5. HITS Algorithm

September 11, 2018

Saumay Agrawal
16BCE1151

```
In [1]: import pandas as pd
        import matplotlib.pyplot as plt
        %matplotlib inline
        import seaborn as sb
        import networkx as nx
```

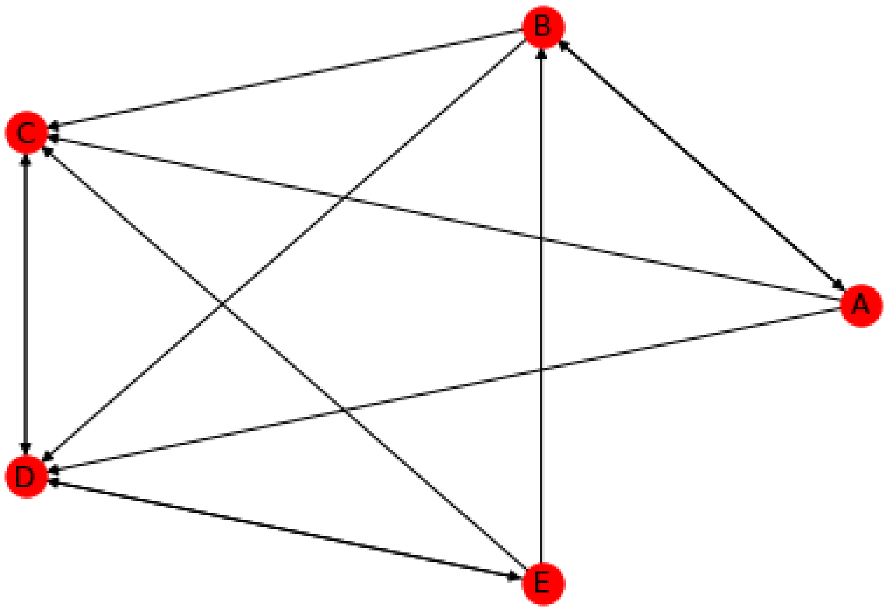
```
/anaconda3/lib/python3.6/importlib/_bootstrap.py:219: RuntimeWarning: numpy.dtype size changed
  return f(*args, **kwds)
/anaconda3/lib/python3.6/importlib/_bootstrap.py:219: RuntimeWarning: numpy.dtype size changed
  return f(*args, **kwds)
/anaconda3/lib/python3.6/importlib/_bootstrap.py:219: RuntimeWarning: numpy.dtype size changed
  return f(*args, **kwds)
/anaconda3/lib/python3.6/importlib/_bootstrap.py:219: RuntimeWarning: numpy.dtype size changed
  return f(*args, **kwds)
```

```
In [2]: # Class to define an object of a web structure

        class Web(object):

            def __init__(self, matrix=[], nnames=[]):
                self.matrix = matrix
                self.nnames = nnames
                self.nnodes = len(self.matrix)
                self.authscore = [1 for _ in range(self.nnodes)]
                self.hubscore = [1 for _ in range(self.nnodes)]
                self.inlinks = [self.getInlinks(node) for node in range(self.nnodes)]
                self.outlinks = [self.getOutlinks(node) for node in range(self.nnodes)]

            # Returns the inlinks of a given node
            def getInlinks(self, node):
                return [row for row in range(self.nnodes) if self.matrix[row][node]==1]

            # Returns the outlinks of a given node
            def getOutlinks(self, node):
```

```
        return [i for i in range(self.nnodes) if self.matrix[node][i]==1 ]

    # Draws the graph for current web structure
    def drawGraph(self):
        G = nx.DiGraph()
        G.add_nodes_from(self.nnames)
        for i in range(self.nnodes):
            for j in range(self.nnodes):
                if self.matrix[i][j]==1:
                    G.add_edge(self.nnames[i], self.nnames[j])
        nx.draw_shell(G, with_labels=True, arrows=True)

    # Normalizes the given vector
    def normalize(self, vector):
       summation = sum(vector)
       vector = [round(i/summation, 3) for i in vector]
       return vector

    # Calculates the authority score for a given node
    def getAuthScore(self, node):
       score = 0
       for inlink in self.inlinks[node]:
          score += self.hubscore[inlink]
       return score

    # Calculates the hub score for a given node
    def getHubScore(self, node):
       score = 0
       for outlink in self.outlinks[node]:
          score += self.authscore[outlink]
       return score

    # Runs the HITS Algorithm on current web structure
    def hits(self, niter):
        for itnum in range(niter):
            print('A #{} '.format(itnum), self.authscore)
            print('H #{} '.format(itnum), self.hubscore)
            print()
            newauthscore = [self.getAuthScore(i) for i in range(self.nnodes)]
            newauthscore = self.normalize(newauthscore)
            newhubscore = [self.getHubScore(i) for i in range(self.nnodes)]
            newhubscore =  self.normalize(newhubscore)
            self.authscore, self.hubscore = newauthscore, newhubscore
```

