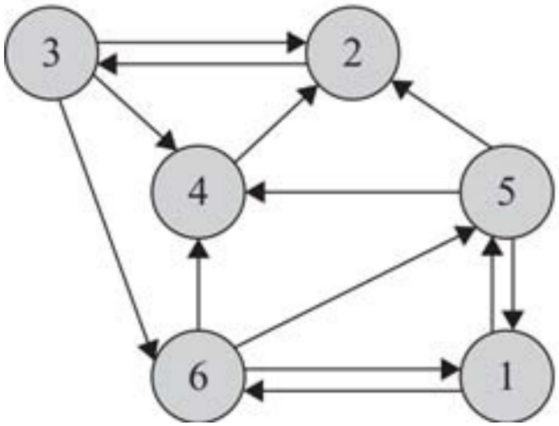
Cascade Maximization: A Greedy Algorithm

Algorithm 7.2 Maximizing the spread of cascades – Greedy algorithm

Require: Diffusion graph $G(V, E)$, budget k

1: **return** Seed set S (set of initially activated nodes)
2: $i = 0$;
3: $S = \{\}$;
4: **while** $i \neq k$ **do**
5: $v = \arg\max_{v \in V \setminus S} f(S \cup \{v\})$;
or `equivalently` $\arg\max_{v \in V \setminus S} f(S \cup \{v\}) - f(s)$
6: $S = S \cup \{v\}$;
7: $i = i + 1$;
8: **end while**
9: Return S;

Cascade Maximization Example



Algorithm 7.2 Maximizing the spread of cascades – Greedy algorithm

Require: Diffusion graph $G(V, E)$, budget k

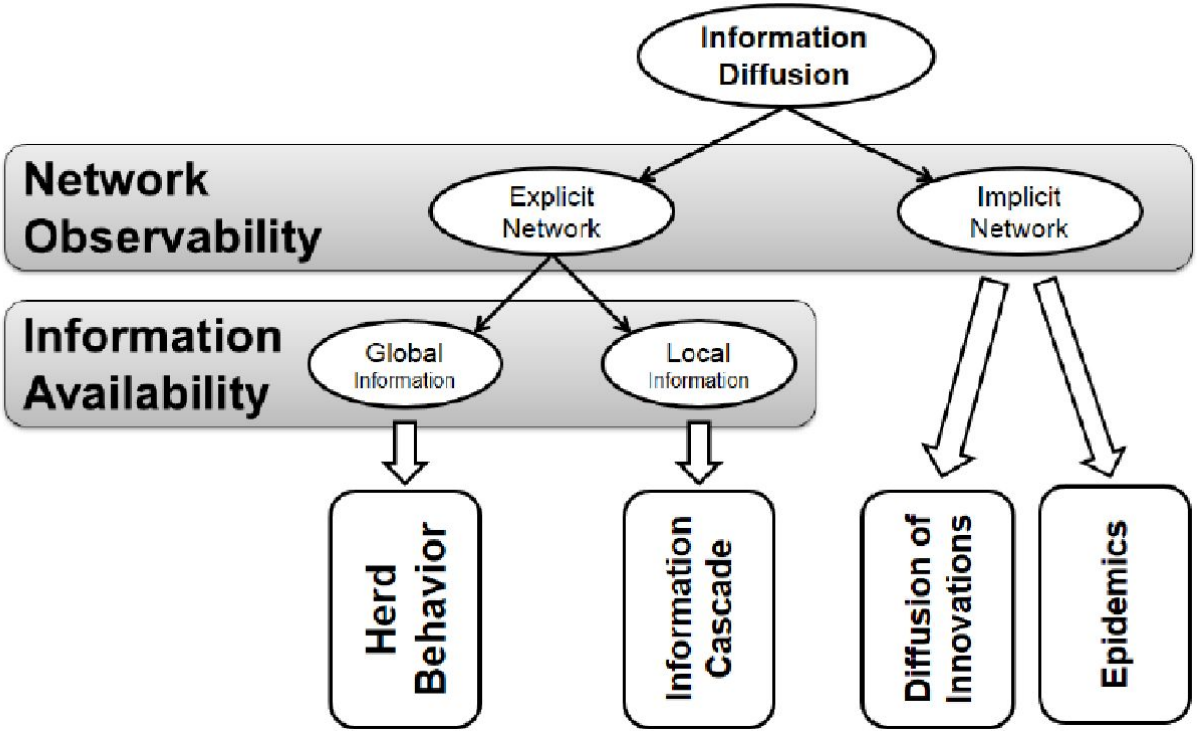
1: **return** Seed set S (set of initially activated nodes)
2: $i = 0$;
3: $S = \{\}$;
4: **while** $i \neq k$ **do**
5: $v = \arg\max_{v \in V \setminus S} f(S \cup \{v\})$;
or `equivalently` $\arg\max_{v \in V \setminus S} f(S \cup \{v\}) - f(s)$
6: $S = S \cup \{v\}$;
7: $i = i + 1$;
8: **end while**
9: Return S;

