Documentation Q2

Problem Statement

In the given problem statement, we have to implement the PageRank and authority-hub algorithm to calculate the most relevant pages in the given dataset. For this, we have used the Wikivote data consisting of 7115 nodes and 103689 edges.

Import and Graph Creation

We import the given data using the standard file reading in python and then we separate the data into information and Graph part. The Information part contains the information and statistics about the data in the variable **meta** and the Graph connection is in the Graph part. This Graph part is sent to pandas dataframe to create a table with 2 columns; ToNode and FromNode.

This pandas dataframe is used to create a graph using the networkx library of python. We implement the **nx.from_pandas_edgelist** function to create the graph and then convert the graph to directed graph.

PageRank

Reference taken from the Wikipedia page. Initially all nodes have the same probability to be the most relevant page. Using the page rank algorithm, we check the number of outgoing edges from the specific node and divide its current probability by the total number of outgoing edges. This value is added to all the nodes which currently have 0 score and some fixed probability. This step is done for all nodes. Summarizing the formula:

$PR(a) = \Sigma PR(i)/L(i)$ where $L(i)$ is the total number of outgoing edges for that node and $PR(i)$ is the current pagerank score.

We also calculate the Mean squared error between the new PageRank scores and old scores. This is done to stop the convergence when the error becomes smaller than 1e-15. By default, the number of iterations are 100 but the algorithm converges at 20-25 iterations.

Authority and Hub

Reference taken from the Wikipedia page. Authority score is for all the pages pointing to the specific page i.e. incoming edges and hub score is for all the pages being pointed by the specific page i.e. outgoing edges. Initially all values for hub and authority score is one as provided in the resource. I also maintain a buffer authority and hub score which will act as the previous pointer. First authority scores are updated using the formula :

$$\text{auth}(p) = \boldsymbol{\Sigma}_{q \epsilon P_q} \text{hub}(q) \text{ where } P_q \text{ is all the nodes which link to P}$$

Then, hub scores are updated using the formula :

hub(p) = $\mathbf{\Sigma}_{q \in Q_p}$ auth(q) where Q_p is all the nodes which are going out from p to q

We then calculate the mean squared error between the values for both authority and hub scores and then update the buffer. This algorithm runs till the mse is above 1e-20 for both the scores and then terminates. By default the maximum number of iterations is 100.

Results

```
top 10 rank scores :
2565    0.0043372949187308815
11      0.003017206269367328
766     0.002968177479349323
457     0.002963411320667381
4037    0.002878218886740526
1549    0.0028581648714845506
1166    0.002669208905008099
2688    0.0023843472728713416
15      0.002163159726354969
1374    0.002131987766043142

top 10 authority scores :
2565    0.15769611748358103
766     0.13015243025685455
1549    0.12938941353080033
1166    0.11950594168986171
2688    0.11008403659853248
457     0.10999186611635883
3352    0.09179709631226124
11      0.08956574261869124
1151    0.08717924518500951
1374    0.08692950770481205

top 10 hub scores :
2565    0.157696117537377
766     0.13015243029945367
1549    0.12938941344572305
1166    0.11950594165584667
2688    0.11008403661789759
457     0.10999186615700852
3352    0.09179709627666102
11      0.08956574247014454
1151    0.08717924513642718
1374    0.08692950771109112
```