

Ranking System

-- English Premier League

Group Member:

001349439 Yifan Chen, 001087410 Jingyu Wang, 001300548 Yiyu Chen

1. Problem Statement

Develop a ranking system which is able to evaluate the following expression where x_i, x_j are elements from a set of competing elements X : $P(x_i, x_j)$. where $P(x_i, x_j)$ is the probability that x_i would beat x_j if they met in a head to head matchup at neutral territory.

A by-product of this expression would be the ability to build a table of elements ordered such that if an element x_i appears above another element x_j , then you may infer that $P(x_i, x_j) > 0$. Note that there may be a set of undecidable cycles (like rock, paper, scissors) in which case you will have to mark those elements as equal in your table.

It is desirable also that the value of P is not just a single number, but a probability density function (pdf), in other words there should perhaps be some sort of bounds on the P , maybe a uniform pdf between two values. As you get more data, the bounds will become narrower (i.e. your precision will improve).

The input to your system will be a set of prior encounters with a result. These results can be win-loss or they can be scores, as in for example the English Premier League (EPL). If appropriate, you may also consider home team advantage if you feel that it matters.

2. Problem Analysis

Many aspects of football can influence the outcome of the forecast, and some of them can give the bottom-scoring teams in the standings a great opportunity to beat the top-scoring teams in the standings.

a. The recent situation of the team

The winning team tends to be in high spirits and has an advantage in the next game. Usually, when a team is in the doldrums, problems arise in the locker room and the problems persist.

b. Recent records between two teams in historical battles

Teams like Arsenal have a poor record against teams like Liverpool, Chelsea and Manchester United. Analyze previous games between these teams to see if there is a pattern.

c. Home and away form

Knowing each team's home and away form is the defining factor of football, with some teams not winning all season and some having 12 unbeaten games at home. Statistically, the average team is much better at home than away. Home-field advantage is an important factor in football matches.

d. Time difference, distance, plateau and other factors.

In the World Cup and other international competitions, these factors often need special consideration, but in English Premier League, we do not need to consider these factors.

3. Mathematical Model Building

We combine mechanism analysis methods and test ingesting methods for the Premier League to establish a mathematical model, first of all, according to the understanding of the reality of football matches to analyze its causality, to find out the law reflecting the internal mechanism, need to have practical significance, according to the factors affecting the football match, we can give each team a ranking value, in the teams category of attributes for rank, the initial value of 100, according to the results of 288 games before the 2020-2021 season, Calculate the rank value of the 20 teams at the end of the current tournament. The rules are as follows:

A:B(A is home team)	Win	Tie	Lose
A.points > B.points	(+5,-7)	(-3,+2)	(-7,+4)
A.points = B.points	(+5,-5)	(-1,+1)	(-7,+7)

A.points < B.points	(+7,-5)	(+0,-0)	(-4,+7)
-------------------------------	---------	---------	---------

Remarks: points are current points for the team(win+3,tie+1,lose do nothing).

We see the two-team game as a "black box" system, because the strength of the two teams can not be directly judged, so the internal mechanism can not be directly sought. Therefore, we can data the data we consider before, such as the source of data, such as rank, points, the number of average goal each teams get accordingly etc., and use statistical analysis methods on this basis, according to the predetermined guidelines in a certain type of model to choose the best fit with the data model.

Then we also calculate the possibility of wins and losses between Team x_i and x_j , which is $P(x_i, x_j)$ when x_i is a home team and x_j is an away team. Considering the factor that the average team is much better at home than away, $P(x_i, x_j)$ is different from (x_j, x_i) . Also we give different weights depending on the year of the game. Because as time goes on, the team's performance changes back, so the more recent the game, the more representative the team's ability. This is described in detail in the following methods. Our group calls it a lottery pool.

4. Algorithm design and selection

According to what we build for the analysis. We choose to use the BP (back propagation) algorithm which is a kind of supervised learning algorithm. Because when we give some input learning sample, use reverse propagation algorithm to adjust the power and deviation of the network repeatedly, so that the output array and the expected array as close as possible, when the error square of the network output layer is less than the specified error when the training is completed, save the network's weight and deviation.

First, use the data we collect and calculate to do the initialization, randomly given each connection right w , v and the threshold θ_i , r_t .

Then the output of the cells of the hidden layer and output layer is calculated by the given input and output mode $b_j = f(w_{jai} - \theta_j)$ $c_t = f(v_{jt} b_j - r_t)$.

b_j is the actual output of the first neuron of the hidden layer, c_t is the actual output of the t neuron in the output layer, w_{ij} is the connection right from the input layer to the hidden layer, and v_{jt} is the connection right from the hidden layer to the output layer.

Lastly the BP algorithm selects the next input mode to return to the previous step and repeat training until the network set output error to reach the required end of training. In this case, the traditional BP algorithm helps us to turn a set of sample input/output problems into a nonlinear optimization problem.

We use java's framework Encog to realize machine learning . In Encog, the creation of neural networks mainly uses the basic Network and BasicLayer classes, and of course other classes such as activation functions.

First, Import related packages using maven.

```
<dependencies>
  <dependency>
    <groupId>org.encog</groupId>
    <artifactId>encog-core</artifactId>
    <version>3.4</version>
  </dependency>
</dependencies>
```

Constructing a neural network: It's especially easy to build a neural network with Encog, first by first you need a New BasicNetwork object and then to add layers to the object. The first parameter of the BasicLayer constructor specifies the activation function, the second parameter specifies the number of neurons by bias neurons, the third parameter specifies the number of neurons, if only one number is passed, means that only the number of neurons is specified, and the default activation function is Sigmoid, with biased neurons.

```
BasicNetwork nw=new BasicNetwork();
nw.addLayer(new BasicLayer(null,true,6));
nw.addLayer(new BasicLayer(new ActivationSigmoid(),true,2));
nw.addLayer(new BasicLayer(new ActivationSigmoid(),false,2));
nw.getStructure().finalizeStructure();
nw.reset();
```

Because paranoid neurons affect the next layer, and the activation function affects the data of the previous layer. So generally the first layer (input layer) does not need to activate the function, and the last layer (output layer) does not need to bias neurons. The call finalizeStructure indicates that the build has been completed and that there is no need to add layers to it, and reset is the random initialization of the connection weights between the layers.