In [3]:

```
# Initialising web structure for question 1

matrix1 = [[0,1,1,1,0],
         [1,0,1,1,0],
         [0,0,0,1,0],
         [0,0,1,0,1],
         [0,1,1,1,0]]
names1 = ['A', 'B', 'C', 'D', 'E']
web1 = Web(matrix1, names1)
web1.drawGraph()
```

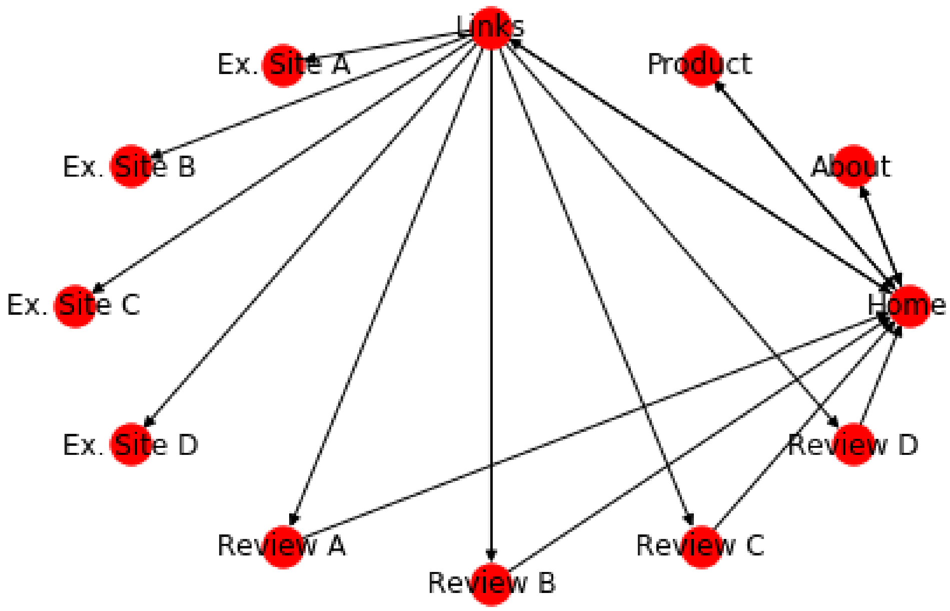


```
In [4]: # HITS Algorithm result for web structure 1

        web1.hits(10)
```

```
A #0  [1, 1, 1, 1, 1]
H #0  [1, 1, 1, 1, 1]

A #1  [0.083, 0.167, 0.333, 0.333, 0.083]
H #1  [0.25, 0.25, 0.083, 0.167, 0.25]

A #2  [0.094, 0.187, 0.344, 0.312, 0.063]
H #2  [0.263, 0.237, 0.105, 0.131, 0.263]

A #3  [0.089, 0.198, 0.337, 0.327, 0.049]
H #3  [0.267, 0.238, 0.099, 0.129, 0.267]

A #4  [0.089, 0.2, 0.337, 0.326, 0.048]
```

```
H #4  [0.27, 0.236, 0.103, 0.121, 0.27]

A #5  [0.088, 0.202, 0.336, 0.329, 0.045]
H #5  [0.271, 0.236, 0.102, 0.121, 0.271]

A #6  [0.088, 0.202, 0.336, 0.329, 0.045]
H #6  [0.271, 0.236, 0.103, 0.119, 0.271]

A #7  [0.088, 0.203, 0.335, 0.329, 0.044]
H #7  [0.271, 0.236, 0.103, 0.119, 0.271]

A #8  [0.088, 0.203, 0.335, 0.329, 0.044]
H #8  [0.271, 0.235, 0.103, 0.119, 0.271]

A #9  [0.088, 0.203, 0.335, 0.329, 0.045]
H #9  [0.271, 0.235, 0.103, 0.119, 0.271]
```

```
In [5]: # Initialising web structure for question 2

        matrix2 = [[0,1,1,1,0,0,0,0,0,0,0,0],
                   [1,0,0,0,0,0,0,0,0,0,0,0],
                   [1,0,0,0,0,0,0,0,0,0,0,0],
                   [1,0,0,0,1,1,1,1,1,1,1,1],
                   [0,0,0,0,0,0,0,0,0,0,0,0],
                   [0,0,0,0,0,0,0,0,0,0,0,0],
                   [0,0,0,0,0,0,0,0,0,0,0,0],
                   [0,0,0,0,0,0,0,0,0,0,0,0],
                   [1,0,0,0,0,0,0,0,0,0,0,0],
                   [1,0,0,0,0,0,0,0,0,0,0,0],
                   [1,0,0,0,0,0,0,0,0,0,0,0],
                   [1,0,0,0,0,0,0,0,0,0,0,0]]
        names2 = ['Home', 'About', 'Product', 'Links', 'Ex. Site A', 'Ex. Site B', 'Ex. Site C
