[6] Big Data: Networks

Matt Taddy, University of Chicago Booth School of Business

faculty.chicagobooth.edu/matt.taddy/teaching

[6] Networks

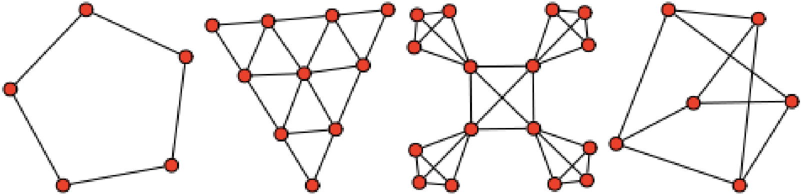
Graphical models provide a language for networks:
0/1 connections between people/sites/covariates.
Think of graphs like a binary version of correlation.

Graph Structure:

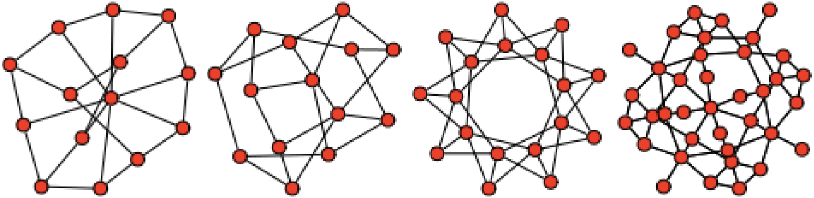
- Summarization: nodes and edges, direction.
- Measuring connectivity and betweenness
- Page Rank for relevance ordering.

Association and networks

- Market Basket Analysis



The network has nodes (vertices), such as a website or worker, and edges are the (directed or undirected) links between nodes.



Network data is connected

A network consists of variables and connections between them.
A connection is discrete: it's either there or it's not.

Data living on a network:

- Word usage in text and language (what words follow?)
- Organization charts and employment (who's boss?)
- Business credit, supply, and competition networks.
- Genes: many SNPs pop if and only if another does.
- Everything on the internet!

Sometimes the network is given,
other times we just glimpse *traffic* on the network.

Graph Models and Network Structure

Start with a given network: you see all connections.

We'll reduce dimension + summarize important properties.

In particular, we'll focus on measures of network connectivity.

Each node has connectivity statistics

Degree: How many other nodes are you connected to?

Betweenness: How many node-to-node paths go through you?

You can also make a lot of cool illustrations for graphs.
These tend to be more pretty than informative,
but that doesn't mean they aren't useful.

Network Graphs in R

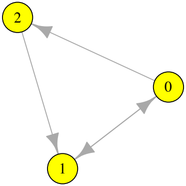
`igraph` is a toolbox for visualizing and summarizing graphs.
It has front-ends for `R` and `Python`. Others: Gephi, Pajek, etc.

Unlike most R packages, `igraph` is well documented.
Type `help(igraph)` to get started.

For most applications, you'll read graphs from an edgelist:

```
0  1
0  2
1  0
2  1
```

$\Rightarrow$




```
edgemat <- as.matrix(read.table("edgelist.txt"))
graph <- graph.edgelist(my_edgelist)
```

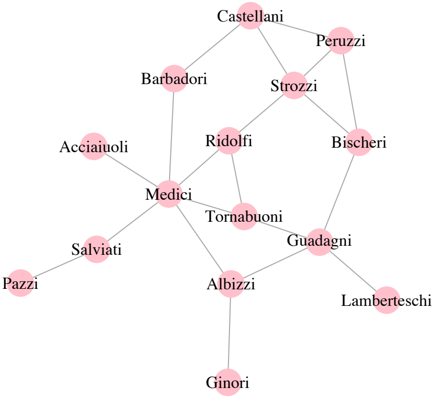
Marriage and Power

Early Renaissance Florence was ruled by an oligarchy of powerful families.

By the 15th century, the Medicis emerged supreme, & Medici Bank became the largest in Europe.

