**INFO 6205**

**Program Structures & Algorithms**

**Spring 2020**

**Ranking System**

**Haimin Zhang 001212733**

**Beiyi Sheng 001306096**

1. **Introduction**
   1. **Background of Elo Ranking System**

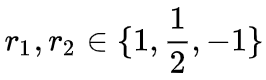
The Elo rating system is a method for calculating the relative skill levels of players in zero-sum games such as chess. It is named after its creator Arpad Elo, a Hungarian-American physics professor.

The difference in the ratings between two players serves as a predictor of the outcome of a match. Two players with equal ratings who play against each other are expected to score an equal number of wins. If one player has a higher rating than another, he is more likely to win the game.

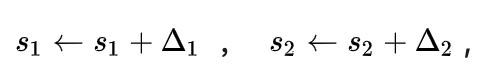
A player's Elo rating is represented by a number which may change depending on the outcome of rated games played. After every game, the winning player gets points while the losing player loses points. The points gained or lost are influenced by the difference between the elo ratings of the two players. If the high-rated player wins, then only a few rating points will be taken from the low-rated player. However, if the lower-rated player scores an upset win, many rating points will be transferred. This means that this rating system is self-correcting.

Elo's central assumption was that the chess performance of each player in each game is a normally distributed random variable. Although a player might perform significantly better or worse from one game to the next, Elo assumed that the mean value of the performances of any given player changes only slowly over time. Elo thought of a player's true skill as the mean of that player's performance random variable.

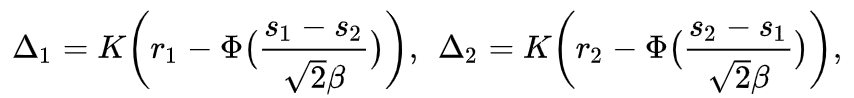
### **Mathematical Details**

Let  be the skill of player #1, and  for player #2. They all follow normal distribution. The result of past game is represented by  : { win, draw and lose}.

After each game, we update s1 and s2,

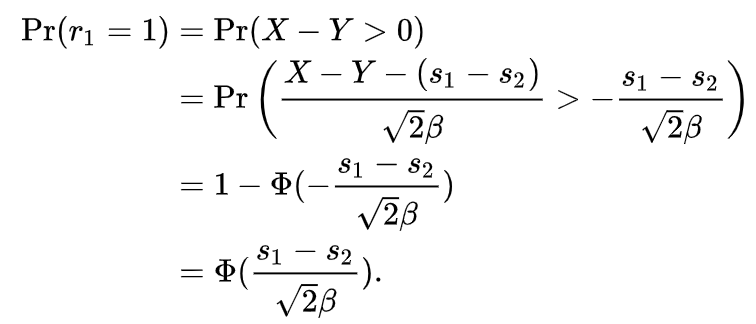


where

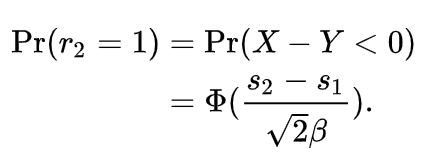


 is the cumulative distribution function for normal distribution. K is used to control the magnitude of parameter updates.

The probability for player #1 to win is

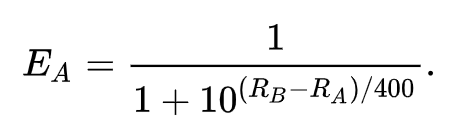


Similarly, the probability for player #2 to win is

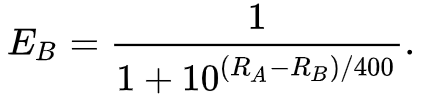


Some Elo ranking systems also use logistic distribution instead of the normal distribution to describe the ability of the evaluation object. The update formula is easier to derive while the principle remains unchanged.

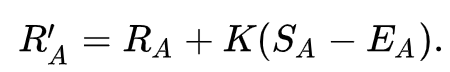
If Player A has a rating of  and Player B a rating of , the exact formula (using the logistic curve) for the expected score of Player A is



Similarly the expected score for Player B is



Supposing Player A was expected to score  points but actually scored  points. The formula for updating that player's rating is



Also, we can introduce G as the factor of goal difference to improve the elo system. The gain/lose of a game becomes



We make the following rules to calculate G.

* If the game is a draw or is won by 1 goal, G = 1.
* If the game is won by two goals, .
* If the game is won by three or more goals, , where N is the goal difference.
  1. **K-Factor**

As mentioned above, there’s a parameter K in the formula. K is called K-factor and controls how much ratings can change. When both players have the same K-factor, the sum of updated ratings is zero. This means that the rating points are conserved.