Processing data:

Ready to enter data and the data we expect to output

Construct an MLDataSet object training set, and we pass our training data to it.

```
MLDataSet ts=new BasicMLDataSet(input,ideal);
```

Input layer:

```
public static double [][]input= new double[6748][6];
```

Output layer:

```
public static double[][] output =new double[6748][2];
```

We assume that [0,6] is a normal point in a match. So we divided the goals by 6 so that the output will be normalized.

```
public static double[][] output1 =new double[6748][2];
```

Hidden layers: In the previous discussion, when neural networks contained the input layer and output, sometimes the output layer and the input layer were the same, but most were usually two separate layers, and there might be other layers between the input layer and the output layer, which is called the hidden layer, and the hidden layer is between the input layer and the output layer. Hidden layers can also take on more complex structures.

Using Elastic Propagation (RPROP) training: We train our network in this program using Elastic Communication Training (RPROP), which is the best training algorithm supported by Encog, which of course offers other training techniques and can solve certain problems with certain training techniques. The code for rPROP training is as follows:

```
MLTrain t=new ResilientPropagation (nw,ts);
int epoch=1;
do {
    t.iteration();
    epoch++;
}while(t.getError()>0.2);
```

The above code passes through multiple iterations, and when iterate to make the neural network error rate below 1, the neural network training is over. Here we can run our code to see the results of the training:

```
Console [x]
Predictor [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe
epoch: 1 error:0.09326706782490767
epoch: 2 error:0.06556486976311048
epoch: 3 error:0.04768347928469557
epoch: 4 error:0.04318143687545494
epoch: 5 error:0.046136467737943575
epoch: 6 error:0.042301856202304707
epoch: 7 error:0.042766018851111955
epoch: 8 error:0.04264146139864331
epoch: 9 error:0.04141631839586956
epoch: 10 error:0.04110295118688753
epoch: 11 error:0.04058149924394296
epoch: 12 error:0.0399392081523832
epoch: 13 error:0.0393148673638024
epoch: 14 error:0.03916052605779564
epoch: 15 error:0.03839425774314036
epoch: 16 error:0.03812632814584137
epoch: 17 error:0.03844791509275413
epoch: 18 error:0.037724261404013484
epoch: 19 error:0.040611444264135585
epoch: 20 error:0.0378490520551049
epoch: 21 error:0.03844217421939231
epoch: 22 error:0.03879365865698156
epoch: 23 error:0.03853178657265089
epoch: 24 error:0.03859929638307023
epoch: 25 error:0.03843904239188276
epoch: 26 error:0.03890237718609074
epoch: 27 error:0.03879137428594205
epoch: 28 error:0.03851504526762889
epoch: 29 error:0.03853144383798943
epoch: 30 error:0.03862715196377156
epoch: 31 error:0.03865069693278187
epoch: 32 error:0.03866251014574778
epoch: 33 error:0.038859585759696176
epoch: 34 error:0.03884844320748463
epoch: 35 error:0.03858260949860396
epoch: 36 error:0.038575121204714125
epoch: 37 error:0.03863511296689443
epoch: 38 error:0.038684269707972764
epoch: 39 error:0.03862056526134206
epoch: 40 error:0.03855711557870822
```

The error rate is high during the first iteration, and then decreases as the number of iterations increases. Each run turns out to be different, but as long as the error rate is decreasing, our neural network is built successfully.

Above is one method we use. (Totally using BP algorithm to predict)

The other method is combining BP algorithm and Lottery pool algorithm.

a. data structure

Creating an array of Team types and store all team information in the array. Team [] teams;

Creating an array of Match types, reading all match information for 20 years than stored in this array. Match [] matches;

(BP) Creating a Two-dimensional array to store the information as an input for training as a neural network error reverse-propagation algorithm. Int [][] input;

Creating a Two-dimensional array as an output to store the result of a neural network error reverse-propagation algorithm. Int [][] output;

(Calculate P(xi,xj)) Creating a hashmap store for the number of the home and away field as the key value, the value is a hashmap that stores all the scores of this team, and key value of inner hashmap is the score, and the value is the number of times the score appears. (Factors like home and away , game time determine the weight of the number of occurrences)

HashMap <String, HashMap<String, Integer>() map;

5. Algorithmic indicators

a. Correctness: We will represent the correctness of the two algorithms in the test (title 7)

b. Efficiency and storage demand:

This is a simple three-tier BP neural network, the number of neurons per layer is n1, n2, n3. Two matrix operations are performed, and the two matrix multiplications (actually vectors and matrix multiplications) are performed n1 n2 and n2 n3 etc calculations, respectively. Since the number of junctions (n1 and n3) of the input layer and the final output layer is determined, it can be considered a constant, and the hidden layer n2 in the middle can be set by itself.

Therefore, the time complexity of calculating a feed-forward for a sample should be O (n1*n2+n2*n3). The time complexity and feed-forward calculation slot in reverse propagation are the same, what here are a total of m training samples,

each of which is trained only once, then the time complexity of training a neural network should be O (92*n^2). (92 is a constant number which can be ignored).

The time of using HashMap to find record is O(1). And predicting matches needs to iterate the outer hashmap and iterate the inner hashmap. Therefore , the time of predicting is O($\frac{1}{2} n^2$).

6. Algorithm implementation

Classes:

Team	Matches
teamName : String; teamNo : int; rank : int=100(default); points : int; Goaldifference : int; totalgoal : int; totalloss : int; avggoal : double; avgloss : double; getGoaldifference() : return int; setGoaldifference(int goaldifference); getTeamNo() : return int; setTeamNo(int teamNo) ; getRank() : return int; setRank(int rank) ; getPoints() : return int; setPoints(int points) ; getAvggoal() : return double; setAvggoal(double avggoal) ; getAvgloss() : return double; setAvgloss(double avgloss) ;	round : int ; dateyear : int ; homeTeam : Team; AwayTeam : Team; homegoal : int; awaygoal : int ; win : int(-1,0,1); getdateyear() : return int; setyear(int dateyear); getRound() : return int; setRound(int round) ; getHomeTeamNo() :return String; setHomeTeamNo(Team homeTeam) ; getAwayTeamNo() :return int ; setAwayTeamNo(Team awayTeamNo); getHomegoal() : return int; setHomegoal(int homegoal) ; getAwaygoal() : return int; setAwaygoal(int awaygoal) ; getWin() return int; setWin(int win) ;

Team:

Field Name	Description
teamName	The current team's name
teamNo	The current team's No
rank	The number of current team's ability
points	The points of current team

Goldifference	The number of current team's goal difference
totalgoal	The number of total goals of current team
totalloss	The number of total conceded goals of current team
avggoal	The number of average goals of current team
avgloss	The number of average conceded goals of current team
played;	The number of matches have played by current team

Match:

Field Name	Description
round	The round of the match
homeTeamNo	The number of the home team
AwayTeamNo	The number of the away team
homegoal	The number of the home team's goal
awaygoal	The number of the away team's goal
win	The result of the match. 1 means home team win, 0 means draw, -1 means away team win