Political ties were established via marriage.
How did Medici win?

Marriage in Florence: 1250-1450



Network links can be used to measure “social capital”.

A node’s degree is its number of edges.

```
> sort(degree(marriage))
Ginori ... Strozzi Medici
     1           4      6
```

Medicis are connected!

This is good enough for many analysis...

Deeper network structure with betweenness

An alternative to degree, betweenness measure the proportion of shortest paths containing a given node.

Shortest path: fewest steps from i to j (direction matters).

Say $s_k(i,j)$ is the proportion of shortest paths from i to j containing node k.

$$\text{betweenness}(k) = \sum_{i,j:i \neq j, k \notin \{i,j\}} s_k(i,j)$$

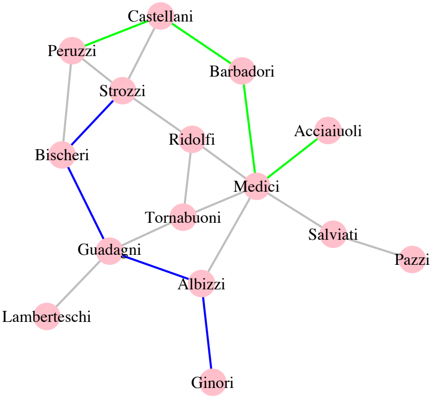
This measures how much influence a node has over connections between others.

Betweenness vs Degree

Medicis have the highest degree, but only by a factor of 3/2 over the Strozzis.

But their betweenness is 5 times higher!

Betweenness measures deep graph connectivity, rather than just counting neighbors.




```
> sort(betweenness(marriage))
 Ginori   ...   Strozzi  Medici
    0.0             9.3    47.5
```

Structural Holes

A structural hole is a low-level node in
an organization chart with high betweenness.

Social Capital in brokerage opportunities.
The name seems pejorative, because it is.

Holes can act like bottlenecks in companies, and lead to unexpected employees having excess power and influence.

But if you’re the employee, it’s a fast track to promotion!

Burt: *Structural Holes and Good Ideas*, AJS 2004.
`igraph` has `constraint` for finding structural holes.

Collaborative Filtering

A common question in data mining:
what do one person's choices say about anothers?
As amazon says: "people who buy this book also bought..."

These types of tasks are referred to as 'collaborative filtering':
using shared choices to predict preferences.

It's a big field, with many tools

- logistic regression of each product on to all other choices.
- principal componenets analysis: underlying taste factors.

Many of the tools from this class apply (projects?).

But as an easy start, there are good fast algorithms
for discovering low dimensional *association rules*.

Association Rules

Consider two binary variables: $x_a + x_b$.

If $x_b = 1$ more often when $x_a = 1$,

then $x_a \Rightarrow x_b$ is an association rule.

Ex: when you buy chips, you need beer to wash them down.

Suppose that beer is purchased 10% of the time in general, but 50% of the time when the consumer grabs chips.

- The *support* for ‘beer’ is 10%
- The *confidence* of this rule is 50%.
- It’s *lift* is 5: 50% is 5 times higher than 10%.

Given this information, you could put some chips by the beer.

Market Basket Analysis

Using purchase coincidence to build association rules.

Our example basket: LHS (chips) $\Rightarrow$ RHS (beer)
Left Hand Side: 'antecedent', Right Hand Side: 'consequent'.

Every event has support: the proportion of times it occurred.
This leads to two measures of association rule strength:

confidence: supp(LHS and RHS)/supp(LHS)
The probability of RHS given LHS.

lift: supp(LHS and RHS)/[supp(RHS) supp(LHS)]
Increase in probability of RHS given LHS occurs.

Support, Association, and Lift

Generally, association rules with high lift are most useful because they tell you something you don't already know.

Low support does not preclude high confidence or high lift.

Chips $\Rightarrow$ Beer is high support, but low lift if everybody always buys beer.

Caviar $\Rightarrow$ Vodka is low support, but high lift if people only buy vodka for their caviar parties.

