INFO6205 Ranking System Project Report

Group: NotAtAll

Qiushi Zhang 001057253
Yu Chen 001385622

Introduction

- **Background**

Due to the spread of COVID19, many industries are impacted and suffering great losses, including football. England Premier League (EPL) has announced that all the matches will be suspended. Therefore, it will be interesting to forecast the final result of EPL in this season. Although there will be a little suspense of champion because Liverpool has been leading the scoreboard too much, the suspense of the Champions League place and relegation. Therefore, we will manage to forecast the rest EPL matches and find some relationship between two teams and their match result.

- **Ranking System**

There are many superior performance ranking systems which utilized in sports. For example, ELO ranking system is one of the most commonly used in sports field such as chess, FIFA national team points and celestial system of League of Legends (LOL). As a matter of fact, ELO Ranking System is our first idea to solve the problem. However, after doing research on ELO and EPL, we found that the existing ranking system of EPL (point +3 if won, +1 if draw and +0 if lost) does not fit the ELO ranking system, that means we have to build up another ranking system to calculate the points of each teams. Furthermore, as far as I am concerned, the essence of ELO ranking system is two points. Firstly, using logistic distribution to estimate the expectation win rate point. Secondly, using a mutable K value to make the point change more reasonable, for example, a better team will not win a lot from a very week team. It might promote us building up a new ranking system to connect with the existing scoreboard of EPL, which will spend too much extra time. Therefore, we will utilize the match data of EPL from Season 00-01 to Season 19-20 to record the relationship between two sides (Home team and Away Team) and the probability of every results (Home Win, Draw and Away win) to estimate the probability of a match result.

Project Summarize

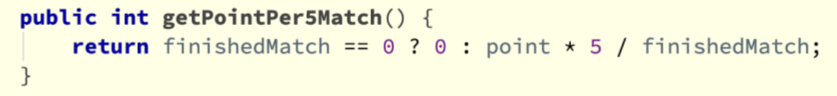
- **Target**

Our target can be summarized in such points. First of all, we will find out a probability of a result of an upcoming EPL match. For example, Everton will play with Liverpool at home in round 30, we will get the probability of the wining rate of home team (Everton), the draw rate and the wining rate of Away team (Liverpool). Furthermore, we will get the expectational point of both teams. We will 'Kick off' every match and get the expectational point of home and away teams, which will be added in their total points in scoreboard. Therefore, we will get a final expectational scoreboard. Finally, we will simulate every next match randomly, using the probability of each match calculated before. For example, if the match Everton vs Liverpool is calculated that the wining rate of Everton is 22%, draw rate is 16% and wining rate of Liverpool is 62%, we will use `java.util.Random` to simulate the match with such probability and get the result randomly.

- **Basic Frame**

The package League is used to simulate the EPL system, which Java classes Team, Match can create every team and match objects. Package Rank is used to build up some basic algorithms and package Main is used to simulate the season and calculate the final point expectation.

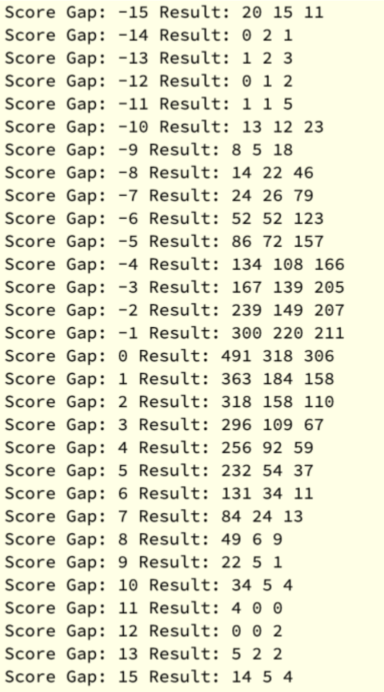
To calculate the ability gap between, we cannot just use the current point gap because it may occur that two teams have finished different count of rounds. For example, Team A has finished 10 matches and its point is 25. Team B has finished 15 matches while its point is also 25. We cannot say that team A and B are at the same level because they have different match count although they have the same point. Therefore, we have to calculate point per match to judge the ability gap between two teams. As a matter of fact, we will use floor of 5 * point / finished match to estimate the team point, because we want a beautiful probability distribution of match results of two teams.



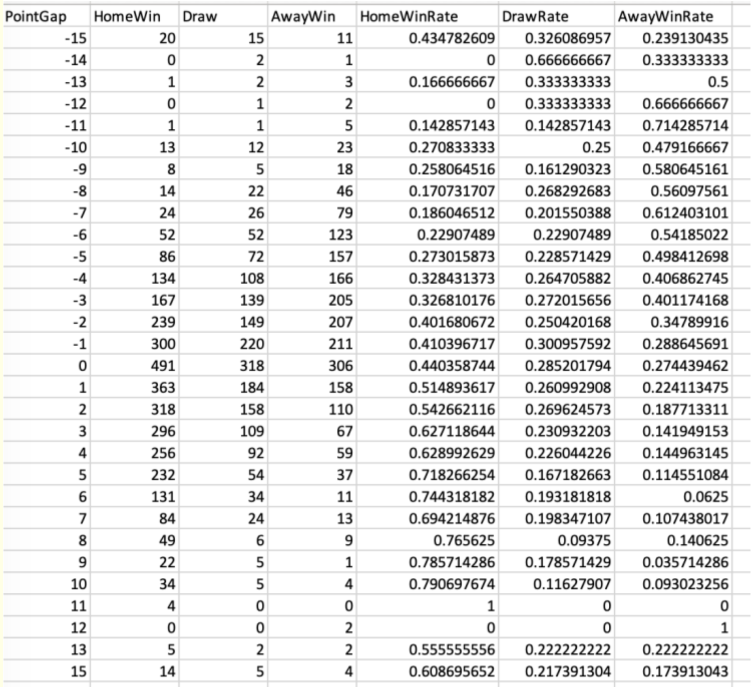
```java
public int getPointPer5Match() {
    return finishedMatch == 0 ? 0 : point * 5 / finishedMatch;
}
```

