Machine Learning
CPSC 340

Tutorial 12
Random Walk on Graph

Page Rank Algorithm
Label Propagation on Graph

- Assume a strongly connected graph $G = (V, A)$
Label Propagation on Graph

- Assume a strongly connected graph $G = (V, A)$
- $V$: set of nodes
Label Propagation on Graph

- Assume a strongly connected graph $G = (V, A)$
- $V$: set of nodes
- $A$: adjacency matrix
Label Propagation on Graph

- Assume a strongly connected graph $G = (V, A)$
- $V$: set of nodes
- $A$: adjacency matrix
- Two type of nodes: labeled and unlabeled
Label Propagation on Graph

- Assume a strongly connected graph $G = (V, A)$
- $V$: set of nodes
- $A$: adjacency matrix
- Two type of nodes: labeled and unlabeled
- Label is either $+1$ or $-1$
Label Propagation on Graph

- Assume a strongly connected graph $G = (V, A)$
- $V$: set of nodes
- $A$: adjacency matrix
- Two type of nodes: labeled and unlabeled
- Label is either $+1$ or $-1$
- Goal: assign a label to unlabeled nodes.
Random Walk for Label Propagation

- Random Walk on Graph: we jump from one node to another one with some probability
Random Walk for Label Propagation

- Random Walk on Graph: we jump from one node to another one with some probability
- Label propagation algorithm
Random Walk for Label Propagation

- Random Walk on Graph: we jump from one node to another one with some probability
- Label propagation algorithm
  - start from an unlabeled node $v$
Random Walk for Label Propagation

- Random Walk on Graph: we jump from one node to another one with some probability
- Label propagation algorithm
  - start from an unlabeled node $\nu$
  - do $k$ times random walk starting from $\nu$ and store the output labels
Random Walk for Label Propagation

- Random Walk on Graph: we jump from one node to another one with some probability
- Label propagation algorithm
  - start from an unlabeled node $v$
  - do $k$ times random walk starting from $v$ and store the output labels
  - do majority vote among the stored labels
Random Walk for Label Propagation

Random Walk Algorithm

- repeat until you find a label
- let $v$ be the node you are in and has $d_v$ neighbours
- if $v$ is unlabeled, with uniform probability $\frac{1}{d_v}$ pick one of its neighbours and jump to that node
- if $v$ is labeled
  - with probability $\frac{1}{d_v+1}$ output its label
  - with uniform probability $\frac{1}{d_v+1}$ pick one of its neighbours and jump to that node
Random Walk for Label Propagation

Random Walk Algorithm

- repeat until you find a label
- let $v$ be the node you are in and has $d_v$ neighbours
Random Walk for Label Propagation

Random Walk Algorithm

- repeat until you find a label
- let \( v \) be the node you are in and has \( d_v \) neighbours
- if \( v \) is unlabeled, with uniform probability \( \frac{1}{d_v} \) pick one of its neighbours and jump to that node
- if \( v \) is labeled, with probability \( \frac{1}{d_v+1} \) output its label, with uniform probability \( \frac{1}{d_v+1} \) pick one of its neighbours and jump to that node
Random Walk Algorithm

- repeat until you find a label
- let $v$ be the node you are in and has $d_v$ neighbours
- if $v$ is unlabeled, with uniform probability $\frac{1}{d_v}$ pick one of its neighbours and jump to that node
- if $v$ is labeled
Random Walk for Label Propagation

Random Walk Algorithm

- repeat until you find a label
- let $v$ be the node you are in and has $d_v$ neighbours
- if $v$ is unlabeled, with uniform probability $\frac{1}{d_v}$ pick one of its neighbours and jump to that node
- if $v$ is labeled
  - with probability $\frac{1}{d_v+1}$ output its label
Random Walk Algorithm

- repeat until you find a label
- let $v$ be the node you are in and has $d_v$ neighbours
- if $v$ is unlabeled, with uniform probability $\frac{1}{d_v}$ pick one of its neighbours and jump to that node
- if $v$ is labeled
  - with probability $\frac{1}{d_v+1}$ output its label
  - with uniform probability $\frac{1}{d_v+1}$ pick one of its neighbours jump to that node
Exercise