There's no deep theory around ARules. We just scan the high-lift or high-confidence rules to find interesting rules.

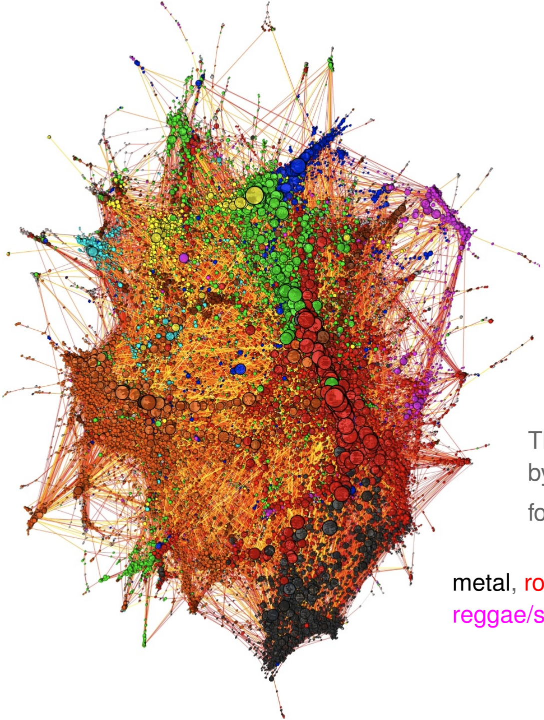
Finding Association Rules with R

To find confidence and lift, just count the number of times RHS and LHS happen, and how often they happen together.

$$\text{supp(event)} = \frac{\text{number of times event occurs}}{\text{number of observations}}$$

However, counting all possible combinations can take forever.
Apriori: algorithm for finding rules over a support threshold.

The `apriori` function is available in the `arules` package.
You need to get the data in a certain format,
but after this it is straightforward to use.



Last.fm Artist Plays

Online radio keeps track of everything you play, for recommending music & focused marketing.

This 'network' shows artists sized by play count, with lines (edges) for shared users.

metal, rock, pop, jazz, electronica, hip-hop reggae/ska, classical, folk/country/world.

Association rules for Music Taste

lhs		rhs	support	confidence	lift
t.i.	=>	kanye west	0.0104	0.5672	8.8544
pink floyd, the doors	=>	led zeppelin	0.0106	0.5387	6.8020
beyonce	=>	rihanna	0.0139	0.4686	10.8810
morrissey	=>	the smiths	0.0112	0.4655	8.8961
megadeth	=>	iron maiden	0.0132	0.4307	7.2677
jimi hendrix	=>	the doors	0.0120	0.3062	5.3170
nelly furtado	=>	madonna	0.0100	0.2750	5.0374
bright eyes	=>	the shins	0.0102	0.2698	5.4623
elliott smith	=>	modest mouse	0.0109	0.2679	5.1732
britney spears	=>	lady gaga	0.0120	0.2612	7.7292
ramones	=>	the clash	0.0104	0.2586	5.9052
franz ferdinand	=>	kaiser chiefs	0.0132	0.2224	7.1153

Example: Given a new user that listens to a lot of Morrissey, we're 46% positive that they'll also like the Smiths;
This is 9 times higher than if we didn't know about Morrissey.

From association to networks

Graphs can be a useful way to summarize all sorts of data.
We can define networks using any measure of connectivity.

For example, an association network:

Say there's an edge between `lhs` and `rhs` if `support` and `confidence` are greater than some thresholds.