Method1:

Field Name	Description
int numTeams	Set to 20, final. The number of total teams from 2001-2020. The first 20 is the team of 2019-2020
Team[] teams	The array that has to be filled up with all of the team names and stats.
Team[] trainteams	The array that has to be filled up with all of the team names and stats for BP training.
int numMatchs	Set to 380, final. The number of total matches of 2019-2020
Matches[] matches	The array that has to be filled up with all of the matches and stats of 2019-2020.
double [][]input	The array that has to be filled up with converted input information for BP training.

double [][]output	The array that has to be filled up with converted output information for BP training.
Int success	The number of correct prediction.
double[][] outputcompressed	Normalize the output.
int[][] points	The array that has to be filled up the ranking of 2018-2019 after the prediction
HashMap<String,HashMap<String,Integer>>	<p>The outer hashmap key is String which contains hometeam number and away team number and the outer hashmap value is a hashmap.</p> <p>And the inner hashmap key is String contains score and value is Number of occurrences multiply weight.</p>

Method Name	Description
public static void readteams()	The method to read the team information form the csv files and put it into the teams array.
public static void readmatch()	The method to read the team information form the csv files and put it into the matches hashmap.
public static HashMap<String,HashMap<String,Integer>> calculatewithweight (Matches1[] matches)	The method to read the remaining matches from the csv files and put the round, home team and away team. Record the record between two teams. If the date of record before 2015, the weight is 2. If the date of record between 2016 and 2010 , the weight is 4. The other weight is 6. Also score should be recorded reversely in reversed team number at the same time the weight should be divided by 2 respectively.
HashMap<String,HashMap<String,Integer>> calculatewithbp(Matches1[] matches)	Combine the method above and BP algorithm when there are no record between two teams

public static ArrayList<Integer> predict1(int a,int b,double[][] input,double[][] ideal)	The method to predict matches through BP. The main factor is the abilities(rank), points and goals difference of the home team and the away team.
public static void renew(Matches2 m)	The method to renew the team information after reading a match's result.
public static void BP(double [][] data , double [][] target	The method to train a network to predict the matches.
static void result(HashMap<String,HashMap<String,Integer>> res)	Read the array which stores the remaining matches. Take out the number of the home team and the away team. Get a hashmap which contains all records of matches between the two teams before. And give a random number to decide what score they would have this time.

Method2:

Field Name	Description
int numTeams	Set to 40, final. The number of total teams from 2001-2020. The first 20 is the team of 2019-2020
Team2[] teams	The array that has to be filled up with all of the team names and stats.
Team2[] trainteams	The array that has to be filled up with all of the team names and stats for BP training.
int numMatches	Set to 380, final. The number of total matches of 2019-2020
Matches2[] matches	The array that has to be filled up with all of the matches and stats of 2019-2020.
Matches2[] trainmatches	The array that has to be filled up with all of the matches and stats for BP training.
double [][]input	The array that has to be filled up with converted input information for BP training.
double [][]output	The array that has to be filled up with converted output information for BP training..
Int success	The number of correct prediction.

int[] stats	The 220 stats, 11 per team, are inputted into this array, as they read into a file.
int[][] points	The array that has to be filled up the ranking of 2018-2019 after the prediction

Method Name	Description
public static void read()	The method to read the team information form the csv files and put it into the teams array.
public static void trainread()	The method to read the team information form the csv files and put it into the trainteams array.
public static void predictmatch()	The method to read the remaining matches from the csv files and put the round, home team and away team into the matches array. And use the BP prediction or elo prediction to predict a result of the match then put it into the array. At last, renew the ranking of the teams.
public static void print() throws IOException	The method to get the total inputs for BP training and get a CSV file
public static ArrayList<Integer> predict1(int a,int b,double[][] input,double[][] ideal)	The method to predict matches through BP. The main factor is the abilities, average goals and average conceded goals of the home team and the away team.
public static int predict(int a, int b)	The method to predict the matches through elo prediction. The main factor is the team's current ranking.
public static void readmatch()	The method to read the match information form the csv files and put it into the matches array.
public static void trainmatch()	The method to read the match information form the csv files and put it into the trainmatches array.

public static void renew(Matches2 m)	The method to renew the ranking after reading a match's result.
public static void renew1(Matches2 m)	The method to renew the team information after reading a match's result.
public static void BP(double [][] data , double [][] target	The method to train a network to predict the matches.

7. Algorithm improvements

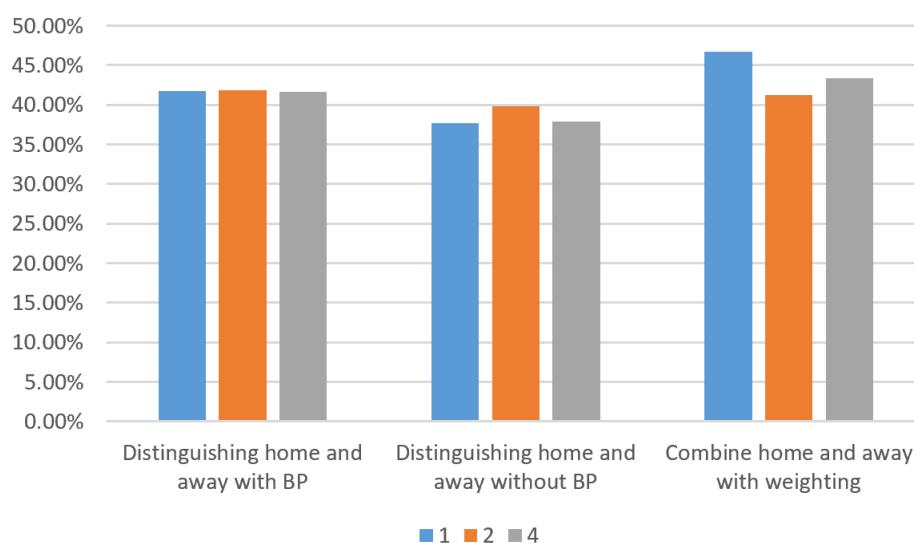
Override the sorting for teams according to their points. When points are the same , using the Goal difference.

8. Testing

We predict the last 10, 20 and 40 matches in 2018-2019 premier league and calculate the accuracy.

