Machine Learning Engineer Nanodegree
Capstone Proposal
Matthew Ignal
January 9th, 2017

NBA Prospects Algorithm Proposal

Domain Background

Every year, in mid-June, the lengthy process of scouting, interviewing, and getting to know young basketball players culminates in the NBA Draft, whereupon these athletes are selected by NBA franchises with the hopes they can develop into rotation players and potentially All-Stars. With each draft, there are players selected who will wildly out-perform their draft position while others will fail to live up to the lofty expectations that come with being selected with an early lottery pick. Nabbing an elite player can change the course of these billion-dollar franchises for 10 or 20 years, while failing to capitalize on a rare high-pick opportunity can lead to a decade of mediocrity.

With such high stakes, all 30 teams invest heavily in their scouting and departments to help inform their brain-trust to try and make the best decision at the time. They even work closely with psychologists to try and determine how players will develop! Nevertheless, the history of the NBA Draft, even in the modern era, suggests that this science is far from perfect. High-profile "busts" have been selected with the top pick (recently, Anthony Bennett), while the last pick of the draft has produced an All-Star (5-foot-9 Isaiah Thomas).

To assist with this process, various player projection algorithms have cropped up recently. Several exist publicly (catalogued at tothemean.com), although I am sure many more are only available internally to teams. Basketball is a passion of mine and I'm particularly interested in prospect projections, so I'd like to throw my hat into the ring while providing a few new wrinkles.

Problem Statement

The problem to be solved is to improve prospect projections so there can be a reasonable estimate of a player's future performance. The overall target variable to evaluate a player will be 'Wins Above Replacement' (WAR), a statistic which, according to Basketball Reference, "translates a player's point differential approximately into wins." (It is calculated simply by multiplying a player's VORP, a statistic maintained by Basketball Reference, by 2.7.) In a given season, an All-Star can have a WAR above 15.0, and it's possible for some players to be below the "replacement level" of 0.0.

I will try to incorporate ordinary box-score stats, player measurables, combine statistics, as well as play-by-play data to assist with the process of creating a regression model.

Datasets and Inputs

The dataset for this project is be a custom collection of stats obtained from the following sources via web-scraping: Basketball Reference and RealGM for historical stats, DraftExpress' Pre-Draft Measurement Database, as well as Hoop-Math.com and NBAMiner.com for play-by-play data.

Basketball Reference has the strongest NBA database, so I will use it to collect historical statistics for all NBA players, including WAR. RealGM has the superior international and college database, collecting per-minute and advanced statistics since 2003. By making use of their entire database, these per-minute statistics can actually be converted to more telling per-possession statistics. This database will also allow for standardization of international stats (see below) and provide the primary inputs on my final model. DraftExpress' database has physical and athletic testing on roughly 75% of players dating from around 2003 as well.

NBAMiner.com has more in-depth play-by-play data since 1997 which can be used to generate scores for individual skills. Hoop-math has similar, albeit less thorough, collegiate play-by-play data dating from 2012.

The main dataset will be limited to players who were playing in non-High School leagues outside the NBA at some point since 2003, who later played in the NBA. There are 1443 players with about 3500 seasons between them. I would estimate that there are about 50-70 features of which to make use.

Following collection of data, the stats for players around the world will have to be standardized since the model will include both college and international prospects. The different quality of leagues around the world means that it's more impressive to put up stats in one league than it is in another. As Layne Vashro details at Nylon Calculus, by comparing stats for players who played in multiple leagues within a certain span of time, we can measure the level of competition in all of the various leagues worldwide and standardize the statistics accordingly.

Solution Statement

The solution to prospect projections is to come up with a machine learning algorithm that puts all this data to use! This will produce estimates of future WAR performance and provide rankings for NBA drafts based on pre-NBA data for every year. If projections are accurate, then the model can be applied to the upcoming NBA draft.

Benchmark Model

Fortunately, the website tothemean.com offers a valuable comparison of various draft models that can put the results of my model in context. The experimental accuracy of the various models over time are displayed on the website, and one can cycle through different models which best predict different "all-in-one" stats (Win Shares is used by default).

The benchmark model will be using the given Win Share values to compute the averaged Draft Rank Value (see the next section for details) for the 'actual draft' selections, which I calculated to be 46.88 for the years 2003-2015. If my model can outperform NBA General Managers or at least come close to it, I know it will have some value.

Evaluation Metrics

Evaluation Metric #1: For evaluating projections of individual player-seasons as well as value over a period of years, I chose root mean-squared error. I chose this over the coefficient of determination because it is more punitive toward misses for high-ranking players (in either projection or actual), which is desirable given the stakes are higher near the top of draft.

Evaluation Metric #2: For comparing my model to the benchmark model, I chose Draft Rank Value. Jesse Fischer's Draft Rank Value metric contains the value $\text{RMSE}_{\text{Ranking}}$. This works as follows: 1) For an NBA Draft model with ranked players, display the name and rank next to the player's actual total Win Shares. 2) Take the Win Shares list and organize it top to bottom and put it alongside the actual Win Shares. This is the expected Win Shares for each draft slot. 3) Take the root mean squared error of the difference: abbreviated $\text{RMSE}_{\text{Ranking}}$.

There is one other value used in the formula. According to Fischer, "$\text{RMSE}_{\text{Worst}}$ is defined as the RMSE value for the worst possible ranking for a given year (optimal ranking reversed)."

Draft Rank Value (DRV) is then calculated by:

$$\textbf{DRV} = 100*(1-(\text{RMSE}_{\text{Ranking}}/\text{RMSE}_{\text{Worst}}))$$

Fischer suggests that this model works because it punishes for both overvaluing and undervaluing prospects, while more severely punishing mistakes at the top of the draft (where the stakes are higher).

