NBA Team Rankings Assessment and NBA Team Wining rate in the Game.

CS 179 – Spring 2019

Hu Haotian

Tan Conghuai

Gan Peilin

June 2019

Abstraction. The task of assessing the skills of players and teams is important in modern games, especially given the fact that online-real time gaming has been more and more popular in recent years. A "fair" game matching system is the key factor determining user's game experience, which involves assessing skills of players and teams. A poor game match strategy would discourage less-skilled player from keep playing. Thus, in this work, we aim to find a probability model to estimate the skills of players or teams base on past game outcomes. Furthermore, we aim to develop a way to predict the outcome of a match according to skill model we train. Specifically, we (1) choose an existing probability model called TrueSkill to train our data, (2) try out the model on data from NBA regular season team match outcomes, (3) assess the skills of each NBA team basing on regular season outcomes then rank all 30 teams according to their skills, (4) compare the predicted team rank with real rank to evaluate the performance of model, (5) improve the model through empirical experiments, and (6) evaluate our model by comparing the prediction rank with real rank.

I. Introduction

Ranking NBA teams is quite challenging due to several reasons. Primarily, we are assessing each team basing on game outcome. However, the match outcome isn't the deterministic factor in evaluating team's skill. If team A wins team B in a match, we can't say A is guaranteed to be the stronger team, which means, there exists a great deal of uncertainty in game outcome and we should reduce the weight of uncertainty in our assessment. The key problem we have is that the TrueSkill probability model only learns from win, lose, or draw outcomes and cannot use additional outcome information such as score, rank, and winning streak. We aim to take into account all potential factors in our skill assessment to improve the model. Degree of uncertainty is another problem we have. When assessing a team's skill, degree of uncertainty determines how strong our ranking system believe its rank to estimate the interval of its skill level. Our model will modify degree of uncertainty for each team according to the game outcomes. The goal of this work is to develop and improve a probability model to assess teams' skill as close as possible to their real skill.

II. Description of Resources

The resource we used in this project is TrueSkill. TrueSkill is a probability model written by Microsoft that estimates the skill of players. The data we used is the 2018-2019 NBA regular season score table. By using the TrueSkill and score table, our model tries to predict the rank of each team.

The TrueSkill uses the Bayesian graphical model (factor graph) to improve the “mu” and “sigma” after each score. For example, in our data, for every score, M = {i, j} which i and j is two different team, o is the outcome: {team 1 win, team 2 win} and l which is vector that represents the rating. We calculate the P (l | o, M) by estimates the posterior distribution according to Bayes’ rule:

$$p(\boldsymbol{l}|o, M) \propto p(o|\boldsymbol{l}, M)p(\boldsymbol{l})$$

The p(l) can be calculated by the Gaussian prior and p(o |l, M) can be calculated by the normal distribution of every model. By getting the p(l| o, M), TrueSkill will update the team I and team 2 by comparing with the outcome of the score.

The team’s power is estimated by the normal distribution which comes from the mean value “mu” – “u”, and confident value “sigma” “s”. Initially, “u” will be 25 and “s” will be 25/3. Because of the normal distribution property, the 99.7% of values are within 3 “s” of the “u”. So, the bottom line of every player will be 0 (25 – 3 * 25/3). After each game, the “mu” and “sigma” of each team will be updated according to game outcomes. At the end, we will use the two method: 1. “u” 2. “u” – 3 * “s” to estimate the rank of teams. Below is how to estimate the ranks:

```python
from trueskill import Rating, quality_1vs1, rate_1vs1, TrueSkill
import pandas as pd
import os

def win_lose(homeTeam, awayTeam, result):
    firstRecord = int(result.split(' - ')[0])
    secondRecord = int(result.split(' - ')[1])
    if (firstRecord > secondRecord):
        return (homeTeam, awayTeam)
    return (awayTeam, homeTeam)

record = pd.read_csv('/Users/tanconghuai/uci/大三spring/CS179/nba-2018-EasternStandardTime.csv', sep=',')

resultDict = {teamName: Rating() for teamName in record['Home Team']}
for index, row in record.iterrows():
    win, lose = win_lose(row['Home Team'], row['Away Team'], row['Result'])
    resultDict[win], resultDict[lose] = rate_1vs1(resultDict[win], resultDict[lose])

sortList = sorted(resultDict.items(), key = lambda x: -(x[1].mu - 3 * x[1].sigma)) # use the mu and sigma to calculate
sortList = [x[0] for x in sortList]
sortList1 = sorted(resultDict.items(), key = lambda x: -x[1].mu) # use only mu
sortList1 = [x[0] for x in sortList1]
actualRank = ['Golden State Warriors', 'Milwaukee Bucks', 'Houston Rockets', 'Toronto Raptors', 'Denver Nuggets', 'Portland Trail Blazers',
```