Method_A (combing lottery pool and BP algorithm) :

Predicted rounds	1	2	4
Distinguishing home and away with BP	41.80%	41.90%	41.70%
Distinguishing home and away without BP	37.70%	39.80%	37.90%
Combine home and away with weighting	46.70%	41.20%	43.40%



Some short cuts:

Screenshot 1 (Top): Eclipse IDE - FinalFor20YEARS/src/EFL1/New.java

```

eclipse-workspace - FinalFor20YEARS/src/EFL1/New.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Pack... Type... JUnit RandomWalk.java Employee.java RandomWalk.java NewtonApprox... QuickFind.java QuickUnion... JUnit.java test.java Team.java match.java New.java
FinalFor20YEARS EFL1
src
  EFL1
    BpDeep.java
    match.java
    New.java
    Team.java
Maven Dependencies
bin
target
  2000-2001.csv
  2002-2003.csv
  2003-2004.csv
  2004-2005.csv
  2005-2006.csv
  2006-2007.csv
  2007-2008.csv
  2008-2009.csv
  2009-2010.csv
  2010-2011.csv
  2011-2012.csv
  2012-2013.csv
  2013-2014.csv
  2014-2015.csv
  2015-2016.csv
  2016-2017.csv
  2017-2018.csv
  2018-2019.csv
  2019-2020.csv
  Book11.csv
  Book2.csv
  BOOK3.csv
  BOOK4.csv
  pom.xml
FirstTestNGProject
  info6205_util
  MultipleThreads
  neuralnet

```

```

62 // System.out.println(matches[i].getdateyear()+" "+matches[i].getHomeTeamNo()+" "+matches[i].getAwayTeamNo());
63 }
64
65 for(int k=0;k<100;k++) {
66
67     HashMap<String,HashMap<String, Integer>> res= calculatewithbp(matches);
68     //HashMap<String,HashMap<String, Integer>> res= calculatewithhomeandawayobp(matches);
69     //HashMap<String,HashMap<String, Integer>> res= calculatewithweight(matches);
70
71     for (String key : res.keySet()) {
72
73         HashMap<String, Integer> pointsmap= res.get(key);
74
75         //System.out.println("Key = " + key[0]+key[1] + ", Value = " + value);
76
77     }
78
79     for(int i=6000;i<7210;i++) {
80         if(matches[i].dateyear==2018) {
81
82             //System.out.println(i);
83             int countmatches=0;

```

Console

```

terminated> New [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Apr 11, 2020, 6:46:32 PM – 6:46:33 PM)
1 1
Tottenham :Everton
4 0
4 0
Watford :West Ham
2 0
2 0
success418
Man City 9800
Liverpool 9586
Chelsea 7313
Tottenham 7201
Arsenal 6986
Man United 6906
Wolves 5814
Everton 5360
Watford 5216
Leicester 5187
West Ham 4957

```

Screenshot 2 (Bottom): Eclipse IDE - FinalFor20YEARS/src/EFL1/New.java

```

eclipse-workspace - FinalFor20YEARS/src/EFL1/New.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Pack... Type... JUnit RandomWalk.java Employee.java RandomWalk.java NewtonApprox... QuickFind.java QuickUnion... JUnit.java test.java Team.java match.java New.java
FinalFor20YEARS EFL1
src
  EFL1
    BpDeep.java
    match.java
    New.java
    Team.java
Maven Dependencies
bin
target
  2000-2001.csv
  2002-2003.csv
  2003-2004.csv
  2004-2005.csv
  2005-2006.csv
  2006-2007.csv
  2007-2008.csv
  2008-2009.csv
  2009-2010.csv
  2010-2011.csv
  2011-2012.csv
  2012-2013.csv
  2013-2014.csv
  2014-2015.csv
  2015-2016.csv
  2016-2017.csv
  2017-2018.csv
  2018-2019.csv
  2019-2020.csv
  Book11.csv
  Book2.csv
  BOOK3.csv
  BOOK4.csv
  pom.xml
FirstTestNGProject
  info6205_util
  MultipleThreads
  neuralnet

```

```

64
65
66 for(int k=0;k<100;k++) {
67
68     // HashMap<String,HashMap<String, Integer>> res= calculatewithbp(matches);
69     HashMap<String,HashMap<String, Integer>> res= calculatewithhomeandawayobp(matches);
70
71     //HashMap<String,HashMap<String, Integer>> res= calculatewithweight(matches);
72
73     for (String key : res.keySet()) {
74
75         HashMap<String, Integer> pointsmap= res.get(key);
76
77         //System.out.println("Key = " + key[0]+key[1] + ", Value = " + value);
78
79     }
80
81     for(int i=6000;i<7210;i++) {
82         if(matches[i].dateyear==2018) {
83             //System.out.println(i);
84             int countmatches=0;

```

Console

```

terminated> New [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Apr 11, 2020, 6:38:29 PM – 6:38:30 PM)
0 0
0 0
Watford :West Ham
2 0
2 0
success377
Man City 9747
Liverpool 9570
Chelsea 7311
Tottenham 7176
Arsenal 6956
Man United 6807
Wolves 5813
Everton 5389
Watford 5185
Leicester 5182
West Ham 4956
...
```

eclipse-workspace - FinalFor20YEARS/src/EFL1/New.java - Eclipse IDE

```
File Edit Source Refactor Navigate Search Project Run Window Help
RandomWalk.java Employee.java RandomWalk.java NewtonApprox... QuickFind.java QuickUnion... JUnit.java test.java Team.java match.java New.java

FinalFor20YEARS
  JRE System Library [JavaSE-1.8]
    src
      EFL1
        BpDeep.java
        match.java
        New.java
        Team.java
      Maven Dependencies
      bin
    target
      2000-2001.csv
      2002-2003.csv
      2003-2004.csv
      2004-2005.csv
      2005-2006.csv
      2006-2007.csv
      2007-2008.csv
      2008-2009.csv
      2009-2010.csv
      2010-2011.csv
      2011-2012.csv
      2012-2013.csv
      2013-2014.csv
      2014-2015.csv
      2015-2016.csv
      2016-2017.csv
      2017-2018.csv
      2018-2019.csv
      2019-2020.csv
      Book11.csv
      Book2.csv
      Book3.csv
      Book4.csv
      pom.xml
FirstTestNGProject
info6205_util
MultipleThreads
neuralnet

RandomWalk.java Employee.java RandomWalk.java NewtonApprox... QuickFind.java QuickUnion... JUnit.java test.java Team.java match.java New.java

public static void main(String[] args) throws IOException {
    // TODO Auto-generated method stub
}

public static void main(String[] args) {
    // readteams();
    for(int i=0;i<41;i++) {
        //System.out.println(teams[0]);
        System.out.println(teams[i].teamName+ " "+teams[i].teamName);
    }
}

public static Map<String,Team> teammap=new HashMap<String,Team>();
public static int bifen[][]=new int [20][2];
public static int success;

readteams();
for(int i=0;i<41;i++) {
    //System.out.println(teams[0]);
    System.out.println(teams[i].teamName+ " "+teams[i].teamName);
}

}

Console
<terminated> New Java Application C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Apr 11, 2020, 9:34:57 PM - 9:34:58 PM)
0 3
0 3
Newcastle :Liverpool
1 0
1 0
Chelsea :Watford
4 0
4 0
Huddersfield :Man United
2 1
2 1
Arsenal :Brighton
2 0
2 0
Man City :leicester
2 1
2 1
success839
Man City :leicester
```