- **Activation rule:** $|i - j| \equiv 2 \pmod 3$
- **Budget:** 2

Diffusion of Innovations

- **The network is not observable**
- **Only public information is observable**

Information Diffusion Types



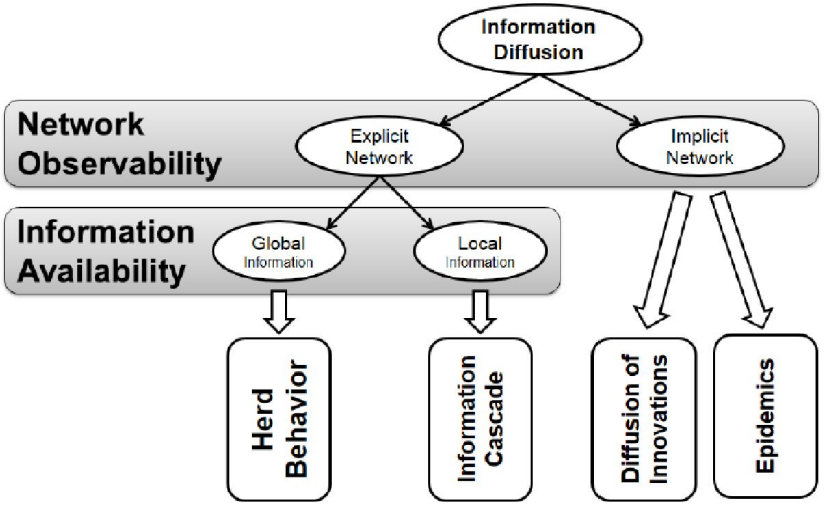
We define the process of interfering with information diffusion by expediting, delaying, or even stopping diffusion as Intervention

Innovation Characteristics

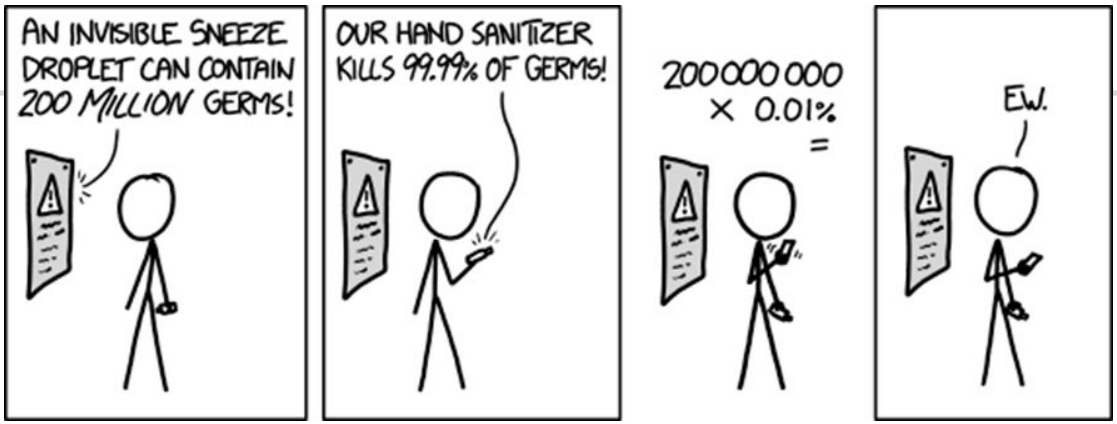
For an innovation to be adopted, the individuals adopting it (adopters) and the innovation must have certain qualities

Innovations must be:

- **Observable**,
  - The degree to which the results of an innovation are visible to potential adopters
- **Have Relative Advantage**
  - The degree to which the innovation is perceived to be superior to current practice
- **Compatible**
  - The degree to which the innovation is perceived to be consistent with socio- cultural values, previous ideas, and/or perceived needs
- **Trialable**
  - The degree to which the innovation can be experienced on a limited basis
- **Not too Complex**
  - The degree to which an innovation is difficult to use or understand.


Information Diffusion
Network Observability
Explicit Network
Implicit Network
Information Availability
Global Information
Local Information
Herd Behavior
Information Cascade
Diffusion of Innovations
Epidemics

Epidemics

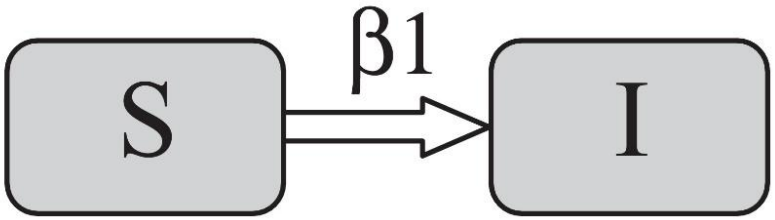

AN INVISIBLE SNEEZE DROPLET CAN CONTAIN 200 MILLION GERMS!
OUR HAND SANITIZER KILLS 99.99% OF GERMS!
200 000 000 × 0.01% =
EW.

