Predicting the optimal location to open a restaurant

Mayank Rastogi

26th June 2020

1. Introduction

1.1 Background

A restaurant or an eatery, is a business that prepares and
serves food and drinks to customers. Meals are generally served and eaten on the premises, but many restaurants also offer take-out and food delivery services. Restaurants vary greatly in appearance and offerings, including a wide variety of cuisines and service models ranging from inexpensive fast food restaurants and cafeterias, to mid-priced family restaurants, to high-priced luxury establishments. Today there are over a hundred thousand restaurants and small eateries in every city to choose from and hence the competitions is very hard. Therefore, it is advantageous for teams to accurately predict whether and where a restaurant should be opened so that it's the most profitable.

1.2 Problem

A restaurant is not run and made profit from solely based on its location. It is done through the food, more specifically, for a restaurant to be popular in an area it should serve the food that is the most popular in an area or a taste that any other restaurant fails to provide.

1.3 Interest

This topic is very interesting as according to an article written by the US STATS, opening a restaurant is the most popular small business one goes for. Therefore all entertainers start from something small and opening a profitable restaurant is one of the best ways to do so.

2.Data acquisition and cleaning

2.1 Data sources

Most player stats, position, age, and draft position data can be found in two Kaggle datasets here and here. These two datasets, however, lack data for certain years. For example, the player stats dataset ends in 2017, and the player draft dataset starts in 1978 and ends in 2015. To complement these two datasets, I scraped cooking-reference.com for player season stats of 2018 and player draft positions of 1965-1977 and 2016-2017 (restaurants drafted in 2018 has yet to play in RESTAURANTS).

2.2 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 cooking was a lot different in the early years from today's game. There are several problems with the datasets. First, restaurants were identified by their names. However, there were different restaurants 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 restaurants only accounted for ~1% of the data. Therefore, restaurants with duplicate names were removed. Second, multiple entries existed for restaurants 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 restaurants and discarded partial season rows. Third, there were two short seasons in recent RESTAURANTS 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 restaurants 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.

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.

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. 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 cooking was a lot different in the early years from today’s game. There are several problems with the datasets. First, restaurants were identified by their names. However, there were different restaurants 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 restaurants only accounted for ~1% of the data. Therefore, restaurants with duplicate names were removed. Second, multiple entries existed for restaurants 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 restaurants and discarded partial season rows. Third, there were two short seasons in recent RESTAURANTS 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 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).

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)

<u>4. Predictive Modeling</u>

There are two types of models, regression and classification, that can be used to predict player improvement. Regression models can provide additional information on the amount of improvement, while classification models focus on the probabilities a player might improve. The underlying algorithms are similar between regression and classification models, but different audience might prefer one over the other. For example, an NBA team executive might be more interested in the amount of improvement (regression models), but a general NBA fan might find the results of classification models more interpretable. Therefore, in this study, I carried out both regression and classification modeling.

4.1 Regression models

4.1.1 Applying standard algorithms and their problems

I applied linear models (linear regression, Ridge regression, and Lasso regression), support vector machines (SVM), random forest, and gradient boost models to the dataset, using root mean squared error (RMSE) as the tuning and evaluation metric. The results all had the same problems. The predicted values had much narrow range than the actual values (Figure 8), and as a result, the prediction errors were larger as the actual values deviated further from zero (Figure 9). These results were not acceptable, because players with large improvement/decline were arguably more important for NBA teams to predict than players with little change in performance. Having larger errors on those predictions was obviously not desirable.

4.1.2 Solution to the problems

The reason behind these problems were the uneven distribution of player improvement, in that players with little improvement/decline were more common than players with big improvement/decline (Figure 8). Therefore, the models tried to prioritize minimizing errors on players with little improvement/decline when RMSE was used as the evaluation metric. My solution to this problem was to assign weights to samples based on the inverse of the abundances of target values. In other words, players with large improvement/decline would have higher weights in model training and evaluation because they were more rare. Using this method, all models predicted target values with similar range and distribution as the actual target values (Figure 10).