If the K-factor is set too large, there will be too much sensitivity, in terms of a large number of points exchanged in each game. But if the K-value is too low, the sensitivity will be minimal, and the system will not respond quickly enough to changes in a player's actual level of performance so that the rating stays inaccurate.

1. **Data Exploration**

Our data source link is [England Football Results Betting Odds | Premiership Results & Betting Odds](http://www.football-data.co.uk/englandm.php). We download 20 year data from 2000 to 2020. The data format is csv and the table below shows the meaning of columns that we used in our project.

|  |  |
| --- | --- |
| Abbreviation | Full name |
| Date | Match Date (dd/mm/yy) |
| HomeTeam | Home Team |
| AwayTeam | Away Team |
| FTHG | Full Time Home Team Goals |
| FTAG | Full Time Away Team Goals |

In our project, we used the past 19 years history data to train the model, and then make predictions on 2019-2020 season data. At the last, we rank teams according to their final elo value(rating value).

1. **Project Description**

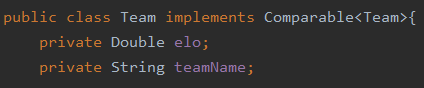
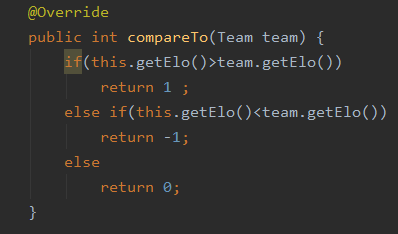
To develop a ranking system which is able to evaluate the following expression where  are elements from a set of competing elements **X**:

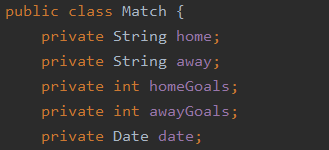


Where  is the probability that xi would beat xj if they met in a head to head matchup at neutral territory.

The input to the 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).

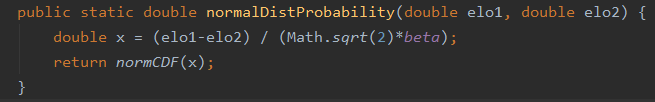
1. **Implementation**
   1. **Model**

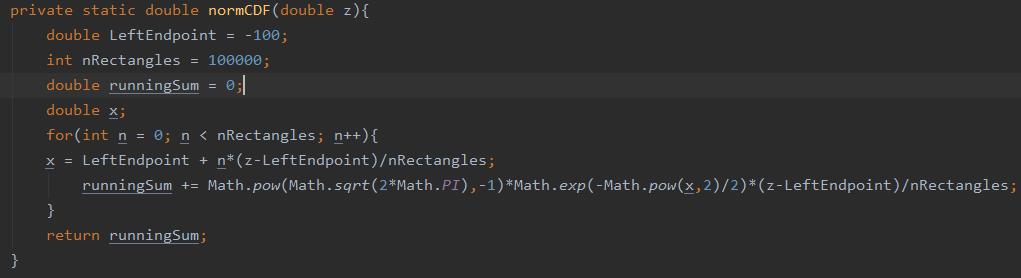
Our model is divided into two parts, Team and Match. In Class Team, there are two private data, elo and teamName and a method to compare with another team.

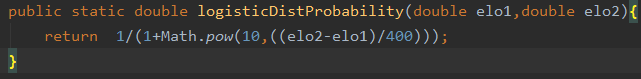
In Class Match, we store the two teams as home and away, and their goals.