Basic Epidemic Models

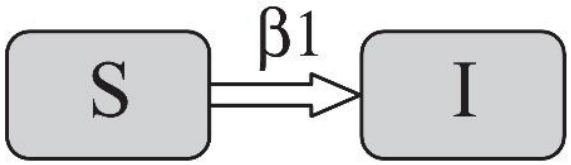
- SI
- SIR
- SIS
- SIRS

SI Model

- At each time stamp, an infected individual will meet βN people on average and will infect βS of them
- Since I are infected, βIS will be infected in the next time step



SI Model: Equations



$$\frac{dS}{dt} = -\beta IS, \qquad \frac{dI}{dt} = \beta IS.$$

$$(S + I = N) \Rightarrow \frac{dI}{dt} = \beta I(N - I) \Rightarrow I(t) = \frac{NI_0 e^{\beta t N}}{N + I_0(e^{\beta t N} - 1)}$$

I_0 is the number of individuals infected at time 0

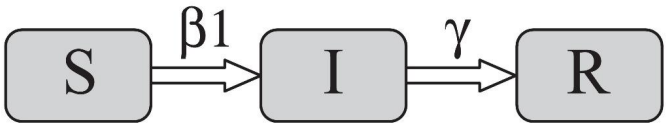
SIR Model

SIR model:

- In addition to the **I** and **S** states, a recovery state **R** is present

- Individuals get infected and some recover

- Once hosts recover (or are removed) they can no longer get infected and are not susceptible



SIR Model, Equations

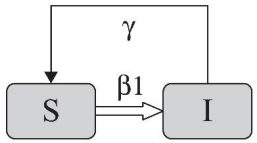
$$I + S + R = N$$

$$\begin{aligned}
\frac{dS}{dt} &= -\beta IS, \\
\frac{dI}{dt} &= \beta IS - \gamma I, \\
\frac{dR}{dt} &= \gamma I.
\end{aligned}$$

γ defines the recovering probability of an infected individual at a time stamp

SIS Model

- The **SIS** model is the same as the **SI** model with the addition of infected nodes recovering and becoming susceptible again

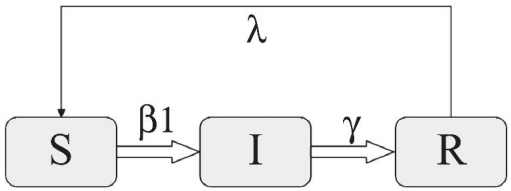


$$\frac{dS}{dt} = \gamma I - \beta IS, \qquad \frac{dI}{dt} = \beta IS - \gamma I$$

$$\Rightarrow \quad \frac{dI}{dt} = \beta I(N - I) - \gamma I = I(\beta N - \gamma) - \beta I^2$$

SIRS Model

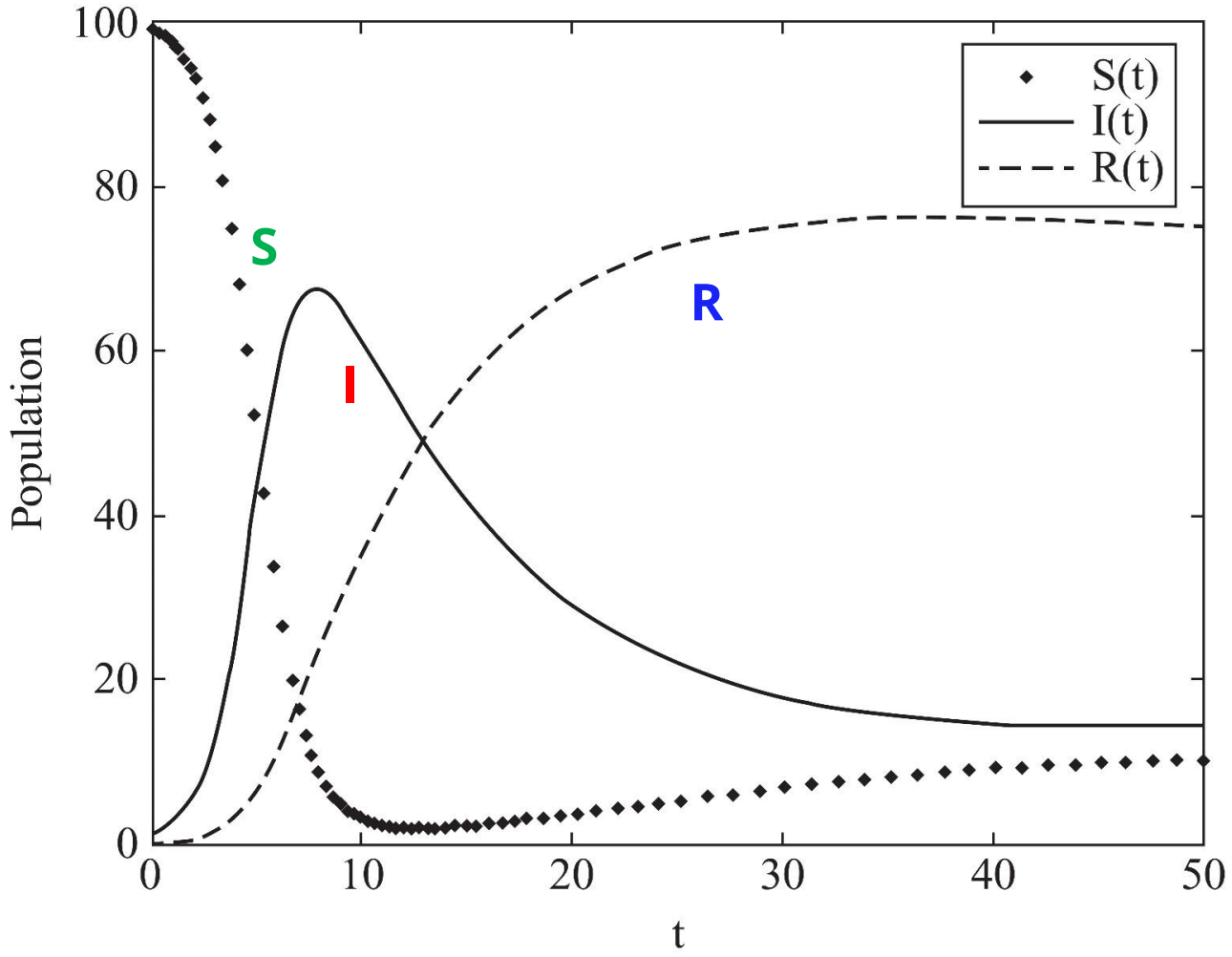
The individuals who have recovered will lose immunity after a certain period of time and will become susceptible again



