INFO 6205

# Ranking System
for EPL teams

Introduction

What is EPL?

The Premier League, often referred to as the English Premier League or the EPL outside England, is the top level of the English football league system. Contested by 20 clubs, it operates on a system of promotion and relegation with the English Football League.

Aim of the Project

Developed a ranking system which will evaluate a team's ranking based on their past performance. We used Win-loss data and incorporated the PageRank Algorithm to compute the rank of each team. The structure of the historical records forms a large graph of teams connected through their matches played. Each team is represented as the vertex in the graph and edges point the match outcomes between the teams i.e the vertex.

What is PageRank?

PageRank is an algorithm used by Google to rank the webpages in their search engine results. It assesses the importance of a website by examining the importance of a website by examining the importance of other websites referring to it. Similar to this our assumption was that a strong team would be determined by having more wins or against other strong teams. A win against a weaker opponent might not be as important as a close win against an equal contender. Taking these factors into account we utilised PageRank.

Project Description

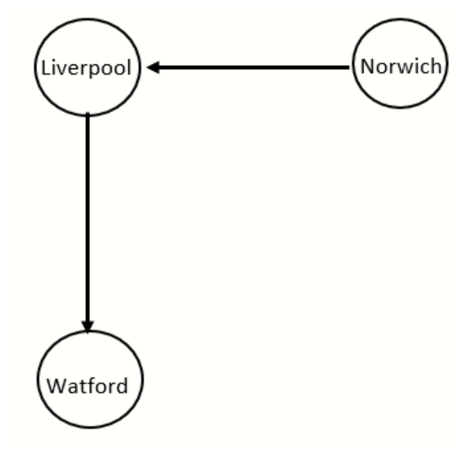
Data Used in the Project

In the project, data of the matches conducted this season have been used. There are a total of 20 teams. Ideally, each team plays 19 matches on home ground and 19 matches away. But due to COVID-19 not all the matches took place so for those matches we tacked this by predicting the outcomes of future football matches by using the past match outcome data.

Implementation

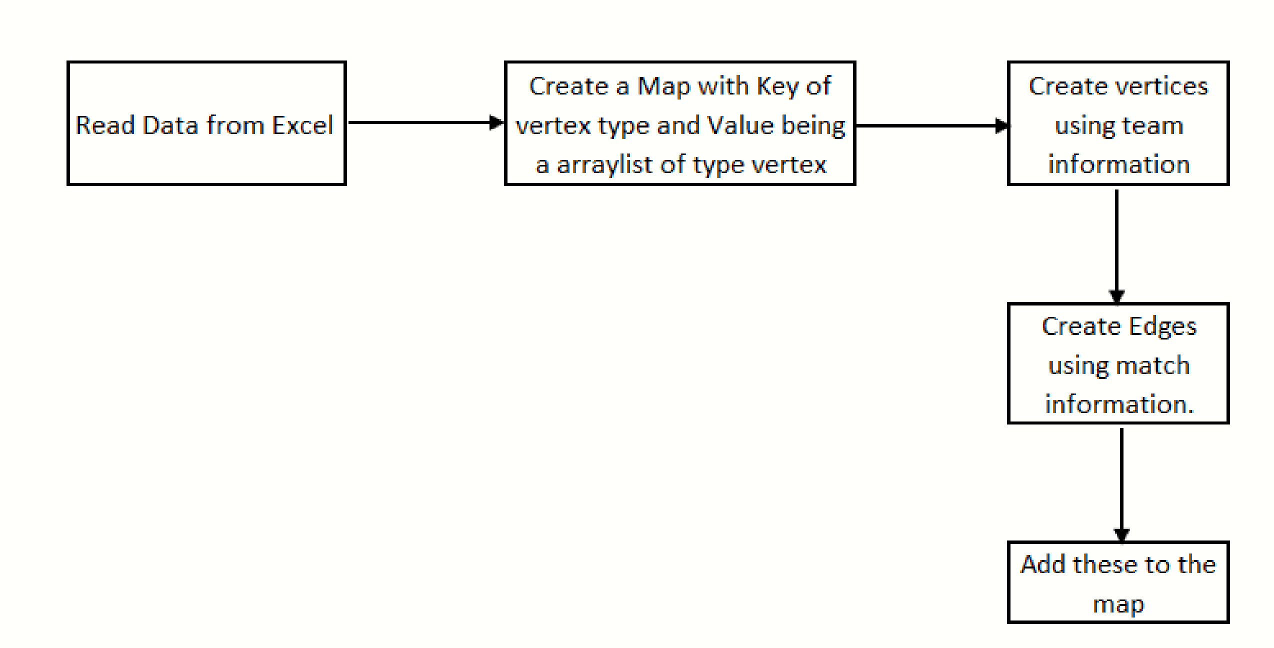
Created an Directed Graph with Vertices representing teams and edges representing a match outcome. The direction of the edge is towards the winning team. From vertex of an edge is a losing team and the To vertex of the same edge is the winning team. Basically the edge points towards the winning team in a match as shown in the figure.

