CDS

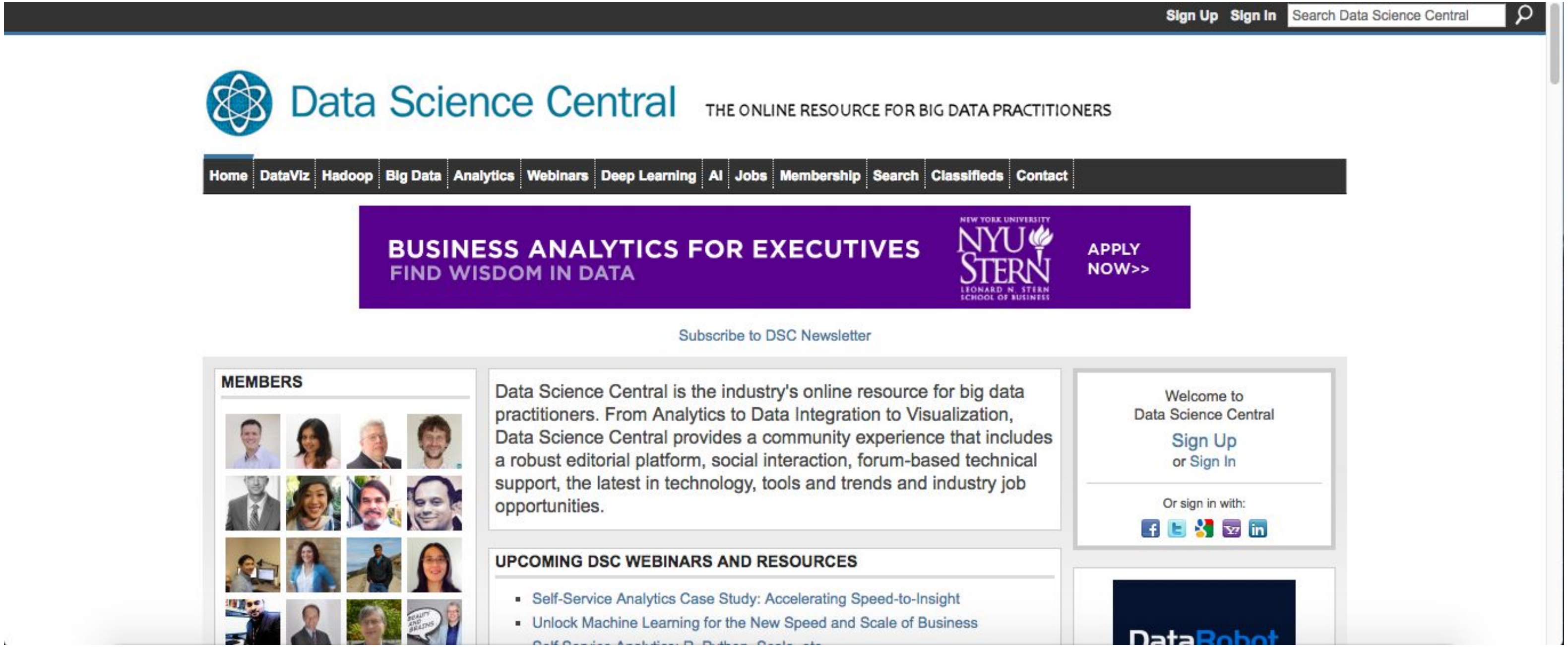
Cornell Data Science

Text Mining and Graphs

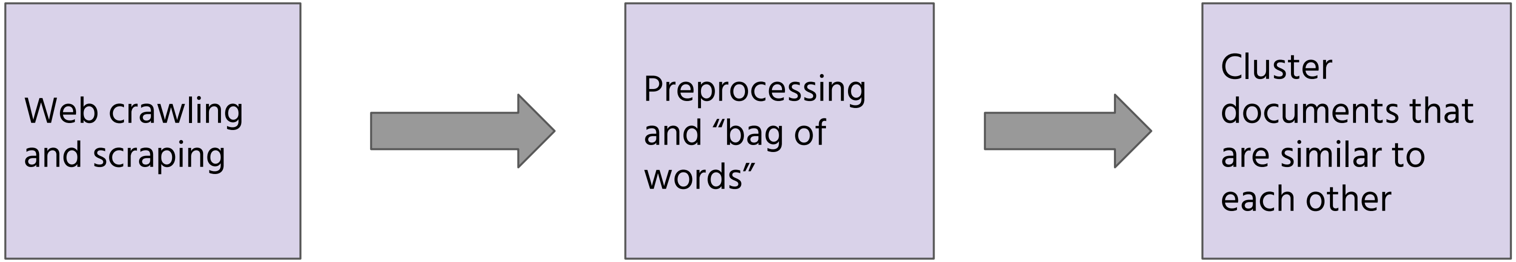
Real-World Application

Suppose you're making a student-led course and are figuring out what topics to include...

...but all you have at your disposal is a single website.

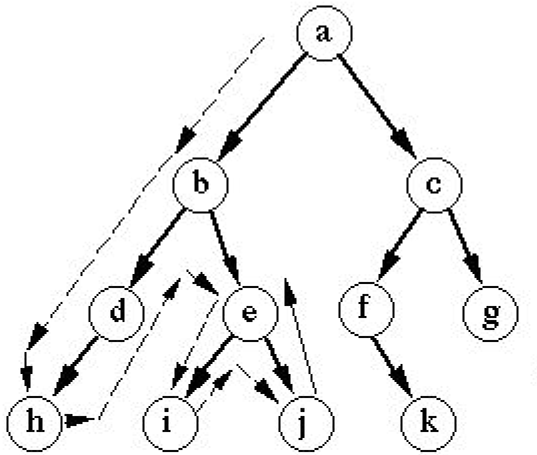


The Approach



Crawling

Web crawling - depth-first search of a set of linked pages. Google uses web crawling to build its search database.



Scraping

Scraping - Converting raw web pages into a usable format.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Example<title>
        <link rel="stylesheet" href="styl
    </head>
    <body>
        <h1>
            <a href="/">Header</a>
        </h1>
        <nav>
            <a href="one/">One</a>
            <a href="two/">Two</a>
            <a href="three/">Three</a>
        </nav>
```

(Be nice and follow the rules of whichever site you're scraping!)

[Source]

Processing Text

Our goal: the **bag of words** format. Each document is converted into the a set of word frequency counts.

```
{“data”: 152,
“science”: 138,
“information”: 99,
“learning”: 89,
“machine”: 85,
“model”: 70,
“build”: 68}
```

Pros:

Easy to interpret
Space-efficient
Can be put into matrix form

Cons:

Ignores word order
May not accurately encode meaning

Text Processing Techniques

“Cornell Data Science is an engineering project team at Cornell that seeks to prepare students for a career in data science.”

Punctuation removal/Lowercasing

“cornell data science is an engineering project team at cornell that seeks to prepare students for a career in data science”

Text Processing Techniques

Stop word removal

“cornell data science engineering project team cornell seeks prepare students career data science”

Lemmatization

“cornell data science engineering project team cornell seek prepare student career data science”

Source

Documents and Matrices

Here is what our document looks like now:

