Lecture 27: Learning from relational data

STATS 202: Data mining and analysis

December 2, 2019

Announcements

- The Kaggle deadline is this Friday (Dec 6) at 11.59 pm. If you haven't already, make a submission to earn credit.
- In order to earn credit for the Kaggle competition, submit "Homework 8" by this Saturday.

Announcements

The class this Wednesday will be a review. There may be no class Friday, depending on when we will finish the review. Instructor and TAs will have their usual office hours.

Relational data

The observations have the form of a graph.

Examples.

- Links between websites.
- Relationships between accounts in social networks.
- Transmission networks for contagious diseases.

The links can be **directed** or **undirected**.

There can be different types of link (friend, follower, followed).

We can observe the graph in time (how do social networks grow?).

Each vertex can have additional features or **metadata**.

PageRank algorithm

- Invented by Sergei Brin and Larry Page of Google.
- Uses a graph of links between websites to rank websites by "importance".
- **Motivation:**
  - Consider the problem of searching the web using the query "birth control".
  - There are millions of pages containing the term.
  - Analyzing the content of each website semantically to infer which one is more likely to interest the user is very expensive.
  - We need a way to rank websites, to filter out all those that are rarely visited. This information is given by links.

PageRank algorithm

Consider a hypothetical **surfer** who is jumping from website to website by clicking on random links. Intuitively, the websites that are visited more frequently can be considered more important in the network of links.

Will the surfer visit every website eventually? No. It is possible to get stuck in a website with no outgoing links, or to be stuck in a loop between two websites, for example.

To avoid this problem, we modify the random walk, such that at every step, with probability $1 - q$, we pick a website at random, and with probability q we go through one of the links in the current website at random.

PageRank algorithm

- The **surfer**'s random walk is a Markov chain on the set of websites.
- It is a fact that the frequency with which the surfer visits any website converges to some limit.
- The PageRank of a website is this limiting frequency.

PageRank algorithm

Let P_{ij} be the probability of jumping from website i to website j, then

$$P_{ij} = (1-q)\frac{1}{n} + q\left[\frac{\#\text{ of links from } i \text{ to } j}{\#\text{ of links out of } i}\right]$$

The limiting frequency of website j, π_j, must satisfy

$$\pi_j = \sum_{i=1}^{n} \pi_i P_{ij}$$

or in matrix notation $\pi = \pi P$. That is, π is an eigenvector of the transition probability matrix P with eigenvalue 1.

Finding the limiting frequencies π

In principle, finding the limiting frequencies could require solving the eigendecomposition of a matrix P which is $n \times n$, and this has a complexity which grows as n^3.

However, it is possible to compute π by starting with the approximation $\pi^{(0)} = (1/n, \ldots, 1/n)$, and iterating:

$$\pi^{(t)} = \pi^{(t-1)} P.$$

The number of iterations necessary for convergence is typically small.

The matrix-vector multiplication in each iteration can be sped up using sparse matrix techniques.

How can PageRank be used in web search?

One idea:

1. Find all websites that contain all query terms.
2. Display them in order of their PageRank.

A more likely approach:

1. Use PageRank to select the 10,000 most important pages which contain the query terms.
2. Rank these 10,000 pages by analyzing their content, integrating information about the user, etc.

Working with graphs in R and Python

The package `igraph` implements a lot of utilities for analyzing graphs in R and Python.

It has a function page.rank, among many others.