Page Rank checks for the number of outgoing links i.e edges from a vertex and deduces a value for the vertex which is equal to the value of the team. Initially each vertex's weighting is equal to 1/total number of teams or vertices. With each iteration when an edge gets added the value of individual vertices gets distributed accordingly. This allows us to consider the factor like if the team has defeated a strong opponent or a weaker opponent. Hence producing a fair ranking.

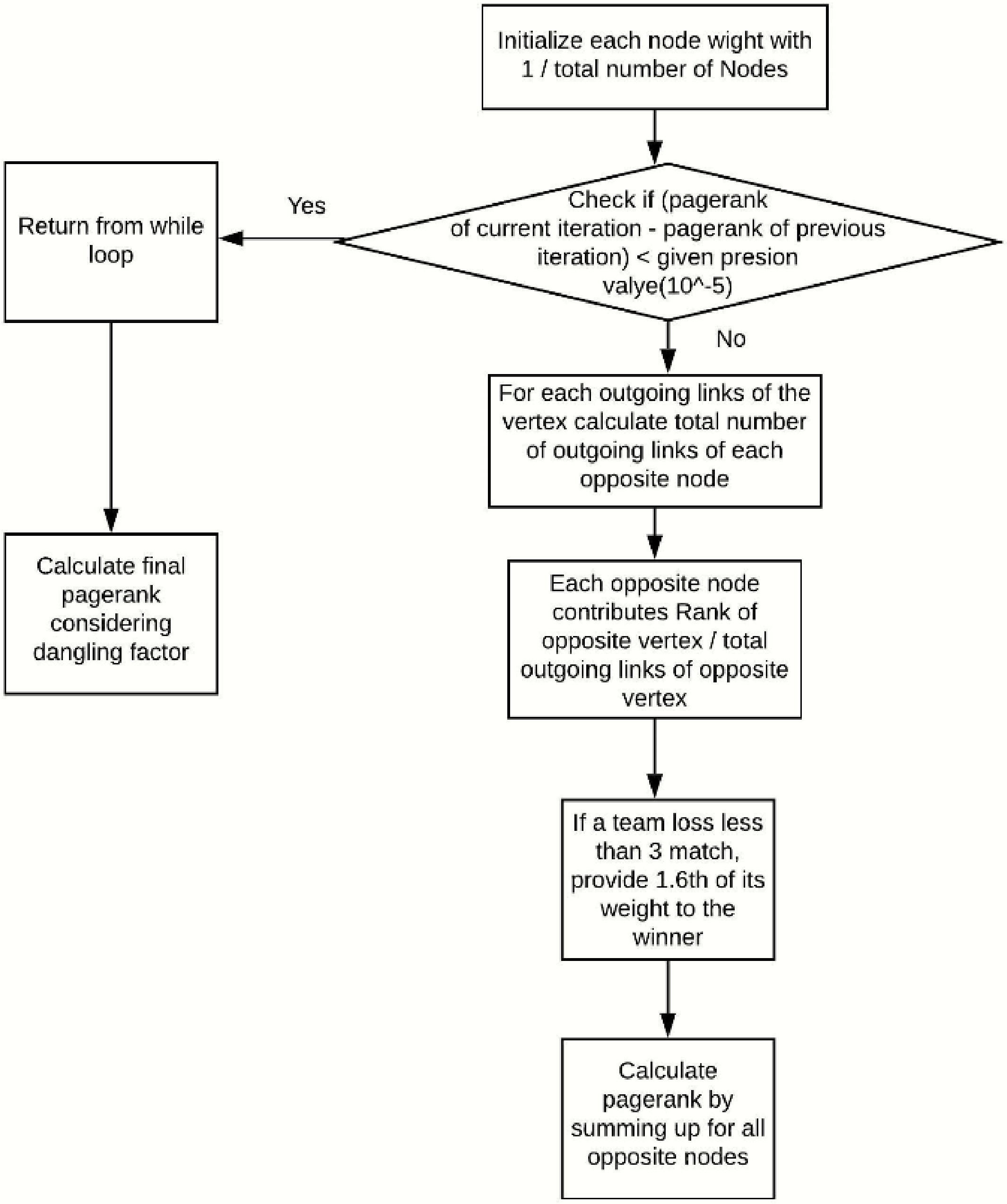


Flow of the Code

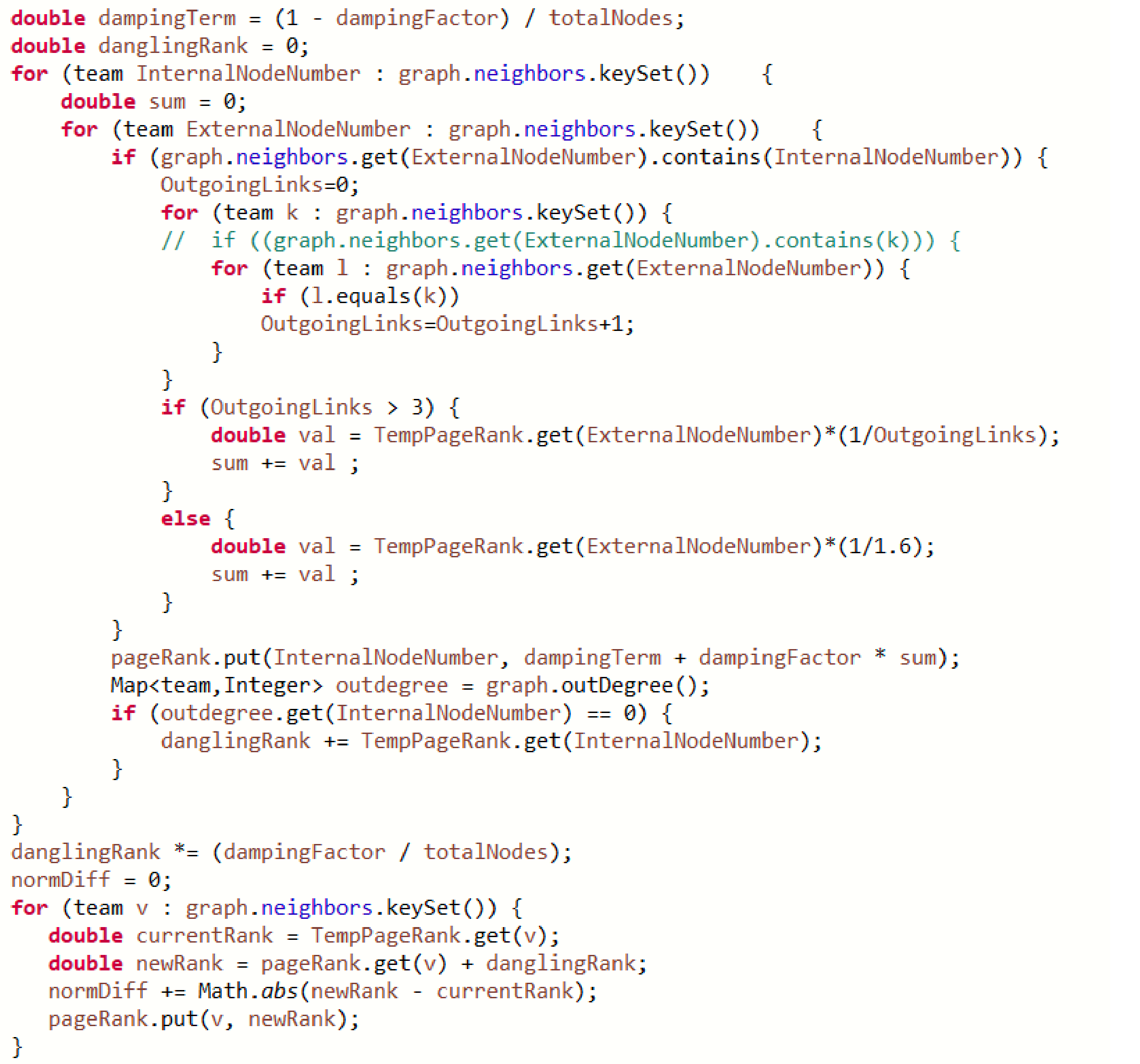
PART 1


Read Data from Excel
Create a Map with Key of vertex type and Value being a arraylist of type vertex
Create vertices using team information
Create Edges using match information.
Add these to the map

Modified PageRank


Initialize each node wight with 1 / total number of Nodes
Check if (pagerank of current iteration - pagerank of previous iteration) < given presion valye(10^-5)
Yes
Return from while loop
Calculate final pagerank considering dangling factor
No
For each outgoing links of the vertex calculate total number of outgoing links of each opposite node
Each opposite node contributes Rank of opposite vertex / total outgoing links of opposite vertex
If a team loss less than 3 match, provide 1.6th of its weight to the winner
Calculate pagerank by summing up for all opposite nodes