```
{"cornell": 2,

"data": 2,

"science": 2,

"engineering": 1,

"project": 1, ...}
```

Document-term matrix

```
[2 2 1 1 1 1 0 0 1 ...]
[0 1 2 1 5 1 3 2 3 ...]
[1 1 9 1 8 0 0 0 10 ..]
…
```

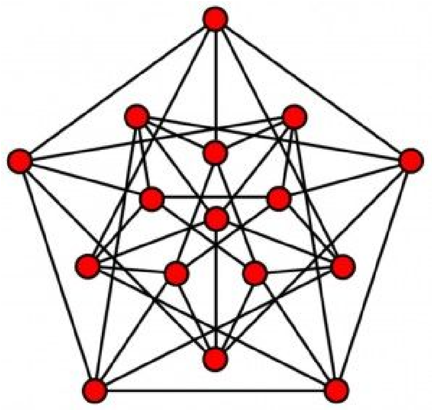
A Network of Documents

We build a **graph** of documents, where:

- Vertices (nodes) are documents
- Two documents are linked by an edge if they have a certain **cosine similarity**

This graph is **undirected**

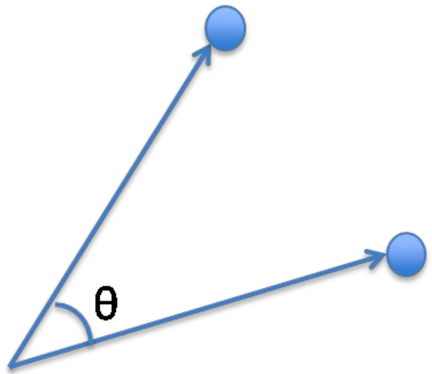
- If X links to Y, Y links to X



Cosine Similarity

Take the two rows (these are vectors) and find the cosine of the angle between the vectors.

$$sim(A,B) = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$



Why cosine similarity? Efficiently computed on sparse matrices. Also controls for length of vectors.

[Source](#)

Building the Network

We take the document-term matrix, which is *d* x *t*, and create a new *d* x *d* matrix. This is the **adjacency matrix** of the graph.

Document-term matrix