$$\frac{dS}{dt} = \lambda R - \beta IS,$$
$$\frac{dI}{dt} = \beta IS - \gamma I,$$
$$\frac{dR}{dt} = \gamma I - \lambda R.$$

Like the SIR, model this model has no closed form solution, so numerical integration can be used

SIRS Model



SIRS model simulated with $S_0 = 99$, $I_0 = 1$, $R_0 = 1$, $\beta = 0.01$, $\lambda = 0.02$, and $\gamma = 0.1$

Influence and Homophily

Kai Shu

Spring 2022

Why are connected people similar?

Influence

- The process by which a user (i.e., influential) affects another user
- The influenced user becomes more similar to the influential figure.
  - **Example:** If most of our friends/family members switch to a cellphone company, we might switch [i.e., become influenced] too.

Homophily

- Similar individuals becoming friends due to their high similarity
  - **Example:** Two musicians are more likely to become friends.

Confounding

- The environment's effect on making individuals similar
  - **Example:** Two individuals living in the same town are more likely to become friends than two random individuals

Source of Assortativity in Networks

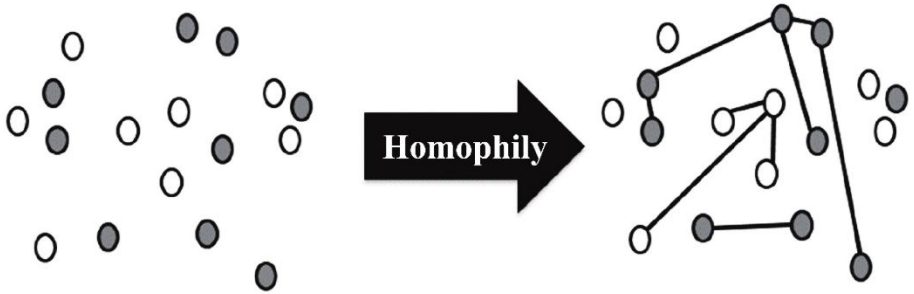
Both influence and Homophily generate similarity in social networks

Influence

Makes connected nodes similar to each other

Homophily

Selects similar nodes and links them together



Measuring Assortativity for Nominal Attributes

- Assume **nominal** attributes are assigned to nodes
  - Example: race

- Edges between nodes of the same type can be used to measure assortativity of the network
  - Same type = nodes that share some attribute value(s)
  - Node attributes could be nationality, race, sex, etc.

$$\frac{1}{m}\sum_{(v_i,v_j)\in E}\delta(t(v_i),t(v_j)) = \frac{1}{2m}\sum_{ij}A_{ij}\delta(t(v_i),t(v_j))$$

$t(v_i)$ denotes type of vertex v_i

$$\delta(x,y) = \begin{cases} 0, & \text{if } x \neq y \\ 1, & \text{if } x = y \end{cases}$$

Kronecker delta function

Measuring Assortativity for Ordinal Attributes

- A common measure for analyzing the relationship between *ordinal* values is *covariance*
- It describes how two variables change together
- In our case, we have a network
  - We are interested in how values assigned to nodes that are connected (via edges) are correlated

Influence

- Measuring Influence
- Modeling Influence

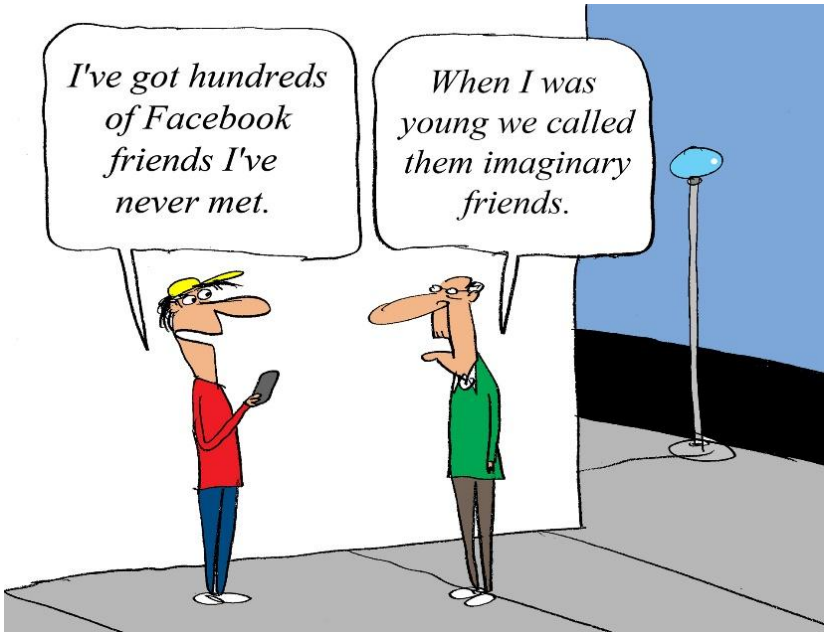
Prediction-based Measurement

We assume that

- an individual's attribute, or
- the way the user is situated in the network

can **predict** how influential the user **will** be

- Example 1
  - The number of *friends* of an individual is correlated with how influential she is
    - It is natural to use any of the c***entrality measures*** discussed (Chapter 3) for prediction-based influence measurements
    - How strong are these friendships?
- Example 2
  - On Twitter, in-degree (number of ***followers***) is a benchmark for measuring influence