eclipse-workspace - FinalFor20YEARS/src/EFL1/New.java - Eclipse IDE

```
File Edit Source Refactor Navigate Search Project Run Window Help
RandomWalk.java Employee.java RandomWalk.java NewtonApprox... QuickFind.java QuickUnion... JUnit.java test.java Team.java match.java New.java

FinalFor20YEARS
  JRE System Library [JavaSE-1.8]
    src
      EFL1
        BpDeep.java
        match.java
        New.java
        Team.java
      Maven Dependencies
      bin
    target
      2000-2001.csv
      2002-2003.csv
      2003-2004.csv
      2004-2005.csv
      2005-2006.csv
      2006-2007.csv
      2007-2008.csv
      2008-2009.csv
      2009-2010.csv
      2010-2011.csv
      2011-2012.csv
      2012-2013.csv
      2013-2014.csv
      2014-2015.csv
      2015-2016.csv
      2016-2017.csv
      2017-2018.csv
      2018-2019.csv
      2019-2020.csv
      Book11.csv
      Book2.csv
      Book3.csv
      Book4.csv
      pom.xml
FirstTestNGProject
info6205_util
MultipleThreads
neuralnet

RandomWalk.java Employee.java RandomWalk.java NewtonApprox... QuickFind.java QuickUnion... JUnit.java test.java Team.java match.java New.java

if(i == 0) {
    i = arg1.getTotalgoal()-arg0.getTotalgoal();
}
return i;
};

for(Team t:teams) {
    System.out.println(t.teamName+" "+t.points);
}

}

public static void readteams() {
    try {
        BufferedReader reader = new BufferedReader(new FileReader("C:\\Users\\lcheny\\eclipse-workspace\\FinalFor20YEARS\\Book2.csv"));
        String line = null;
        int n =0;
        while((line=reader.readLine())!=null){
            String item[] = line.split(",");
}

Console
New Java Application C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Apr 11, 2020, 9:19:33 PM)
0 1
0 1
Newcastle :Southampton
2 2
2 2
Everton :Man United
1 3
1 3
Arsenal :Crystal Palace
2 0
2 0
Cardiff :Liverpool
1 4
1 4
Chelsea :Burnley
1 1
1 1
success1736
Liverpool :OFC
```

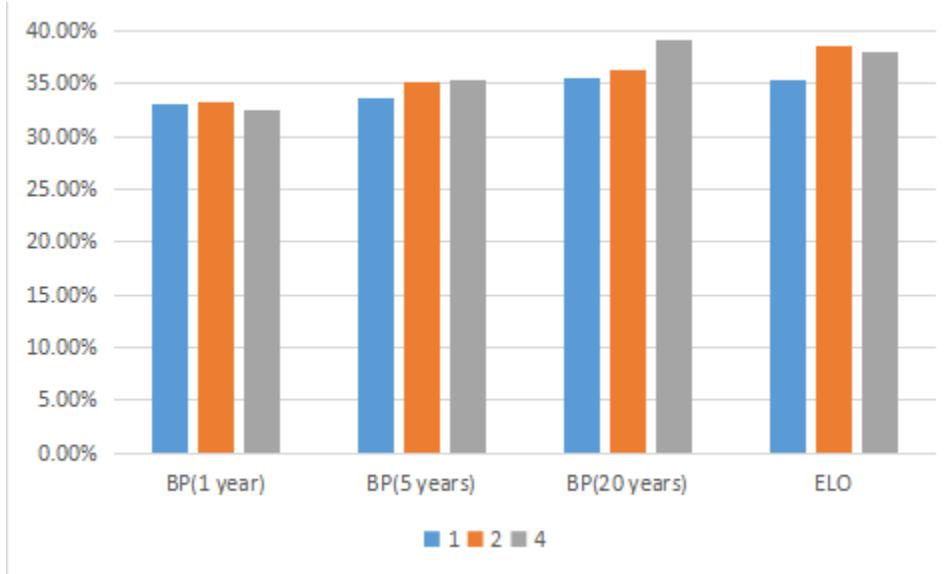
Conclusion:

1. **The more games you record, the fewer games you predict, the more accurate.**
2. **Combination makes the algorithms more stable**

Method_B (Using BP algorithm):

Test Result of BP (which learning 1 year ,5 years and 20 years accordingly) and ELO

Round(count backwards)	1	2	4
BP(1 year)	33.10%	33.20%	32.55%
BP(5 years)	33.70%	35.25%	35.30%
BP(20 years)	35.50%	36.37%	39.10%
ELO	35.30%	38.50%	38.05%



File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer EFL JRE System Library [JavaSE-1.8] EPL/pom.xml Predictor.class BasicNetwork.class

```

131//      }
132//    }
133//  }
134//  for(Team t : teams)
135//  {
136//    System.out.println(t.teamName);
137//    System.out.println(t.getGoals());
138//  }
139//}
140// BP(input1,output2);
141//}
142//Int n = success();
143System.out.println(success);
144}
145}
146}
147}
148public static void read()
149{
150try {
151  BufferedReader reader = new BufferedReader(new FileReader("C:\\Users\\Yifan Chen\\eclipse-workspace\\EFL\\train1.csv"));
152  String line;
153  int n=0;
154  while((line=reader.readLine())!=null){
155    String last = item[1].length-1; //0x00E0A000-0x00000000-0x00000000-0x00000000
156    String[] item = line.split(",");
157    int homeTeamNo = Integer.parseInt(item[0]);
158    int awayTeamNo = Integer.parseInt(item[1]);
159    int goalsDifference = Integer.parseInt(item[2]);
160    //System.out.println(item[0]);
161    //System.out.println(item[1]);
162    Team team = new Team();
163    team.teamNo = Integer.parseInt(item[0]);
164    team.teamName = item[1];
165    team.goalsDifference = 0;
166    //team.points = Integer.parseInt(item[2]);
167    n++;
168  }
169}