Actually, we can distribute the point gap between two against teams from -15 to 15 according to the equation $\left\lfloor \frac{5 * pointOfHome}{finishedMatchOfHome} - \frac{5 * pointOfAway}{finishedMatchOfAway} \right\rfloor$. To simplify, we will define $\left\lfloor \frac{5 * pointOfHome}{finishedMatchOfHome} - \frac{5 * pointOfAway}{finishedMatchOfAway} \right\rfloor$ as gp.

The next step of the system is read the data from past 19 seasons. We will ignore the irrelevant information such as index of the gambling and keep only home team, away team and the match result (H, D or A, which represent home win, draw and away win). Because of lack of scoreboard in this data, we have to calculate the point of each team before match began. Afterwards, we will put the point gaps between two against teams as the key and an integer array as the value in a HashMap. The array contains three numbers, which respectively correspond to home won counts, draw counts and away won counts when the point gaps between two against teams is the key integer. The result will be written in relationship.csv file, in which the corresponding Home Win / Draw / Away Win rate will be calculated.

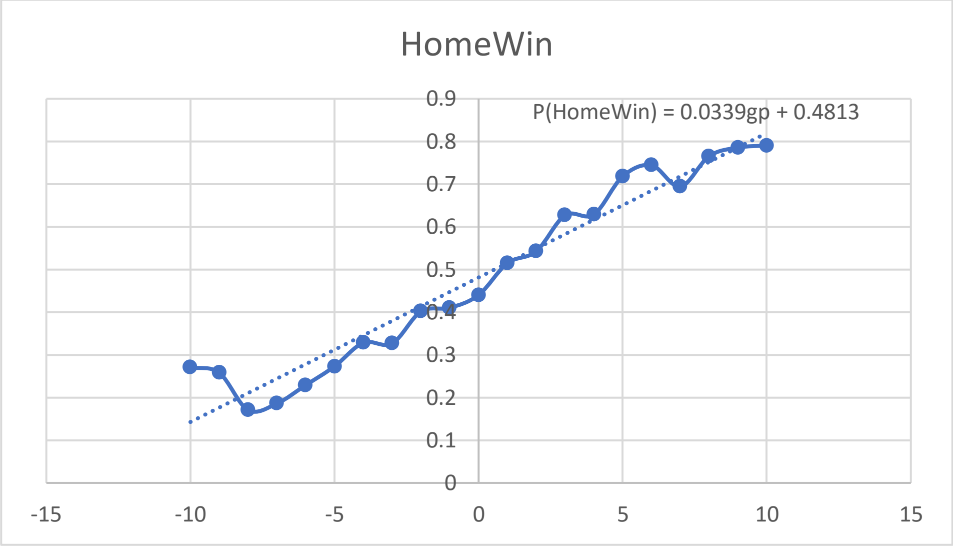


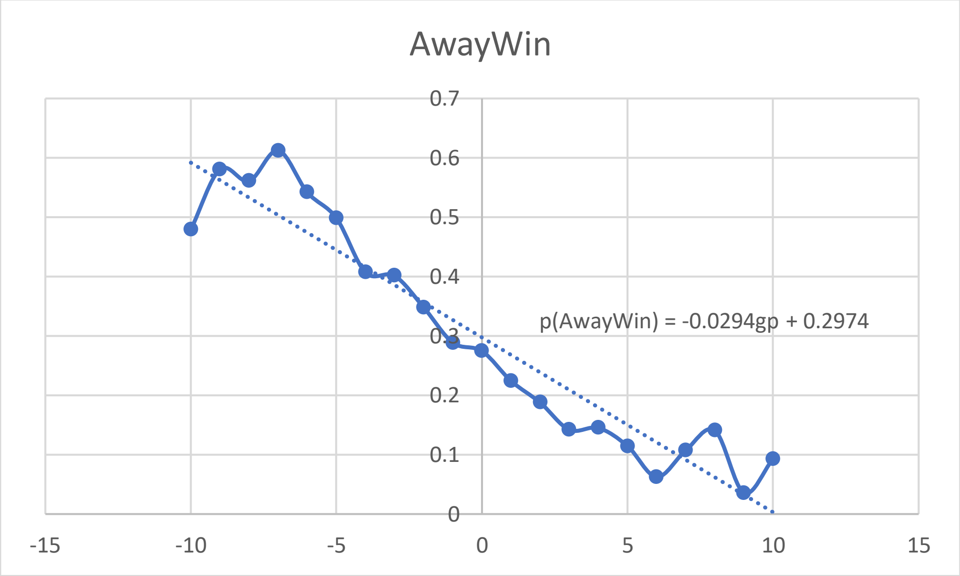
```
Score Gap: -15 Result: 20 15 11
Score Gap: -14 Result: 0 2 1
Score Gap: -13 Result: 1 2 3
Score Gap: -12 Result: 0 1 2
Score Gap: -11 Result: 1 1 5
Score Gap: -10 Result: 13 12 23
Score Gap: -9 Result: 8 5 18
Score Gap: -8 Result: 14 22 46
Score Gap: -7 Result: 24 26 79
Score Gap: -6 Result: 52 52 123
Score Gap: -5 Result: 86 72 157
Score Gap: -4 Result: 134 108 166
Score Gap: -3 Result: 167 139 205
Score Gap: -2 Result: 239 149 207
Score Gap: -1 Result: 300 220 211
Score Gap: 0 Result: 491 318 306
Score Gap: 1 Result: 363 184 158
Score Gap: 2 Result: 318 158 110
Score Gap: 3 Result: 296 109 67
Score Gap: 4 Result: 256 92 59
Score Gap: 5 Result: 232 54 37
Score Gap: 6 Result: 131 34 11
Score Gap: 7 Result: 84 24 13
Score Gap: 8 Result: 49 6 9
Score Gap: 9 Result: 22 5 1
Score Gap: 10 Result: 34 5 4
Score Gap: 11 Result: 4 0 0
Score Gap: 12 Result: 0 0 2
Score Gap: 13 Result: 5 2 2
Score Gap: 15 Result: 14 5 4
```