* 1. **Implementation of Elo**

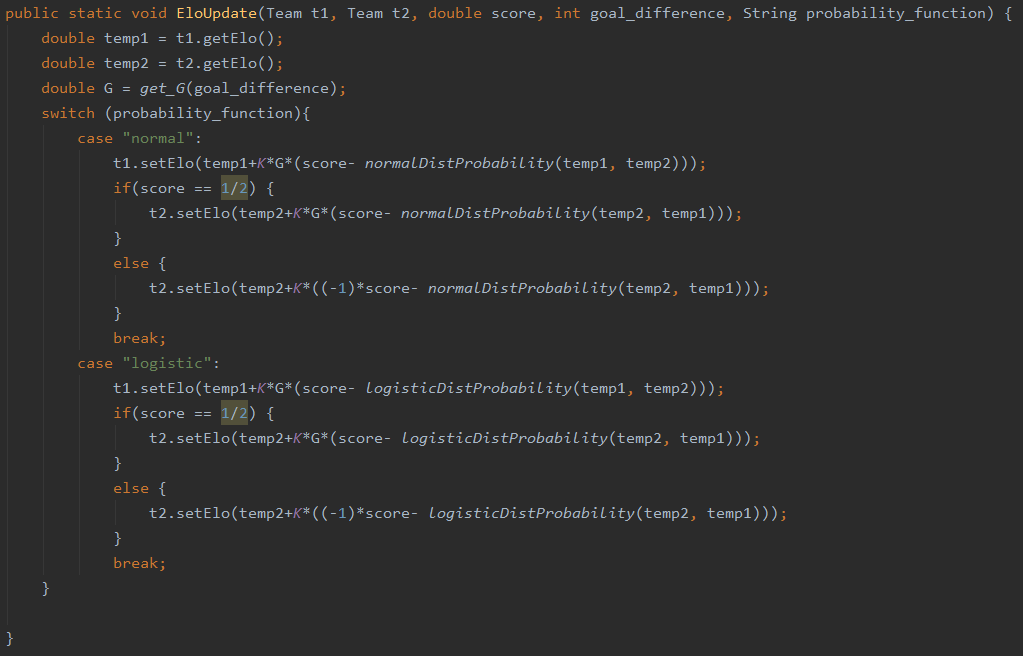
There are two parameters in our elo implementation, K and beta.

The method normalDistProbability is used to calculate the probability that the team with elo1 wins. The method normCDF will calculate the cumulative distribution function by simulating integral.

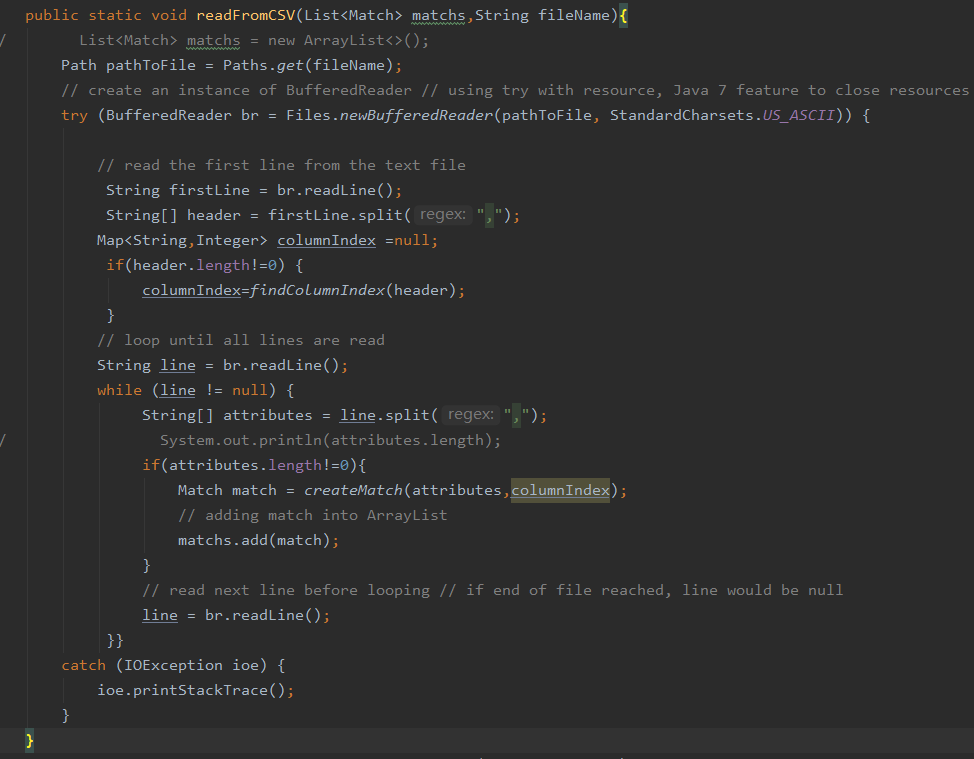


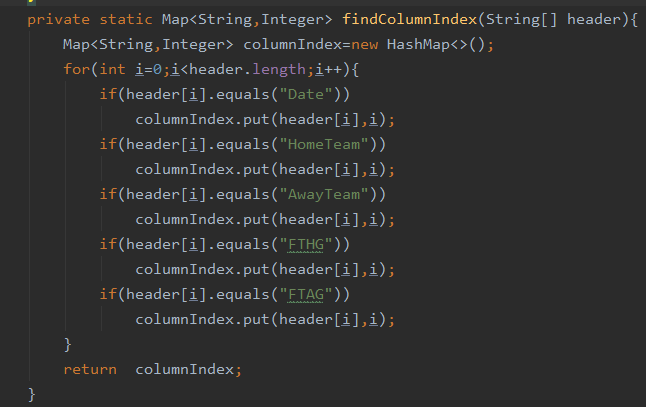
The method logisticDistProbability is based on the logistic curve.

