15TH ANNUAL
MIT SLOAN
SPORTS ANALYTICS CONFERENCE
APRIL 8-9, 2021 | VIRTUAL
presented by ESPN 42 ANALYTICS

Predicting NBA Talent from Enormous Amounts of College Basketball Tracking Data

A.N. Patton, M. Scott, N. Walker, A. Ottenwess, P. Power, A. Cherukumudi & P. Lucey
Stats Perform (Chicago, USA & London, UK)

1. Introduction

In 2010, the SportVU optical player and ball tracking system was first deployed in select NBA arenas by teams wanting to obtain an edge in player and team analysis. Due to the value of the tracking data, the NBA then adopted SportVU league-wide prior to the 2013-14 season. Since then, nearly all analysis and decision-making for NBA teams has been data-driven, utilizing not only the raw positional data, but tactical insights derived from the markings detected automatically by machine learning algorithms (e.g., screens, isolations, drives, etc.).

However, when it comes to analyzing NCAA players for an upcoming draft, NBA teams are severely limited in their decision-making ability as they do not have the same detailed tracking data of NCAA players. In-venue hardware solutions are impractical for the NCAA, with over 300 Division I schools alone in addition to the numerous exhibition/tournament and post-season games not played at NCAA venues. Additionally, for an NBA front office to model a college player's future potential output, they will need historical tracking data of current NBA players to build a training set for modeling - something that in-venue solutions cannot achieve.

To circumvent this issue, we have utilized state of the-art computer vision techniques to capture player and ball tracking data from thousands of historical NCAA D-I Men's basketball games directly from broadcast video (see Figure 1). This volume of data equates to **more than 650,000 possessions and *over 300 million frames of broadcast video.*** From the tracking data, we automatically detect events such as ball-screens, drives, isolations, post-ups, off-ball screens and defensive match-ups using our actor-action attention neural network system achieving recall and precision rates of 0.8 and 0.7 respectively.

Even though the generation of tracking data from broadcast for college basketball is in itself a massive breakthrough in the field of basketball analytics – it is not enough. To showcase the value of the generated data, it is best to gauge the value through a predictive task. In this paper, we focus on the task of predicting the talent of future NBA players. ***We do this by predicting the probability of a player making the NBA directly from college data***. We show using tracking data enables us to obtain more accurate forecasts compared to current data sources (tracking log-loss: 0.30 vs play-by-play log-loss: 0.40). The additional benefit of our approach is that we apply "interpretable machine learning" techniques (i.e., Shapley values) to not only create accurate predictions but also identify the strengths and weaknesses of a specific player.

The rest of the paper is as follows: in Section 2 we describe the steps we took to generate tracking and event data from broadcast video; in Section 3 highlight the contextual features such as defensive match-ups and coverage types as well as creating rich features to describe each player; and then in Section 4 we describe how we can predict NBA talent from college data.

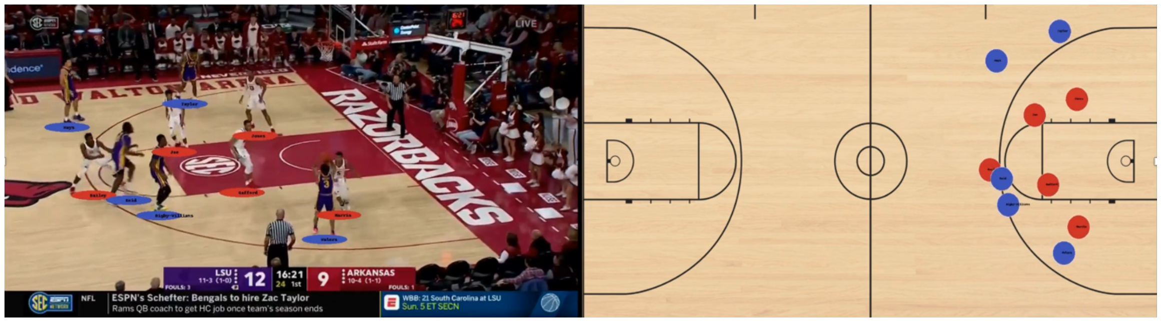


Figure 1: *Example of broadcast footage frame and corresponding player identification tracking data.*

2. Creating Tracking Data from College Broadcast Video

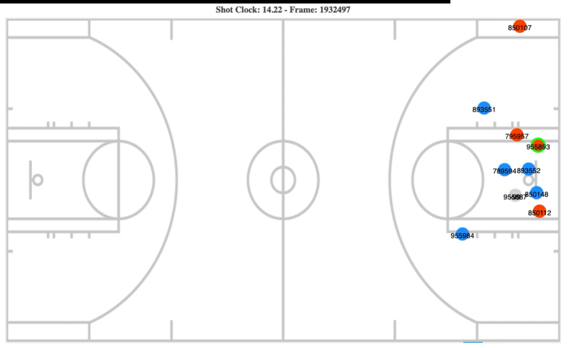
To get the tracking data into a usable form, there are two key steps that are required: i) Mapping pixels to dots (e.g., Figure 1), and ii) Transforming the dots to a semantically meaningful event layer which can be used to describe player attributes but also as an input into our NBA talent prediction model.

