Predicting NBA Draft Prospect Value with Learning to Rank

Jack McCarthy

Abstract

Evaluating NBA draft prospect value prior to the draft is a difficult and highly variable process. Historically, NBA teams had trouble identifying value at both high and low draft positions. This project proposes the implementation of a supervised learning to rank model–LambdaMART–which uses NBA player college stats to predict their known value in the NBA. Comparisons with the true draft order show the model achieves similar performance to NBA teams in predicting ultimate player value.

1 Introduction

Given the degree to which an NBA team's success hinges on its ability to draft well, assessing draft prospect value is a crucial step in the team-building process. Current methods of doing so range from traditional scouting, centered around physical attributes and observable skills, to more advanced statistical analyses. The aim of this project is to provide a new data point for consideration when evaluating prospects by producing player rankings via machine learning.

2 Data

2.1 Data Collection

All data used in this project was sourced from Sports Reference, which contains comprehensive data from most major American sports including both college and professional basketball [3]. Player information and statistics were scraped using BeautifulSoup–a versatile HTML parser–in conjunction with pandas. The resulting data tables are indexed by the unique player ID supplied by NBA Reference.

Players were included in the data set if they were drafted between 1998 and 2018 and have college statistics available. This analysis excludes players that were drafted out of high school or from international leagues.

2.2 Feature Engineering

Along with the raw statistics and physical attributes scraped from Basketball Reference, the following features were generated to capture additional information about the players:

Feature	Type	Definition
years	numeric	Number of years in college
bling	numeric	Number of awards earned in college
top_conf	binary	Player played for a top-5 conference
top_school	binary	Player played for a top-5 school
rsci_10	binary	Player was a top 10 high school recruit
rsci_50	binary	Player was a top 50 high school recruit
rsci_100	binary	Player was a top 100 high school recruit
guard	binary	Player is a guard
forward	binary	Player is a forward
center	binary	Player is a center

Top 5 schools and conferences were those which produced the most NBA players historically. If a player had multiple seasons worth of stats, the stats from each player's best college season (according to win shares) were chosen to ensure that the feature vector was 1-dimensional.

2.3 Grouping Data

Traditionally, learning to rank algorithms frame the data as the 3-tuple (q_i, d_{ij}, r_{ij}), where q_i is the query label, d_{ij} is the j^{th} document in the query, and r_{ij} is said document's associated relevancy score [2]. In our case, q_i represents the i^{th} draft from 1998 to 2018, and d_{ij} is the j^{th} player picked in draft q_i. The relevancy score r_{ij} was defined in terms of a player's impact in the NBA, which can be roughly summarized by the Value Over Replacement Player (VORP) metric.

3 Modeling

3.1 Response Definition

For the purposes of this project, we needed to define the relevancy score in such a way that we capture a player's potential. VORP provides a clean single-number metric upon which we can construct our relevancy scores, there are multiple values for each player across seasons. As such, we set relevancy to be

$$r_{ij} = \lfloor F_{X_{(n_i)}}(X_{ij}) \rceil \ / \ 10$$

where

$$F_{X_{(n_i)}}(X_{ij}) = \text{Prob}(\max\{X_{ik}\}_{k=1}^{n_i} \leq X_{ij})$$

with X_{ij} being the mean of the largest 3 VORP values for player d_{ij} and n_i being the number of players in draft q_i. Averaging the best 3 values encodes some information about the consistency of a player over time. This provides an integer higher-the-better relevancy score for each player from 0 to 10.

3.2 Model Definition

LambdaMART is a gradient boosted tree algorithm designed specifically for ranking tasks. Its ability to rank comes from its specialized objective function, which incorporates information about the quality of the rankings it produces during training.

The generalized formulation for the XGBoost objective is

$$\text{obj}^{(t)} = \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t)$$

where g_i and h_i are defined as

$$g_i = \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$$
$$h_i = \partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$$

which means all we need to do to specify a custom objective is define the loss function l, and from it, derive g_i and h_i [1].

Suppose I is the set of all pairs of documents (across all queries) which has been sorted according to predicted score. Leting $s_i = \hat{y}_i^{(t-1)}$, the LambdaMART loss function is defined to be

$$l = \sum_{\{i,j\} \rightleftharpoons I} |\Delta Z_{ij}| \log \left(1 + e^{-\sigma(s_i - s_j)} \right)$$

which gives

$$g_i = \sum_{j:\{i,j\} \rightleftharpoons I} -\sigma \, |\Delta Z_{ij}| \, \rho_{ij}$$

$$h_i = \sum_{j:\{i,j\} \rightleftharpoons I} \sigma^2 \, |\Delta Z_{ij}| \, \rho_{ij} (1 - \rho_{ij})$$

where

$$\rho_{ij} = \frac{1}{1 + e^{\sigma(s_i - s_j)}}$$

In our case, Z is the Normalized Discounted Cumulative Gain (NDCG), which for relevancy scores sorted by their associated predicted score is defined as

$$\text{NDCG}_k = \frac{\text{DCG}_k}{\text{IDCG}_k}$$

where the Discounted Cumulative Gain (DCG) for rank position k is given by

$$\text{DCG}_k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i+1)}$$

and IDCG_k is the maximal value of DCG_k reached by the optimal ordering [2]. ΔZ_{ij} represents the change in NDCG by swapping documents i and j.