        web2 = Web(matrix2, names2)
        web2.drawGraph()
```



```
In [6]: # HITS Algorithm result for web structure 2

        web2.hits(10)

A #0  [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
H #0  [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

A #1  [0.389, 0.056, 0.056, 0.056, 0.056, 0.056, 0.056, 0.056, 0.056, 0.056, 0.056, 0.056]
H #1  [0.167, 0.056, 0.056, 0.5, 0.0, 0.0, 0.0, 0.0, 0.056, 0.056, 0.056, 0.056]

A #2  [0.157, 0.031, 0.031, 0.031, 0.094, 0.094, 0.094, 0.094, 0.094, 0.094, 0.094, 0.094]
H #2  [0.05, 0.117, 0.117, 0.251, 0.0, 0.0, 0.0, 0.0, 0.117, 0.117, 0.117, 0.117]

A #3  [0.306, 0.016, 0.016, 0.016, 0.081, 0.081, 0.081, 0.081, 0.081, 0.081, 0.081, 0.081]
H #3  [0.048, 0.081, 0.081, 0.468, 0.0, 0.0, 0.0, 0.0, 0.081, 0.081, 0.081, 0.081]

A #4  [0.197, 0.01, 0.01, 0.01, 0.097, 0.097, 0.097, 0.097, 0.097, 0.097, 0.097, 0.097]
H #4  [0.017, 0.108, 0.108, 0.336, 0.0, 0.0, 0.0, 0.0, 0.108, 0.108, 0.108, 0.108]

A #5  [0.264, 0.005, 0.005, 0.005, 0.09, 0.09, 0.09, 0.09, 0.09, 0.09, 0.09, 0.09]
H #5  [0.014, 0.09, 0.09, 0.445, 0.0, 0.0, 0.0, 0.0, 0.09, 0.09, 0.09, 0.09]

A #6  [0.215, 0.003, 0.003, 0.003, 0.097, 0.097, 0.097, 0.097, 0.097, 0.097, 0.097, 0.097]
```

```
H #6  [0.006, 0.102, 0.102, 0.381, 0.0, 0.0, 0.0, 0.0, 0.102, 0.102, 0.102, 0.102]

A #7  [0.245, 0.001, 0.001, 0.001, 0.094, 0.094, 0.094, 0.094, 0.094, 0.094, 0.094, 0.094]
H #7  [0.004, 0.094, 0.094, 0.433, 0.0, 0.0, 0.0, 0.0, 0.094, 0.094, 0.094, 0.094]

A #8  [0.223, 0.001, 0.001, 0.001, 0.097, 0.097, 0.097, 0.097, 0.097, 0.097, 0.097, 0.097]
H #8  [0.001, 0.099, 0.099, 0.404, 0.0, 0.0, 0.0, 0.0, 0.099, 0.099, 0.099, 0.099]

A #9  [0.236, 0.0, 0.0, 0.0, 0.095, 0.095, 0.095, 0.095, 0.095, 0.095, 0.095, 0.095]
H #9  [0.001, 0.095, 0.095, 0.427, 0.0, 0.0, 0.0, 0.0, 0.095, 0.095, 0.095, 0.095]
```