Pseudo Code



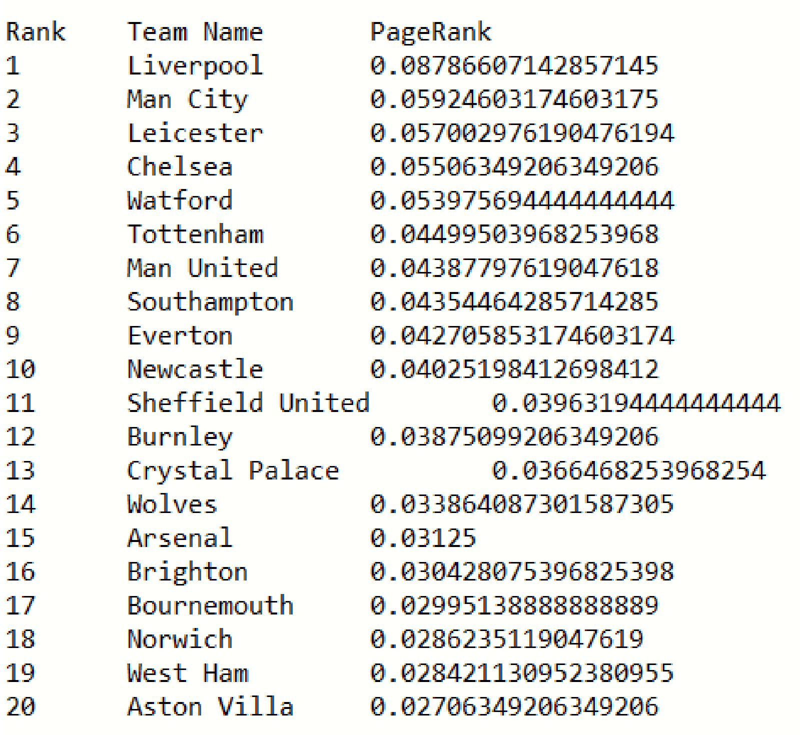
```
double dampingTerm = (1 - dampingFactor) / totalNodes;
double danglingRank = 0;
for (team InternalNodeNumber : graph.neighbors.keySet())    {
    double sum = 0;
    for (team ExternalNodeNumber : graph.neighbors.keySet())    {
        if (graph.neighbors.get(ExternalNodeNumber).contains(InternalNodeNumber)) {
            OutgoingLinks=0;
            for (team k : graph.neighbors.keySet()) {
            //  if ((graph.neighbors.get(ExternalNodeNumber).contains(k))) {
                for (team l : graph.neighbors.get(ExternalNodeNumber)) {
                    if (l.equals(k))
                    OutgoingLinks=OutgoingLinks+1;
                }
            }
            if (OutgoingLinks > 3) {
                double val = TempPageRank.get(ExternalNodeNumber)*(1/OutgoingLinks);
                sum += val ;
            }
            else {
                double val = TempPageRank.get(ExternalNodeNumber)*(1/1.6);
                sum += val ;
            }
        }
        pageRank.put(InternalNodeNumber, dampingTerm + dampingFactor * sum);
        Map<team,Integer> outdegree = graph.outDegree();
        if (outdegree.get(InternalNodeNumber) == 0) {
            danglingRank += TempPageRank.get(InternalNodeNumber);
        }
    }
}
danglingRank *= (dampingFactor / totalNodes);
normDiff = 0;
for (team v : graph.neighbors.keySet()) {
   double currentRank = TempPageRank.get(v);
   double newRank = pageRank.get(v) + danglingRank;