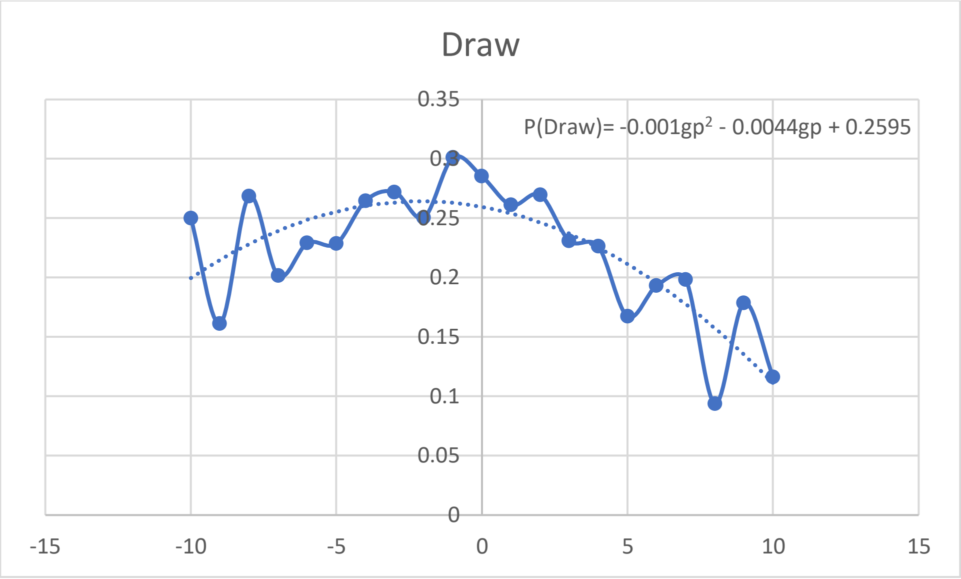


PointGap	HomeWin	Draw	AwayWin	HomeWinRate	DrawRate	AwayWinRate
-15	20	15	11	0.434782609	0.326086957	0.239130435
-14	0	2	1	0	0.666666667	0.333333333
-13	1	2	3	0.166666667	0.333333333	0.5
-12	0	1	2	0	0.333333333	0.666666667
-11	1	1	5	0.142857143	0.142857143	0.714285714
-10	13	12	23	0.270833333	0.25	0.479166667
-9	8	5	18	0.258064516	0.161290323	0.580645161
-8	14	22	46	0.170731707	0.268292683	0.56097561
-7	24	26	79	0.186046512	0.201550388	0.612403101
-6	52	52	123	0.22907489	0.22907489	0.54185022
-5	86	72	157	0.273015873	0.228571429	0.498412698
-4	134	108	166	0.328431373	0.264705882	0.406862745
-3	167	139	205	0.326810176	0.272015656	0.401174168
-2	239	149	207	0.401680672	0.250420168	0.34789916
-1	300	220	211	0.410396717	0.300957592	0.288645691
0	491	318	306	0.440358744	0.285201794	0.274439462
1	363	184	158	0.514893617	0.260992908	0.224113475
2	318	158	110	0.542662116	0.269624573	0.187713311
3	296	109	67	0.627118644	0.230932203	0.141949153
4	256	92	59	0.628992629	0.226044226	0.144963145
5	232	54	37	0.718266254	0.167182663	0.114551084
6	131	34	11	0.744318182	0.193181818	0.0625
7	84	24	13	0.694214876	0.198347107	0.107438017
8	49	6	9	0.765625	0.09375	0.140625
9	22	5	1	0.785714286	0.178571429	0.035714286
10	34	5	4	0.790697674	0.11627907	0.093023256
11	4	0	0	1	0	0
12	0	0	2	0	0	1
13	5	2	2	0.555555556	0.222222222	0.222222222
15	14	5	4	0.608695652	0.217391304	0.173913043