Assume we are given adjacency matrix, a labelList, a matrix where the first column contains node numbers and the second column contains class labels and starting node, write code for random walk algorithm.
function [y] = runRandomWalk(A, labelList, v)

while 1
    if any(labelList(:,1) == v)
        neighbours = find(A(v,:));
        nNeighbours = length(neighbours);
        ind = ceil(rand*(nNeighbours+1));
        if ind == nNeighbours+1
            ind = find(labelList(:,1)==v);
            y = labelList(ind,2);
            return
        else
            v = neighbours(ind);
        end
    else
        neighbours = find(A(v,:));
        nNeighbours = length(neighbours);
        ind = ceil(rand*nNeighbours);
        v = neighbours(ind);
    end
end
end
The ranking problem:

- Input: a large set of objects (and possibly a query object).
- Output option 1: score of each object (and possibly for query).
- Output option 2: ordered list of most relevant objects (possibly for query).
The ranking problem:
- Input: a large set of objects (and possibly a query object).
- Output option 1: score of each object (and possibly for query).
- Output option 2: ordered list of most relevant objects (possibly for query).

Examples:
- Country comparisons (Global Hunger Index)
- Academic journals (Impact factor).
- Sports/gaming (Elo and TrueSkill)
- Internet search engines
PageRank

- Goal: ranking webpages based on some score or weight
PageRank

- Goal: ranking webpages based on some score or weight
- Assuming that we have $n$ webpages and let the score of webpage $i$ be $p_i$, and $P = (p_i)$ be a vector of size $n$
PageRank

- Goal: ranking webpages based on some score or weight
- Assuming that we have $n$ webpages and let the score of webpage $i$ be $p_i$, and $P = (p_i)$ be a vector of size $n$
- Make the directed webpage graph $G$
  - node $i$ has an edge into node $j$ if there is a link from page $i$ to $j$ i.e. $i \rightarrow j$.
  - Assume this graph is strongly connected and aperiodic and does not have absorbing node
PageRank

- Goal: ranking webpages based on some score or weight
- Assuming that we have $n$ webpages and let the score of webpage $i$ be $p_i$, and $P = (p_i)$ be a vector of size $n$
- Make the directed webpage graph $G$
  - node $i$ has an edge into node $j$ if there is a link from page $i$ to $j$ i.e. $i \rightarrow j$.
  - Assume this graph is strongly connected and aperiodic and does not have absorbing node
- Let $m_j$ be the number of outgoing edges from node $j$ and $m = (m_j)$ be a vector of size $n$
PageRank

- Goal: ranking webpages based on some score or weight
- Assuming that we have \( n \) webpages and let the score of webpage \( i \) be \( p_i \), and \( P = (p_i) \) be a vector of size \( n \)
- Make the directed webpage graph \( G \)
  - node \( i \) has an edge into node \( j \) if there is a link from page \( i \) to \( j \) i.e. \( i \to j \).
  - Assume this graph is strongly connected and aperiodic and does not have absorbing node
- Let \( m_j \) be the number of outgoing edges from node \( j \) and \( m = (m_j) \) be a vector of size \( n \)
- Let \( A \) be the adjacency matrix for \( G \) i.e. \( A_{ij} = 1 \) if \( i \to j \) o.w. \( A_{ij} = 0 \)
Goal: ranking webpages based on some score or weight

Assuming that we have \( n \) webpages and let the score of webpage \( i \) be \( p_i \), and \( P = (p_i) \) be a vector of size \( n \)

Make the directed webpage graph \( G \)
  - node \( i \) has an edge into node \( j \) if there is a link from page \( i \) to \( j \) i.e. \( i \rightarrow j \).
  - Assume this graph is strongly connected and aperiodic and does not have absorbing node

Let \( m_j \) be the number of outgoing edges from node \( j \) and \( m = (m_j) \) be a vector of size \( n \)

Let \( A \) be the adjacency matrix for \( G \) i.e. \( A_{ij} = 1 \) if \( i \rightarrow j \) o.w. \( A_{ij} = 0 \)

Let \( Z = A^T (\text{diag}(m))^{-1} \)
Goal: ranking webpages based on some score or weight

Assuming that we have \( n \) webpages and let the score of webpage \( i \) be \( p_i \), and \( P = (p_i) \) be a vector of size \( n \)

Make the directed webpage graph \( G \)

- node \( i \) has an edge into node \( j \) if there is a link from page \( i \) to \( j \) i.e. \( i \rightarrow j \).
- Assume this graph is strongly connected and aperiodic and does not have absorbing node