   normDiff += Math.abs(newRank - currentRank);
   pageRank.put(v, newRank);
}
```

Probability Density Function

For Predicting the outcome of matches that did not take place. We tried to implement a Probability density function which will take the mean of goal differences of a team in all the past matches and standard deviation to compute a normal distribution variable. The value of this variable is the deciding factor of the match outcome for each team.

Output

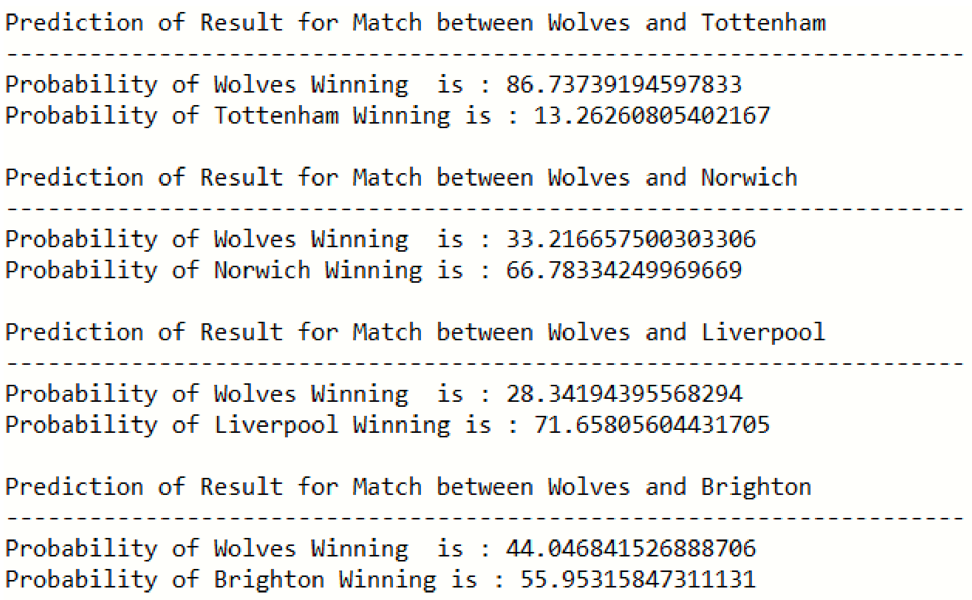
Team Rankings



```
Rank    Team Name       PageRank
1       Liverpool       0.08786607142857145
2       Man City        0.05924603174603175
3       Leicester       0.057002976190476194
4       Chelsea         0.05506349206349206
5       Watford         0.053975694444444444
6       Tottenham       0.04499503968253968
7       Man United      0.04387797619047618
8       Southampton     0.04354464285714285
9       Everton         0.042705853174603174
10      Newcastle       0.04025198412698412
11      Sheffield United        0.039631944444444444
12      Burnley         0.03875099206349206
13      Crystal Palace          0.0366468253968254
14      Wolves          0.033864087301587305
15      Arsenal         0.03125
16      Brighton        0.030428075396825398
17      Bournemouth     0.02995138888888889
18      Norwich         0.0286235119047619
19      West Ham        0.028421130952380955
20      Aston Villa     0.02706349206349206
```