However, there is some “unreasonable” data. For example, we can notice that when *gp* is -15, which means that the away team is too much stronger than home team, there are more circumstances that home team defeated away team. However, it’s easy to understand. We can imagine that after Round 1 of a season, team A won the match in Round 1. Therefore, its score is 3, meanwhile, team B lost the match in Round 1 so its score is 0. In this case, if B would encounter A at home in Round 2, we can calculate the *gp* between B and A is -15. Therefore, when the absolute value of *gp* is big enough (in this case, larger than 10), it will stand a good chance that it happened in the first few rounds of the season, which will course extreme *gp*. Therefore, we will choose *gp* only from -10 to 10, which will be much easy to fit the data.







$$P(HomeWin) = 0.0339gp + 0.4813$$
$$P(AwayWin) = -0.0294gp + 0.2974$$
$$P(Draw) = -0.001gp^2 - 0.0044gp + 0.2595$$

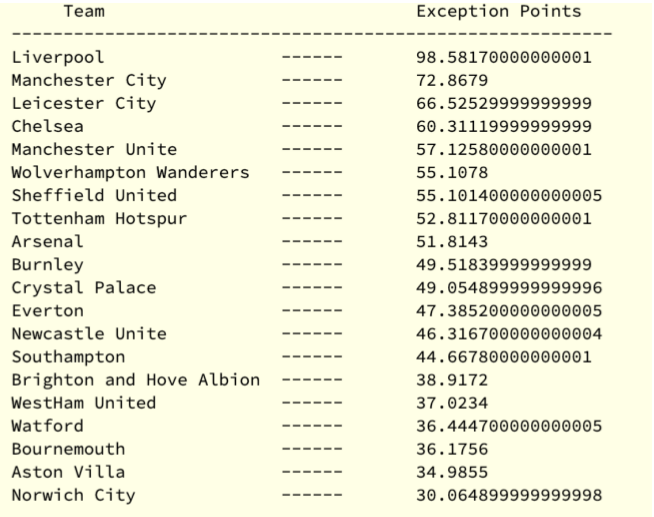
From the data and the corresponding fitting curve, we can find out that the change tendency of home win rate and away win rate are easy to understand. With the growth of *gp*, the ability gap between home and away teams will increase. Of course, the home win rate will increase, too. The same reason is suitable for the away win rate. However, although the fitting curve of draw rate is also easy to understand, which is when *gp* between home and away team tend to 0, the more similar their strength are, the more possible the game result would be draw. However, we can observe that the degree of fitting is not very high. The fitting curve can only represent the general trend of primitive curve. As a matter of fact, this is the problem of different style of different teams. For some teams such as Arsenal, the attack is weak, but the defense is very stable. The ability for Arsenal to draw the game is much higher than some other teams which is more likely to devote their energy to attack. Therefore, we can only get a tendency of the relationship between *gp* and draw rate. Because it will be much more complete to analyze different teams by their style. We can fix that in this project by calculate the draw rate of a match by the equation: $P(Draw) = 1 - P(HomeWin) - P(AwayWin)$. However, in such case, the draw rate will be a linage tendency, which cannot fit our fitting curve. In order to satisfy the basic equation: $P(HomeWin) + P(AwayWin) + P(Draw) = 1$, we decide to use equation: $P(Draw) = 1 - P(HomeWin) - P(AwayWin)$, because we find that the squared coefficient of the P(Draw) equation is only -0.001, which means the tendency curve approach to a linear line.

$$P(Draw) = -0.0045gp + 0.2213$$

For every match which has been scheduled but haven't kickoff, we can estimate the possibility of home team win, draw or away team win by utilizing the equation above. According to the possibility, we can calculate the expectational point of each team in each match. The home team's expectational point is $3 * P(HomeWin) + 1 * P(Draw)$, while the away team's expectational point is $3 * P(AwayWin) + 1 * P(Draw)$.

$$E(H) = 3 * P(HomeWin) + 1 * P(Draw)$$
$$E(A) = 3 * P(AwayWin) + 1 * P(Draw)$$

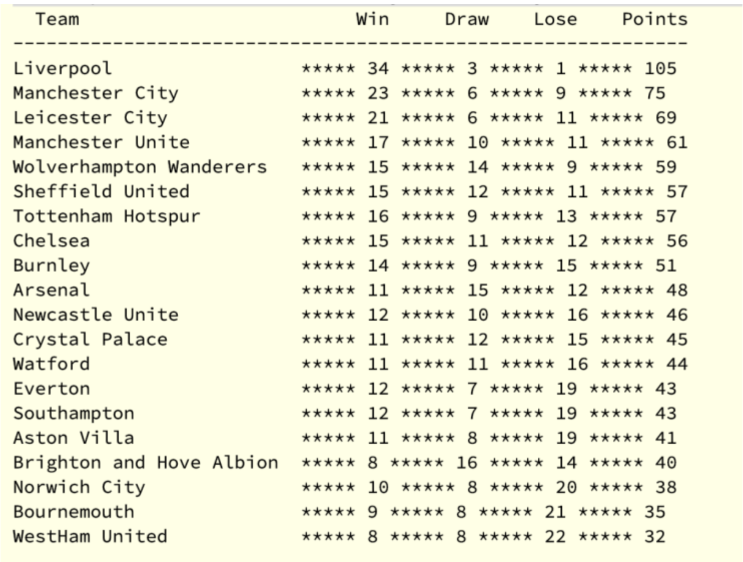
With the expectational point equation, we can calculate the expectational point in the end of the season by calculate every match's expectational point of both teams.



```
     Team                              Exception Points
----------------------------------------------------------
Liverpool                 ------       98.58170000000001
Manchester City           ------       72.8679
Leicester City            ------       66.52529999999999
Chelsea                   ------       60.31119999999999
Manchester Unite          ------       57.12580000000001
Wolverhampton Wanderers   ------       55.1078
Sheffield United          ------       55.101400000000005
Tottenham Hotspur         ------       52.81170000000001
Arsenal                   ------       51.8143
Burnley                   ------       49.51839999999999
Crystal Palace            ------       49.054899999999996
Everton                   ------       47.385200000000005
Newcastle Unite           ------       46.316700000000004
Southampton               ------       44.66780000000001
Brighton and Hove Albion  ------       38.9172
WestHam United            ------       37.0234
Watford                   ------       36.444700000000005
Bournemouth               ------       36.1756
Aston Villa               ------       34.9855
Norwich City              ------       30.064899999999998
```