The method EloUpdate is used to update the elo points of the two teams after a game. It takes two teams t1 and t2, the score of the game, the goal difference as input.



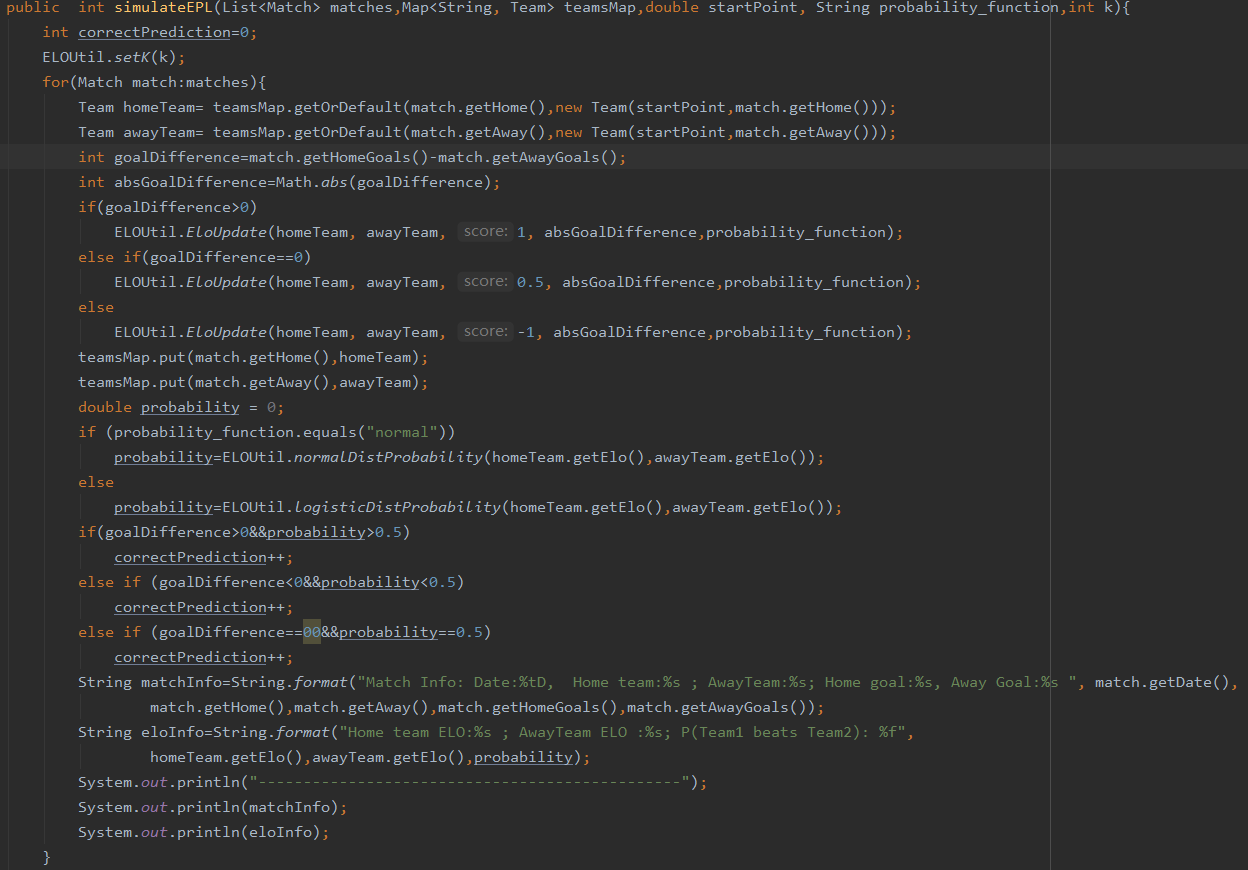
* 1. **File I/O**

We created ReadUtil.java to load and process our data. We loaded history data line by line and put these data into the Match model. There exists a meaningless empty line at the end of the file, we also cleaned up these empty lines. We used Standard US\_ASCII for the buffer reader at the beginning to avoid the crash.