Observation-based Measurement

We quantify influence of an individual by measuring the amount of influence ***attributed*** to the individual

I. When an individual is the role model

- Influence measure: size of the audience that has been influenced

II. When an individual spreads information

- Influence measure: the size of the cascade, the population affected, the rate at which the population gets influenced

III. When an individual increases values

- Influence measure: the increase (or rate of increase) in the value of an item or action
  - The second person who bought the fax machine increased its value dramatically

Linear Threshold Model (LTM)

A node i would become active if incoming influence ($w_{j,i}$) from friends exceeds a certain threshold

$$\sum_{v_j \in N_{\text{in}}(v_i)} w_{j,i} \leq 1$$

- Each node i chooses a threshold θ_i randomly from a uniform distribution in an interval between 0 and 1
- At time t, all nodes that were active in the previous steps $[0 \ldots t-1]$ remain active, but only nodes activated at time $t-1$ get the chance to activate
- Nodes satisfying the following condition will be activated

$$\sum_{v_j \in N_{\text{in}}(v_i), v_j \in A_{t-1}} w_{j,i} \geq \theta_i$$

LTM Algorithm

Algorithm 1 Linear Threshold Model (LTM)

Require: Graph $G(V, E)$, set of initial activated nodes A_0

return Final set of activated nodes A_∞
i=0;
Uniformly assign random thresholds θ_v from the interval [0, 1];
while $i = 0$ or $(A_{i-1} \neq A_i, i \geq 1)$ **do**
    $A_{i+1} = A_i$
    inactive $= V - A_i$;
    for all $v \in$ inactive **do**
        if $\sum_{j \text{ connected to } v, j \in A_i} w_{j,v} \geq \theta_v$. **then**
            activate v;
            $A_{i+1} = A_{i+1} \cup \{v\}$;
        end if
    end for
    $i = i + 1$;
end while
$A_\infty = A_i$;
Return A_∞;

Homophily

“Birds of a feather flock together”

Definition

Homophily: the tendency of individuals to associate and bond with similar others

- i.e., love of the same

- People interact more often with people who are "*like them*" than with people who are dissimilar

What leads to Homophily?

- Race and ethnicity, Sex and Gender, Age, Religion, Education, Occupation and social class, Network positions, Behavior, Attitudes, Abilities, Beliefs, and Aspirations

Homophily Model

Algorithm 1 Homophily Model

Require: Graph $G(V, E)$, $E = \emptyset$, similarities $sim(v, u)$

1: **return** Set of edges E
2: **for all** $v \in V$ **do**
3: θ_v = generate a random number in [0,1];
4: **for all** $(v, u) \notin E$ **do**
5: **if** $\theta_v < sim(v, u)$ **then**
6: $E = E \cup (v, u)$;
7: **end if**
8: **end for**
9: **end for**
10: Return E;

Distinguishing Influence and Homophily

- Shuffle Test
- Edge-Reversal Test
- Randomization Test

I. Shuffle Test (Influence)

IDEA:

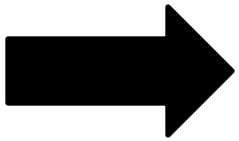
- Influence is temporal
- If u influences v, then u should have been activated before v.
- Define a temporal assortativity measure
- If there is no influence, then *a shuffling of the activation timestamps* should not affect the temporal assortativity measurement

Shuffle Test of Influence

If influence does not play a role, the timing of activations should be independent of users.

Even if we randomly shuffle the timestamps of user activities, we should obtain a similar temporal assortativity value

User	A	B	C
Time	1	2	3



User	A	B	C
Time	2	3	1

Test of Influence

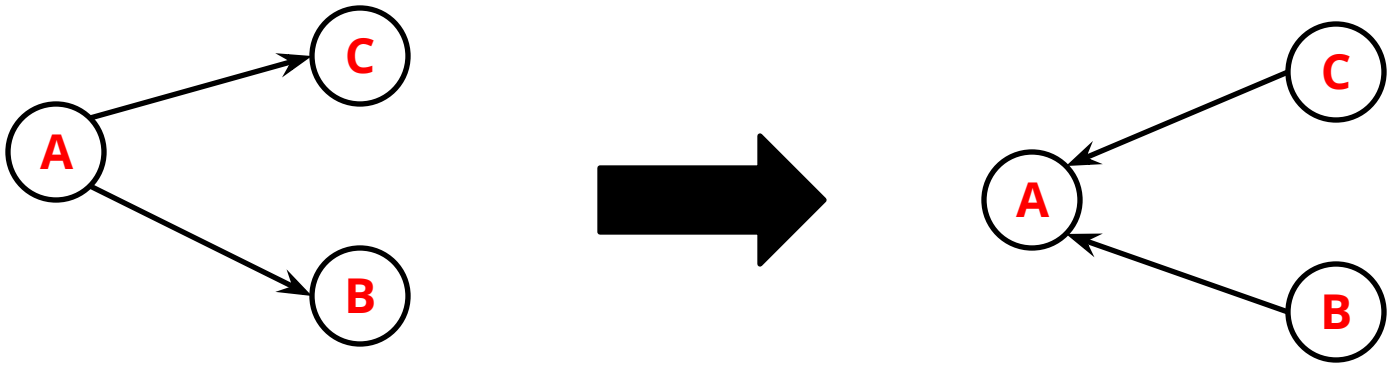
After we shuffle the timestamps of user activities, if the new estimate of temporal assortativity is significantly different from the original estimate based on the user's activity log,