Let \( m_j \) be the number of outgoing edges from node \( j \) and \( m = (m_j) \) be a vector of size \( n \)

Let \( A \) be the adjacency matrix for \( G \) i.e. \( A_{ij} = 1 \) if \( i \rightarrow j \) o.w. \( A_{ij} = 0 \)

Let \( Z = A^T (\text{diag}(m))^{-1} \)

\((Z)_{ij}\): probability of jumping from page \( j \) to page \( i \) via a link
PageRank

- Goal: ranking webpages based on some score or weight
- Assuming that we have $n$ webpages and let the score of webpage $i$ be $p_i$, and $P = (p_i)$ be a vector of size $n$
- Make the directed webpage graph $G$
  - node $i$ has an edge into node $j$ if there is a link from page $i$ to page $j$ i.e. $i \rightarrow j$.
  - Assume this graph is strongly connected and aperiodic and does not have absorbing node
- Let $m_j$ be the number of outgoing edges from node $j$ and $m = (m_j)$ be a vector of size $n$
- Let $A$ be the adjacency matrix for $G$ i.e. $A_{ij} = 1$ if $i \rightarrow j$ o.w. $A_{ij} = 0$
- Let $Z = A^T (\text{diag}(m))^{-1}$
- $(Z)_{ij}$: probability of jumping from page $j$ to page $i$ via a link
- But we can go directly from page $j$ to $i$ by entering the address of page $i$ in the address-bar of browser
  - add some small amount to all $(Z)_{ij}$ and normalize!
For some $d \in (0, 1)$ let $E = \text{ones}(n, n)$ and $T = \frac{1-d}{n} E + dZ$
For some \( d \in (0, 1) \) let \( E = \text{ones}(n, n) \) and \( T = \frac{1-d}{n} E + dZ \).

Now with transition matrix \( T \) for a webpage graph \( G \), we have

\[
P = TP
\]
For some $d \in (0, 1)$ let $E = \text{ones}(n, n)$ and $T = \frac{1-d}{n} E + dZ$

Now with transition matrix $T$ for a webpage graph $G$, we have

$$P = TP$$

Each $p_i$ is a probability, so $\sum_{i=1}^{n} p_i = 1$
PageRank

- For some $d \in (0, 1)$ let $E = \text{ones}(n, n)$ and $T = \frac{1-d}{n} E + dZ$
- Now with transition matrix $T$ for a webpage graph $G$, we have $P = TP$
- Each $p_i$ is a probability, so $\sum_{i=1}^{n} p_i = 1$
- With two above equations, we can find all $p_i$s by solving corresponding linear system
PageRank: Exercise

Let \( d = \frac{3}{4} \). For the following webpage graph, find \( A, m, Z \) and \( T \). Then make the linear system and solve it and find \( P \).
PageRank: Solution

\[
m = \begin{pmatrix} 2 \\ 2 \\ 2 \\
\end{pmatrix}
\]
\[
A = \begin{pmatrix} 0 & 0 & 1 & 1 \\
1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 \\
\end{pmatrix}
\]
\[
Z = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
\frac{1}{2} & 0 & \frac{1}{2} & 0 \\
\end{pmatrix}
\]
\[
T = \frac{1}{16} * \begin{pmatrix} 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
\end{pmatrix} + \frac{1}{4} * Z
\]
\[
P = TP \rightarrow (T - I)P = 0 \quad (I \text{ is Identity matrix})
\]
\[
\sum_{i=1}^{4} p_i = 1
\]
PageRank: Solution

Combining two equations, we get

\[
\begin{pmatrix}
\frac{1}{16} & \frac{7}{16} & \frac{1}{16} & \frac{7}{16} \\
\frac{16}{7} & \frac{1}{16} & \frac{16}{7} & \frac{1}{16} \\
\frac{16}{7} & \frac{1}{16} & \frac{16}{7} & \frac{16}{7} \\
\frac{16}{1} & \frac{1}{16} & \frac{1}{16} & \frac{16}{1}
\end{pmatrix}
\begin{pmatrix}
p_1 \\
p_2 \\
p_3 \\
p_4
\end{pmatrix}
= 
\begin{pmatrix}
0 \\
0 \\
0 \\
1
\end{pmatrix}
\]

\[
P = 
\begin{pmatrix}
0.25 \\
0.25 \\
0.25 \\
0.25
\end{pmatrix}
\]