Because the contingency will be excluded by calculating the expectational point, which means the score board will not looks more dispersedly. However, it's enough for us to estimate the general ranking in the end of the incomplete season. There is no doubt that Liverpool will win the champion, Norwich City and Aston Villa will fall on eval days, they cannot stay in EPL next season. Bournemouth is very likely to face promotion and relegation play-off.

Next step we will use the `java.util.Random` to simulate the matches left. Because of uncertainty is getting involved, the scoreboard will look more differently:

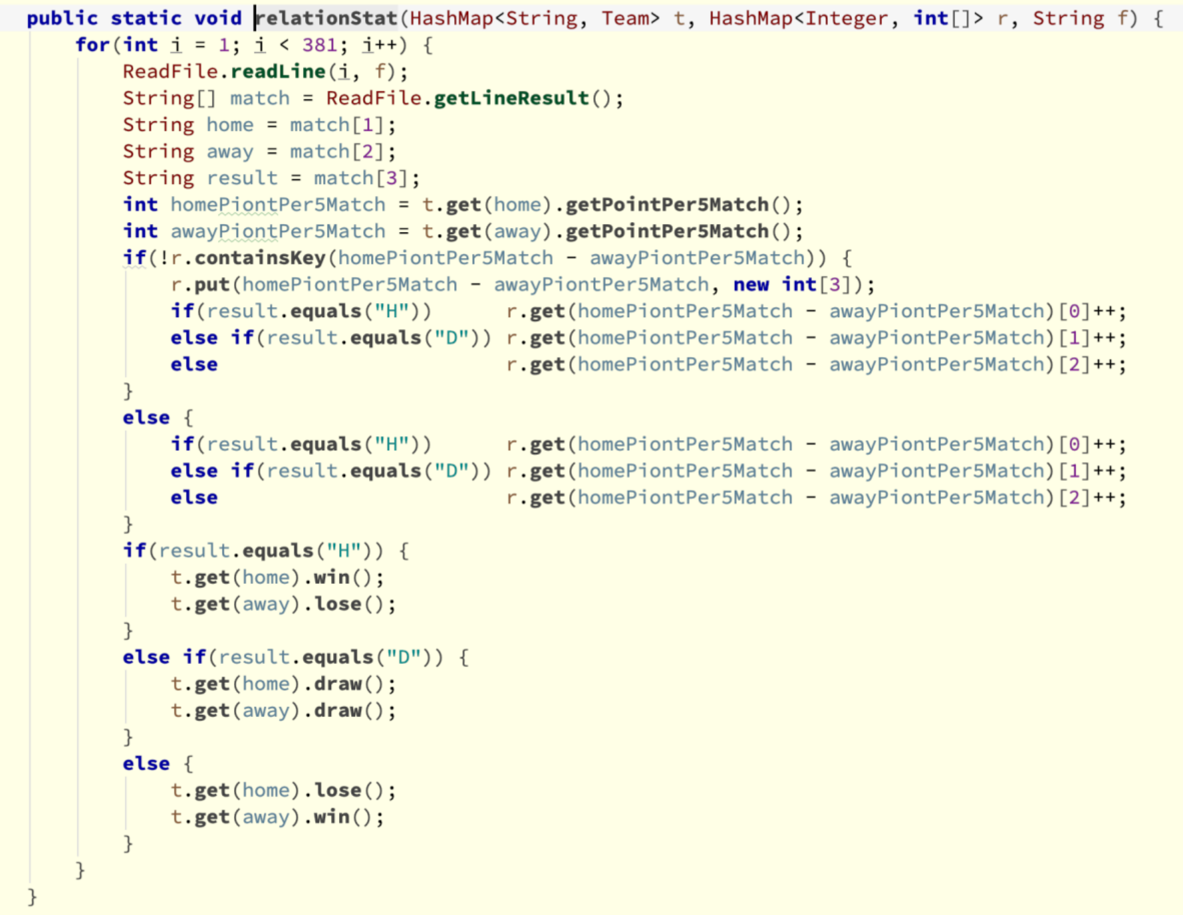


```
  Team                         Win     Draw     Lose     Points
-------------------------------------------------------------
Liverpool                 ***** 34 ***** 3 ***** 1 ***** 105
Manchester City           ***** 23 ***** 6 ***** 9 ***** 75
Leicester City            ***** 21 ***** 6 ***** 11 ***** 69
Manchester Unite          ***** 17 ***** 10 ***** 11 ***** 61
Wolverhampton Wanderers   ***** 15 ***** 14 ***** 9 ***** 59
Sheffield United          ***** 15 ***** 12 ***** 11 ***** 57
Tottenham Hotspur         ***** 16 ***** 9 ***** 13 ***** 57
Chelsea                   ***** 15 ***** 11 ***** 12 ***** 56
Burnley                   ***** 14 ***** 9 ***** 15 ***** 51
Arsenal                   ***** 11 ***** 15 ***** 12 ***** 48
Newcastle Unite           ***** 12 ***** 10 ***** 16 ***** 46
Crystal Palace            ***** 11 ***** 12 ***** 15 ***** 45
Watford                   ***** 11 ***** 11 ***** 16 ***** 44
Everton                   ***** 12 ***** 7 ***** 19 ***** 43
Southampton               ***** 12 ***** 7 ***** 19 ***** 43
Aston Villa               ***** 11 ***** 8 ***** 19 ***** 41
Brighton and Hove Albion  ***** 8 ***** 16 ***** 14 ***** 40
Norwich City              ***** 10 ***** 8 ***** 20 ***** 38
Bournemouth               ***** 9 ***** 8 ***** 21 ***** 35
WestHam United            ***** 8 ***** 8 ***** 22 ***** 32
```