there is evidence of influence.

2. The Edge-reversal Test (Influence)

If influence resulted in activation, then the direction of edges should be important (who influenced whom).

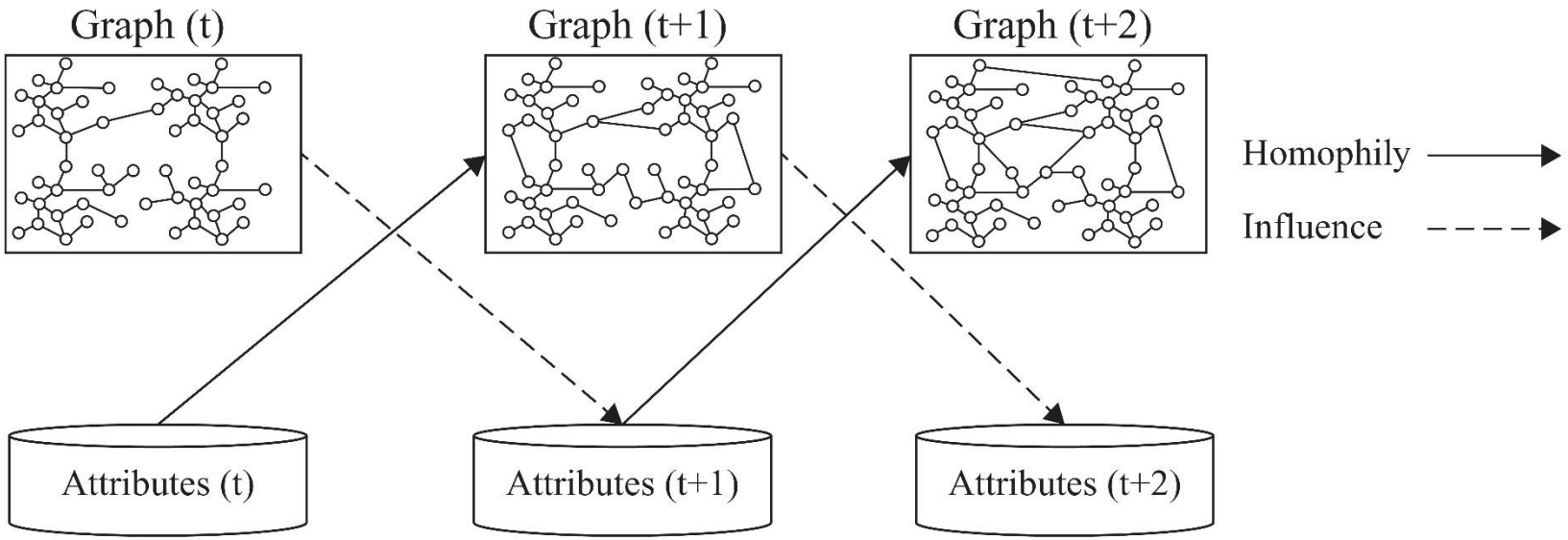
- Reverse directions of all the edges
- Run the same logistic regression on the data using the new graph
- If correlation is not due to influence, then α should not change



3. Randomization Test (Influence/Homophily)

- Capable of detecting both Influence and Homophily in networks

- *Influence* changes *attributes* and *Homophily* changes *connections*



Influence Significance Test

Algorithm 1 Influence Significance Test

Require: G_t, G_{t+1}, X_t, X_{t+1}, number of randomized runs n, α

1: **return** Significance
2: $g_0 = G_{Influence}(t)$;
3: **for all** $1 \leq i \leq n$ **do**
4: $\quad XR^i_{t+1} = randomize_I(X_t, X_{t+1})$;
5: $\quad g_i = A(G_t, XR^i_{t+1}) - A(G_t, X_t)$;
6: **end for**
7: **if** g_0 larger than $(1 - \alpha/2)\%$ of values in $\{g_i\}_{i=1}^n$ **then**
8: $\quad$ return significant;
9: **else if** g_0 smaller than $\alpha/2\%$ of values in $\{g_i\}_{i=1}^n$ **then**
10: $\quad$ return significant;
11: **else**
12: $\quad$ return insignificant;
13: **end if**

