第21章 PageRank算法

1. PageRank是互联网网页重要度的计算方法，可以定义推广到任意有向图结点的重要度计算上。其基本思想是在有向图上定义随机游走模型，即一阶马尔可夫链，描述游走者沿着有向图随机访问各个结点的行为，在一定条件下，极限情况访问每个结点的概率收敛到平稳分布，这时各个结点的概率值就是其 PageRank值，表示结点相对重要度。
2. 有向图上可以定义随机游走模型，即一阶马尔可夫链，其中结点表示状态，有向边表示状态之间的转移，假设一个结点到连接出的所有结点的转移概率相等。转移概率由转移矩阵M表示 $M=[m_{ij}]_{n\times n}$ 第i行第j列的元素m_{ij}表示从结点j跳转到结点i的概率。
3. 当含有n个结点的有向图是强连通且非周期性的有向图时，在其基础上定义的随机游走模型，即一阶马尔可夫链具有平稳分布，平稳分布向量R称为这个有向图的 PageRank。若矩阵M是马尔可夫链的转移矩阵，则向量R满足$MR=R$向量R的各个分量称 PageRank为各个结点的值。 $R=\begin{bmatrix} PR(v_1) \\ PR(v_2) \\ \vdots \\ PR(v_n) \end{bmatrix}$ 其中$PR(v_i), i=1,2,\cdots,n$，表示结点v_i的 PageRank值。这是 PageRank的基本定义。
4. PageRank基本定义的条件现实中往往不能满足，对其进行扩展得到 PageRank的一般定义。任意含有n个结点的有向图上，可以定义一个随机游走模型，即一阶马尔可夫链，转移矩阵由两部分的线性组合组成，其中一部分按照转移矩阵M，从一个结点到连接出的所有结点的转移概率相等，另一部分按照完全随机转移矩阵，从任一结点到任一结点的转移概率都是$1/n$。这个马尔可夫链存在平稳分布，平稳分布向量R称为这个有 PageRank向图的一般，满足 $R=dMR+\frac{1-d}{n}1$

其中$d(0\leq d\leq 1)$是阻尼因子，1是所有分量为1的n维向量。

5. PageRank的计算方法包括迭代算法、幂法、代数算法。

幂法将 PageRank的等价式写成$R=(dM+\frac{1-d}{n}E)R=AR$ 其中d是阻尼因子，E是所有元素为1的n阶方阵。

PageRank算法可以看出R是一般转移矩阵A的主特征向量，即最大的特征值对应的特征向量。 幂法就是一个计算矩阵的主特征值和主特征向量的方法。

步骤是：选择初始向量x_0；计算一般转移矩阵A；进行迭代并规范化向量 $y_{t+1}=Ax_t$ $x_{t+1}=\frac{y_{t+1}}{\|y_{t+1}\|}$ 直至收敛。

在实际应用中许多数据都以图(graph)的形式存在，比如，互联网、社交网络都可以看作是一个图。图数据上的机器学习具有理论与应用上的重要意义。pageRank算法是图的链接分析 (link analysis)的代表性算法，属于图数据上的无监督学习方法。

pageRank算法最初作为互联网网页重要度的计算方法，1996年由page和Brin提出，并用于谷歌搜索引擎的网页排序。事实上，pageRank可以定义在任意有向图上，后来被应用到社会影响力分析、文本摘要等多个问题。

pageRank算法的基本想法是在有向图上定义一个随机游走模型，即一阶马尔可夫链，描述随机游走者沿着有向图随机访问各个结点的行为。在一定条件下，极限情况访问每个结点的概率收敛到平稳分布，这时各个结点的平稳概率值就是其 pageRank值，表示结点的重要度。 pageRank是递归定义的，pageRank的计算可以通过迭代算法进行。

```
#https://gist.github.com/diogojc/1338222/84d767a68da711a154778fb1d00e772d65322187

import numpy as np
from scipy.sparse import csc_matrix


def pageRank(G, s=.85, maxerr=.0001):
    """
    Computes the pagerank for each of the n states
    Parameters
    ----------
    G: matrix representing state transitions
       Gij is a binary value representing a transition from state i to j.
    s: probability of following a transition. 1-s probability of teleporting
       to another state.
    maxerr: if the sum of pageranks between iterations is bellow this we will
            have converged.
    """
    n = G.shape[0]

    # transform G into markov matrix A
    A = csc_matrix(G, dtype=np.float)
    rsums = np.array(A.sum(1))[:, 0]
    ri, ci = A.nonzero()
    A.data /= rsums[ri]

    # bool array of sink states
    sink = rsums == 0

    # Compute pagerank r until we converge
    ro, r = np.zeros(n), np.ones(n)
    while np.sum(np.abs(r - ro)) > maxerr:
        ro = r.copy()
        # calculate each pagerank at a time
        for i in range(0, n):
            # inlinks of state i
            Ai = np.array(A[:, i].todense())[:, 0]
            # account for sink states
            Di = sink / float(n)
            # account for teleportation to state i
            Ei = np.ones(n) / float(n)

            r[i] = ro.dot(Ai * s + Di * s + Ei * (1 - s))
```

```
    # return normalized pagerank
    return r / float(sum(r))
```

```
# Example extracted from 'Introduction to Information Retrieval'
G = np.array([[0,0,1,0,0,0,0],
              [0,1,1,0,0,0,0],
              [1,0,1,1,0,0,0],
              [0,0,0,1,1,0,0],
              [0,0,0,0,0,0,1],
              [0,0,0,0,0,1,1],
              [0,0,0,1,1,0,1]])
print(pageRank(G,s=.86))
```

```
[0.12727557 0.03616954 0.12221594 0.22608452 0.28934412 0.03616954
 0.16274076]
```

本章代码来源：https://github.com/hktxt/Learn-Statistical-Learning-Method

本文代码更新地址：https://github.com/fengdu78/lihang-code

中文注释制作：机器学习初学者公众号：ID:ai-start-com

配置环境：python 3.5+

代码全部测试通过。