INFO 6205 Ranking System - Project Report

Problem Statement

Your task is to develop a ranking system which is able to evaluate the following expression where xi, xj are elements from a set of competing elements X: P(xi, xj) where P(xi, xj) is the probability that xi would beat xj 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 xi appears above another element xj, then you may infer that P(xi, xj) > 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. Your choice of element is entirely up to you. There may be bonus points for originality. However, My recommendation would be to concentrate on the EPL, especially in light of COVID-19 which has abruptly terminated (or at least postponed) the season. Although Liverpool must end the season at the top of the table (it is mathematically impossible for any other team to pass them), the next five positions are important for the summer, as are the last three positions, which teams will be relegated.

It is desirable that the order of data input not affect the result. Please note that this is a very open-ended project. As a byproduct, you will learn about statistics and probability (more, if you already know some). There may be many “right” answers, even if they appear different.

Finally, there is a particular project which I’m interested in and that is a ranking system for bridge players. I have data for this and, if you’re interested in taking that on, please slack me. It will be harder than the EPL (or similar) problem but it will also carry some bonus points. For most of you, the EPL (or similar) problem will be your best bet.

And, even more finally, if you really want to do the traffic simulation (a harder problem), I would love that. Again, it will carry bonus points, but will involve much more work, I think. I wouldn’t recommend it if you are on your own.

Introduction

Created a system to predict the EPL 2019-20 rankings. Used data from 2000-01 to 2019-20 and predicted the remaining matches using probability density function to form the final standings table.

What is Probability Density Function?

In probability theory, a probability density function (PDF), or density of a continuous random variable, is a function whose value at any given sample (or point) in the sample space (the set of possible values taken by the random variable) can be interpreted as providing a relative likelihood that the value of the random variable would equal that sample. In other words, while the absolute likelihood for a continuous random variable to take on any particular value is 0 (since there are an infinite set of possible values to begin with), the value of the PDF at two different samples can be used to infer, in any particular draw of the random variable, how much more likely it is that the random variable would equal one sample compared to the other sample.

In a more precise sense, the PDF is used to specify the probability of the random variable falling within a particular range of values, as opposed to taking on any one value. This probability is given by the integral of this variable's PDF over that range—that is, it is given by the area under the density function but above the horizontal axis and between the lowest and greatest values of the range. The probability density function is nonnegative everywhere, and its integral over the entire space is equal to 1.

Calculating probability between two intervals using PDF

A probability density function is most commonly associated with absolutely continuous univariate distributions. A random variable X has density f_X, where f_X is a non-negative Lebesgue-integrable function, if:

$$\Pr[a \leq X \leq b] = \int_a^b f_X(x)\,dx.$$

Hence, if F_X is the cumulative distribution function of X, then:

$$F_X(x) = \int_{-\infty}^{x} f_X(u)\,du,$$

and (if f_X is continuous at x)

$$f_X(x) = \frac{d}{dx}F_X(x).$$

Intuitively, one can think of $f_X(x)\,dx$ as being the probability of X falling within the infinitesimal interval $[x, x + dx]$.

Aim of the Project

- Given two teams predict the winner
- To develop a ranking system for EPL season 2019-20

Data Set used in the project

- Match results from previous seasons are obtained from http://www.football-data.co.uk/englandm.php
- Data from the last 20 seasons are used

Implementation Description

Analysis

Below are the main two factors considered for predicting the results between two teams

- Previous results against each other for the last 20 years
- Home team performance against the opponent
- Average Mean goals scored against all opponents

Loading data from previous matches

Data from csv files are read line by line. Home team, Away Team and Goal Difference are obtained and stored in the objects. The structure of classes is as below

TeamDirectory will have a list of **Team**

Team consists of a Map of ProbabilityDensityFunctions with key being the away team and value being the **ProbabilityDensityFunction**

ProbabilityDensityFunction has map - occurrence. The key for this map is the goal difference and value will be the number of such occurrences from the data

The data is stored in the objects of the above classes

Building Probability Density Function

Once we have the data of the goal differences and their number of occurrences for each combination of teams, the probability is calculated for the goal difference. Mean and Standard Deviation is calculated as follows

Mean is the average of all values

The formula for standard deviation (SD) is

$$\mathrm{SD} = \sqrt{\frac{\sum |x - \mu|^2}{N}}$$

where $\sum$ means "sum of", x is a value in the data set, μ is the mean of the data set, and N is the number of data points in the population.

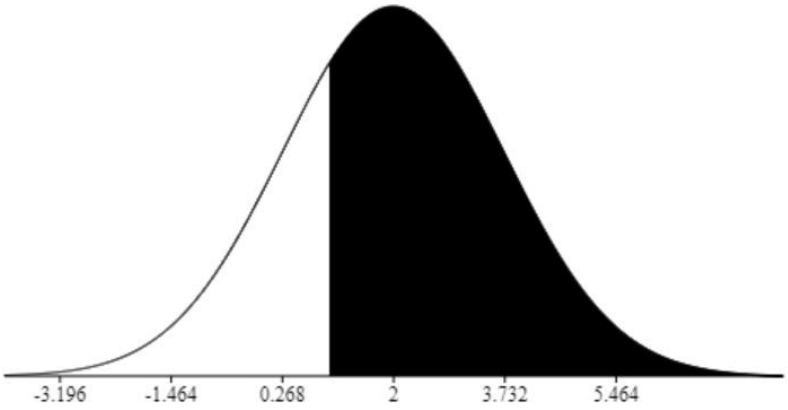
Predicting result between the two teams

Given a home team and away team, we fetch the Probability Density Function between those teams. By using Mean and Standard Deviation of the Probability Density Function, we calculate the area under the curve where x>1. This gives the probability of the home team winning. The area under the curve where -1<x<1 will give the probability of the draw. And the area under the curve x<-1 will give the probability of the home team losing. Whichever probability is highest among the three will be considered as the result.