```

Tasks Console

Predictor [Java Application] C:\Program Files\Java\jre1.8.0_241\bin\javaw.exe (2020年4月11日下午3:57:50)

```

Manchester United Cardiff 2 4
Southampton Huddersfield 2 0
Tottenham Hotspur Everton 4 1
West Bromwich Albion 1 3
Brighton Manchester City 1 3
Burnley Arsenal 2 3
Crystal Palace Southampton 1 4
Fulham Newcastle United 1 3
Leicester City Chelsea 4 1
Liverpool Wolverhampton Wanderers 2 2
Manchester United Cardiff 2 1
Southampton Huddersfield 1 0
Tottenham Hotspur 1 1
Watford West Ham United 1 4
337

```

Writable Smart Insert 142:9:4171

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer EFL JRE System Library [JavaSE-1.8] EPL/pom.xml Predictor.class BasicNetwork.class

```

224String home = teams[matchIndex].homeTeamNo.getTeamName();
225int a = matches[matchIndex].homeTeamNo;
226int b = matches[matchIndex].awayTeamNo;
227String away = teams[matchIndex].awayTeamName();
228int b1 = matches[matchIndex].awayTeamNo;
229
230double output[1][1]=new double [1000][2];
231for(int i=0;i<1000;i++)
232{
233  output[1][0]=output[1][1]/6;
234  output[1][1]=output[1][1]/6;
235}
236
237ArrayList<Integer> l=predict(a,b,input1,output2);
238
239matches[matchIndex].win=0;
240//predict(matches[n].homeTeamNo,matches[n].AwayTeamNo);
241//team[matches[n].homeTeamNo].goalsDifference=predict(a,b,input1,output2).get(1);
242//team[matches[n].awayTeamNo].goalsDifference=predict(a,b,input1,output2).get(2);
243//matches[n].homeGoal=l.get(1);
244//matches[n].awayGoal=l.get(2);
245//predict(a,b,input1,output2).get(0);
246//System.out.println("home "+home);
247//int c = predict(a,b);
248//matches[n].win = c;
249System.out.println(home+" "+away+" "+matches[n].homegoal+" "+matches[n].awaygoal+" ");
250//new(matches[n]);
251n++;
252}
253}
254} catch (Exception e) {
255e.printStackTrace();
256}
257}
258}
259public static void input(match m,int n)
260{
261  input[n][0]= trainteams[n].homeTeamNo.rank;
262}

```

Tasks Console

terminated Predictor [Java Application] C:\Program Files\Java\jre1.8.0_241\bin\javaw.exe (2020年4月11日下午4:06:43 - 下午4:07:20)

```

Wolverhampton Wanderers Fulham 3 2
Arsenal Burnley 3 2
Chelsea Watford 3 2
Huddersfield Manchester United 3 1
Henslowe Leicester City 4 1
Brighton Manchester City 1 2
Burnley Arsenal 1 2
Crystal Palace Southampton 4 2
Fulham Newcastle United 1 3
Leicester City Chelsea 1 3
Liverpool Wolverhampton Wanderers 2 2
Manchester United Cardiff 1 3
Southampton Huddersfield 4 1
Tottenham Hotspur Everton 3 1
Watford West Ham United 1 4
262

```

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer ERL

- ERL
 - BpDeep.java
 - matchjava
 - Predictor.java
 - Team.java
 - EPL/pom.xml
 - Predictor.class
 - BasicNetwork.class
- Maven Dependencies
- bin
- target
- Book1.csv
- Book2.csv
- BOOK3.csv
- Book4.csv
- Book5.csv
- Book6.csv
- EPL-Score-Predictor [EPL-Score-Predictor master]
- FirstTestNGProject [FirstTestNGProject11]
- MachineLearning [MachineLearning master]
- NeuralNetwork
- newproject
- SoccerLeaguePredictor [SoccerLeaguePredictor master]
- WebdriverTest

Outline Task List Find All Activate...

```

577     if(matches[n].homeTeamNo == trainteams[n].homeTeamNo) {
578         trainteams[n].homeTeamNo.points += 3;
579     }
580     else if(matches[n].win < 0) {
581         trainteams[n].homeTeamNo.rank -= 2;
582         trainteams[n].homeTeamNo.points -= 2;
583         trainteams[n].homeTeamNo.points += 1;
584         trainteams[n].awayTeamNo.rank += 2;
585         trainteams[n].awayTeamNo.rank -= 4;
586         trainteams[n].awayTeamNo.rank += 4;
587         trainteams[n].awayTeamNo.rank += 7;
588     }
589     else {
590         trainteams[n].homeTeamNo.rank += 2;
591         trainteams[n].awayTeamNo.rank += 4;
592     }
593     trainteams[n].homeTeamNo.rank += 3;
594     trainteams[n].homeTeamNo.rank += trainteams[n].awayTeamNo.points;
595     trainteams[n].homeTeamNo.rank += 1;
596     trainteams[n].awayTeamNo.rank += 1;
597     else if(trainteams[n].homeTeamNo.points < trainteams[n].awayTeamNo.points) {
598         trainteams[n].homeTeamNo.rank += 4;
599         trainteams[n].awayTeamNo.rank += 3;
600         trainteams[n].awayTeamNo.rank += 0;
601     }
602     else {
603         trainteams[n].homeTeamNo.rank += 2;
604         trainteams[n].awayTeamNo.rank += 2;
605     }
606     trainteams[n].homeTeamNo.rank += 1;
607     trainteams[n].awayTeamNo.rank += 1;
608 }
609 public static void rate() {
610     if(matches[348].win == 0) {
611         success += 1;
612     }
613     if(matches[341].win == 0) {
614         success += 1;
615     }

```

Tasks Console

```

terminated: Predictor [Java Application] C:\Program Files\Java\jre1.8.0_241\bin\javaw.exe (2020年4月11日 下午4:19:54 - 下午4:20:02)
Wolverhampton wanderers Fulham 1 3
Arsenal Brighton 0 1
Chelsea Manchester United 0 0
Huddersfield Manchester United 3 2
Manchester City Leicester City 1 2
Brighton & Hove Albion 1 1
Burnley Arsenal 3 2
Crystal Palace Bournemouth 2 3
Fulham Newcastle United 1 1
Leicester City Chelsea 1 1
Liverpool Wolverhampton Wanderers 3 3
Manchester United Cardiff City 1 1
Southampton Huddersfield 2 1
Tottenham Hotspur Everton 2 1
Watford West Ham United 3 0
1412

```

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer ERL

- ERL
 - BpDeep.java
 - matchjava
 - Predictor.java
 - Team.java
 - EPL/pom.xml
 - Predictor.class
 - BasicNetwork.class
- Maven Dependencies
- bin
- target
- Book1.csv
- Book2.csv
- BOOK3.csv
- Book4.csv
- Book5.csv
- Book6.csv
- EPL-Score-Predictor [EPL-Score-Predictor master]
- FirstTestNGProject [FirstTestNGProject11]
- MachineLearning [MachineLearning master]
- NeuralNetwork
- newproject
- SoccerLeaguePredictor [SoccerLeaguePredictor master]
- WebdriverTest

