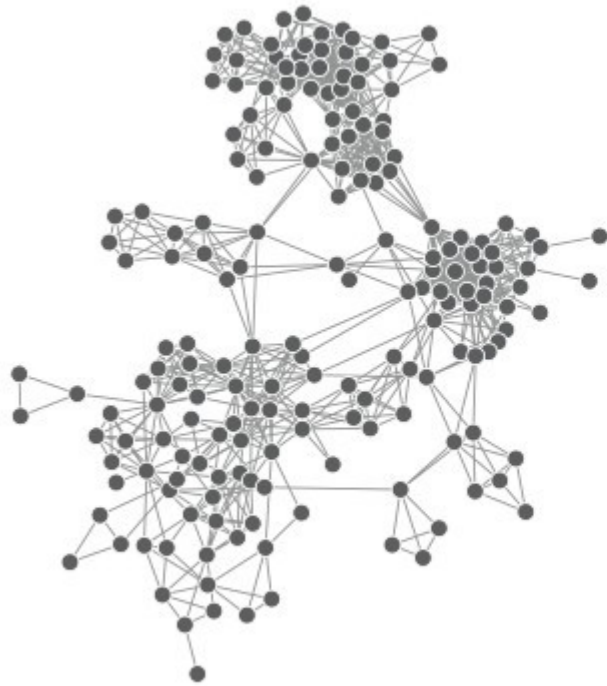


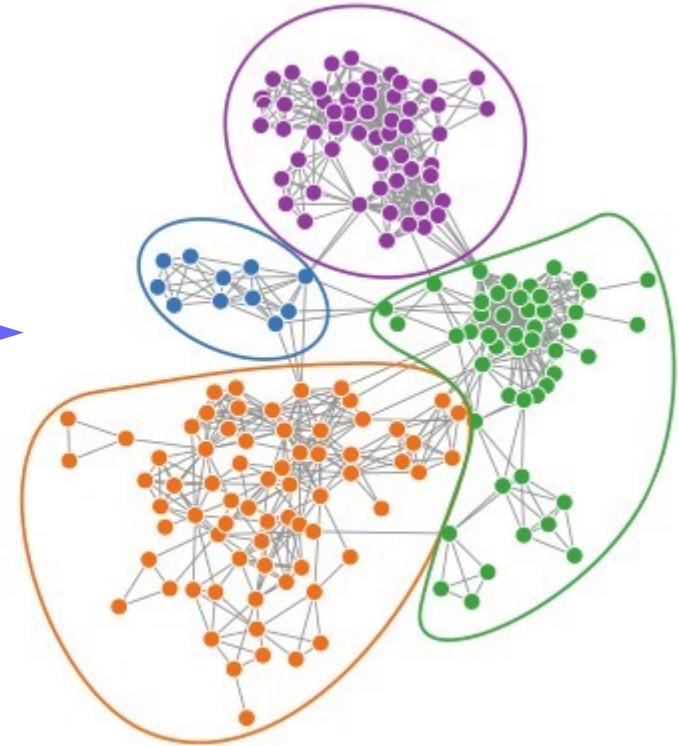
Community detection

Leto Peel
l.peel@maastrichtuniversity.nl
@PiratePeel

Community detection



Network



Communities

Supervised vs unsupervised

Supervised vs unsupervised

The supervised learner doesn't know much



Supervised vs unsupervised

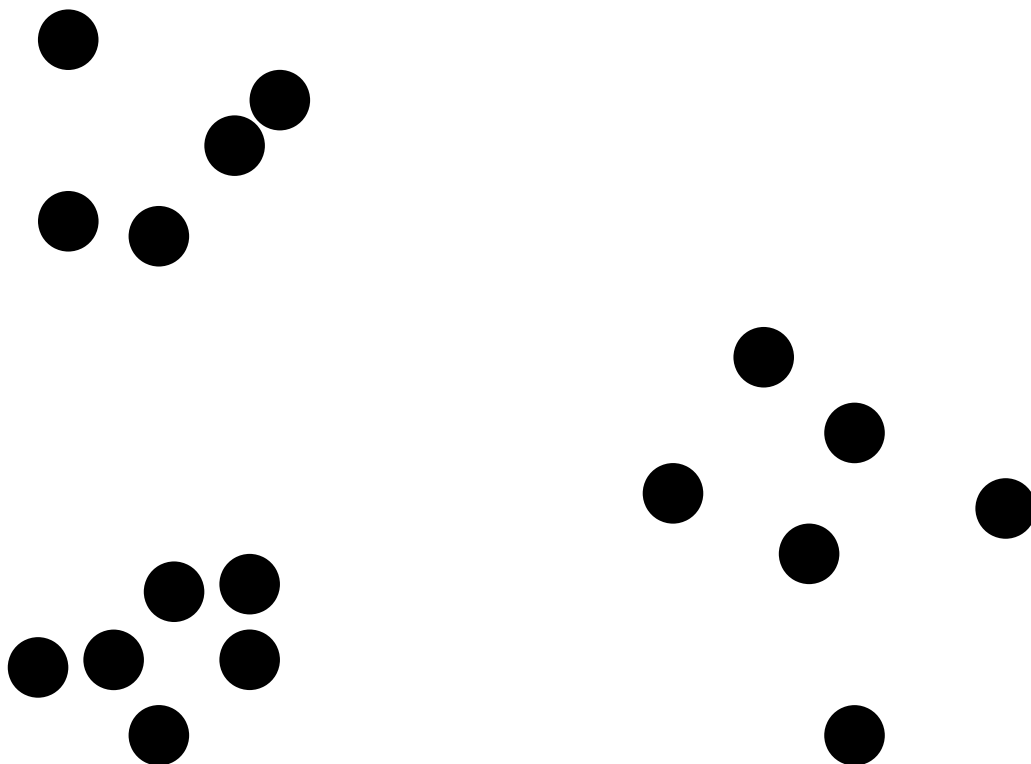
The supervised learner doesn't know much



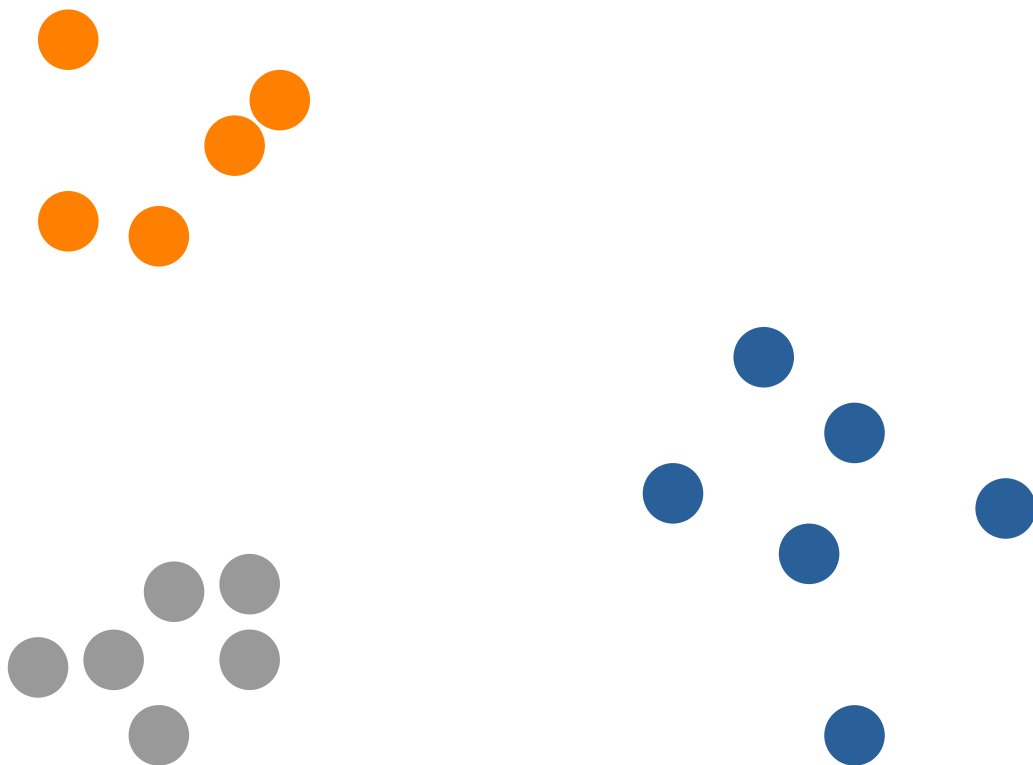
The unsupervised learner knows what it is doing



