PageRank

James Quacinella

April 19, 2017

Transition Matrix

```
n <- 6
A <- matrix(c(0,.5,.5,0,0,0,
              1/6,1/6,1/6,1/6,1/6,1/6,
              1/3,1/3,0,0,1/3,0,
              0,0,0,0,.5,.5,
              0,0,0,.5,0,.5,
              0,0,0,1,0,0), byrow=T, nrow=n)
A
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
## [1,] 0.0000000 0.5000000 0.5000000 0.0000000 0.0000000 0.0000000
## [2,] 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667
## [3,] 0.3333333 0.3333333 0.0000000 0.0000000 0.3333333 0.0000000
## [4,] 0.0000000 0.0000000 0.0000000 0.0000000 0.5000000 0.5000000
## [5,] 0.0000000 0.0000000 0.0000000 0.5000000 0.0000000 0.5000000
## [6,] 0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000
```

Lets confirm the rows add to 1:

```
rowSums(A)
```

```
## [1] 1 1 1 1 1 1
```

Lets update our matrix with a decay matrix as per the notes:

```
d <- 0.85

B <- d*A + ((1-d) / n)
B
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
## [1,] 0.0250000 0.4500000 0.4500000 0.0250000 0.0250000 0.0250000
## [2,] 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667
## [3,] 0.3083333 0.3083333 0.0250000 0.0250000 0.3083333 0.0250000
## [4,] 0.0250000 0.0250000 0.0250000 0.0250000 0.4500000 0.4500000
## [5,] 0.0250000 0.0250000 0.0250000 0.4500000 0.0250000 0.4500000
## [6,] 0.0250000 0.0250000 0.0250000 0.8750000 0.0250000 0.0250000
```

Lets create a page rank vector, which represents probabilities:

```
r <- rep(1/n, n)
r
```

```
## [1] 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667
```

Lets do one iteration:

```
r_new <- B %*% r
r_new
```

```
##           [,1]
## [1,] 0.1666667
## [2,] 0.1666667
## [3,] 0.1666667
## [4,] 0.1666667
## [5,] 0.1666667
## [6,] 0.1666667
```

Wait … why didn't r change?? If you think about how B is constructed with it rows summing to 1, then r via multiplication never changes! In order to see any changes in R, I had to swap the matrix multiplcaition order:

```
r_new <- r %*% B
r_new
```

```
##            [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
## [1,] 0.09583333 0.1666667 0.1194444 0.2611111 0.1666667 0.1902778
```

Lets do this repeatedly and find the page rank vector:

```
library(expm)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'expm'
```

```
## The following object is masked from 'package:Matrix':
##
##     expm
```

```
r_final <- r %*% ( B %^% 30 )
r_final
```

```
##            [,1]       [,2]       [,3]      [,4]      [,5]      [,6]
## [1,] 0.05170475 0.07367927 0.05741242 0.3487037 0.1999038 0.2685961
```

Eigenvector Method

Lets try to solve this using eigenvectors:

```
eigen(B)
```

```
## $values
## [1]  1.00000000  0.57619235 -0.42500001 -0.42499999 -0.34991524 -0.08461044
##
## $vectors
##            [,1]       [,2]          [,3]          [,4]         [,5]
## [1,] -0.4082483 -0.7278031 -5.345224e-01  5.345225e-01 -0.795670150
## [2,] -0.4082483 -0.3721164 -6.463633e-09 -6.463634e-09  0.059710287
## [3,] -0.4082483 -0.5389259  5.345225e-01 -5.345225e-01  0.602762996
## [4,] -0.4082483  0.1174605 -2.672613e-01  2.672612e-01  0.002611877
## [5,] -0.4082483  0.1174605 -2.672613e-01  2.672612e-01  0.002611877
## [6,] -0.4082483  0.1174605  5.345225e-01 -5.345224e-01  0.002611877
##              [,6]
## [1,] -0.486246420
## [2,]  0.673469294
## [3,] -0.556554233
## [4,]  0.009145393
## [5,]  0.009145393
## [6,]  0.009145393
```

Uh oh, the eigenvector with eigenvalue 1 doesn't look right at all! One thing to notice is that our method above is using matrix B in a way thats NOT an eigenvector problem. We need to do some math to switch the order of the terms:

$$r = r * B$$
$$r^T = (r * B)^T$$
$$r^T = B^T * r^T$$

Hence, it seems like we need to look at eigenvectors of the transpose of this matrix:

```
eigens <- eigen( t(B) )
eigens
```

```
## $values
## [1]  1.00000000  0.57619235 -0.42500001 -0.42499999 -0.34991524 -0.08461044
##
## $vectors
##           [,1]       [,2]          [,3]          [,4]        [,5]
## [1,] 0.1044385  0.2931457 -6.217336e-16  5.848969e-16 -0.06471710
## [2,] 0.1488249  0.5093703  2.004273e-16 -1.475988e-16  0.01388698
## [3,] 0.1159674  0.3414619  5.562880e-16 -5.709060e-16  0.07298180
## [4,] 0.7043472 -0.5890805 -7.071068e-01  7.071068e-01 -0.66058664
## [5,] 0.4037861 -0.1413606  7.071068e-01 -7.071068e-01  0.73761812
## [6,] 0.5425377 -0.4135367  1.029092e-08  1.029092e-08 -0.09918316
##              [,6]
## [1,] -0.212296003
## [2,]  0.854071294
## [3,] -0.363638739
## [4,]  0.018399984
## [5,] -0.304719509
## [6,]  0.008182973
```

```
r_final_eigen <- eigens$vectors[,1]
r_final_eigen
```

```
## [1] 0.1044385 0.1488249 0.1159674 0.7043472 0.4037861 0.5425377
```

Issue here is that this is not a pagerank vector since it does not represent probabilities that sum to 1. If r_final_eigen is a eigenvector solution to the equation, then so is any scalar multiple of it. We need to scale this eigen vector such that the components add to 1. NOTE: this is not 'normalizing' a vector

```
r_final - r_final_eigen / sum(r_final_eigen)
```

```
##              [,1]         [,2]         [,3]          [,4]          [,5]
## [1,] 5.717987e-09 9.935579e-09 6.660422e-09 -1.149143e-08 -2.756269e-09
##               [,6]
## [1,] -8.066286e-09
```

```
r_final_eigen <- r_final_eigen / sum(r_final_eigen)
r_final_eigen
```

```
## [1] 0.05170475 0.07367926 0.05741241 0.34870369 0.19990381 0.26859608
```