Outline Task List Find All Activate...

```

217     //int value = Integer.parseInt(last);
218     //System.out.println(item[0]);
219     //System.out.println(item[1]);
220     matches[n] = new match();
221     matches[n].item = item[0];
222     matches[n].item1 = item[1];
223     matches[n].homeTeamNo = Integer.parseInt(item[0]);
224     matches[n].awayTeamNo = Integer.parseInt(item[1]);
225     String home = teams[matches[n].homeTeamNo].getTeamName();
226     String away = teams[matches[n].awayTeamNo].getTeamName();
227     int b = matches[n].awayTeamNo;
228
229     double output[1][new double [1000][2]];
230     for(int i=0;i<1000;++) {
231         output[1][0]=output[1][0]/6;
232         output[1][1]=output[1][1]/6;
233     }
234     //ArrayList<Integer> l = predict(a,b,input1,output3);
235     //matches[n].l.set(0);
236     //predict(matches[n].homeTeamNo,matches[n].awayTeamNo);
237     //team[matches[n].homeTeamNo].goalDifference.predict(a,b,input1,output3).get(1);
238     //team[matches[n].awayTeamNo].goalDifference.predict(a,b,input1,output3).get(2);
239     //matches[n].homegoal1.set(1);
240     //matches[n].awaygoal1.set(2);
241     //team[matches[n].homeTeamNo].input1(output3).get(0);
242     //System.out.println();
243     int c = predict(a,b);
244     System.out.println(home + " " + away+ " "+matches[n].homegoal+ " " +matches[n].awaygoal + " ");
245     renew(matches[n]);
246     n++;
247 }
248
249 
```

Tasks Console

```

terminated: Predictor [Java Application] C:\Program Files\Java\jre1.8.0_241\bin\javaw.exe (2020年4月11日 下午4:45:12 - 下午4:45:15)
Wolverhampton wanderers Fulham 0 0
Arsenal Brighton 0 0
Chelsea Manchester United 0 0
Huddersfield Manchester United 0 0
Manchester City Leicester City 0 0
Brighton & Hove Albion 0 0
Burnley Arsenal 0 0
Crystal Palace Bournemouth 0 0
Fulham Newcastle United 0 0
Leicester City Chelsea 0 0
Liverpool Wolverhampton Wanderers 0 0
Manchester United Cardiff City 0 0
Southampton Huddersfield 0 0
Tottenham Hotspur Everton 0 0
Watford West Ham United 0 0
1523

```

```

File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer [EPLDeep.java] Predictor.java Team.java EPL/pom.xml Predictor.class BasicNetwork.class
src
  EPLDeep.java
  Predictor.java
  Team.java
  EPL/pom.xml
  Predictor.class
  BasicNetwork.class
bin
target
Book1.csv
Book2.csv
BOOK3.csv
Book4.csv
Book5.csv
EPL-Score-Predictor [EPL-Score-Predictor master]
FirstTestNGProject [FirstTestNGProject11]
MachineLearning [MachineLearning master]
NeuralNetwork
newproject
SoccerLeaguePredictor [SoccerLeaguePredictor master]
WebdriverTest
src
  EPLDeep.java
  Predictor.java
  Team.java
  EPL/pom.xml
  Predictor.class
  BasicNetwork.class
Find All Activate...
Outline Task List Find All Activate...
Tasks Console
EPLDeep.java Predictor.java Team.java EPL/pom.xml Predictor.class BasicNetwork.class
int n = 370;
while((line=reader.readLine())!=null){
    String[] item = line.split(",");
    int n = 370;
    while((line=reader.readLine())!=null){
        String[] item = line.split(",");
        int n = 370;
        while((line=reader.readLine())!=null){
            String[] item = line.split(",");
            int n = 370;
            while((line=reader.readLine())!=null){
                String[] item = line.split(",");
                int n = 370;
                while((line=reader.readLine())!=null){
                    String[] item = line.split(",");
                    int n = 370;
                    while((line=reader.readLine())!=null){
                        String[] item = line.split(",");
                        int n = 370;
                        while((line=reader.readLine())!=null){
                            String[] item = line.split(",");
                            int n = 370;
                            while((line=reader.readLine())!=null){
                                String[] item = line.split(",");
                                int n = 370;
                                while((line=reader.readLine())!=null){
                                    String[] item = line.split(",");
                                    int n = 370;
                                    while((line=reader.readLine())!=null){
                                        String[] item = line.split(",");
                                        int n = 370;
                                        while((line=reader.readLine())!=null){
                                            String[] item = line.split(",");
                                            int n = 370;
                                            while((line=reader.readLine())!=null){
                                                String[] item = line.split(",");
                                                int n = 370;
                                                while((line=reader.readLine())!=null){
                                                    String[] item = line.split(",");
                                                    int n = 370;
                                                    while((line=reader.readLine())!=null){
                                                        String[] item = line.split(",");
                                                        int n = 370;
                                                        while((line=reader.readLine())!=null){
                                                            String[] item = line.split(",");
                                                            int n = 370;
                                                            while((line=reader.readLine())!=null){
                                                                String[] item = line.split(",");
                                                                int n = 370;
                                                                while((line=reader.readLine())!=null){
                                                                    String[] item = line.split(",");
                                                                    int n = 370;
                                                                    while((line=reader.readLine())!=null){
                                                                        String[] item = line.split(",");
                                                                        int n = 370;
                                                                        while((line=reader.readLine())!=null){
                                                                            String[] item = line.split(",");
                                                                            int n = 370;
                                                                            while((line=reader.readLine())!=null){
                                                                                String[] item = line.split(",");
                                                                                int n = 370;
                                                                                while((line=reader.readLine())!=null){
                                                                                    String[] item = line.split(",");
                                                                                    int n = 370;
                                                                                    while((line=reader.readLine())!=null){
                                                                                        String[] item = line.split(",");
                                                                                        int n = 370;
                                                                                        while((line=reader.readLine())!=null){
                                                                                            String[] item = line.split(",");
                                                                                            int n = 370;
                                                                                            while((line=reader.readLine())!=null){
                                                                                                String[] item = line.split(",");
                                                                                                int n = 370;
                                                                                                while((line=reader.readLine())!=null){
                                                                                                    String[] item = line.split(",");
                                                                                                    int n = 370;
                                                                                                    while((line=reader.readLine())!=null){
                                                                                                        String[] item = line.split(",");
................................................................