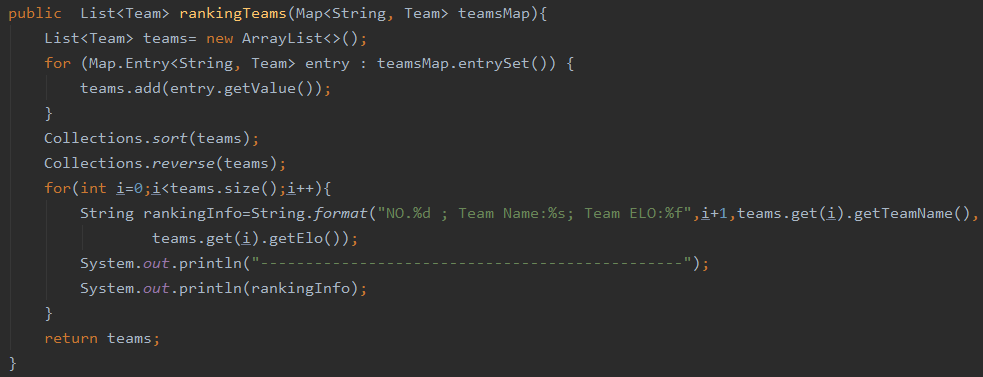
In order to find the correct index corresponding to the correct name, we created a method to keep track of the tab content, no matter how random the positions of columns are. 

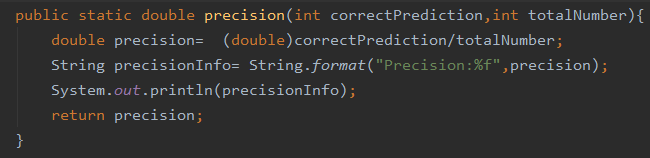
* 1. **Simulator**

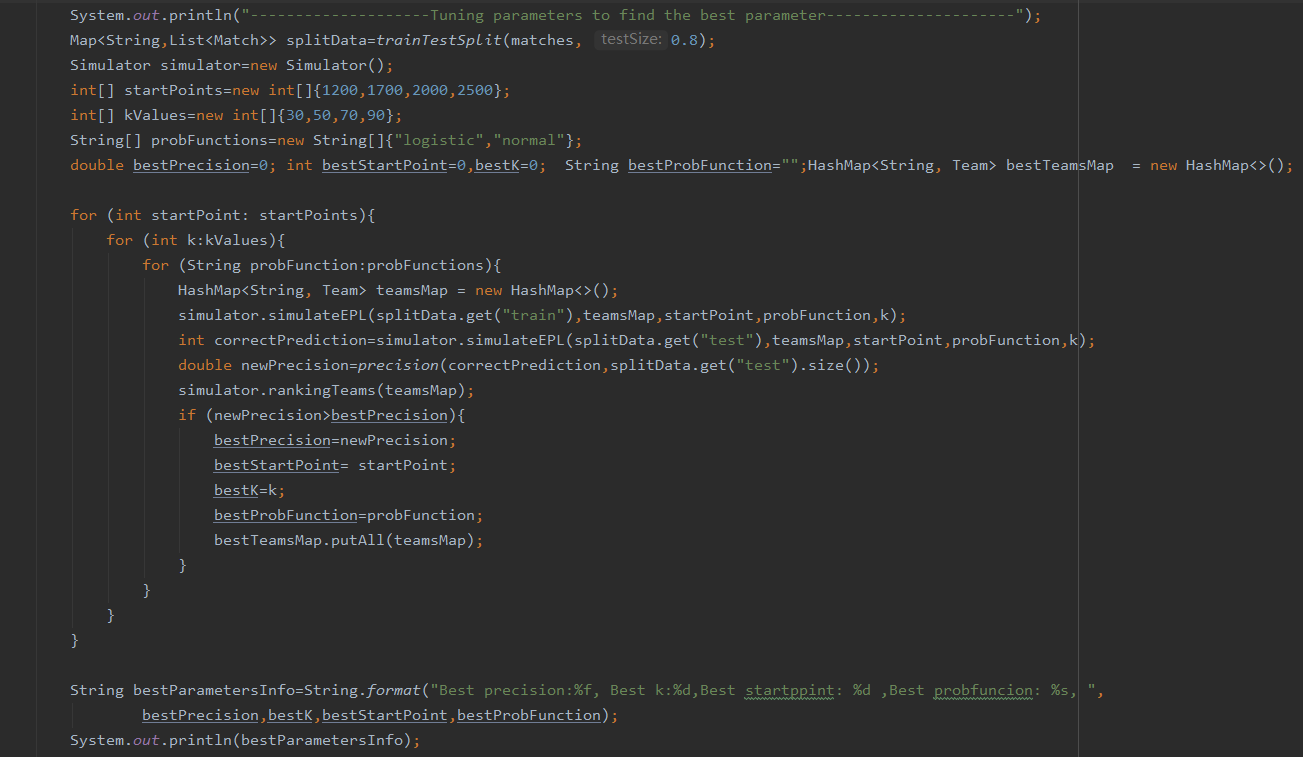
We used Map to store all the teams we needed. The method simulateEPL takes all the matches as input, and uses elo methods to update the elo points of each team.