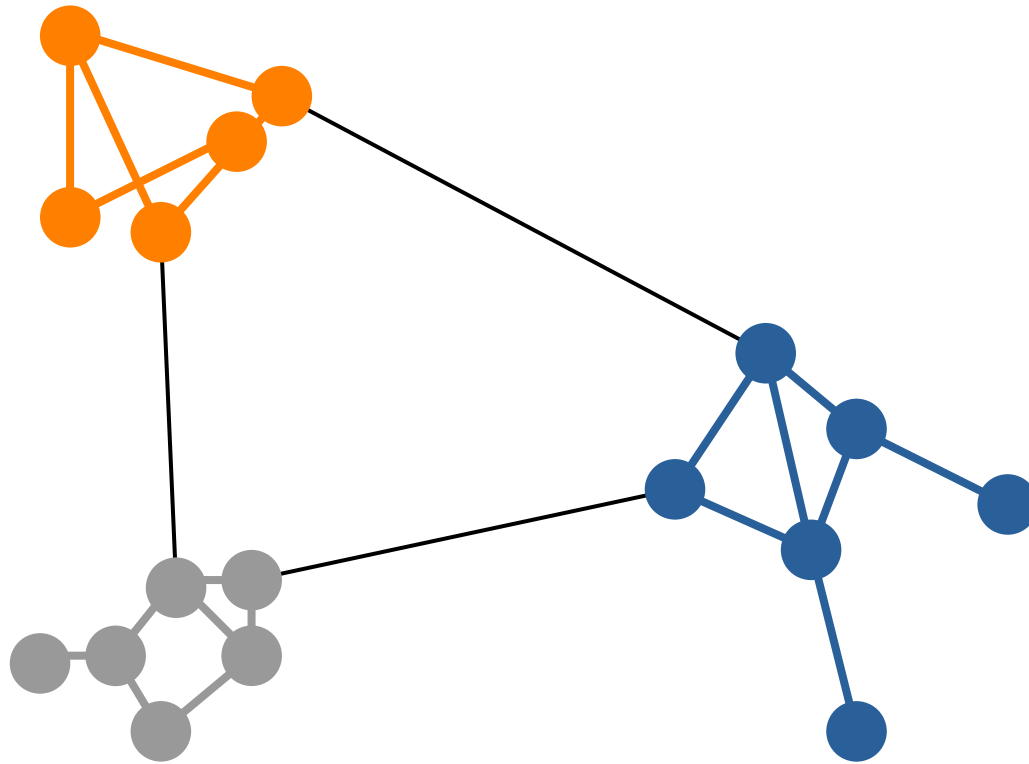
Clustering



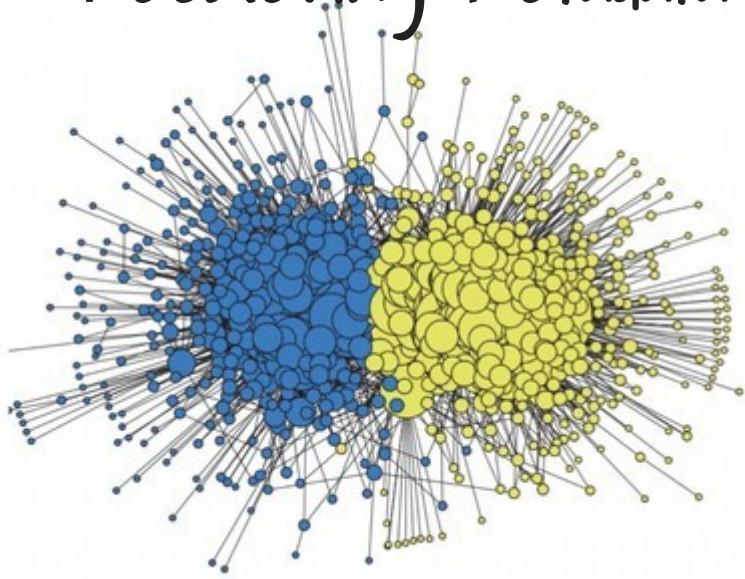
Clustering



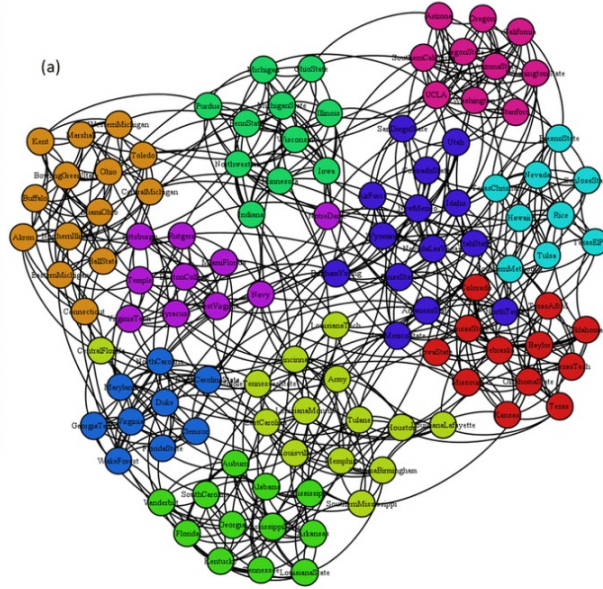
Community detection



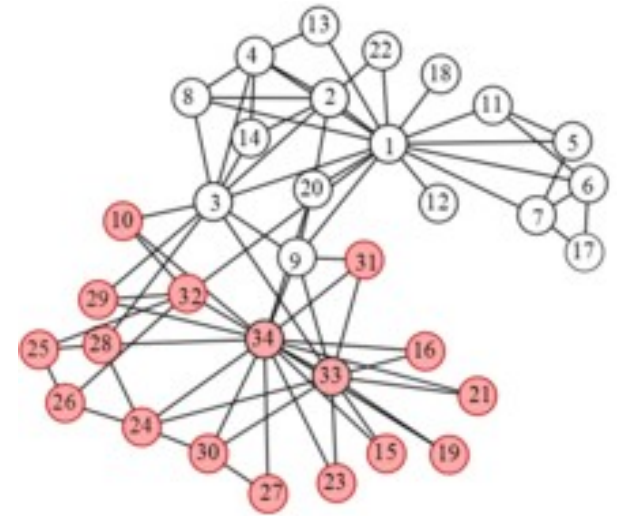
Recovering metadata implies sensible methods



US political blogs



American football



karate club*

*this network is so popular for community detection it has its own prize associated with it, see <http://networkkarate.tumblr.com/>

"Cluster" these objects



"Cluster" these objects



Red

Not Red



"Cluster" these objects



Cannot Fly

Can Fly



"Cluster" these objects



Transport



Not Transport

"Cluster" these objects



Not Alive



Alive

There are no ground truth communities!

Modularity

A partition better than random chance

$$Q = \sum_i (e_{ii} - a_i^2)$$

Modularity

A partition better than random chance

$$Q = \sum_i (e_{ii} - a_i^2)$$

Fraction of links
inside community i



Modularity


A partition better than random chance

$$Q = \sum_i (e_{ii} - a_i^2)$$

Fraction of links
inside community i



Expected fraction of
links in community i if
the network had no
community structure



Modularity

A partition better than random chance

$$a_i = \sum_j e_{ij}$$

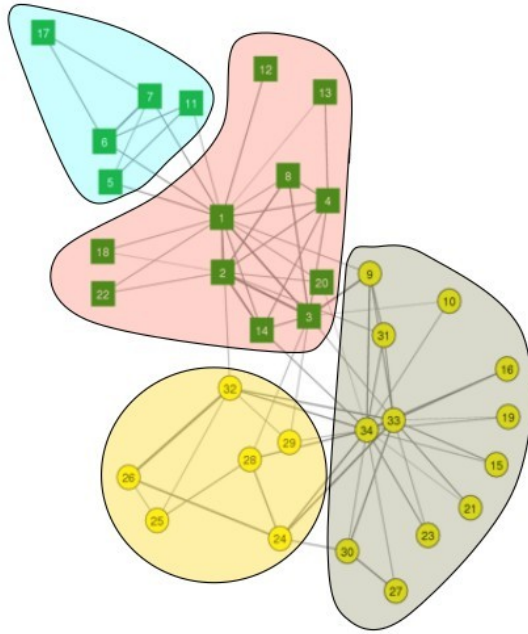
fraction of edges that
go to community i

$$Q = \sum_i (e_{ii} - a_i^2)$$

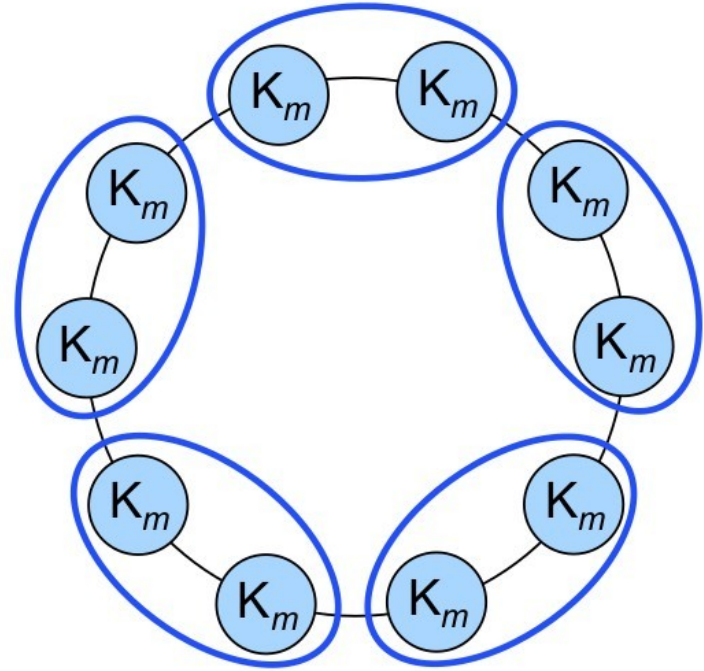
Fraction of links
inside community i

Expected fraction of
links in community i if
the network had no
community structure

Problems with modularity



Finds spurious communities
(overfitting)



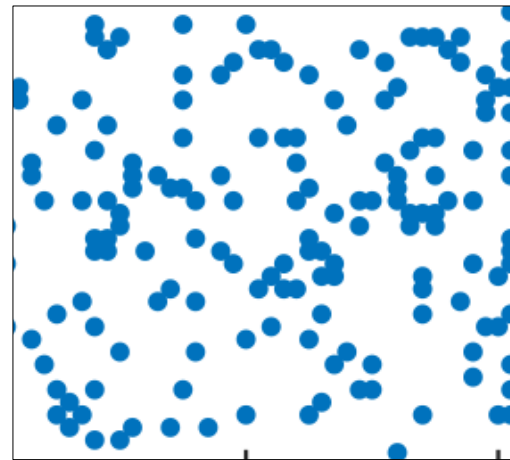
Resolution limit
(underfitting)

Practical Q1 and Q2

Erdos-Renyi random graphs

Flip biased coin for every node pair

$$\Pr(A|\theta) = \prod_{ij} \theta^{A_{ij}} (1 - \theta)^{(1-A_{ij})}$$



Adjacency Matrix

Heads or Tails?



Heads or Tails?



[01010100100000100011111101100]

is just as likely as:

[000000000000000000111111111111]

What is the likelihood that we will observe this sequence of events?

$\mathbf{s} = [01010100100000100011111101100]$

What is the likelihood that we will observe this sequence of events?

$\mathbf{s} = [01010100100000100011111101100]$

$$\Pr(s_i = 1) = \theta = 0.5$$

$$(1-0.5) * 0.5 * (1-0.5) * 0.5 * (1-0.5) * 0.5 * (1-0.5) * (1-0.5) \dots$$

What is the likelihood that we will observe this sequence of events?