The XGBoost algorithm is then carried out as normal with the newly defined functions for g_i and h_i.

3.3 Hyperparameter Tuning

Hyperparameter tuning was done using a grid search over parameters in conjunction with model evaluation via leave-one-out cross-validation. The values of the parameters that gave the most performant model were then used to conduct testing.

4 Results

As a benchmark for testing, we can create predicted values based on a player's actual draft position. We define the "pick score" for player i picked at position p_i to be $s_i = -\log p_i$. Grouping by actual relevancy score, we can investigate the distributions of the predicted value for the model and actual pick position.

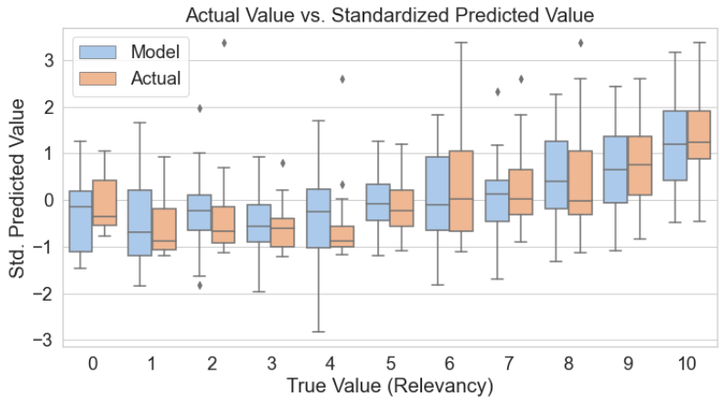

Figure 1: Predicted value distributions

We note that the distributions look quite similar, with the model generally being slightly more optimistic about ultimately worse players than the NBA teams. The model predictions for worse players also appear to vary more than the NBA teams.

We may also look at the NDCG@20 for the predicted rankings. This plot includes the mean NDCG for simulated random orderings to serve as a control.

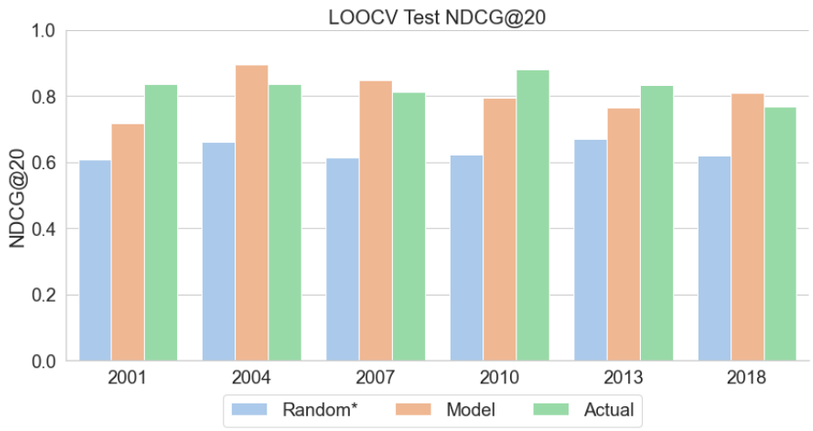

Figure 2: Random order vs. model predictions vs. true picks

In this plot we can see that the model actually performs better than true draft order based on NDCG@20 in three out of the six test years. It can also be seen that both the model and teams were significantly better at identifying player value than randomly determined orders.

5 Conclusion

The learning to rank methodology outlined above has demonstrated great promise in evaluating draft prospects with only publicly available data. The ability of the model to predict player value as compared to actual NBA draft positions is particularly notable. Further investigation into acquiring a wider range of data, developing new features, and finding alternatives to the current relevancy definition is still needed to fully ascertain the potential utility of this approach. This work serves as a proof of concept for a purely machine learning-based method in the ranking of NBA prospects.

References

[1] Chen, T. Guestrin, C., 2016. XGBoost: A Scalable Tree Boosting System. *In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD x27;16. New York, NY, USA: ACM, pp. 785–794.

[2] Chris J.C. Burges. 2010. From RankNet to LambdaRank to LambdaMART: An Overview. *Microsoft Research Technical Report MSR-TR-2010-82*

[3] Sports Reference LLC. Basketball-Reference.com - Basketball Statistics and History. https://www.basketball-reference.com/. (2022)