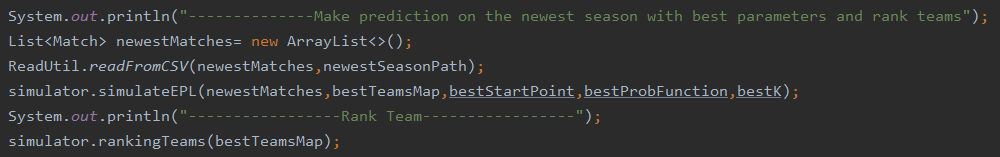
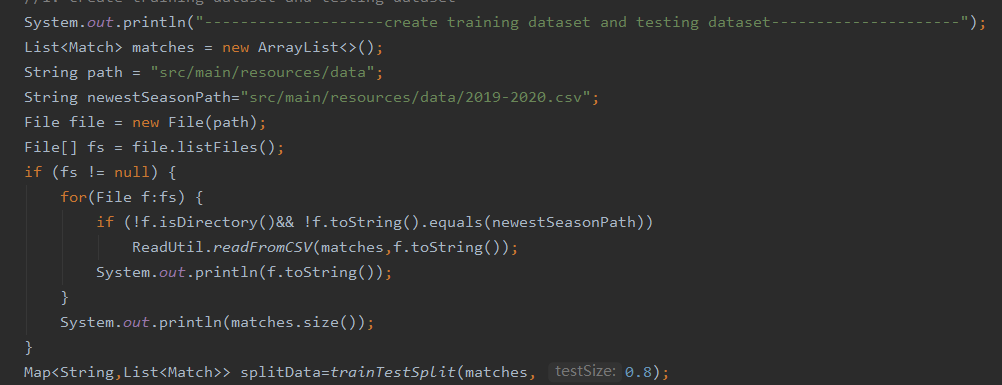


This method will sort the teams by their elo points and print the ranking of each team.