III. Technique Evaluation

To evaluate the technique that we chose to predict rankings for NBA team, we have our first predict data “Trained Rankings with u”. Each team’s “u” refers to its average skill value, the ranking in “Trained Rankings by u” is ranked by the value of “u”. We compared the “Trained Rankings by u” with the team “Power Rankings” that provided by NBA official then formed the histogram graph to visualize the comparison.

In the histogram graph, the x-axis is the names of NBA teams and the y-axis is the rank of each team. Figure 1 is the graph of the comparison, red color represents “Power Rankings” and green color represents “Trained Rankings by u”.

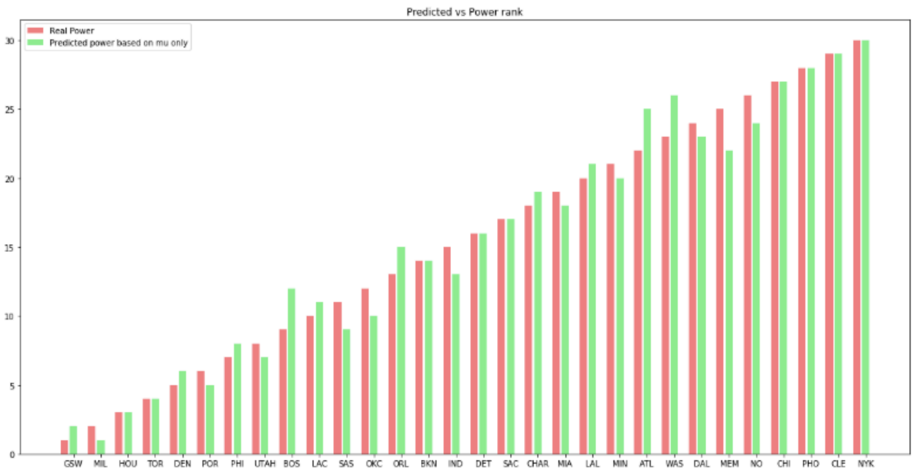

Fig.1

From the predicted data and training data, we get the mean square error of:

```
MSE base on mu only =  2.2666666666666666
```

MSE value tells us the accuracy of our model of prediction. To improve our model, we improve model by adding uncertainty factor in assessment. In our second trained dataset, we used the “mu” and “sigma” at the same time. We know that “u” is the value of average skill, “s” here represents the uncertainty of the data, by the property of normal distribution, 99.7% of values are within [“u” +/- 3 * “sigma”]. In this way, we decide to use “mu” – 3 * “sigma” to use as the rank because this value is the conservative or bottom line value of the team. By using this value, we will know the bottom line of each team in case there is no enough data. Fig.2 below is the histogram graph of comparison between “Trained Rankings by u and s” and “Power Rankings”.