Complete Console Output.pdf

Prediction of outcome of some of the matches using goal differences in their past matches.



```
Prediction of Result for Match between Wolves and Tottenham
---------------------------------------------------------------------
Probability of Wolves Winning  is : 86.73739194597833
Probability of Tottenham Winning is : 13.26260805402167

Prediction of Result for Match between Wolves and Norwich
---------------------------------------------------------------------
Probability of Wolves Winning  is : 33.216657500303306
Probability of Norwich Winning is : 66.78334249969669

Prediction of Result for Match between Wolves and Liverpool
---------------------------------------------------------------------
Probability of Wolves Winning  is : 28.34194395568294
Probability of Liverpool Winning is : 71.65805604431705

Prediction of Result for Match between Wolves and Brighton
---------------------------------------------------------------------
Probability of Wolves Winning  is : 44.046841526888706
Probability of Brighton Winning is : 55.95315847311131
```

Observations

During the implementation of PageRank, it was observed that if a weaker team won against a very strong team weak team once there was a very significant increase in its ranking. Now, it is true that there should be an increase in the ranking but it was observed that the weaker team jumped to a position in top five from which is not really ideal. So to eliminate this issue the code checks if a verted has more than 3 outgoing links the value of the vertex is given as it is. But otherwise it's existing page rank value is divided by 1.6 and this value is added the vertex

During the implementation of the continuous probability density function using goal differences of the team it was observed that the probability of a team ranked lower had a relatively better chance of winning a match. This made us look at the real time rankings of the teams. From this we concluded that a team's ranking cannot entirely depend on the team's match outcomes and that other factors should also be taken into consideration.

Analysis

To produce rankings of the teams in the data set we used PageRank Algorithm. This ranking is purely based on the number of matches won and lost without taking into consideration the draw matches. As shown in the table ranking is very close to the real rankings of the team. We observed that when a weaker team defeats a stronger team in that case the weaker team has a significant increase in it's ranking making the system more fair. In case of the match predictions we observed that teams who have higher ranking have more chances to win based on the goal difference i.e how many goals the team won a match by and their match outcomes.

Conclusions and Future Implementations

To make this system more efficient we believe implementing an edge weighted directed graph should be used. The weightage of the edge could be decided by the no of goals a team has won by or if the match was at the home or away ground. Improvements like giving weightage to matches who's outcome was a tie could be implemented to make this system make more informed decisions.

References

https://web.wpi.edu/Pubs/E-project/Available/E-project-030119-012745/unrestricted/Graph-Based_Sports_Rankings.pdf

https://web.wpi.edu/Pubs/E-project/Available/E-project-031818-152659/unrestricted/GraphBasedSportsRankings_FinalReport.pdf