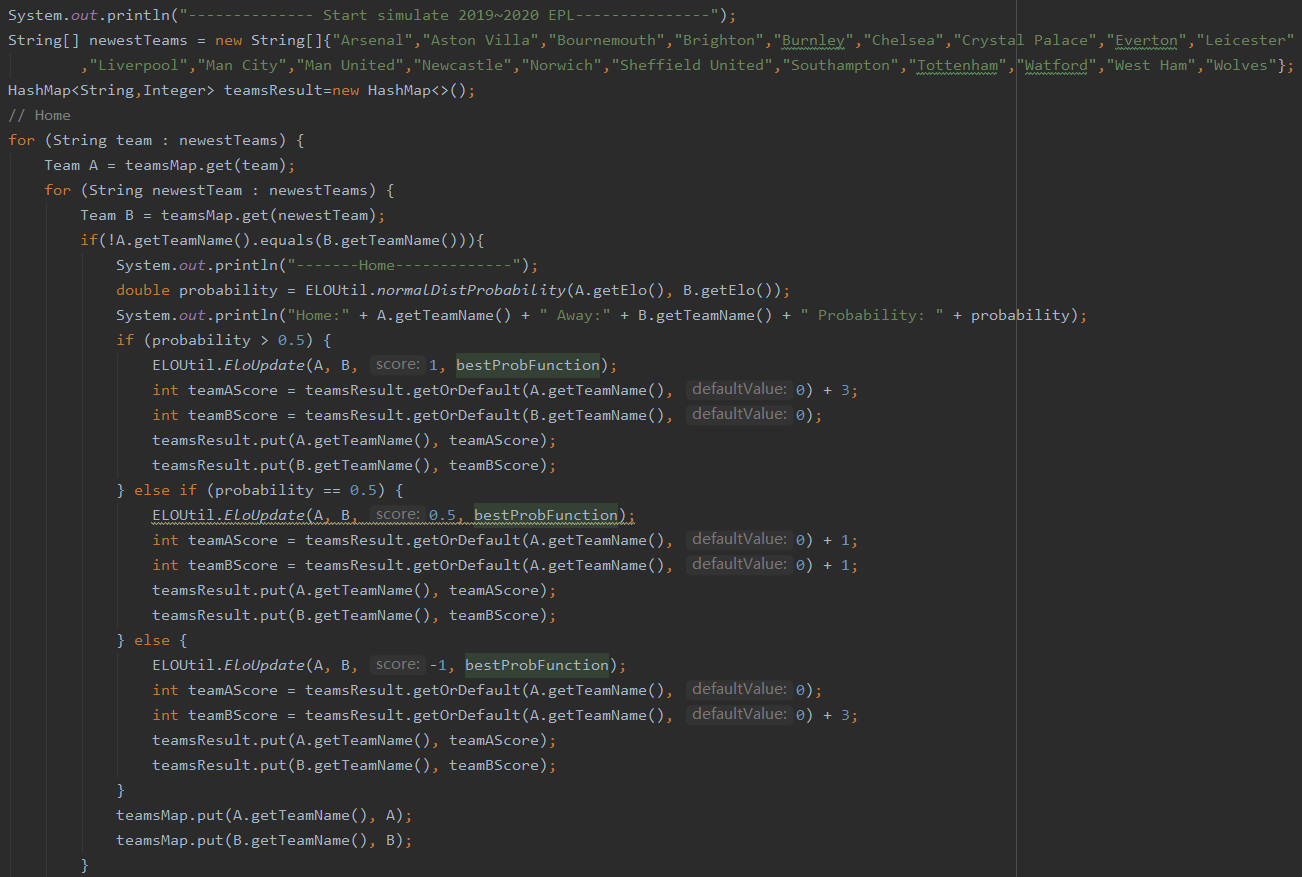


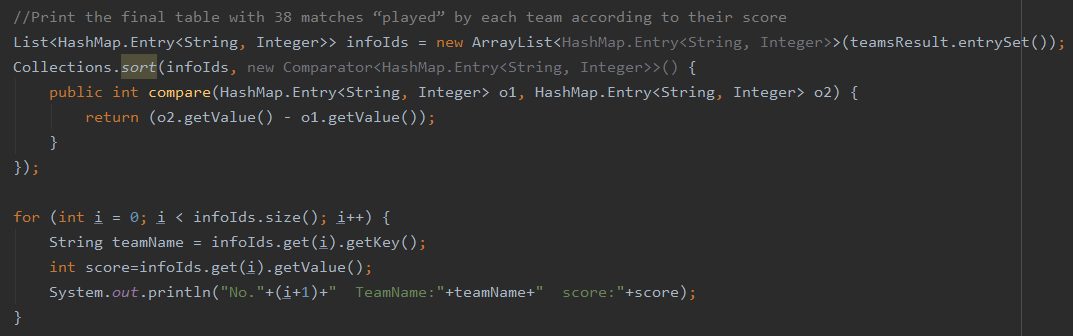
The precision method is to calculate the percent of correct prediction .

And then we started to simulate all the matches except 2019~2020 from history data. First, we splitted these data into training and testing dataset. Then, we ran simulateEPL method on splitting dataset and tried to find the best parameters according to the best precision. After we find the best parameters, we make predictions on the newest season with the best parameters. FInally, we ranked teams according to their elo points.



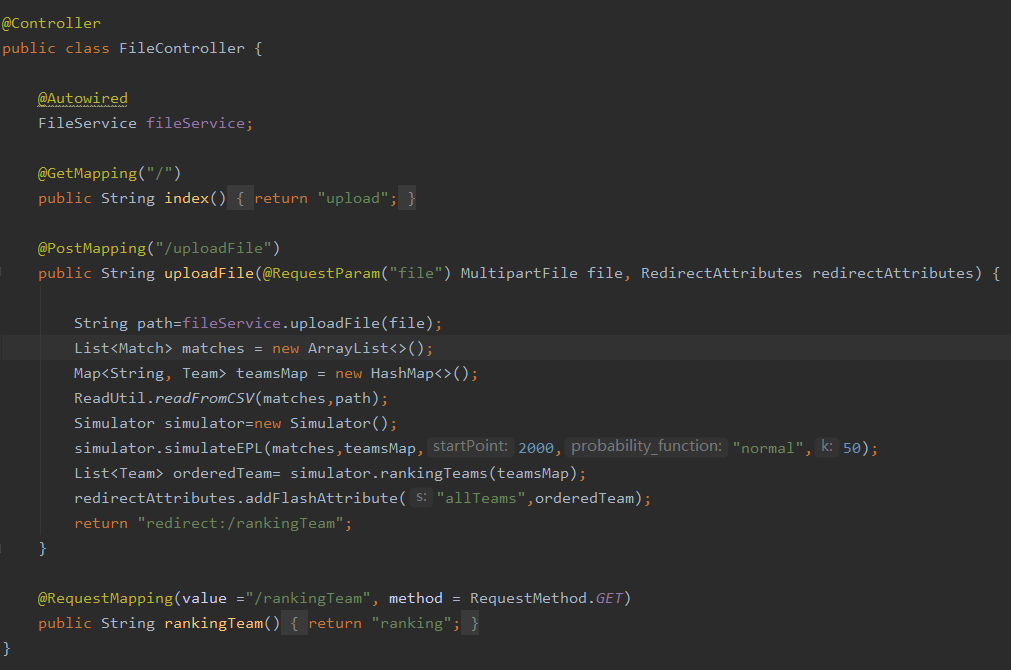
After getting the best parameters, we will simulate EPL(2019~2020). EPL is a double-loop system, divided into home and away games, each team played a total of 38 games.