We can see that Liverpool can almost win the champion in every simulation, because its advantage accumulated is very huge. However, when we focus on the relegation zone, which is the last three teams, we can find that it may change because of the uncertainty. Even from the expectational point table, the gap of the expectational points of the relegation zone is only several points, which means that the situation may change a lot if one of the teams won one more match. As the prober says: The football is round, everything could happen. Therefore, the expectational scoreboard can just provide us a possibility of the last result. We cannot deny that the result must be filled with complexity.

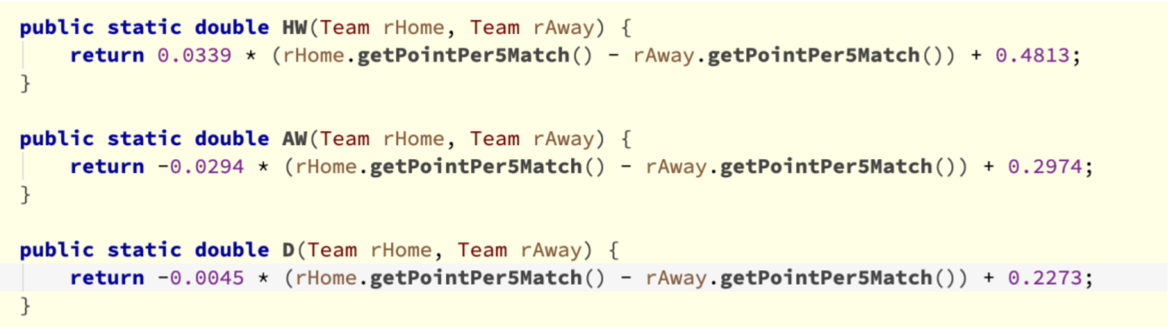
Java Implementation

In order to statistic the relationship between *gp* and the match result, we will put every relationship in a HashMap whose key is *gp* and value is an array with counts of three different results:



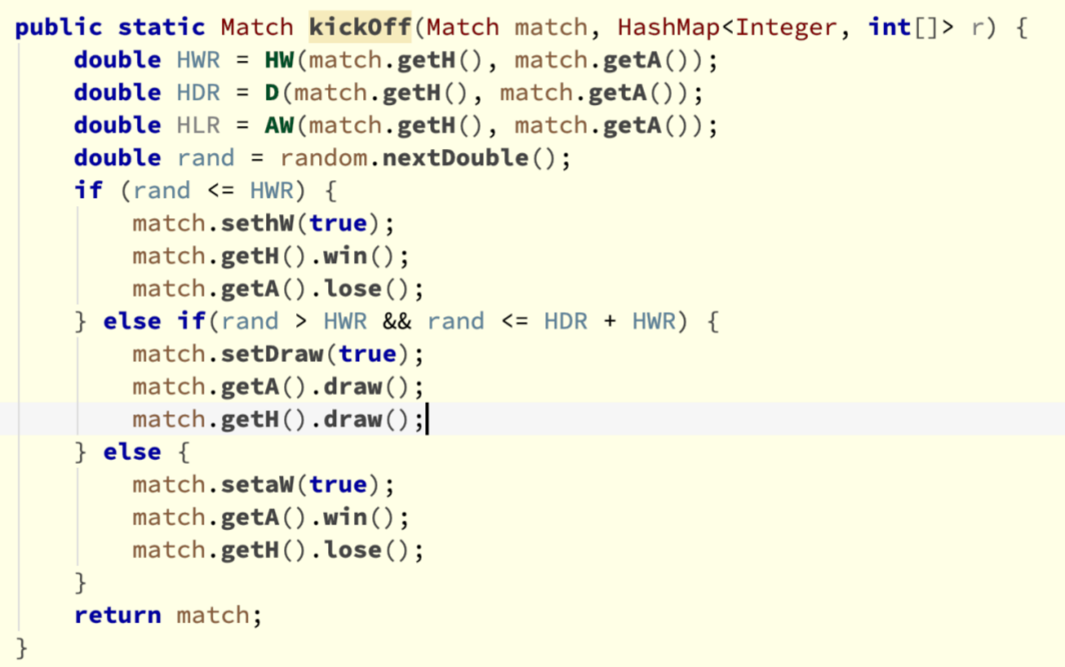
```java
public static void relationStat(HashMap<String, Team> t, HashMap<Integer, int[]> r, String f) {
    for(int i = 1; i < 381; i++) {
        ReadFile.readLine(i, f);
        String[] match = ReadFile.getLineResult();
        String home = match[1];
        String away = match[2];
        String result = match[3];
        int homePiontPer5Match = t.get(home).getPointPer5Match();
        int awayPiontPer5Match = t.get(away).getPointPer5Match();
        if(!r.containsKey(homePiontPer5Match - awayPiontPer5Match)) {
            r.put(homePiontPer5Match - awayPiontPer5Match, new int[3]);
            if(result.equals("H"))      r.get(homePiontPer5Match - awayPiontPer5Match)[0]++;
            else if(result.equals("D")) r.get(homePiontPer5Match - awayPiontPer5Match)[1]++;
            else                        r.get(homePiontPer5Match - awayPiontPer5Match)[2]++;
        }
        else {
            if(result.equals("H"))      r.get(homePiontPer5Match - awayPiontPer5Match)[0]++;
            else if(result.equals("D")) r.get(homePiontPer5Match - awayPiontPer5Match)[1]++;
            else                        r.get(homePiontPer5Match - awayPiontPer5Match)[2]++;
        }
        if(result.equals("H")) {
            t.get(home).win();
            t.get(away).lose();
        }
        else if(result.equals("D")) {
            t.get(home).draw();
            t.get(away).draw();
        }
        else {
            t.get(home).lose();
            t.get(away).win();
        }
    }
}
```