$\mathbf{s} = [010101001000001000111111101100]$

$$\Pr(s_i = 1) = \theta = 0.5$$

$$(1-0.5) * 0.5 * (1-0.5) * 0.5 * (1-0.5) * 0.5 * (1-0.5) * (1-0.5) \dots$$

$$\Pr(s|\theta) = \prod_i \theta^{s_i} (1 - \theta)^{(1-s_i)}$$

What is the probability of heads?



$$\theta_{\text{ML}} = \max_{\theta} \Pr(s|\theta)$$

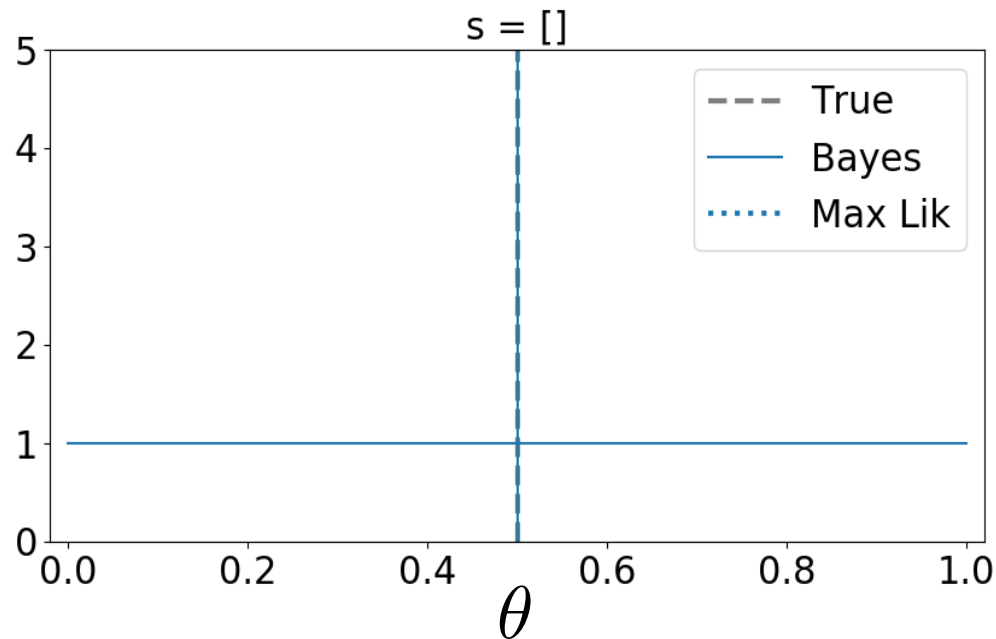
$$\theta_{\text{Bayes}} \propto \Pr(s|\theta)\Pr(\theta)$$

Probabilistic generative model

What is the probability of heads?



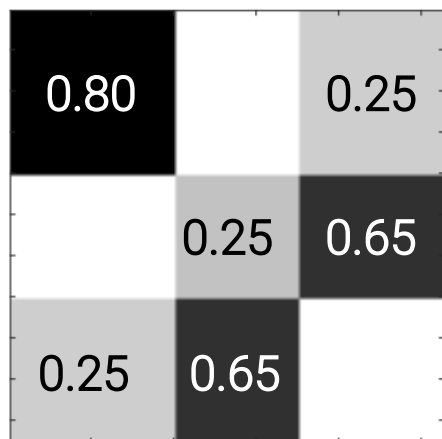
Probabilistic generative model



"I've never seen a coin before"

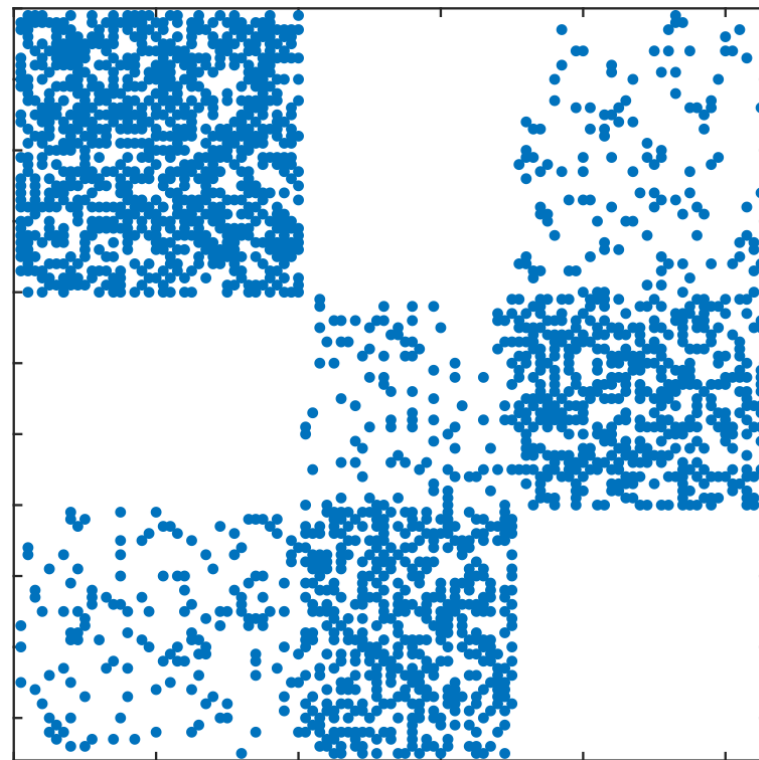
The stochastic block model

Just a bunch of coins!



Mixing Matrix

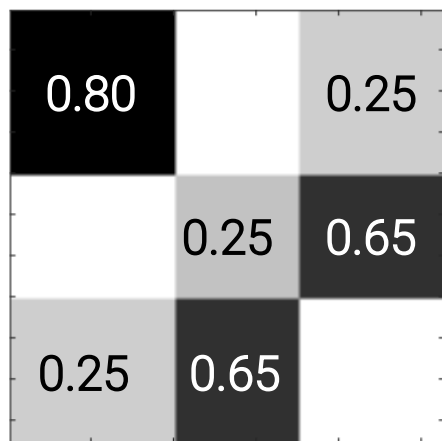
generation



Adjacency Matrix

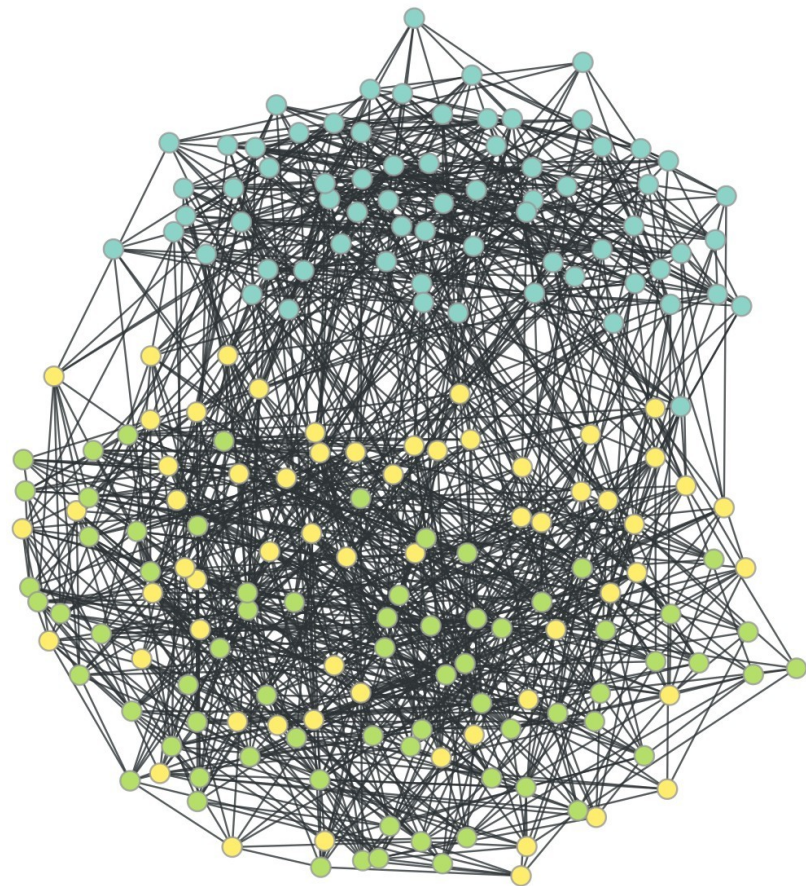
The stochastic block model

Just a bunch of coins!



Mixing Matrix

generation



The stochastic block model

Just a bunch of coins!

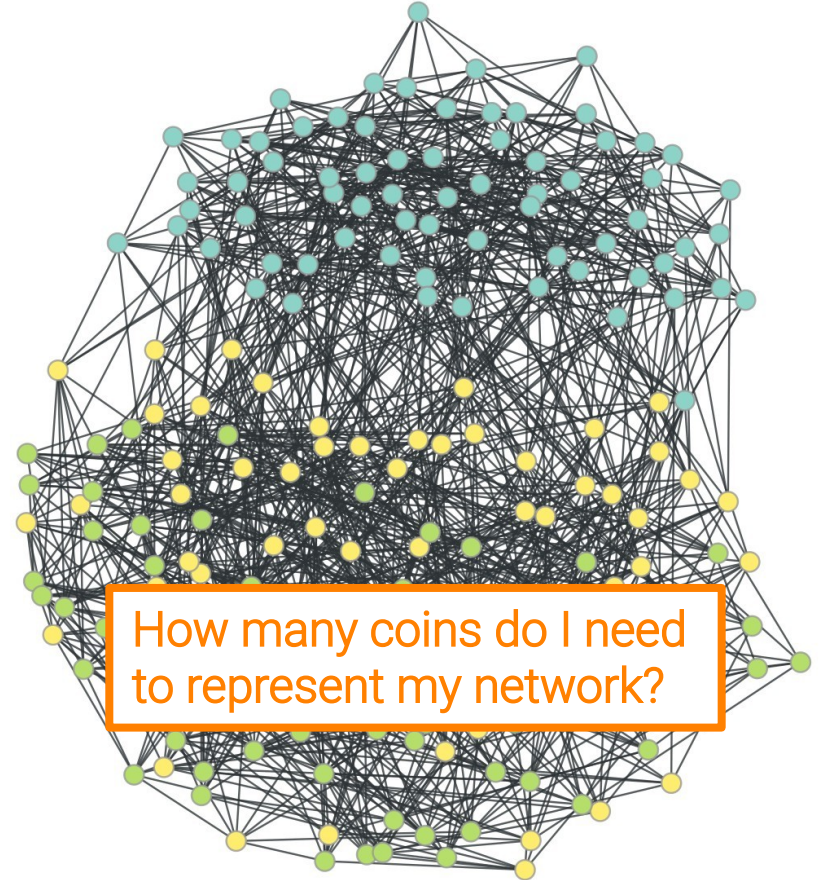


Mixing Matrix

generation



inference



How many coins do I need to represent my network?