```
[2 2 1 1 1 1 0 0 1 ...]
[0 1 2 1 5 1 3 2 3 ...]
[1 1 9 1 8 0 0 0 10 ..]
```

...

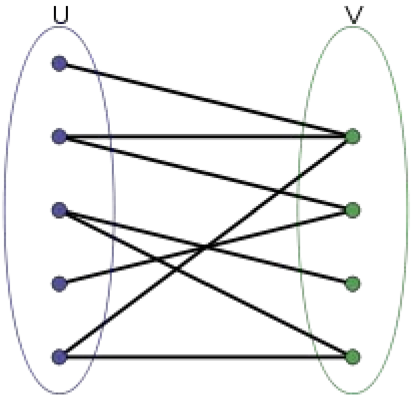
Adjacency matrix

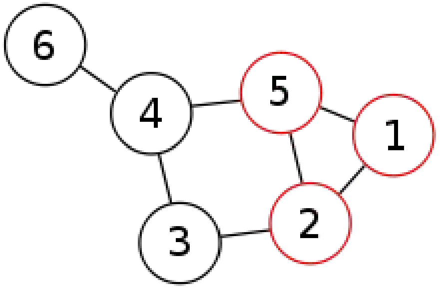
```
[0 0 0 1 1]
[0 0 0 1 0]
[0 0 0 0 0]
[1 1 0 0 1]
[1 0 0 1 0]
```

Graph Structures

Bipartition (bipartite graph) - two sets of vertices that are not internally connected but connect to each other

Clique - set of vertices that are all connected to each other



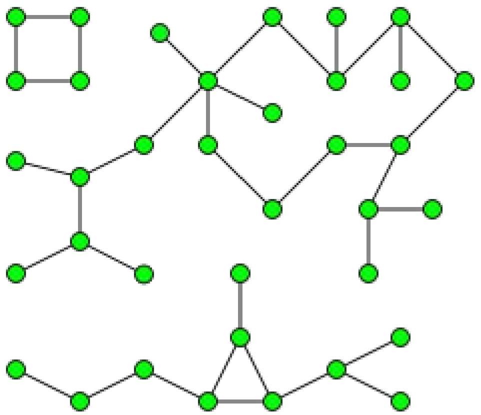


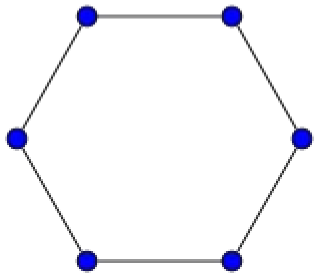
Source Source

Graph Structures

Connected component - maximum-size sets of vertices that are all reachable from each other

Cycle - a path (sequence of edges) that reaches the vertex it originated from





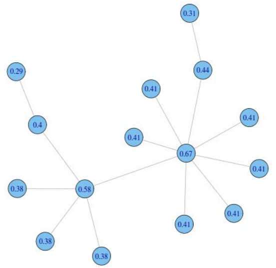
Source Source

Graph Properties

Centrality - how "important" a vertex is. Can be computed in various ways (degree, betweenness, closeness)

Connectivity (cohesion) - how many vertices are reachable from other vertices

Diameter - max of how long it takes to get from one vertex to another

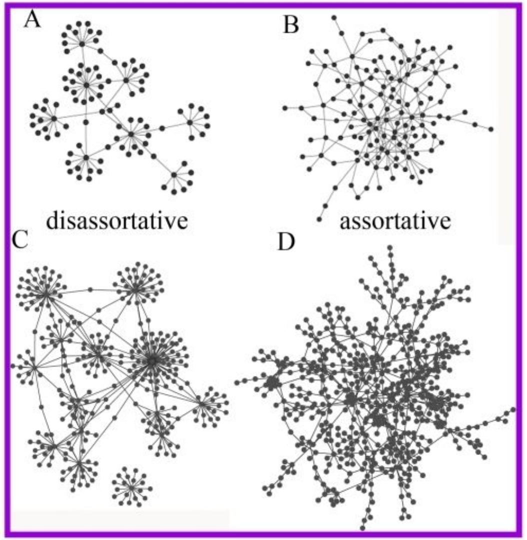


Source

Graph Properties

Assortativity - similarity (in terms of degree) of pairs of linked vertices

Rich-club coefficient - frequency with which nodes with high degree also connect to each other

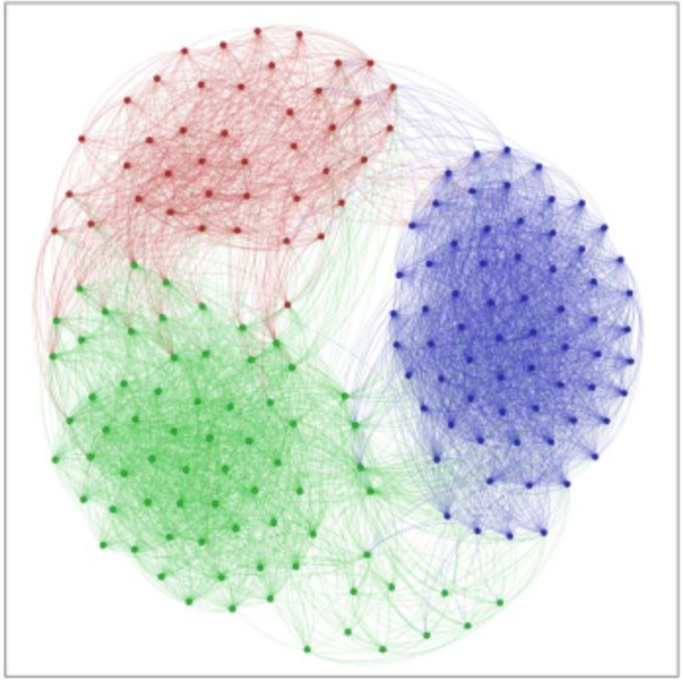


Source

Clustering on Graphs

We've seen *k*-means and hierarchical clustering.

Spectral clustering is another method of clustering, specifically for networked data.



Spectral Clustering

To perform spectral clustering on a graph [1]:

1. Generate the **Laplacian matrix** of the graph. L = D - A (D has degree values on diagonal, A is the adjacency matrix).
2. Find the first *k* eigenvectors of this matrix.
3. Create a matrix where the columns are these eigenvectors. Let the rows be interpreted as data points in space.
4. Plug these points into some other clustering algorithm like *k*-means.

[1] Source

Coming Up

Your assignment: Project 2

Next week: Big data tools and wrap-up

See you then!

Cornell Data Science