```

Tasks Console

EPLDeep.java Predictor.java Team.java EPL/pom.xml Predictor.class BasicNetwork.class

```

<terminated> Predictor [Java Application] C:\Program Files\Java\jre1.8.0_241\bin>javaw.exe (2020年4月11日 下午5:03:09 - 下午5:03:09)
Manchester United Cardiff 0 0
Southampton Huddersfield 0 0
Tottenham Hotspur 0 0
Watford West Ham United 0 0
Brighton Manchester City 0 0
Burnley Newcastle United 0 0
Crystal Palace Bournemouth 0 0
Fulham Newcastle United 0 0
Leicester City Chelsea 0 0
Liverpool Manchester United Warrington 0 0
Manchester United Cardiff 0 0
Southampton Huddersfield 0 0
Tottenham Hotspur Watford 0 0
Watford West Ham United 0 0
353

```

Conclusion:

- The larger the BP training sample size, the higher the prediction accuracy.**
- The more prediction fields, the higher the prediction accuracy. The final accuracy rate is around 40%. (It may be because the more games played, the lower the chance of upset.)**
- The prediction of Premier League 2019-2020 is as the picture below, Liverpool will be the champion, Man City will be the second, and Liverpool, Leicester City, Chelsea and Man United will attend the UEFA Champions League.**

Bournemouth, Aston Villa and Norwich will downgrade

Test result :

The screenshot shows the Eclipse IDE interface with the following details:

- Top Bar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.
- Project Explorer:** Shows files like RandomWalk.java, Employee.java, RandomWalk.java, NewtonApprox..., Team.java, New.java, test.java, Testclass1.java, Testclass2.java, Testclass3.java.
- Run View:** Displays the test results:
 - Finished after 3.966 seconds
 - Runs: 3/3 Errors: 0 Failures: 0
 - test [Runner: JUnit 5] (3.785 s)
 - test0 (1.194 s)
 - test0 (0.773 s)
 - test0 (0.818 s)
- Console View:** Shows the output of the test run:

```
<terminated> test(!) [JUnit] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Apr 12, 2020, 3:08:37 PM – 3:08:42 PM)
Cardiff :liverpool
3 6
3 6
Chelsea :Burnley
3 0
3 0
success: 1481
```
- Bottom Bar:** Taskbar with icons for File, Home, e-mail, Shopping, etc., and system status (3:09 PM, 4/12/2020).

9.Results

Method1:

```

Aston Villa :Sheffield United
3 0
Man City :Arsenal
1 3
Bournemouth :Crystal Palace
2 2
Aston Villa :Chelsea
0 4
Brighton :Arsenal
2 1
Man City :Burnley
1 1
Newcastle :Sheffield United
0 1
Norwich :Southampton
1 1
Watford :Leicester
2 1
Tottenham :Man United
1 2
West Ham :Wolves
0 1
Everton :Liverpool
1 3
Tottenham :West Ham
2 0
Burnley :Watford
1 2
Chelsea :Man City
1 1
Liverpool :Crystal Palace
1 3
Newcastle :Aston Villa
1 1
Norwich :Everton
1 1

```

Liverpool	98
Man City	78
Leicester	66
Chelsea	65
Man United	63
Arsenal	57
Tottenham	56
Wolves	55
Sheffield United	53
Burnley	50
Crystal Palace	49
Everton	49
Southampton	46
Newcastle	45
Brighton	38
West Ham	37
Watford	36
Aston Villa	34
Bournemouth	34
Norwich	27

eclipse-workspace - FinalFor20YEARS/src/EFL1/Method1.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Pack... Type... JUnit RandomWalk.java Employee.java RandomWalk.java NewtonApprox... QuickFindJava... QuickUnion... Method1.java Method2.java Teams2.java FinalFor20...

```

211 // 
212 // 
213 // 
214 // for(Integer in: total.keySet()) {
215 //     System.out.println(teams[in].teamName + "+"+total.get(in)/100);
216 
217 Arrays.sort(teams, new Comparator<Teams1>() {
218 @Override
219 public int compare(Teams1 arg0, Teams1 arg1) {
220     // TODO Auto-generated method stub
221     int i = arg1.getPoints()-arg0.getPoints();
222     if(i == 0) {
223         i = arg1.getGoalDifference()-arg0.getGoalDifference();
224     }
225     if(i == 0) {
226         i = arg1.getTotalgoal()-arg0.getTotalgoal();
227     }
228 }
229 }
230 }
```

Console

```

<terminated> Method1 [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Apr 13, 2020, 12:04:11 AM - 12:04:13 AM)
Liverpool 98
Man City 78
Leicester 66
Chelsea 65
Man United 63
Arsenal 57
Tottenham 56
Wolves 55
Sheffield United 53
Burnley 50
Crystal Palace 49
Everton 49
Southampton 46
Newcastle 45
Brighton 38
West Ham 37
Watford 36
Aston Villa 34
Bournemouth 34
Norwich 27
Birmingham 0
Blackburn 0
```

Method2:

Liverpool	99
Manchester City	73
Leicester City	66
Chelsea	60
Wolverhampton Wanderers	59
Manchester United	58
Sheffield United	56
Arsenal	53
Tottenham Hotspur	53
Crystal Palace	49
Burnley	48
Everton	48
NewCastle United	46
Southampton	45
Brighton	41
West Ham United	38
Watford	37
Bournemouth	37
Aston Villa	36
Norwich City	30

File Edit Source Refactor Navigate Search Project Run Window Help

```

package EFL1;
import java.io.BufferedReader;
public class Method2 {
    public static final int numTeams = 40;
    public static Teams2[] teams = new Teams2[numTeams];
    public static Teams2[] trainteams = new Teams2[40];
    public static final int numMatches = 380;
    public static Matches2[] matches = new Matches2[numMatches];
    public static Matches2[] trainmatches = new Matches2[6748];
    public static double [][] input= new double[6748][6];
    public static double [][] output =new double[6748][2];
    public static double [][] input1= new double[6748][6];
    public static double [][] output1= new double[6748][2];
}

```

Console :

```

<terminated> Method2 [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Apr 12, 2020, 9:58:52 PM - 9:58:54 PM)
99
Liverpool 99
Manchester City 73
Leicester City 66
Chelsea 60
Wolverhampton Wanderers 59
Manchester United 58
Sheffield United 56
Arsenal 53
Tottenham Hotspur 53
Crystal Palace 49
Burnley 48
Everton 48
NewCastle United 46
Southampton 45
Brighton 41
West Ham United 38
Watford 37
Bournemouth 37
Aston Villa 36
Norwich City 30

```