Minimum description length

Description length
(entropy)

$$\Sigma = -\ln P(\mathbf{A}|\boldsymbol{\theta}, \mathbf{b}) - \ln P(\boldsymbol{\theta}, \mathbf{b})$$

Minimum description length

Description length
(entropy)

$$\Sigma = - \ln P(\mathbf{A}|\boldsymbol{\theta}, \mathbf{b}) - \ln P(\boldsymbol{\theta}, \mathbf{b})$$

data description
length

Minimum description length

Description length
(entropy)

$$\Sigma = \underbrace{-\ln P(\mathbf{A}|\boldsymbol{\theta}, \mathbf{b})}_{\text{data description length}} - \underbrace{\ln P(\boldsymbol{\theta}, \mathbf{b})}_{\text{model description length}}$$

Minimum description length

Description length
(entropy)

$$\Sigma = - \underbrace{\ln P(\mathbf{A}|\boldsymbol{\theta}, \mathbf{b})}_{\substack{\text{data description} \\ \text{length}}} - \underbrace{\ln P(\boldsymbol{\theta}, \mathbf{b})}_{\substack{\text{model description} \\ \text{length}}}$$

likelihood prior

The diagram illustrates the Minimum Description Length (MDL) principle. It features the equation $\Sigma = - \ln P(\mathbf{A}|\boldsymbol{\theta}, \mathbf{b}) - \ln P(\boldsymbol{\theta}, \mathbf{b})$. The first term, $-\ln P(\mathbf{A}|\boldsymbol{\theta}, \mathbf{b})$, is annotated with 'likelihood' above it and 'data description length' below it. The second term, $-\ln P(\boldsymbol{\theta}, \mathbf{b})$, is annotated with 'prior' above it and 'model description length' below it. To the left of the equation, the text 'Description length (entropy)' is present. Orange lines connect the labels to the corresponding parts of the equation.

Practical Q3

Many extensions and
applications...

Link prediction

Network reconstruction

Many extensions

Many extensions and applications...

Link prediction

Network reconstruction

Many extensions

- degree correction
- mixed membership
- hierarchical
- edge weights/types
- node metadata
- temporal models

Many extensions and applications...

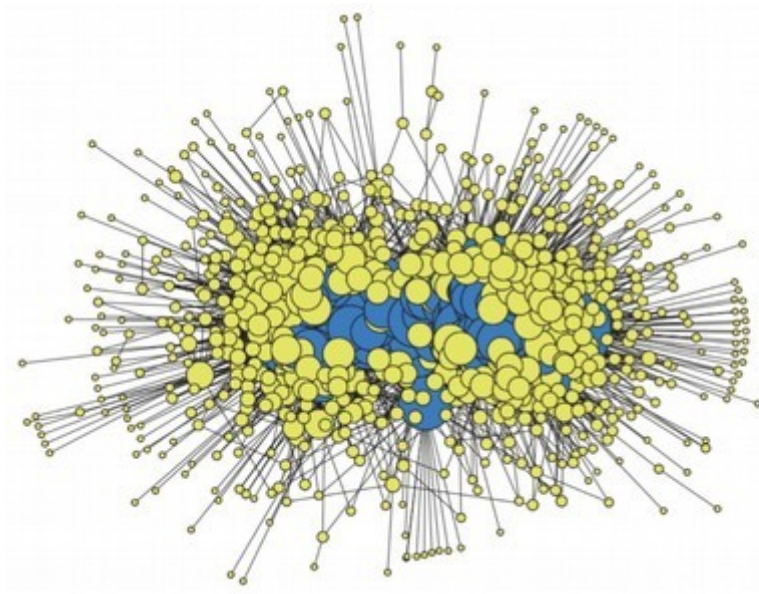
Link prediction

Network reconstruction

Many extensions

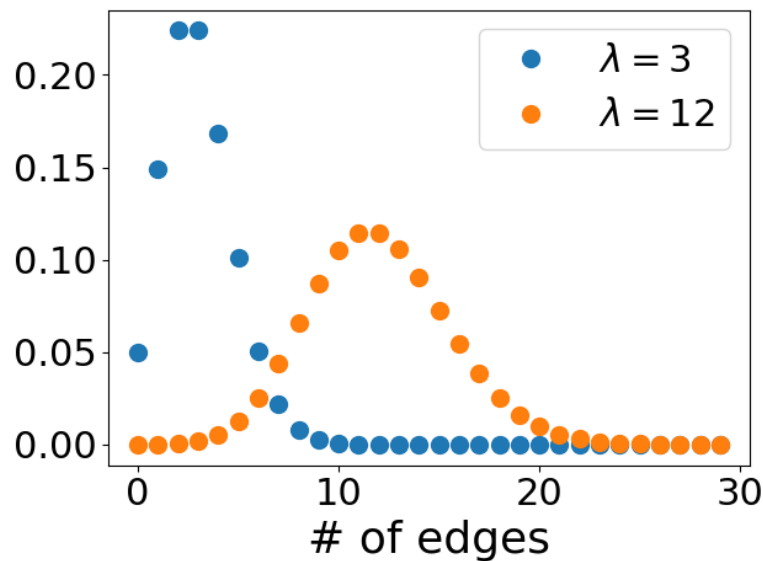
- degree correction
- mixed membership
- **hierarchical**
- edge weights/types
- node metadata
- temporal models

Degree-corrected SBM

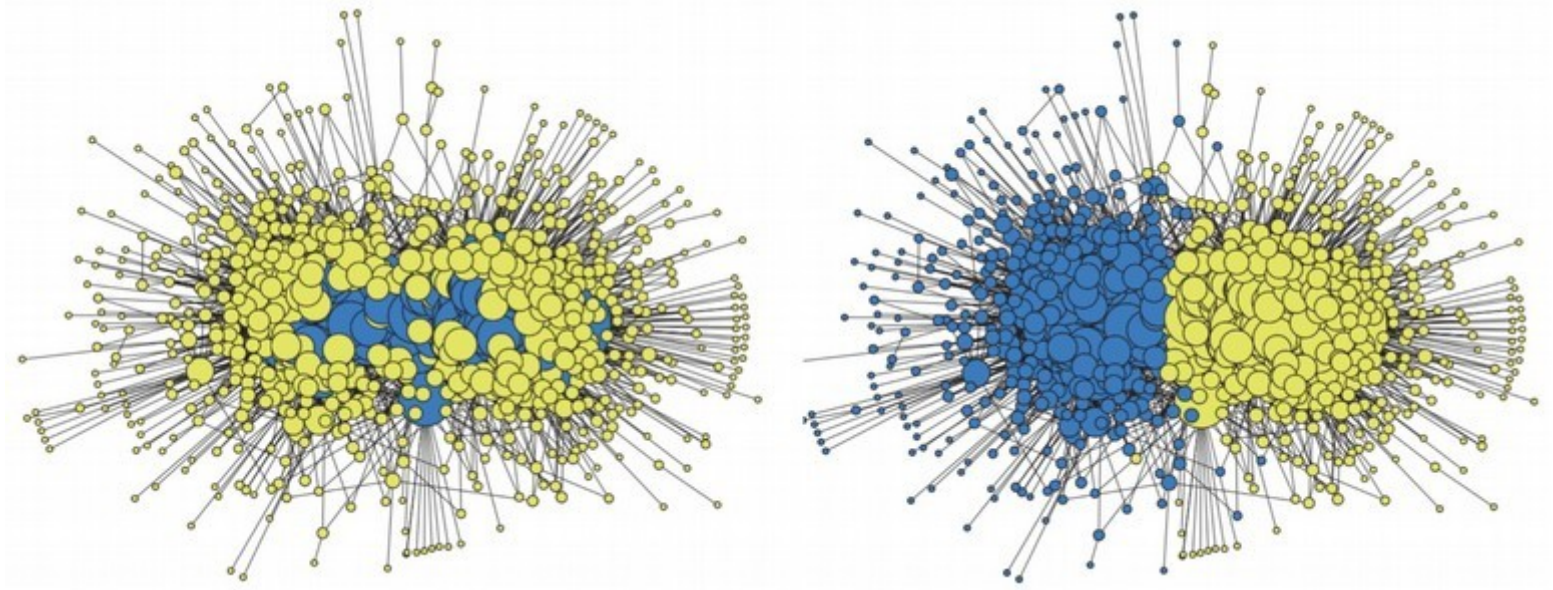


stochastic block model

SBM assumes Poisson distributed degree



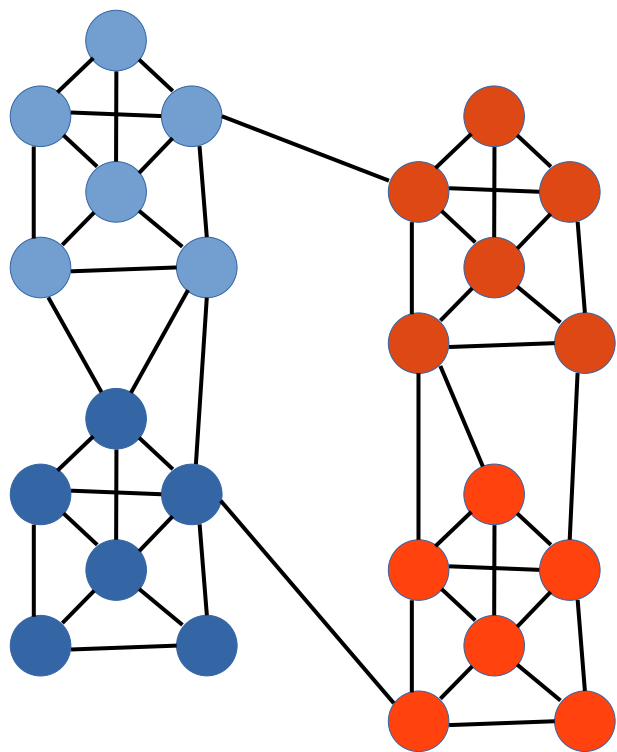
Degree-corrected SBM



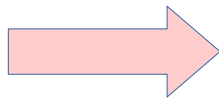
stochastic block model

Karrer, Newman. Stochastic blockmodels and community structure in networks. Phys. Rev. E 83, 016107 (2011).
Adamic, Glance. The political blogosphere and the 2004 US election: divided they blog. 36–43 (2005).

