

INFO 6205 Ranking System Project

Yuanxin Zhang 001405356

Yongjun Ou 001023025

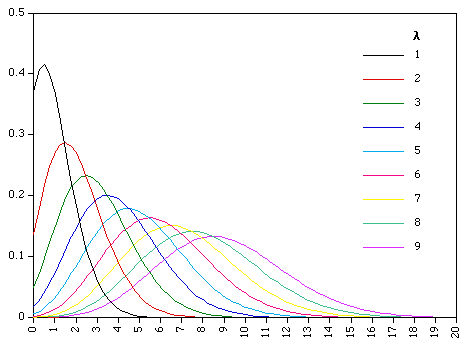


Figure1: Poisson Distribution

# Introduction

## What is Ranking System?

In statistics, ranking is the data transformation in which numerical or ordinal values are replaced by their rank when the data are sorted[1]. For example, we have the numerical data 27.5, 10.9, 17.8, 21.9, the ranks of these data items would be 1, 4, 3 and 2 (from large to small) respectively. In this project, we will based on the English Premier League, use our own method and proper program structure to rebuild a new analysis mechanism to rank all the teams. Finally, we will show the team ranking result in tables.

## What is Poisson Distribution?

*The Poisson distribution*, named after French mathematician Simeon Denis Poisson, is a discrete probability distribution that expresses the probability of a given number of events occurring in a fixed interval of time or space if these events occur with a known constant mean rate and independently of the time since the last event[2]. We will use it and the *Cumulative Distribution Function* to calculate the theoretical calculation distribution of the number of goals, instead of *Possibility Density Function of Normal Distribution*. We will discuss it more detailed in the following Implementation section.

## About the Premier League

The Premier League, often referred to as the English Premier League or the EPL outside England, is the top level of the English football league system. Contested by 20 clubs, it operates on a system of promotion and relegation with the English Football League (EFL). Seasons run from August to May with each team playing 38 matches (playing all 19 other teams both home and away)[3]. Here are the origin terms that we use in our project:

### Relevant Terms

* FTHG: The total number of goals scored by the home team during the match at full time.
* FTAG: The total number of goals scored by the away team during the match at full time.
* FTR: The full time result, denoted as 'H' for home team win, 'A' for away team win, or 'D' for draw

### 2019–2020 Premier League

The 2019–20 Premier League is the 28th season of the Premier League. Different from the past, it is currently halted until at least 30 April 2020, following a decision on 13 March 2020 by the Premier League to suspend the league after a number of players and other club staff became ill due to the coronavirus pandemic[4].

# Problem Statement

Designed a Ranking System for the EPL

* Get the EPL season data from official website, and convert it to usable data
* Build Flexible, Dynamic Data Model from source file for the following rank step.
* Simulate reliable data for predicting the probability to each team pairs for the result.
* Generate final ranking table with meaningful and logical ranking method.
* Generate table for each team showing their winning probability to other teams.

# Project Description

This project is mainly about developing 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.

We take teams as an object to complete the rank table. First, we counted the results of all matches and ranked team according to the number of wins. Then, we counted the goal difference of each team, and then sort them from large to small. Additionally, this project also includes the evaluation mentioned above, which is about pair comparison. We depends on

# Datasets used in the Project

Json Data Set: season-1920.json (from www.football-data.co.uk/englandm.php)

A group of people

Description automatically generated

Figure2: Part of season-1920.jason file screenshot

In order to test our system’s correctness, we also use other season’s data like season-1819.json:



Figure2: Part of season-1819.jason file screenshot

# Implementation

1. Build Data Model

Here is the outline of project in Eclipse IDE:

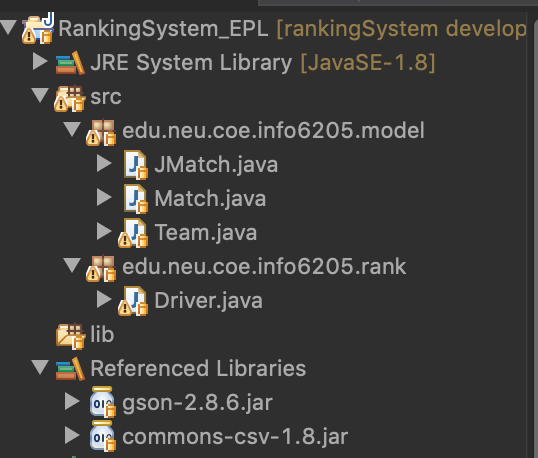


Figure4: Project outline screenshot

As known from the screenshot, we has two package, one is for the data model, the other is in charge of drive the program and do the rank. In model section, JMatch.java is use to input the origin data from json file, Match.java is the match object we exactly used in data process, Team.java is the team object, which includes various attributes like the following pictures show.