To calculate the probability of three results:



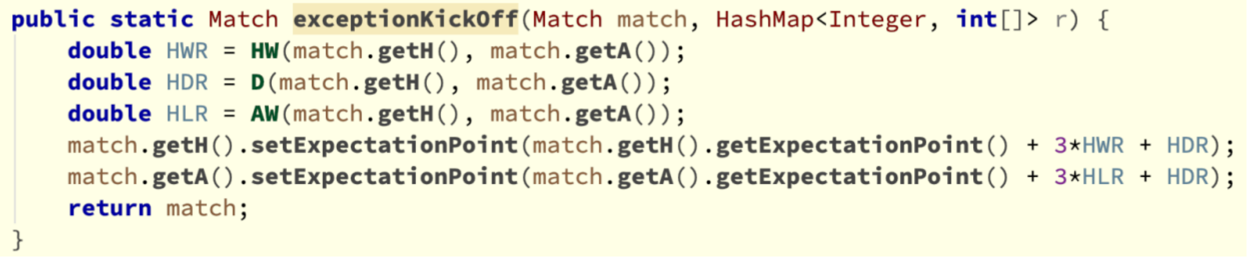
```java
public static double HW(Team rHome, Team rAway) {
    return 0.0339 * (rHome.getPointPer5Match() - rAway.getPointPer5Match()) + 0.4813;
}

public static double AW(Team rHome, Team rAway) {
    return -0.0294 * (rHome.getPointPer5Match() - rAway.getPointPer5Match()) + 0.2974;
}

public static double D(Team rHome, Team rAway) {
    return -0.0045 * (rHome.getPointPer5Match() - rAway.getPointPer5Match()) + 0.2273;
}
```

To simulate one match with calculated probability of three results:



```java
public static Match kickOff(Match match, HashMap<Integer, int[]> r) {
    double HWR = HW(match.getH(), match.getA());
    double HDR = D(match.getH(), match.getA());
    double HLR = AW(match.getH(), match.getA());
    double rand = random.nextDouble();
    if (rand <= HWR) {
        match.sethW(true);
        match.getH().win();
        match.getA().lose();
    } else if(rand > HWR && rand <= HDR + HWR) {
        match.setDraw(true);
        match.getA().draw();
        match.getH().draw();
    } else {
        match.setaW(true);
        match.getA().win();
        match.getH().lose();
    }
    return match;
}
```

To calculate the expectational scoreboard:

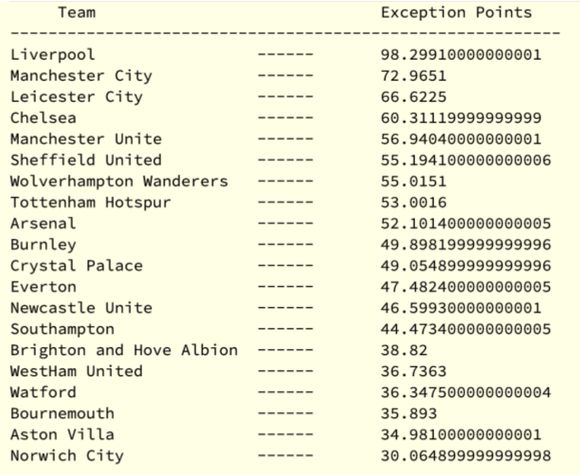


```java
public static Match exceptionKickOff(Match match, HashMap<Integer, int[]> r) {
    double HWR = HW(match.getH(), match.getA());
    double HDR = D(match.getH(), match.getA());
    double HLR = AW(match.getH(), match.getA());
    match.getH().setExpectationPoint(match.getH().getExpectationPoint() + 3*HWR + HDR);
    match.getA().setExpectationPoint(match.getA().getExpectationPoint() + 3*HLR + HDR);
    return match;
}
```