Homophily Significance Test

Algorithm 1 Homophily Significance Test

Require: G_t, G_{t+1}, X_t, X_{t+1}, number of randomized runs n, α

1: **return** Significance
2: $g_0 = G_{Homophily}(t)$;
3: **for all** $1 \leq i \leq n$ **do**
4: $GR^i_{t+1} = randomize_H(G_t, G_{t+1})$;
5: $g_i = A(GR^i_{t+1}, X_t) - A(G_t, X_t)$;
6: **end for**
7: **if** g_0 larger than $(1 - \alpha/2)\%$ of values in $\{g_i\}_{i=1}^n$ **then**
8: return significant;
9: **else if** g_0 smaller than $\alpha/2\%$ of values in $\{g_i\}_{i=1}^n$ **then**
10: return significant;
11: **else**
12: return insignificant;
13: **end if**

Recommendation in Social Media

Kai Shu

Reading Chapter 9

Spring 2022

Recommendation vs. Search

- One way to get answers is using search engines
- Search engines find results that match the query provided by the user
  - You have to search!
- The results are generally provided as a list ordered with respect to the relevance of the item to the given query
- Consider the query "best 2014 movie to watch"
  - The same results for an 8 year old and an adult

Search engines' results are not personalized

Classical Recommendation Algorithms

- Content-based algorithms
- Collaborative filtering

Content-Based Methods

- Content-based recommendation systems are based on the fact that a user's interest should match the description of the items that she should be recommended by the system.
- In other words, the more similar the item's description to that of the user's interest, the more likely that the user finds the item's recommendation interesting.

Finding the similarity between the user and all of the existing items is the core of this type of recommender systems

Content-Based Methods

- To formalize a content-based method, we first represent both user profiles and item descriptions by vectorizing them using a set of *k* keywords
- We can vectorize (e.g., using TF-IDF) both users and items and compute their similarity

$$I_j = (i_{j,1}, i_{j,2}, \ldots, i_{j,k}) \qquad U_i = (u_{i,1}, u_{i,2}, \ldots, u_{i,k}).$$

$$sim(U_i, I_j) = cos(U_i, I_j) = \frac{\sum_{l=1}^{k} u_{i,l} i_{j,l}}{\sqrt{\sum_{l=1}^{k} u_{i,l}{}^2} \sqrt{\sum_{l=1}^{k} i_{j,l}{}^2}}$$

- We can recommend the top most similar items to the user

Content-Based Recommendation Algorithm

Algorithm 9.1 Content-based recommendation

Require: User i's Profile Information, Item descriptions for items $j \in \{1, 2, \ldots, n\}$, k keywords, r number of recommendations.

1: **return** r recommended items.
2: $U_i = (u_1, u_2, \ldots, u_k)$ = user i's profile vector;
3: $\{I_j\}_{j=1}^{n} = (i_{j,1}, i_{j,2}, \ldots, i_{j,k})$= item j's description vector;
4: $s_{i,j} = sim(U_i, I_j)$;
5: Return top r items with maximum similarity $s_{i,j}$.

- In content-based recommendation, we compute the topmost similar items to a user *i* and then recommend these items in the order of similarity

Collaborative Filtering

- Collaborative filtering is the process of selecting information or patterns using techniques involving collaboration among multiple agents, viewpoints, data sources, etc.
- The main advantage of this method is that the recommender system *does not need to have additional information about the users or content of the items*
- Users' rating or purchase history is the only information that is needed to work

Collaborative Filtering

Types of Collaborative Filtering Algorithms:

- **Memory-based**: Recommendation is directly based on previous ratings in the stored matrix that describes user-item relations

- **Model-based**: Alternatively, one can assume that an underlying model (hypothesis) governs the way users rate items.
  This model can be approximated and learned. The model is then used to recommend ratings.
  A model, for example, is that users rate low budget movies poorly

Memory-Based Collaborative Filtering

The most important assumptions of collaborative filtering are:

- **User-based CF**

  Users with similar **previous** ratings for items are likely to rate future items similarly

	I1	I2	I3	I4
U1	1	2	4	4
U2	1	2	4	?
U3	2	5	2	2
U4	5	2	3	3

- **Item-based CF**

  Items that have received similar ratings **previously** from users are likely to receive similar ratings from future users (*item-based CF)*

	I1	I2	I3	I4
U1	1	2	4	4
U2	1	2	4	?
U3	2	5	2	2
U4	5	2	3	3

Collaborative Filtering: Algorithm

1. Weigh all users/items with respect to their similarity with the current user/item
2. Select a subset of the users/items (neighbors) as recommenders
3. Predict the rating of the user for specific items using neighbors' ratings for the same (or similar) items
4. Recommend items with the highest predicted rank

Measure Similarity between Users (or Items)

- Cosine Similarity

$$sim(U_i, U_j) = cos(U_i, U_j) = \frac{U_i \cdot U_j}{\|U_i\| \, \|U_j\|} = \frac{\sum_k r_{i,k} r_{j,k}}{\sqrt{\sum_k r_{i,k}^2} \sqrt{\sum_k r_{j,k}^2}}.$$

- Pearson Correlation Coefficient

$$sim(U_i, U_j) = \frac{\sum_k (r_{i,k} - \bar{r}_i)(r_{j,k} - \bar{r}_j)}{\sqrt{\sum_k (r_{i,k} - \bar{r}_i)^2} \sqrt{\sum (r_{j,k} - \bar{r}_j)^2}}.$$

User-based CF

Updating the ratings:

User v's mean rating

User u's mean rating

$$r_{u,i} = \bar{r}_u + \frac{\sum_{v \in N(u)} sim(u,v)(r_{v,i} - \bar{r}_v)}{\sum_{v \in N(u)} sim(u,v)},$$

Predicted rating of user *u* for item *i*

Observed rating of user *v* for item *i*

Item-based CF

Calculate the similarity between items and then predict new items based on the past ratings for similar items

$$r_{u,i} = \bar{r}_i + \frac{\sum_{j \in N(i)} sim(i,j)(r_{u,j} - \bar{r}_j)}{\sum_{j \in N(i)} sim(i,j)},$$

Item i's mean rating

i and j are two items

Model-Based Collaborative Filtering

- **In memory-based methods** (either item-based or user-based), we aim to predict the missing ratings based on similarities between users or items.
- **In model-based collaborative filtering**, we assume that an underlying model governs the way users rate.
- We aim to learn the model and then use that model to predict the missing ratings.
  - Among a variety of model-based techniques, we focus on a well-established model-based technique that is based on singular value decomposition (SVD).

Aggregation Strategies

- **Maximizing Average Satisfaction**
  - Average everyone's ratings and choose the max

$$R_i = \frac{1}{n} \sum_{u \in G} r_{u,i}$$

- **Least Misery**
  - This approach tries to minimize the dissatisfaction among group's members (Max of the mins of all)

$$R_i = \min_{u \in G} r_{u,i}$$

- **Most Pleasure**
  - The maximum of individuals' maximum ratings is taken as group's rating

$$R_i = \max_{u \in G} r_{u,i}$$

Recommendation Using Social Context

- Recommendation using social context alone
- Extending classical methods with social context
- Recommendation constrained by social context

Modeling Social Information in Recommendation

- The taste for user i is close to that of all his friends $j \in F(i)$

$$\sum_{i=1}^{n} \sum_{j \in F(i)} sim(i, j) \|U_i - U_j\|_F^2$$

- sim(i, j) denotes the similarity between user i and j (e.g., cosine similarity or Pearson correlation between their ratings) and F(i) denotes the friends of i

$$\min_{U,V} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{m} I_{ij} (R_{ij} - U_i^T V_j)^2 + \beta \sum_{i=1}^{n} \sum_{j \in F(i)} sim(i, j) \|U_i - U_j\|_F^2 + \frac{\lambda_1}{2} \|U\|_F^2 + \frac{\lambda_2}{2} \|V\|_F^2$$

Recommendation Constrained by Social Context

- In classical recommendation, to estimate ratings of an item, we determine similar users or items. In other words, any user similar to the individual can contribute to the predicted ratings for the individual.
- We can limit the set of individuals that can contribute to the ratings of a user to the set of **friends** of the user.
  - S(i) is the set of *k* most similar friends of an individual

$$r_{u,i} = \bar{r}_u + \frac{\sum_{v \in S(u)} sim(u,v)(r_{v,i} - \bar{r}_v)}{\sum_{v \in S(u)} sim(u,v)}$$

Predictive accuracy - Metrics measure error rate

- Mean Absolute Error (*MAE*) measures the average absolute deviation between a predicted rating (p) and the user's true rating (r)
  - NMAE = MAE/($r_{max} - r_{min}$)

$$MAE = \frac{\sum_{ij} |\hat{r}_{ij} - r_{ij}|}{n}$$

- Root Mean Square Error (*RMSE*) is similar to *MAE*, but places more emphasis on larger deviation

$$RMSE = \sqrt{\frac{1}{n} \sum_{i,j} (\hat{r}_{ij} - r_{ij})^2}$$

Relevance: Precision and Recall

- **Precision:** a measure of exactness, determines the fraction of relevant items retrieved out of all items retrieved

$$P = \frac{N_{rs}}{N_s}$$

- **Recall:** a measure of completeness, determines the fraction of relevant items retrieved out of all relevant items

$$R = \frac{N_{rs}}{N_r}$$

	Selected	Not Selected	Total
Relevant	N_{rs}	N_{rn}	N_r
Irrelevant	N_{is}	N_{in}	N_i
Total	N_s	N_n	N

Evaluating Ranking of Recommendation

- Spearman's Rank Correlation
  - $\rho = 1 - \frac{6 \sum_{i=1}^{n} (x_i - y_i)^2}{n^3 - n}$
- Kendall's τ
  - It checks the concordant the items of the recommended ranking list against the ground truth ranking list
    - If the two orders are consistent, it is concordant
    - For top 4 items in ranking list, there are 4*3/2=6 pairs
  - $\tau = \frac{c-d}{\binom{n}{2}}$

    where c is the number of concordants and d of disconcordants

Ranking, Example

Consider a set of four items I = {i_1 , i_2 , i_3 , i_4} for which the predicted and true rankings are as follows

	Predicted Rank	*True Rank*
i_1	*1*	*1*
i_2	*2*	*4*
i_3	*3*	*2*
i_4	*4*	*3*

Pair of items and their status {concordant/discordant} are

(i_1, i_2) : *concordant*

(i_1, i_3) : *concordant*

(i_1, i_4) : *concordant*

(i_2, i_3) : *discordant*

(i_2, i_4) : *discordant*

(i_3, i_4) : *concordant*

$$\tau = \frac{4-2}{6} = 0.33$$