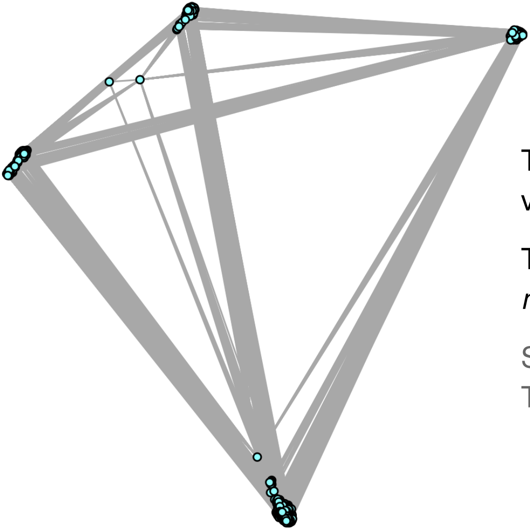
If we just look at any shared membership in a playlist, we get our monster graph from the beginning.

For example, in the `lastfm.R` code we use rules from

```
apriori(playtrans,
  parameter=list(support=.001, confidence=.1, maxlen=2))
```

to define a network with 1k nodes and 36k edges.

0.1% support and 10% confidence lastfm network

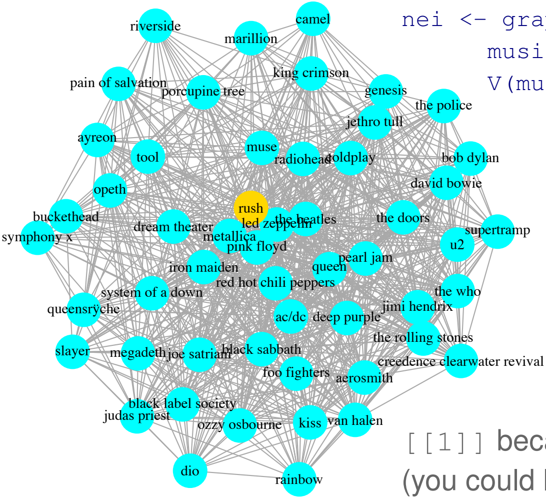


The network has four very distinct cliques.

These look something like *metal*, *hip-hop*, *alt*, *pop*.

See code for plotting. There's lots you can do.

You can also focus in: a *neighborhood* of order *k* for some node contains all vertices no more than *k* steps away.




```
nei <- graph.neighborhood(
    musicnet, order=1,
    V(musicnet)["rush"])[[1]]
```

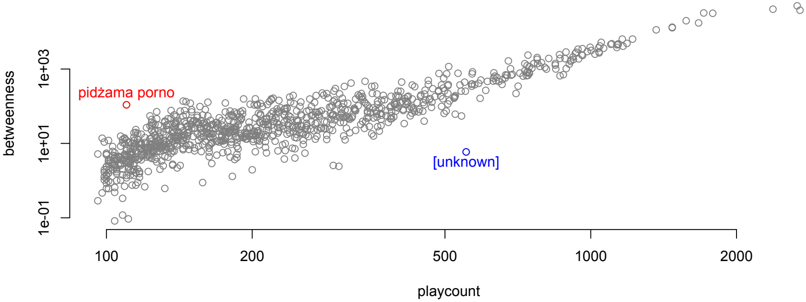
`[[1]]` because it returns a list (you could look at multiple node neighborhoods at once)

Connectivity in Music Choice

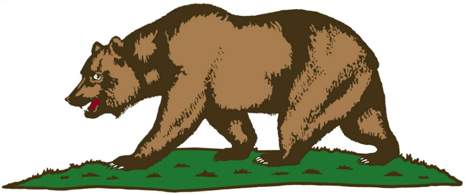
Popularity and connectivity are related but not the same

```
> which.max(mbetween)    > which.max(playcount)
   the beatles                 radiohead
```

There is a strong connection, with some big outliers



Search for "california"



Search is one great example of network analysis;
Consider ranking sites for the query "california".

Expanding the response set:

- Take 200 pages with heavy traffic and high term-frequency for "california".
- Follow links to build a neighborhood.

We're left with about 10,000 sites, with links, to rank.