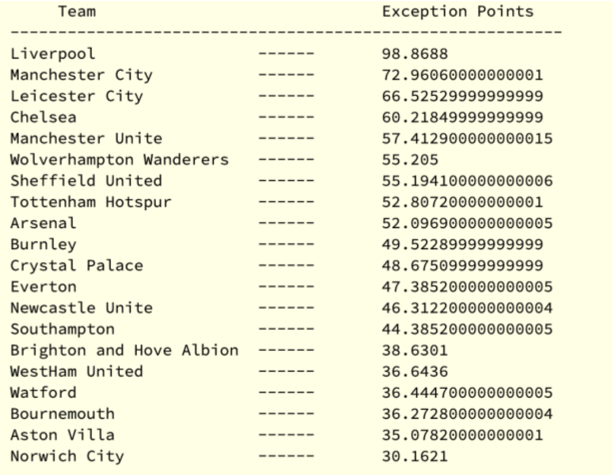
Limitation of the Algorithm

- **Computational accuracy**

When Java calculating float and double numbers, it will cause precision loss because its own malpractice. For example, when calculating the expectational scoreboard, we can find that several teams who has very similar expectational points will get different results, like these:



```
     Team                                Exception Points
----------------------------------------------------------
Liverpool                 ------       98.29910000000001
Manchester City           ------       72.9651
Leicester City            ------       66.6225
Chelsea                   ------       60.31119999999999
Manchester Unite          ------       56.94040000000001
Sheffield United          ------       55.194100000000006
Wolverhampton Wanderers   ------       55.0151
Tottenham Hotspur         ------       53.0016
Arsenal                   ------       52.101400000000005
Burnley                   ------       49.898199999999996
Crystal Palace            ------       49.054899999999996
Everton                   ------       47.482400000000005
Newcastle Unite           ------       46.59930000000001
Southampton               ------       44.473400000000005
Brighton and Hove Albion  ------       38.82
WestHam United            ------       36.7363
Watford                   ------       36.347500000000004
Bournemouth               ------       35.893
Aston Villa               ------       34.98100000000001
Norwich City              ------       30.064899999999998
```



```
     Team                                Exception Points
----------------------------------------------------------
Liverpool                 ------       98.8688
Manchester City           ------       72.96060000000001
Leicester City            ------       66.52529999999999
Chelsea                   ------       60.21849999999999
Manchester Unite          ------       57.412900000000015
Wolverhampton Wanderers   ------       55.205
Sheffield United          ------       55.194100000000006
Tottenham Hotspur         ------       52.80720000000001
Arsenal                   ------       52.096900000000005
Burnley                   ------       49.52289999999999
Crystal Palace            ------       48.67509999999999
Everton                   ------       47.385200000000005
Newcastle Unite           ------       46.312200000000004
Southampton               ------       44.385200000000005
Brighton and Hove Albion  ------       38.6301
WestHam United            ------       36.6436
Watford                   ------       36.444700000000005
Bournemouth               ------       36.272800000000004
Aston Villa               ------       35.07820000000001
Norwich City              ------       30.1621
```

We can find that the position of Sheffield United and Wolverhampton Wanderers are different.

To fix this problem, we can use `java.math.BigDecimal` to get more accurate calculation. However, it is proved that precision loss will still happen when we handle double value with `java.math.BigDecimal`. Therefore, we have to accept the error.

- **Not Suitable for Initial Phase**

Our estimation is based on current scoreboard after round 29, which means we can get a general level of each team. That means, we can get a more reasonable *gp* because the current scoreboard and finished matches can embody the general level of each team. However, try to image that if we want to use this algorithm from the first several rounds of one season. We will find that it is hard for us to get a reasonable result, because there will be more contingency. For example, the defending champion Liverpool lost the round 1 unfortunately while Norwich City won the first match luckily. If these two teams met in round two, we will get an extreme *gp* which is 15, which means it’s very hard for Liverpool to win. However, the truth tells us that Liverpool would still have a very high probability to win because the ability gap. Therefore, we can conclude that the algorithm is suitable when there are enough training data which can embody the level of each teams.

Conclusion

When facing an upcoming EPL match, we can get such equations to estimate the probability of result:

$$P(HomeWin) = 0.0339\left\lfloor\frac{5\,*\,pointOfHome}{finishedMatchOfHome} - \frac{5\,*\,pointOfAway}{finishedMatchOfAway}\right\rfloor + 0.4813$$

$$P(AwayWin) = -0.0294\left\lfloor\frac{5\,*\,pointOfHome}{finishedMatchOfHome} - \frac{5\,*\,pointOfAway}{finishedMatchOfAway}\right\rfloor + 0.2974$$

$$\begin{aligned} P(Draw) = -0.001gp&\left\lfloor\frac{5\,*\,pointOfHome}{finishedMatchOfHome} - \frac{5\,*\,pointOfAway}{finishedMatchOfAway}\right\rfloor^2 \\ &- 0.0044\left\lfloor\frac{5\,*\,pointOfHome}{finishedMatchOfHome} - \frac{5\,*\,pointOfAway}{finishedMatchOfAway}\right\rfloor + 0.2595 \end{aligned}$$