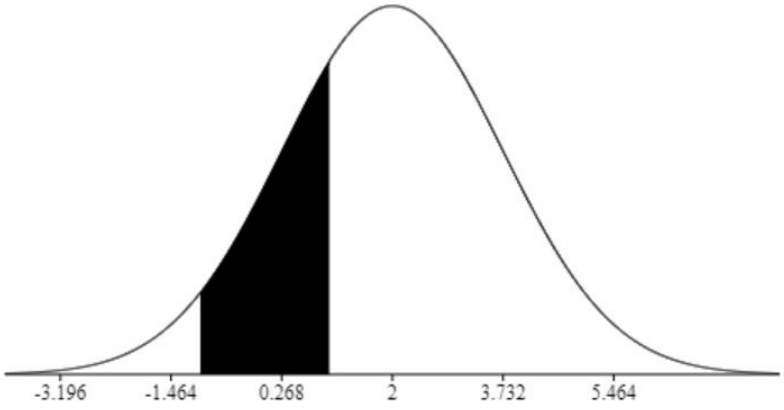
It's calculated using the probability method provided by NormalDistribution class. Used commons-math3 library to achieve this.

For instance, the Mean and SD for the Probability Density Function between Liverpool vs Brighton at Liverpool home is 2.0 and 1.732 respectively.

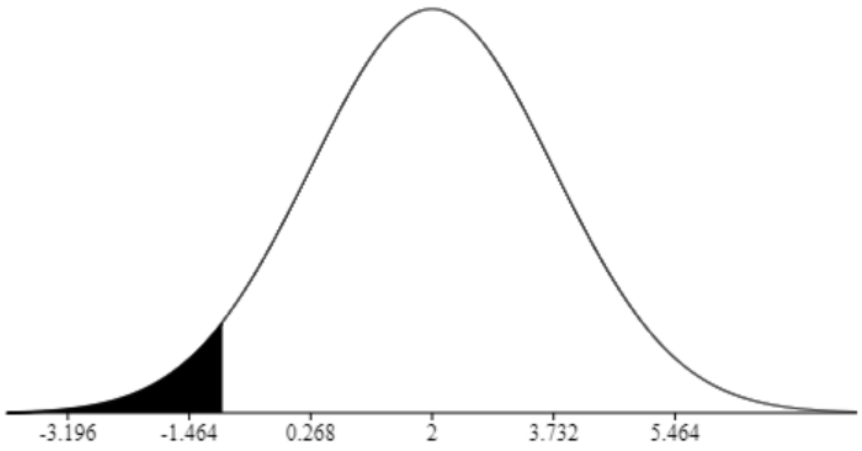
The area of the curve where x>1 will give the probability of Liverpool winning



The area of the curve where -1<x<1 will give the probability of draw



The area of the curve where x<-1 will give the probability of Liverpool losing



Constructing Ranking Table

Parse 2019-20 played matches results and build the table. The class structure is as follows

Table class consists of the list of **TableEntry**

TableEntry contains team name and points

Read each match result and calculate goal difference (homeTeamGoals - awayTeamGoals). If it's more than 0, add 3 points to the home team. If less than 0, 3 points to away team, otherwise 1 point to both teams.

Read upcoming fixtures form a csv file and predict the result using the logic mentioned in the previous topic. Update the table based on the predicted results.

We will get the final predicted table of EPL 2019-20.

Data Structures and Design Pattern Used

Map<String, ProbabilityDensityFunction> to store ProbabilityDensityFunction for a home team against each of its opponents

Singleton Pattern to create only one instance of TeamDirectory and Table object

- Made the constructor private
- Used a static method to get the instance of the singleton classes

Exceptional Scenarios

If teams haven't played against each other

- Considered the average mean number of goals scored by both against other opponents
- If the difference is greater than 1, consider the team with a higher value as winner
- Else considered the match as a draw

If one of the team is new and haven't played a single EPL match

- Consider the average mean goals scored by the other team against opponents
- If it's greater than 1, the new team loses
- If it's less than -1, the new team wins
- If it's between -1 and 1, considered as a draw

Further Improvements

- Considering the goal difference between the teams when the points scored are the same
- Considering other factors such as team momentum (current streak of wins/losses)
- Considering critical injury to the important players in the team

Output

You can either opt of predicting the result between a particular game or simulate the prediction for the remainder of the 2019-20 season

Predict result between two matches

```
Select one of the choices
1.Predict result between two teams
2.Predict EPL Standings for 2019-2020
3.Exit
1
Enter two teams (1st team entered will be considered as the home team)
Liverpool
Brighton
Winning Probability 0.7181485691746134
Liverpool will win the match
```

Predict 2019-20 ranking table

```
Select one of the choices
1.Predict result between two teams
2.Predict EPL Standings for 2019-2020
3.Exit
2
1     LIVERPOOL                98
2     MAN CITY                 81
3     CHELSEA                  71
4     MAN UNITED               65
5     ARSENAL                  62
6     LEICESTER                61
7     TOTTENHAM                57
8     SHEFFIELD UNITED         56
9     EVERTON                  51
10    CRYSTAL PALACE           50
11    BURNLEY                  49
12    WOLVES                   49
13    NEWCASTLE                47
14    SOUTHAMPTON              40
15    BRIGHTON                 35
16    BOURNEMOUTH              34
17    WEST HAM                 33
18    WATFORD                  31
19    ASTON VILLA              30
20    NORWICH                  29
```

Liverpool winning the Premier League 2019-20