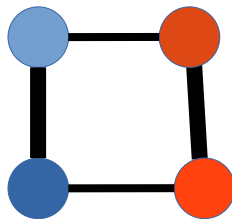
Building the hierarchy



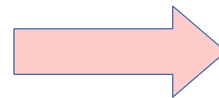
Observed network



infer
communities



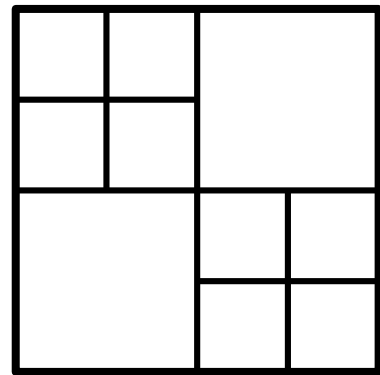
Multigraph

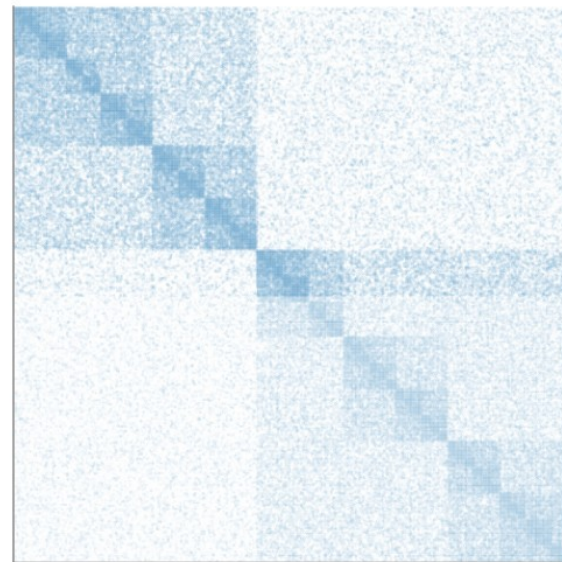
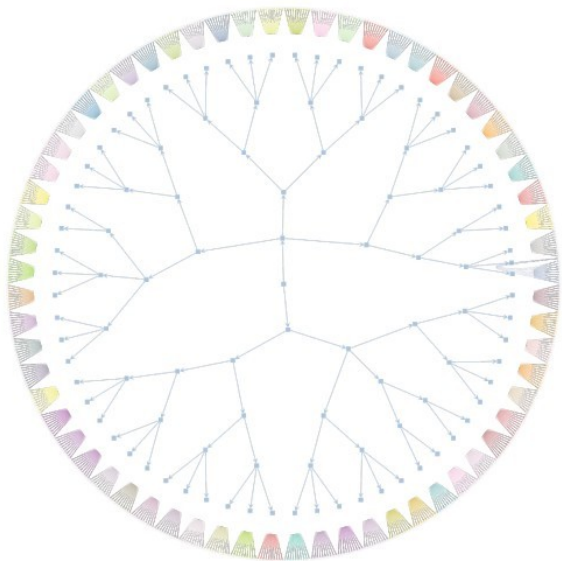