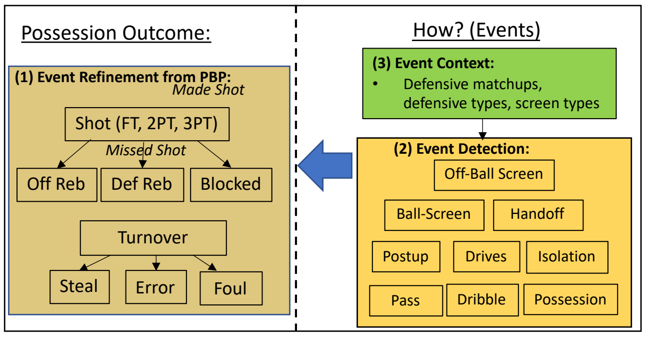
The steps involved are:

1. Ingest broadcast video
2. Categorize frames into trackable/non-trackable clips
3. Calibrate moving camera (i.e., loc)
4. Detect players using skeleton tracking
5. Track and re-identify players over time
6. Ball detection and tracking
7. Optical character recognition
8. Event detection
9. Merge tracking data with play-by-play data

These steps are similar to those described in [1]. However, it is worth emphasizing the key difference between an in-venue solution like SportVU and a broadcast-based tracking solution are down to two key breakthroughs: a) calibration of a moving camera (step 3), and b) body-pose detection (step 4) which enables player re-identification (step 5). For the sake of brevity and to retain the focus on the core contribution of the paper of predicting NBA talent from college tracking data, we point the reader to the following two papers which describe our calibration method [2], as well as our method used to estimate body-pose location which utilizes the Open-Pose technology [3].

A snapshot of the generated player and ball positions is given in Figure 2a. As mentioned previously, we employ this data capture method to generate tracking data across over 650,000 college basketball possessions, which is over 300 million broadcast frames. However, generating the raw tracking data is not enough. To provide both descriptive analysis, as well as a useful feature representation for our prediction task we next have to map the tracking data to a semantic layer (i.e., events). The steps involved of transforming the tracking data to the semantic layer are depicted in Figure 2b.





(a) (b)

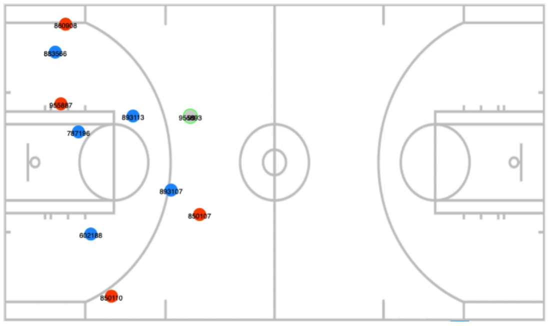
Figure 2: *(a) Single frame representation of the generated player and ball tracking data (blue is offensive team, red is defensive team and gray is the ball (b) From the tracking data, we then merge/align to the play-by-play (PBP) feed to first segment data into possessions. Once segmented, we then conduct the following three steps: (1) We use the PBP data to refine the end of possession event positions (i.e., shot/rebound location), (2) Detect events automatically from the tracking data, and (3) Enhance the events with defensive contextual information.*

Even though the goal of computer vision technology is to capture all data directly from the video stream, the referee is the ultimate decision maker in the successful outcome of an event. For example, in basketball whether a basket was a 2pt shot or 3pt shot (or is valid – a travel did occur or a defensive/offensive foul) is determined by the referee. As such, to capture these data points, we have to rely on humans to manually annotate these points as they have to wait for the referee's ultimate adjudication, and this is documented via the play-by-play (PBP) feed.

As such, a necessary step to automatically detecting events is to first merge/align the play-by-play with the raw tracking stream which also contains the game and shot clock. To do this, we use a fuzzy matching algorithm which combines PBP, OCR and player/ball positions to get the aligned data-source. Once segmented, we then conduct the following three steps (see Figure 2(b)): (1) We use the PBP data to refine the positions and precise frame of the end of possession events (i.e., shot/rebound location), (2) Detect events automatically from the tracking data, and (3) Enhance the events with defensive contextual information.

For the automatic event detection, we first use an actor-action attention neural network system to detect/refine the basic events (i.e., shots, rebounds, passes, dribbles and possessions) in a sequential manner. We then build a host of specialist event detectors for the higher-level events such as postups, drives, isolations, ball-screens, handoffs and off-ball-screens, achieving recall and precision rates of 0.8 and 0.7 respectively, using a neural network approach. Again, for the sake of brevity and to retain focus on the key contribution on the predictive value of the college tracking data, we will have a full paper describing our event detection process in a subsequent paper. An example of the player and ball tracking data, with the event detection output is shown in Figure 3.