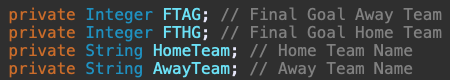


Figure5: JMatch’s attributes screenshot

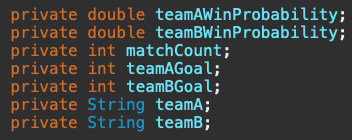


Figure6: Match’s attributes screenshot

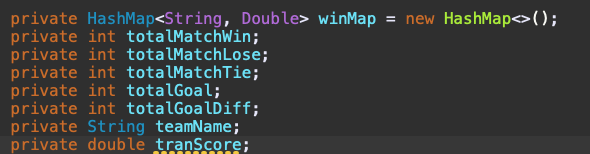


Figure7: Match’s attributes screenshot

1. Poisson distribution:
   1. If the value of discrete variable X can be 0,1,2…..and the probability of each value is
   2. A picture containing object, clock

      Description automatically generated(  )，we call  X  obey the Poisson Distribution with variable  .  is the expectation.
2. We have known that the goal of football game match Poisson Distribution. Then we using the Poisson distribution model and the average number of goals per game, calculate the theoretical calculation distribution of the number of goals.
3. As we have calculated the number of goals through the previous process. And we know the exact distribution of the event that we are going to simulate. We can then do the Monte Carlo method. Randomly generate gaming goals as well as result following Poisson Distribution of every , and calculating the probability of win, tie and lose.

## Solution Flowchart

Extract every data model & convert into Team Model

Modeling the data to original model data

Read the original data from json

Loop to do 100K simulation through the Possion simulation

Insert data to the getPossionVariable(), get the simulation data

Iterate the data extract from team model

Ranking teams by sorting the gaming result table by (TranScore, Wins, GoalDiff)

Count all team pairs gaming result from their probability

Calculate the result probability for each team pairs

Figure8: Implementation step in Flowchart

### Implementation of get Poisson Variable from Lamda

A screenshot of a cell phone

Description automatically generated

### Implementation of Monte Carlo method

A screenshot of a social media post

Description automatically generated

### Implementation of Sorting

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

# Result & Analysis

Here are the result based on season 19-20 dataset, we divided it into 3 final table according to different rank method

### Ranking By Match wins

A screenshot of a cell phone

Description automatically generated

Figure9: Rank Table in Match Wins

### Ranking By Game Goal difference

A screenshot of a cell phone

Description automatically generated

Figure10: Rank Table in Goal Difference

### Ranking By TranScore (Add weight to match: 1 for Wins, 0.5 for Ties, 0 for Loses)

A screenshot of a cell phone

Description automatically generated

Figure10: Rank Table in TranScore

The following table shows the win possibility for one each team, against to the other 19 teams:

### Team table (Ex: Liverpool vs. Other Teams)

A screenshot of a cell phone

Description automatically generated

Figure11: Rank Table in Win Probability

Output csv file:

1. Ranking tables sorting by Number of game wins, number of Goal difference, Transfer Score. (Transfer Score = 1\*wins + 0.5\*tie + 0\* loss)
2. Tables showing for each team. The probability result of gaming with other teams are shown.
3. By analyzing the data table below, EPL season 2019-2020, the team Liverpool ranks #1 on the 3 kinds of final ranking table. But it will lost some games with high probability. The team Norwich lies on the bottom of the ranking tables.

# Reference

[1] <https://en.wikipedia.org/wiki/Ranking>

[2] <https://en.wikipedia.org/wiki/Poisson_distribution>

[3] <https://www.premierleague.com/>

[4] <https://en.wikipedia.org/wiki/2019%E2%80%9320_Premier_League>

[5] <https://en.wikipedia.org/wiki/Monte_Carlo_method>