infer
communities



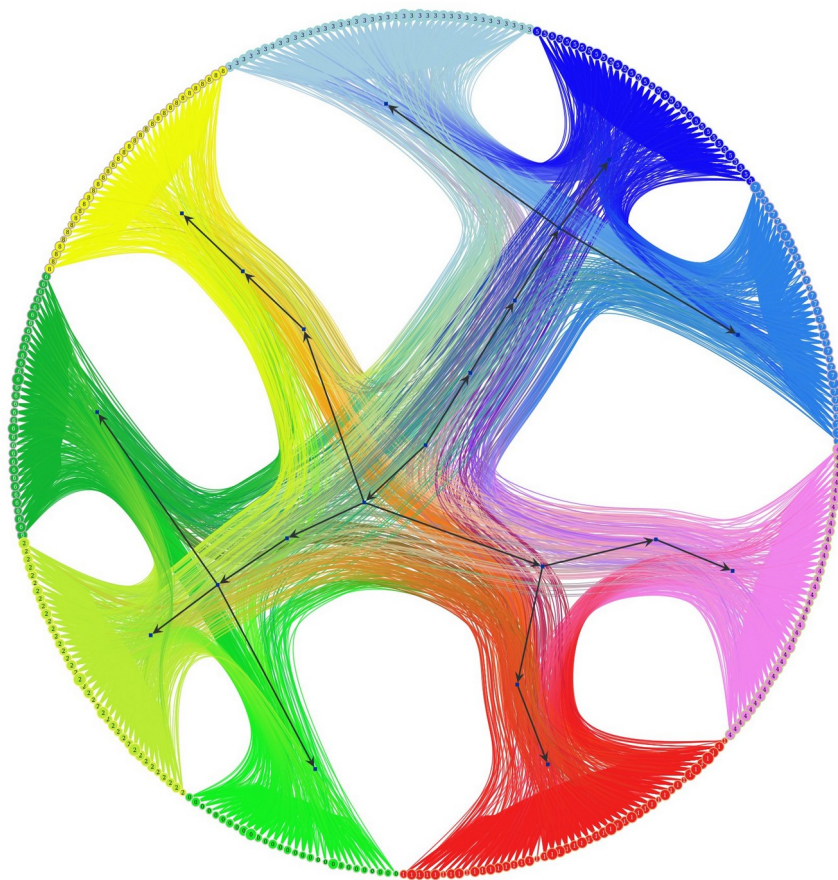
Multigraph



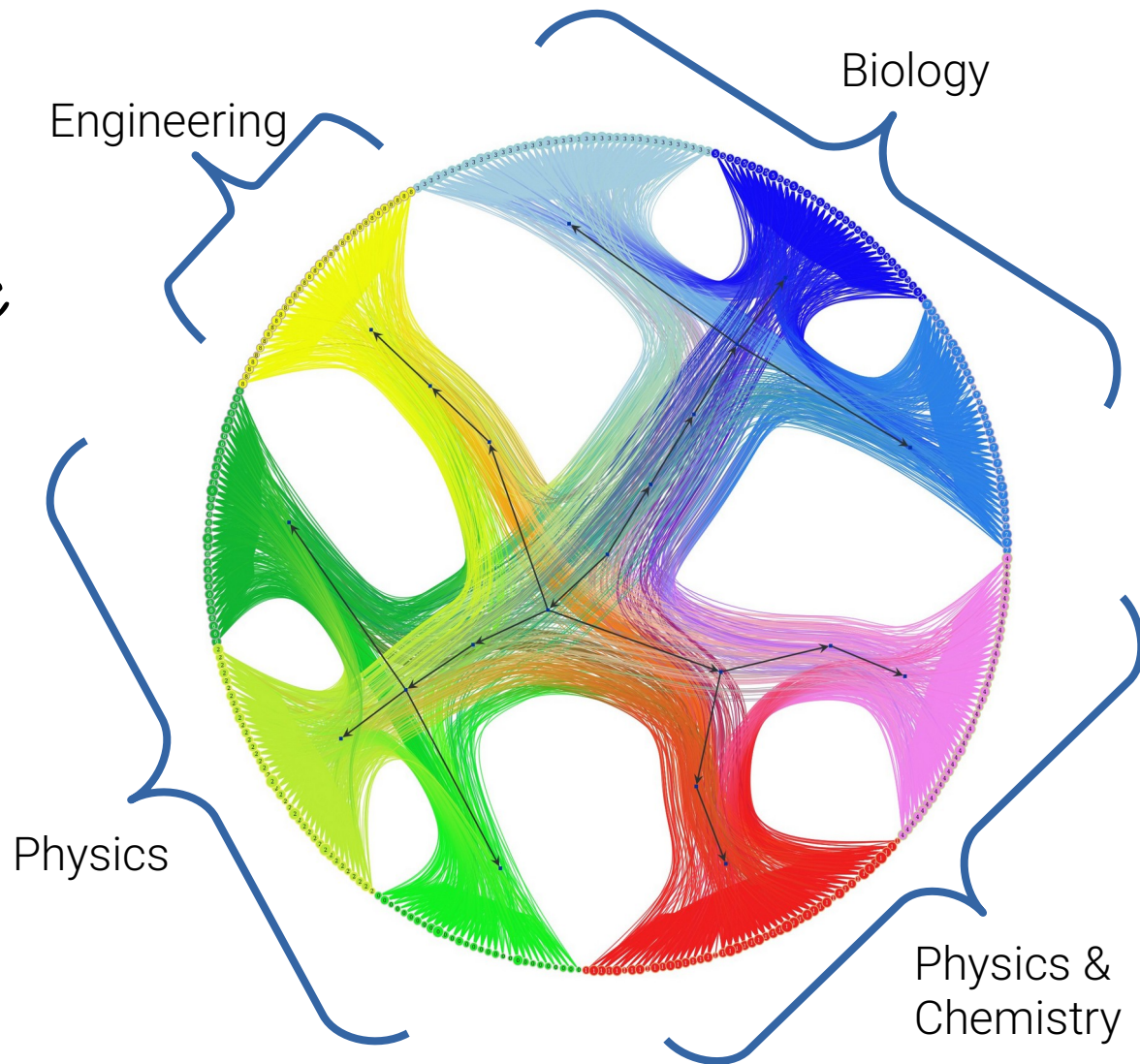


Adjacency matrix with hierarchy

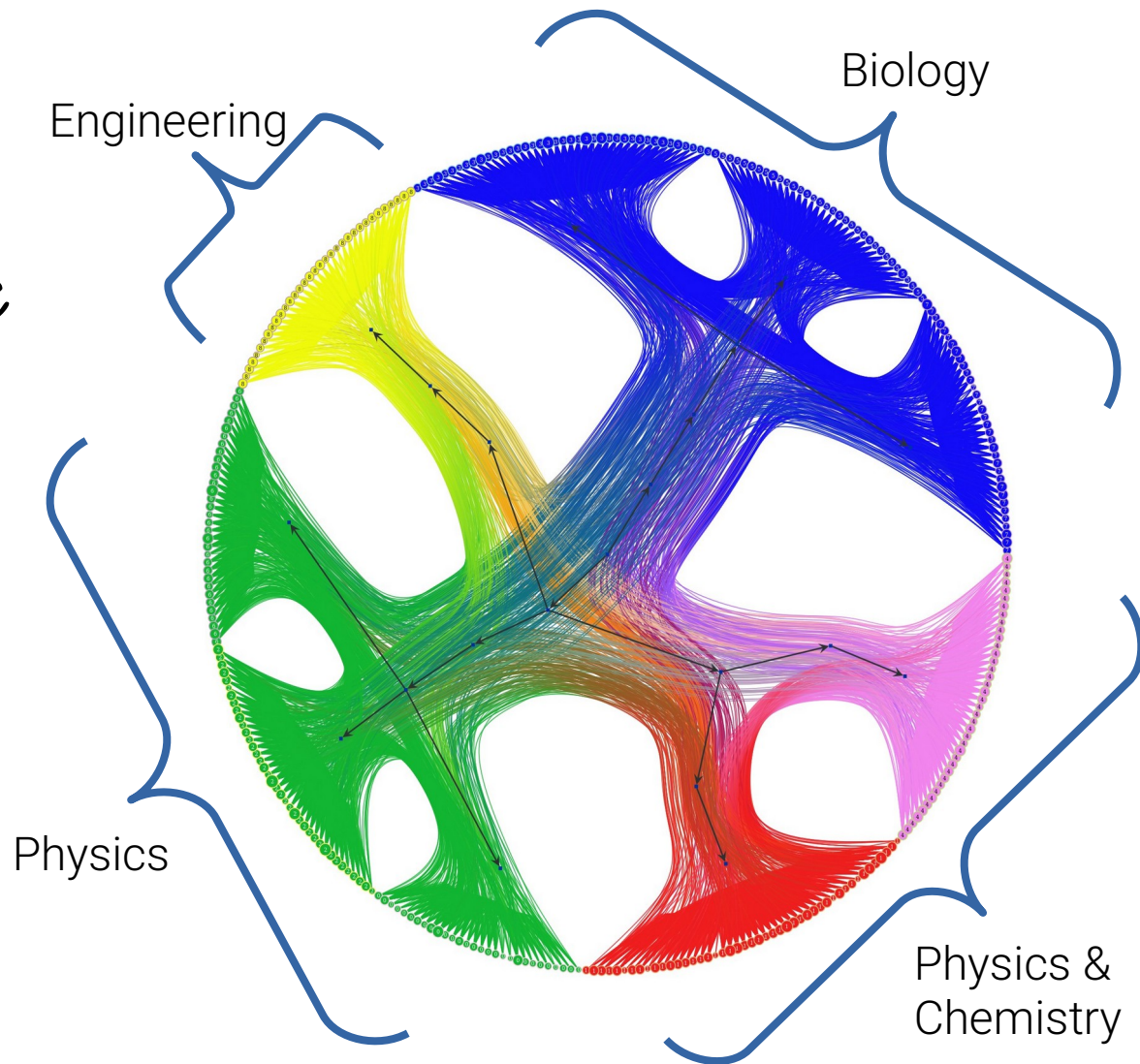
Face-to-face
contacts



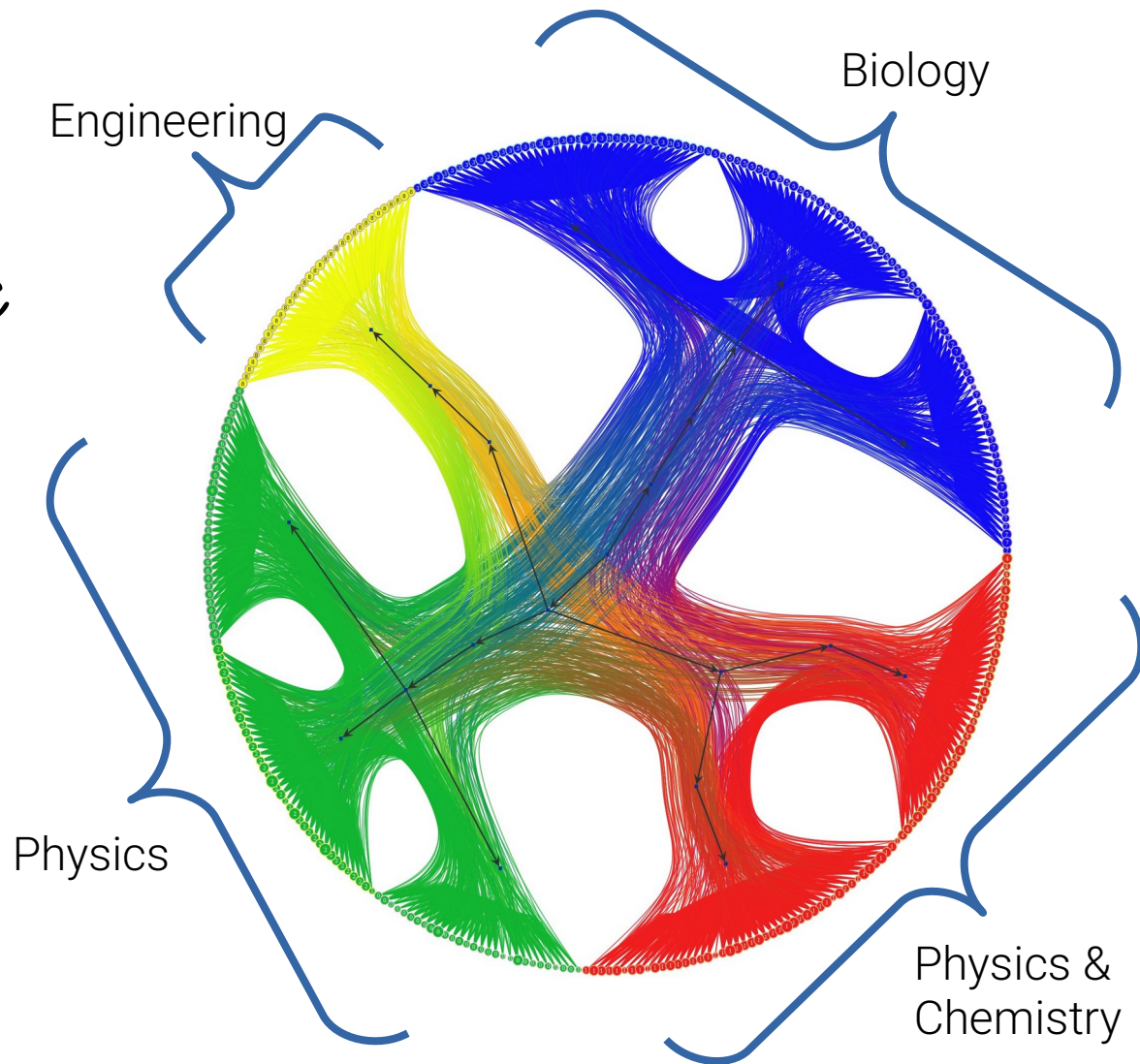
Face-to-face
contacts



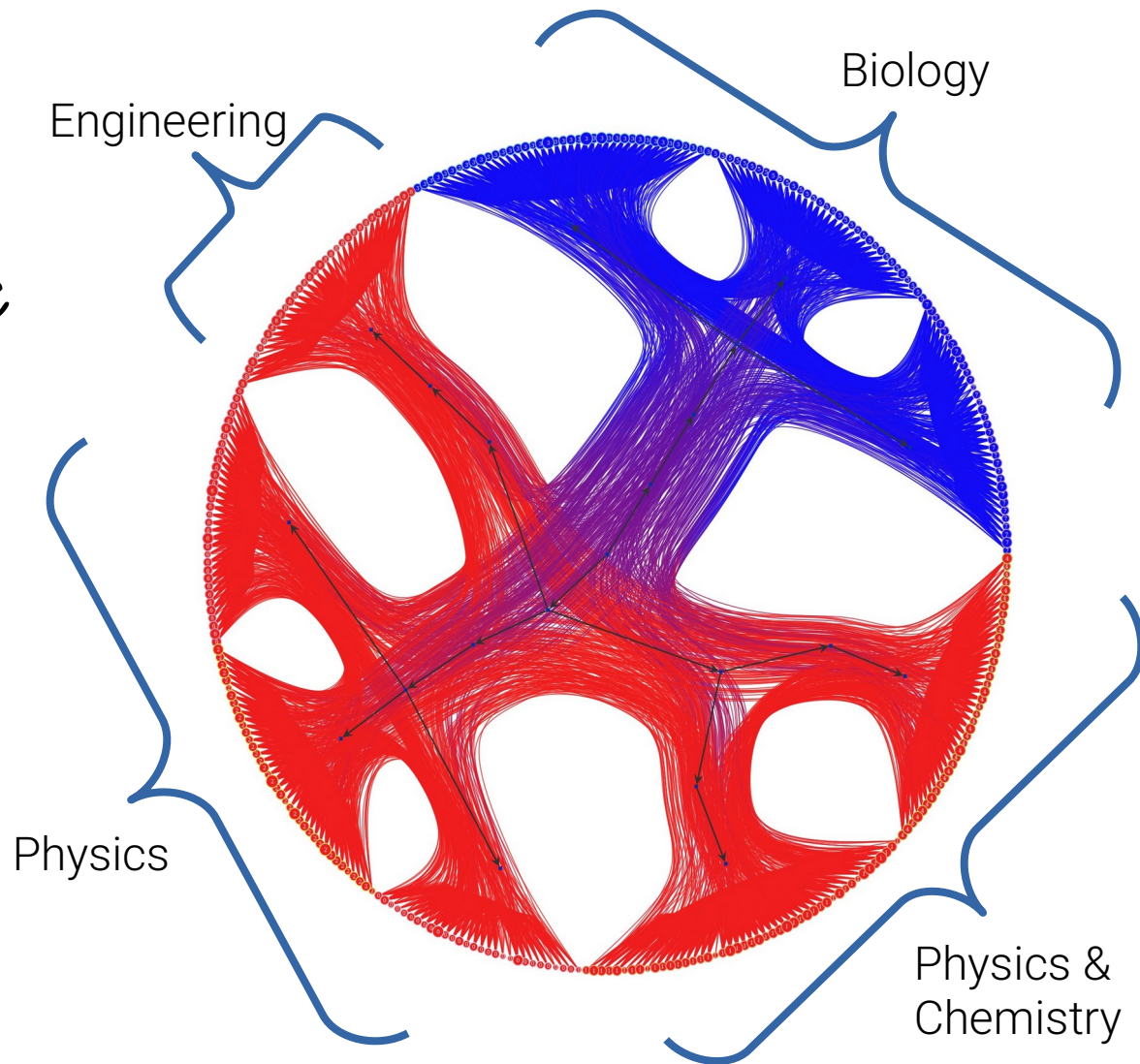
Face-to-face
contacts



Face-to-face
contacts



Face-to-face
contacts

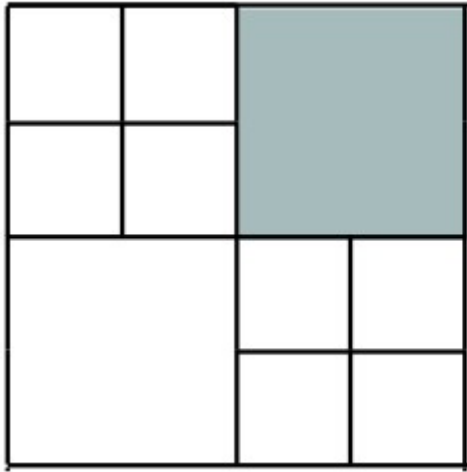


Practical Q4 and Q5

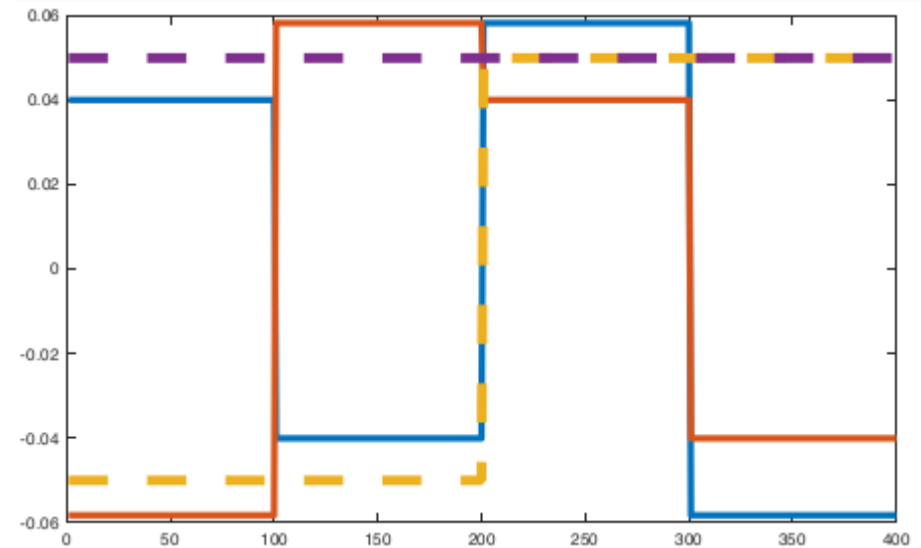
Spectral clustering

Spectral properties

$$\mathbb{E}[A]$$

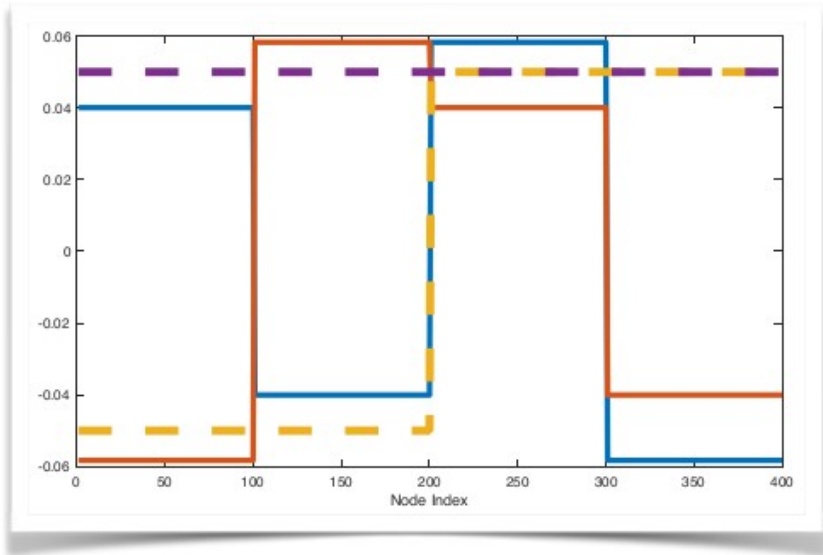