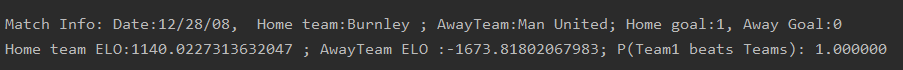
* 1. **Spring boot website**

We created a simple website based on the simulator that can upload csv files and then analyze csv files to rank teams.

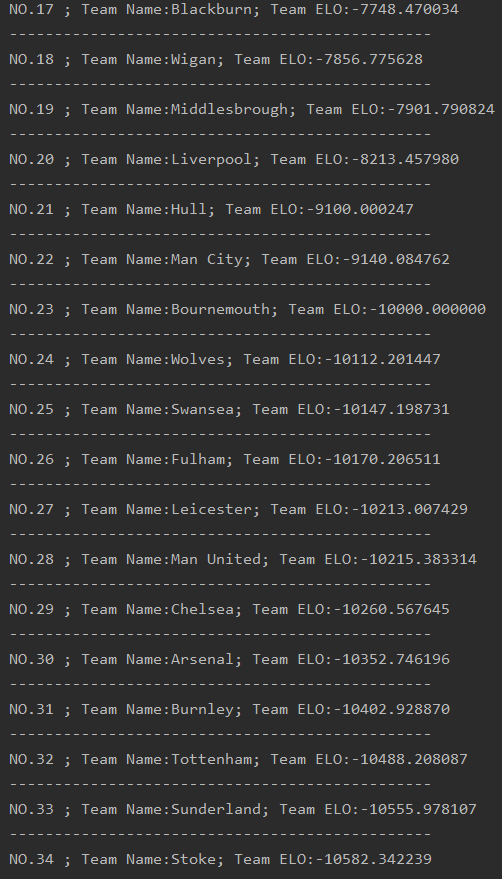


1. **Output and Observations**

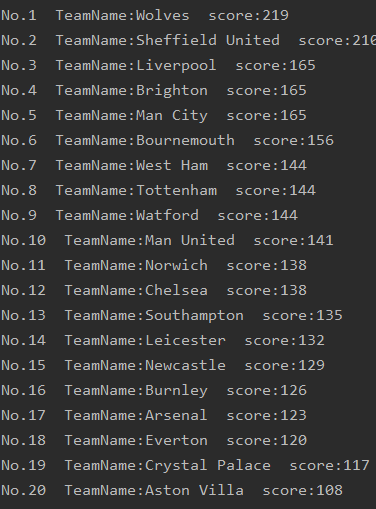
After running the main method of the simulator, we get some outputs about matches info, ranking info and precision info.



Photos above show the match info , updated elo points of each team and the probability after the match.

****Almost at the end of running, we got the best precision and best parameters based on all parameters experiments. The following photos show the rank of teams according ****to their elo points.

After finding best parameters, we started to simulate the EPL(2019~2020) by its rules and got final tables with 38 matches “played” by each team according to their score.



Based on the simulator, we also created spring boot websites to rank teams by the uploaded csv file. The following photos show the upload web page and ranking page.

1. **Conclusions**

* Different K-factors do influence the result of the ranking. In our case, K = 50 performs better, and normal distribution fits better than logistic curve.
* Normal distribution measures the probability of winning than logistic distribution does.
* The elo system is all about the gain and lose of elo points, so the starting point of the elo point doesn’t matter.
* The precision increases when the amount of data increases. So the more matches the teams played, the more accurate our prediction is.

1. **Reference**

* https://www.wikiwand.com/en/World\_Football\_Elo\_Ratings
* https://github.com/wxd/s3-2017-forecasting/blob/master/who\_is\_winning\_it.ipynb
* https://github.com/jnd18/elo-regression/blob/master/slides.pdf