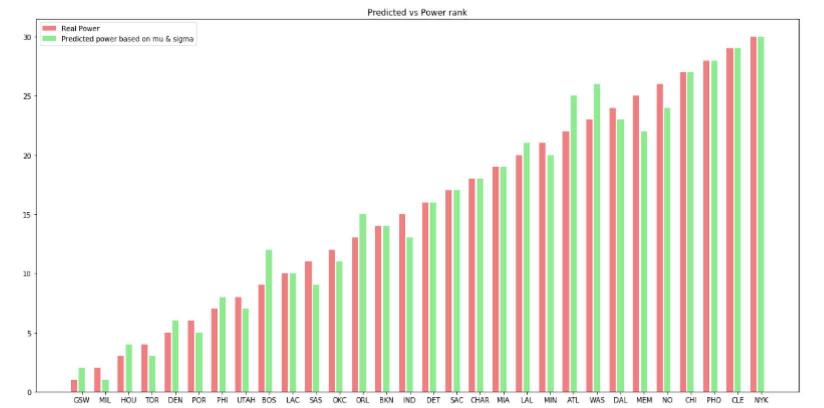

Fig.2

The mean square error is:

```
MSE base on mu & sig =  2.1333333333333333
```

Apparently, the performance of the model is much better since now we have smaller MSE.

Although the model has better performance now, the mean square error is still higher than we expected. We can tell from the graph that half of the predictions are wrong. Then we figure out another way to improve the model. Since one team’s strength is increasing or decreasing not only based on win or lose, the change of strength should also consider the difference of two teams’ score and their ranks. In this way, we decide to add some weight for the winning and losing team. For example, after we receive the score, we will also update the mu inside the winning and losing team by +/- log10 (score different + opponent team's rank). In this way, we not only considered about the win or lose situation, but also considered about the score difference and opponent ranking. Thus, we had our third trained data "Trained Rankings by u&s&w".

Below is the comparison of "Trained Rankings by u&s&w" and "Power Rankings".

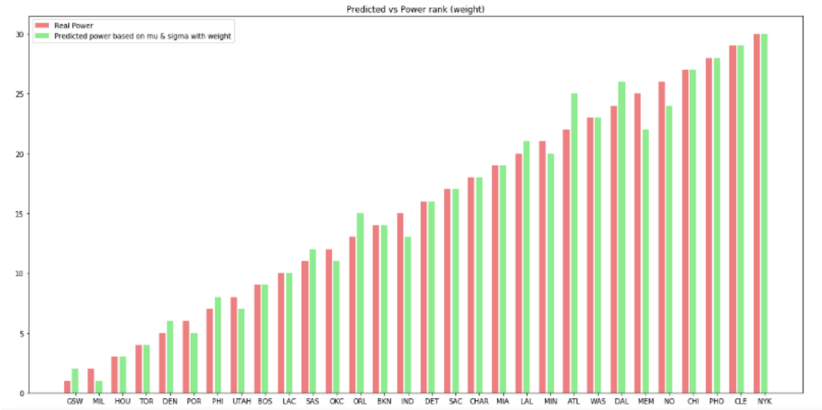

Fig.3

The MSE_based_on_by_u&s&w is:

```
MSE base on mu & sig =  1.4666666666666666
```

We have successfully improved our model by reducing the MSE to 1.46, which indicates better performance than previous models.

IV. Conclusion and Result

Through the paper *Score-based Bayesian Skill Learning*, we have found the probability model TrueSkill using the Gaussian Distribution model and Bayesian's graphical models to learn and predict whether the players/teams fairly match each other in the game playing. However, TrueSkill model only cares about the result of games, win or lose, but doesn't care about any other factors such as one team may much improve by defeating a team that has much higher rank compare to its own.

As a conclusion, we agree most of the claims in the paper we read but not all the claims. We did some improvements in the model to perfect the model to increase the predict accuracy. Adding weight into the ranking standards, in our dataset, we considered "won by amount of points" and "ranking differences between two teams" as weight, and we successfully improve the accuracy of our predicting model.

Citation

Score-based Bayesian Skill Learning - microsoft.com. Retrieved June 14, 2019, from https://www.microsoft.com/en-us/research/wp-content/uploads/2012/01/sbsl_ecml2012.pdf