First 4 Eigenvectors of the Laplacian

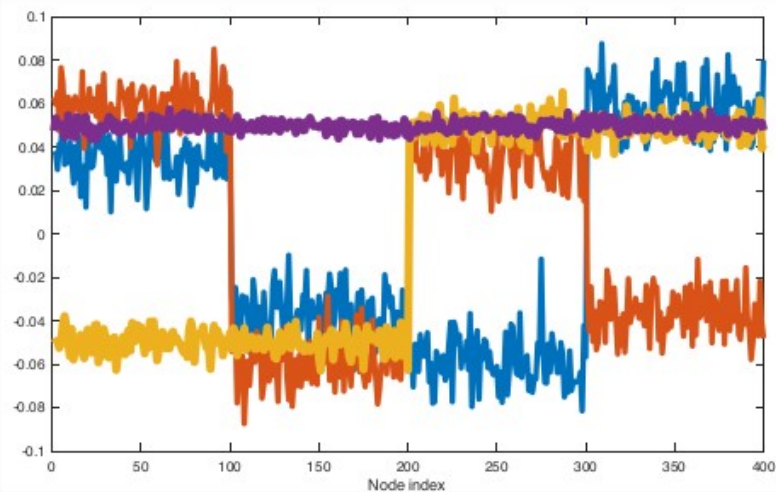
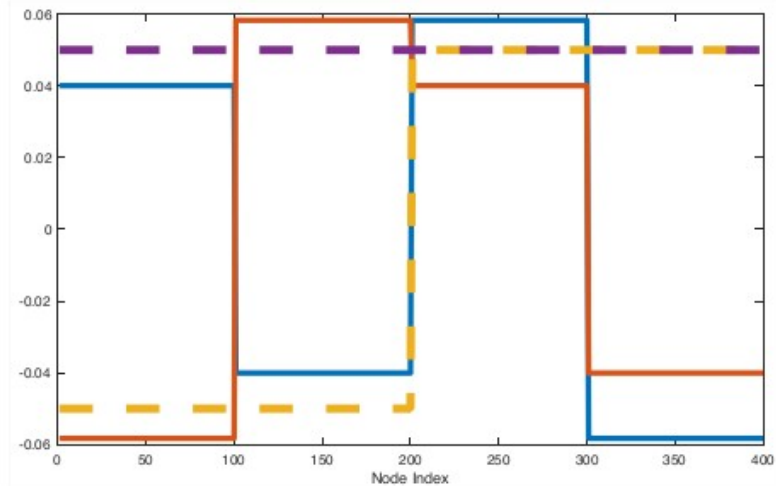


Node index

If we could just “see” the expected adjacency matrix,
then we could just look for constant eigenvectors



If we could just “see” the expected adjacency matrix,
then we could just look for constant eigenvectors



K-means clustering

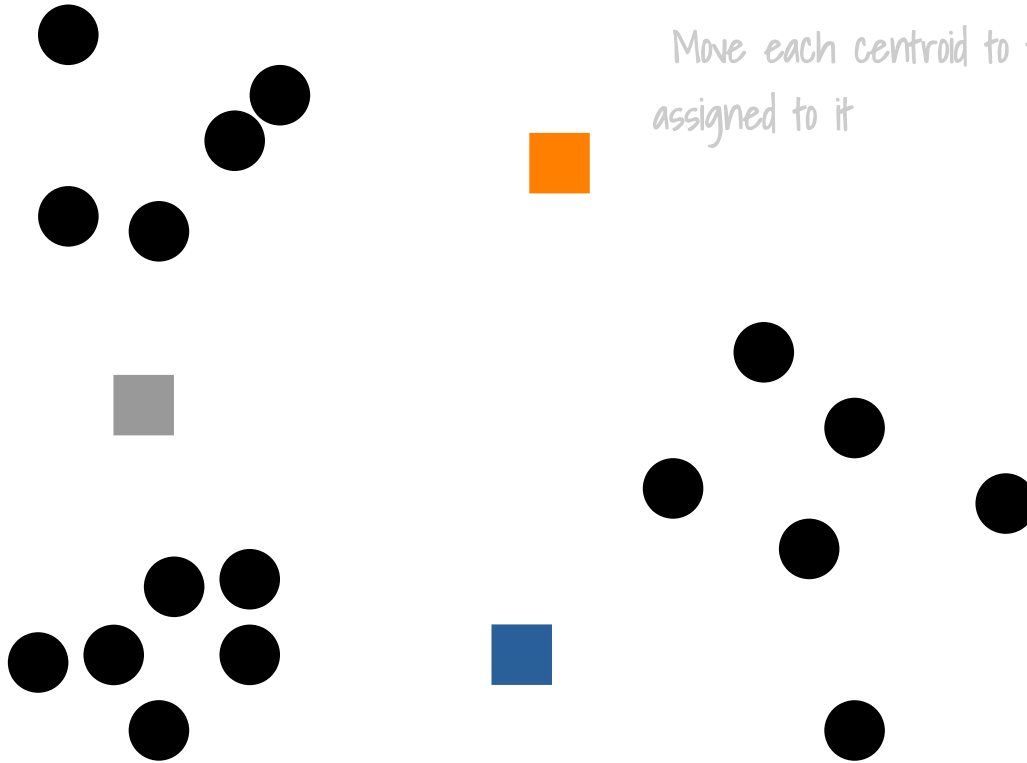
- 1) Randomly initialise centroids (one per cluster)
- 2) Iterate:
 - a) Assign each data point to the nearest centroid
 - b) Move each centroid to the mean of the data points assigned to it

Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

Move each centroid to the mean of the data points assigned to it

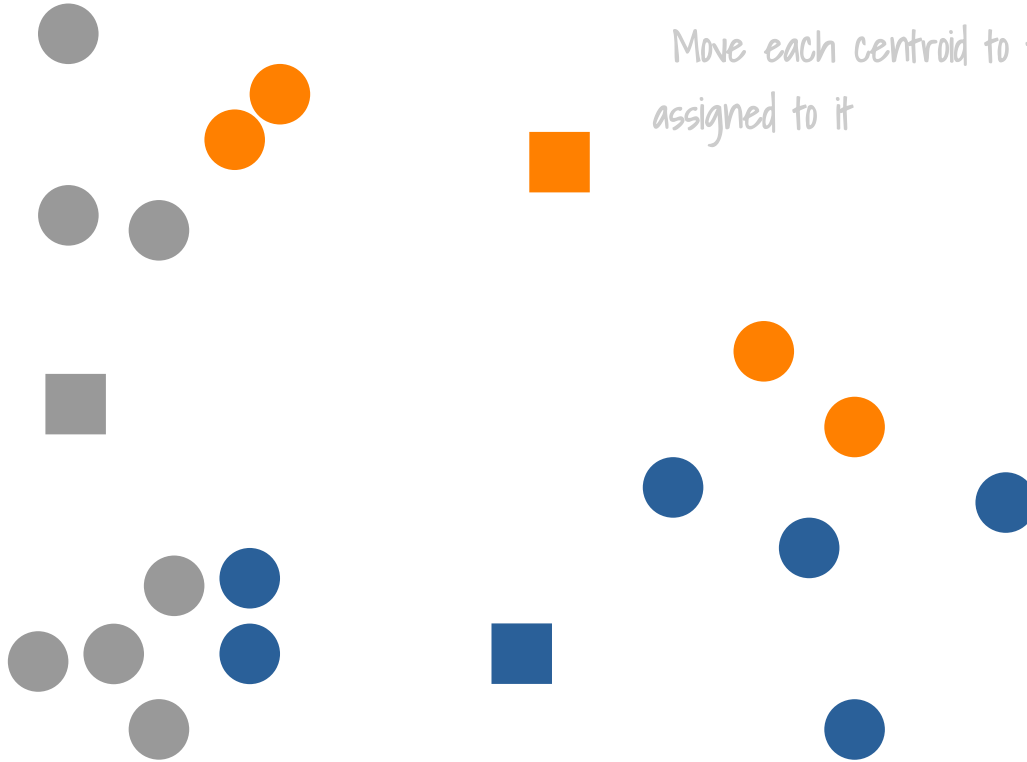


Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

Move each centroid to the mean of the data points assigned to it

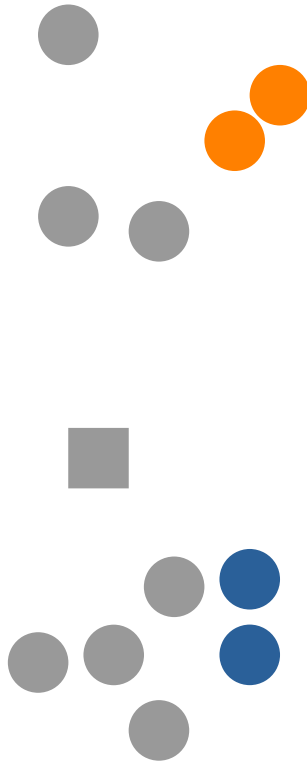


Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

Move each centroid to the mean of the data points assigned to it

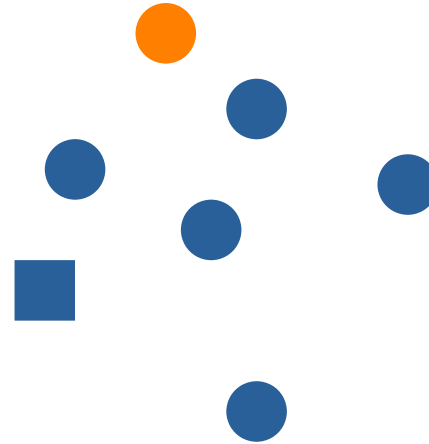
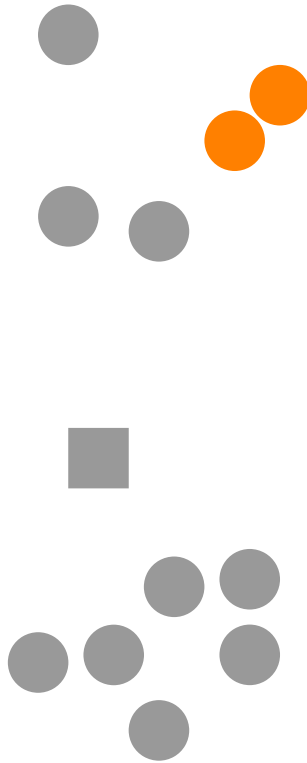


Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

Move each centroid to the mean of the data points assigned to it

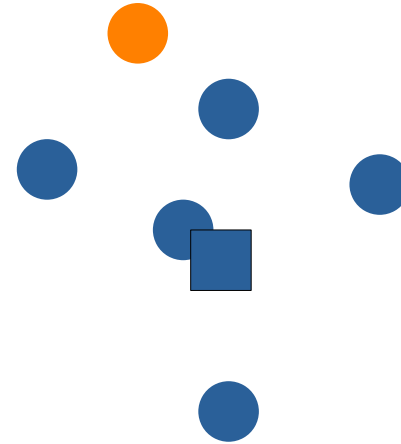
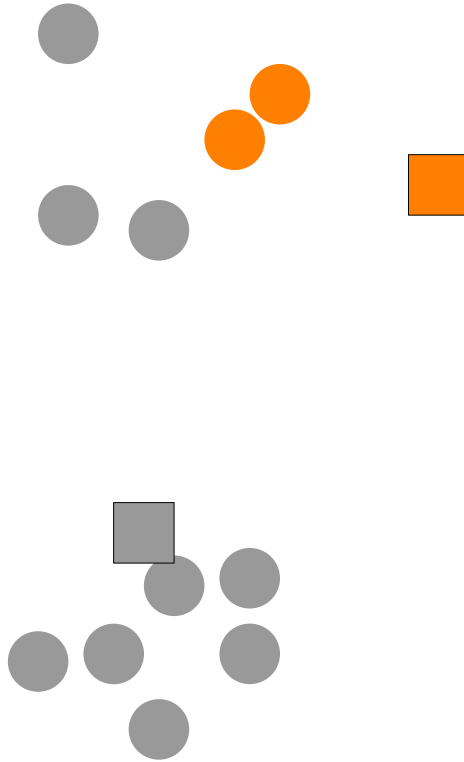


Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

Move each centroid to the mean of the data points assigned to it

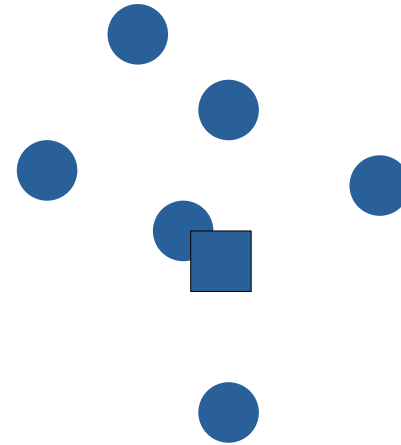
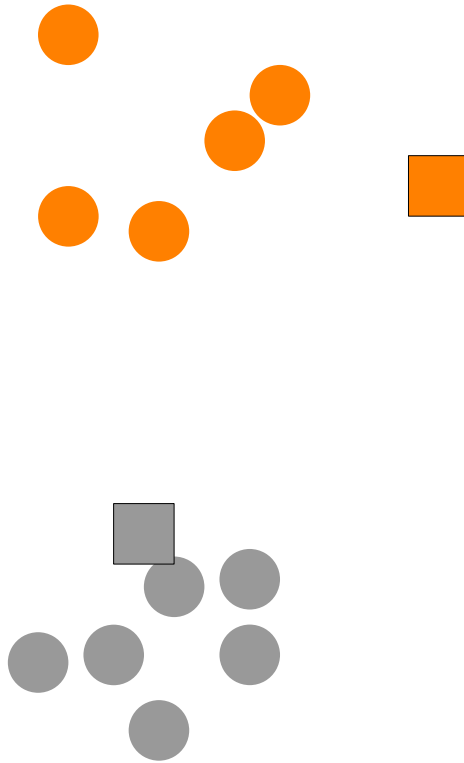


Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

Move each centroid to the mean of the data points assigned to it

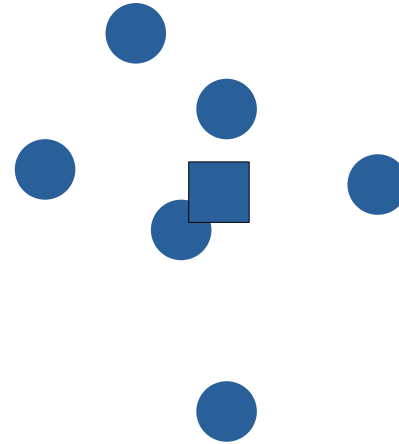


Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

Move each centroid to the mean of the data points assigned to it



Summary...

Summary...

Community detection finds nodes with similar connectivity patterns

- Nodes often have similar properties or functions

Summary...

Community detection finds nodes with similar connectivity patterns

- Nodes often have similar properties or functions

There can be multiple good ways to partition a network

Summary...

Community detection finds nodes with similar connectivity patterns

- Nodes often have similar properties or functions

There can be multiple good ways to partition a network

It's unsupervised!

- Your model must know how to partition the network