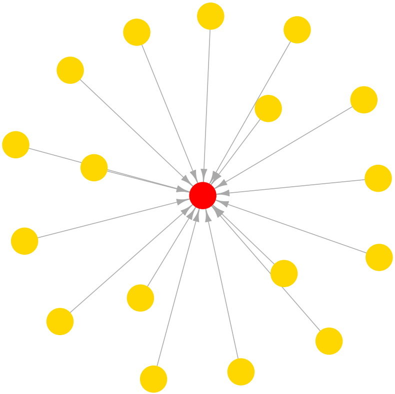
```
caedges <- read.csv("CaliforniaEdges.csv")
casites <- scan("CaliforniaNodes.txt", "character")
edgemat <- cbind(casites[caedges$from], casites[caedges$to])
```

The query provides a very large directed network

Look at neighborhoods to get a workable plot.

```
latimes <- graph.neighborhood(calink, order=1, ...)
```

LA Times network

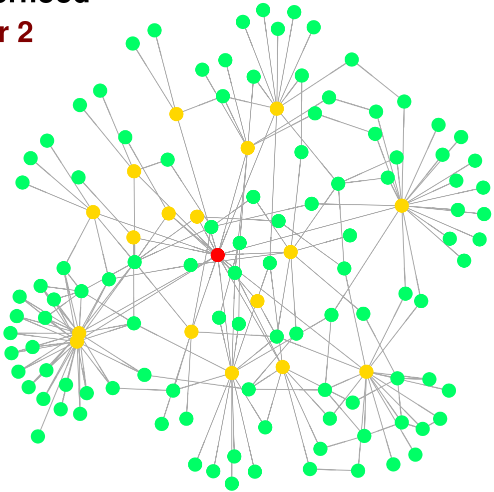


Neighborhood order is the number of included steps away from the node.

At order=1, we just have sites pointing to `latimes.com`.

LA Times Neighborhood

Expanded to order 2



Just going one extra step creates a much bigger network.

Yellow points are the first-order connections from before.

Larry and Sergei's Page Rank

Google has been pretty successful.
Page Rank stat is a key ingredient.

Paper: The Anatomy of a Large-Scale Hypertextual Web Search Engine, 1998

Page Rank labels a site more important if many sites link to it.
But the weight of each link is determined by its own rank.

A recursive calculation: $r_i = \sum_{j=1}^{n} \frac{e_{ij}}{c_j} r_j$

r_i is the page rank, e_{ij} is a binary edge indicator, and $c_j = \sum_i e_{ij}$ is the number of nodes pointed at by node j.

Page rank of "california" search response

We can run PageRank to organize our list of sites.

```
> search <- page.rank(calink)$vector
> casites[order(search, decreasing=TRUE)]
 [1] "http://www.calgold.com/"
 [2] "http://www.sancarlos-homes.com/info.asp"
 [3] "http://spectacle.berkeley.edu/"
 [4] "http://www.graddiv.ucr.edu/"
 [5] "http://www.chico-homes.com/info.asp"
 [6] "http://www.webb.pvt.k12.ca.us/~webb/WSCPrograms.html"
 [7] "http://www.berkeley.edu/"
 [8] "http://www.calfund.org/"
 [9] "http://www.ca-probate.com/"
[10] "http://www.ppconline.com/"
```

I don't think this alone would have made google famous!
Nodes need to be weighted by traffic and by successful clicks.

Homework Due Next Week: Connectivity in Hollywood

We'll explore casts for 'drama' movies from 1980-1999.
See *actors* example code and data.
I've limited the data to actors in more than ten productions over this time period (and to movies with more than ten actors).

[1] The actors network has an edge if the two actors were in the same movie. Plot the entire actors network.

[2] Plot the neighborhoods for "Bacon, Kevin" at orders 1-3.
How does the size of the network change with order?

[3] Who were the most common actors? Who were most connected?
Pick a pair of actors and describe the shortest path between them.

[4] Find pairwise actor-cast association rules with at least 0.01% support and 10% confidence. Describe what you find.

[+] What would be a regression based alternative to ARules?
Execute it for a single RHS actor.