Project Design

Feature Selection/Scaling

We can add additional features to those in the original dataset. A single statistic over the course of a college season (~30 games) might not be as helpful as an ensemble of related statistics. For example, a player's shooting ability might not be captured solely by their 3-point percentage, which tends to be highly variable season-to-season. We should also take into account their 3-point attempts, long-2 %, and their free throw % as well to generate overall shooting scores. The weights of each statistic can be determined by training a regressor onto a target. (My early investigations suggest that a linear regression provides superior performance.)

What will the target of this regression be? Well, there is no single definitive shooting statistic, so I will use the robust NBA shooting data on NBAMiner.com to create overall shooting scores in the NBA. I can adjust the coefficients of the formula until they pass a quick sanity test. (In this case, are the great historical shooting seasons at the top of the list?) I can then train pre-NBA shooting statistics on a player's rookie-year outside shooting score to generate coefficients for each feature in a linear model. Since college play-by-play data is only maintained since 2012, I will have to create two different linear regressions (one on the whole dataset and one with college players since 2012) to generate shooting scores, with the post-2012 regression taking precedence.

The athleticism attribute will be an exception to score creation. This would be poorly captured by rookie-year statistics. Instead, I will have a friend assign athleticism scores, A, B, C, D or F, to each player under the guideline of producing a roughly half-normal distribution (e.g. mostly Fs, few As). I can then convert these to numbers and train regressors on these inputs. I suspect the relationship between the features will be more complex than outside shooting scores, so tree methods will likely work best. Because some of the features for a player will be identical year-to-year (height, for example), using a grouped cross-validation technique will be necessary to reduce overfitting.

While ensembles of trees with cross-validation are generally resistant to overfitting, the method of generating scores means that pre-NBA athleticism predictions will be made on the trained ensemble, meaning that the prediction will be influenced by the target score. To my mind, this is actually desirable since combine data is incomplete, and the generated feature scores will be more accurate.

After generating scores for outside shooting, inside shooting, defense, playmaking, and athleticism, we will want to transform the data so it has normal or half-normal distribution, and then employ a preprocessing scaler to standardize the data prior to testing a variety of regressors on how well they predict a player's WAR.

Training Regressors

The dataset only stretches back to 2003. This might seem like a long time, but it means that players drafted since ~2010 or later probably haven't reached their peak. To solve this, I could either A) incorporate more (spotty) historical data, or B) impute data based off predictions for young players. Option A would limit the value of my features while option B creates some projected targets in lieu of real ones. I decided that B would preserve the overall method of the model and score creation.

One of the main benefits of creating these scores is that it allows for the use of fewer features, which reduces the effect of model bias. I can use the handful of scores along with physical profile and age to fit a cross-validated algorithm onto the data. So, a two-step process occurs:

projecting current NBA players' futures and then fitting a model based on the actual performance or projection. Fortunately the same general process can be applied to both.

It occurs to me that there are several competing ideas when it comes to prospect projections:

A) Players who have a good statistics will tend to be successful.
B) Having a crucial weakness can be devastating while having a crucial strength can be a huge advantage.
C) Players who have similar features will have similar career trajectories.

Idea A intuitively makes sense and implies employing a regression model to predict future WAR. Here we are more interested in overall fit, so we can use the (adjusted) coefficient of determination as a guide to develop a linear regression formula. However, this method is too simple on its own: the regression model(s) will likely overrate many "types" of players will underrating others.

In examining the patterns of the regression results, we might encounter scenarios where a player's skill (or lack thereof) in a particular area takes on a increased level of importance for their projection. Players are in many ways primarily defined by their strengths or their weaknesses. If a player projects as a historically great shooter or defender, that is tremendously important because these types of players are some of the most valuable players in the NBA. Similarly, a player's lack of ability to shoot or defend can keep them off the floor entirely. Idea B then implies a type of Decision Tree Learning. Extremely randomized trees, gradient boosted trees, and random forests tend to produce good results and tend to be resistant to overfitting.

Idea C is the basis of Nate Silver's CARMELO and Kevin Pelton's SCHOENE, two well-known projection models. A Nearest Neighbors algorithm would compare the features of all players and take a weighted average of the k-most similar ones. This would work well in the cases where there are close matches and less so where a player is more unique.

Blending an Ensemble of Regressors

Ultimately, all these ideas might best be used in conjunction. Each of them would appear to be insufficient on their own, but together they could produce more accurate results. One way of accomplishing this is the method of stacking, where the prediction generated from each algorithm is treated as a feature in a new supervised meta-algorithm. This has an added bonus of providing a check on overfitting.

Another way is feature-weighted linear stacking, which stacks linear functions of meta-features (e.g. the return distance of the nearest neighbors or a player's height) along with results of each model to generate dynamic coefficients in a linear regression model.

Yet another method might be to use A and B to generate a baseline projection and then use C to estimate how a player might perform relative to this expectation. There are surely more possibilities out there, but for both steps the RMSE can be tested on all.

With the data imputed and the final model with the lowest RMSE chosen, we can finally calculate the DRV and see how it compares to the benchmark.

References

Draft Models - http://www.tothemean.com/tools/draft-models/#/2016

Worldwide Competition -

Draft Rankings - http://www.tothemean.com/2015/07/26/how-to-compare-draft-rankings.html

Feature-Weighted Linear Stacking - https://arxiv.org/pdf/0911.0460.pdf

Wins Above Replacement - http://www.basketball-reference.com/about/bpm.html