Predicting the Improvement of NBA players

Zhenfeng Liu

October 18, 2018 1.

1. Introduction
   1. Background

Our client is a startup craft beer brewer, they look for a distribution network of their craft beer in one of the area in Toronto. Since the supply of craft beer is limited, high target selling price, and the special flavor of the craft beer, they would like to find out the most suitable area to maximize their profit.

* 1. Challenge

There are some challenge that client would like to study, i) Craft beer with limited supply and “best tasting period“, ii) High target selling price (i.e. 60% more expensive than branded beers e.g. Heineken, Budweiser) and iii) Special flavour like herbs, sours, salty lemon, etc. (Asian flavour)

* 1. Requirement from client

Client looks for an area with lots of bar/pubs/restaurants, to ensure that people in that area are willing to spend money on foods and drinks. Besides, area with Asian restaurants is preferred, as client thinks that it would be a selling point of craft beers with special flavor from Asia. Lastly, people are willing and affordable to spend money on the beers.

2. Data acquisition and cleaning

* 1. Data sources

Data could be found from below URL / API

* FourSquare developer API
* https://en.wikipedia.org/wiki/List\_of\_postal\_codes\_of\_Canada:\_M (List of postal codes of Canada)
* http://cocl.us/Geospatial\_data (Latitude and Longitude data)
  1. Data cleaning

Data downloaded or scraped from multiple sources were combined into one table. There were a lot of missing values from earlier seasons, because of lack of record keeping. I decided to only use data from 1980 season and after, because of later seasons have fewer missing values and basketball was a lot different in the early years from today’s game. There are several problems with the datasets. First, players were identified by their names. However, there were different players with the same names, which cause their data to mix with each other’s. Though it was possible to separate some of them based on the years, teams, and positions they played, I decided that it was not worth the large effort to do so, because such players only accounted for ~1% of the data. Therefore, players with duplicate names were removed. Second, multiple entries existed for players who changed teams mid-season. This cause their seasonal data to represent multiple samples with incomplete data. I wrote script to extract total season stats for these players, and discarded partial season rows. Third, there were two short seasons in recent NBA history, during which less than the normal 82 games were played. This has caused stats in those seasons to be artificially smaller than other seasons. To correct that, I normalized cumulative features such as points, rebounds, etc. as if 82 games were played. After fixing these problems, I checked for outliers in the data. I found there were some extreme outliers, mostly caused by some types of small sample size problem. For example, some players had only played a few games or a few minutes the entire season, and had performed extremely well or poor in those minutes. Therefore, seasons during which less than 20 games or 100 minutes were played were dropped from the dataset. Similarly, there were players who only took one 3-point shot, but made it, therefore had 100% shot accuracy. I changed the shot accuracies for players who shot less than 10 shots to missing values. There were 4 features which had missing values. Games started were imputed from minutes played because starters usually play more minutes. Missing 3-point accuracies were imputed with a very small value (0.05) because if a player rarely shoots 3s, it is probably because he is not very good at it. Missing free throw accuracies were imputed using the mean of all players. Missing draft positions, meaning undrafted, were imputed using position 61 (the position after the last position in the draft, 60th). 2.3 Feature selection After data cleaning, there were 13,378 samples and 49 features in the data. Upon examining the meaning of each feature, it was clear that there was some redundancy in the features. For example, there was a feature of the number of rebounds a player collected, and another feature of the rate of rebounds he collected. These two features contained very similar information (a player’s ability to rebound), with the difference being that the former feature increased with playing time, while the latter feature did not. Such total vs. rate relationship also existed between other features. These features are problematic for two reasons: (1) A player’s certain abilities were duplicated in two features. (2) A player’s playing time were duplicated in multiple features. In order to fix this, I decided to keep all features that were rates in nature, and drop their cumulative counterparts (Table 1). There were also other redundancies, such as that total rebounds are the sum of offensive rebounds and defensive rebounds. For features that can be calculated by sum of other features, I decided to drop them (Table 1). After discarding redundant features, I inspected the correlation of independent variables, and found several pairs that were highly correlated (Pearson correlation coefficient > 0.9). For example, shots attempted, shots made, and points scored were highly correlated. This makes sense, after all, you score points by making shots. From these highly correlated features, only one was kept, others were dropped from the dataset. After all, 24 features were selected. Table 1. Simple feature selection during data cleaning. Kept features Dropped features Reason for dropping features TRB%, ORB%, AST%, STL%, BLK%, TOV%, TRB, ORB, AST, STL, BLK, TOV Two similar features (one being total, one being rates) depicting the same ability of players. TRB%, ORB%, WS, OWS DRB%, DRB, DWS Total = offense + defense. Dropped defense. TS%, FGA, 3P%, 3PA 2PA, 2P, 2P% Field goal = 2-point shots + 3-point shots. Dropped 2-point shots. TS%, WS FG%, eFG%, VORP, BPM, OBPM, DBPM Slightly different features that depict the same overall abilities of players. 3. Exploratory Data Analysis 3.1 Calculation of target variable Player improvement year over year was not a feature in the dataset, and had to be calculated. I chose to calculate the difference of win shares between two consecutive years as the target variable. Win shares were chosen out of a few metrics because it is the most interpretable, after all, we play basketball to win. Calculated player improvement had a normal distribution centered around 0, with most values between -6 and 6. To verify if this calculation is consistent with people’s eye-test of player improvement, I plotted the rank of improvement of past Most Improved Players winners among all players, and found that in most cases, they were among the most improved players (Figure 1). This suggested that the chosen metric of player improvement, was a reasonable one. 3.2 Relationship between improvement and age It is widely accepted that younger players are more likely to improve than older players, and it was indeed supported by our data. Players’ median improvement declined as players’ age increased (Figure 2), and the mean improvement of different age groups (35) were all significantly different from each other (z-test, p35, p=0.002). Figure 1. Rank of delta-win-share of Most Improved Players winners among all players of each year Figure 2. Box plot of improvement of players of different ages. 3.3 Relationship between improvement and overall ability The hypothesis here is that players who are already stars don’t have much room to improve, while a mediocre player can still improve. Our data were consistent with this hypothesis. Using win share per 48 minutes (WS/48) as a measure of a player’s overall ability, I observed a negative relationship between a player’s overall ability and his improvement next season (Figure 3). The mean improvement of star players (WS/48 > 0.2), solid players (WS/48 between 0.1 and 0.2), rotational players (WS/48 between 0 and 0.1), and “scrubs” (WS/48 below 0) were significantly different from each other (z-test, p=0 or