In the next section we will describe how we both create contextual information around these events, as the feature representation creation process for our NBA talent prediction task.



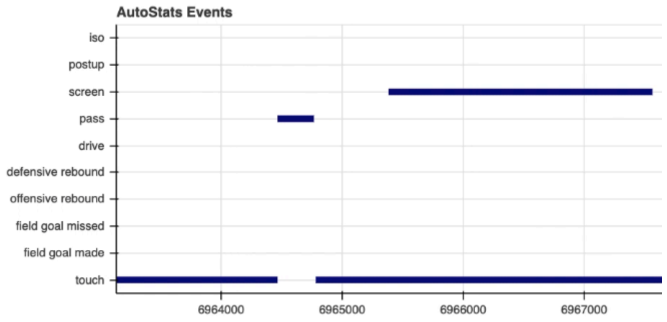


***Figure 3:** (Left) Single frame representation of the generated player and ball tracking data (blue is offensive team, red is defensive team and gray is the ball) (Right) Associated event detection timeline for the frame from (a).*

3. Feature Representation and Enrichment

Mapping tracking data to events as highlighted in the previous section, enables a player representation to be captured, however it still does not contain all the necessary contextual information to build out the best possible player representation (especially for the task of predicting NBA talent). To do this, we need to add in more contextual information such as defensive matchup information (i.e., who is guarding who at each frame), as well as other defensive information such as coverages for ball-screens.

To measure defense, we utilized a measure which we call our "influence score" which captures the influence a defender has on each offensive player on a scale of 0-100. The value for the score was based on common basketball defensive principles such as proximity to player, distance from the basket, passing lanes, and lanes to the basket. To assign such a score at every frame, we utilized a supervised learning approach, where we had basketball experts initially label each defensive match-up in a frame across thousands of frames. These examples formed our initial classifier (which took the form of a multi-layer perceptron (MLP)) which we ran across a multitude of historical games.

As our system collected more tracking data, we then ran our initial influence score classifier across more games to tweak the parameters of the classifier until we were happy with the output. The model produces influence scores for all 25 offensive/defensive matchups at every frame. The results are used to determine defensive matchups, ball screen/off ball screen player roles, ball screen/off ball screen coverages, contested shots, and other features. An example frame of broadcast footage converted to tracking data with matchups derived from the influence scores is shown in Figure 4.

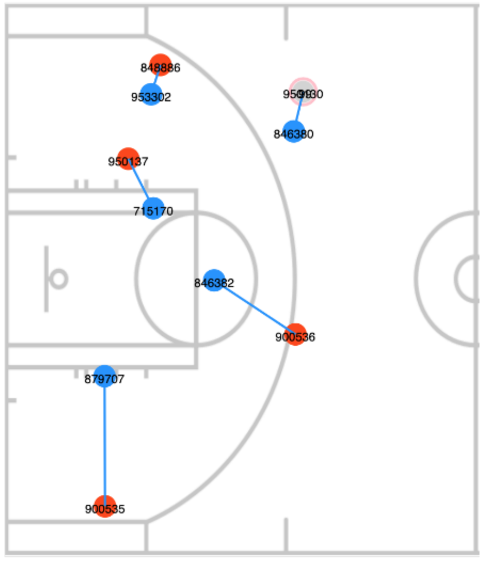
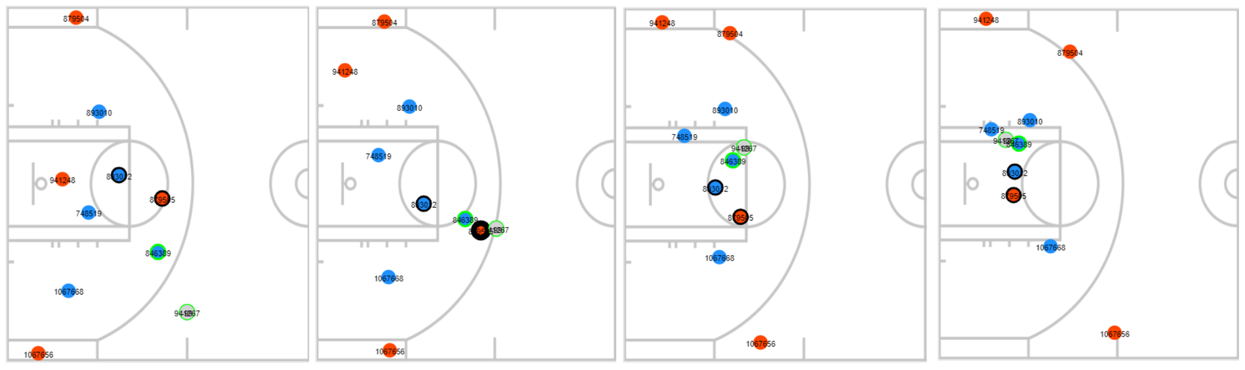


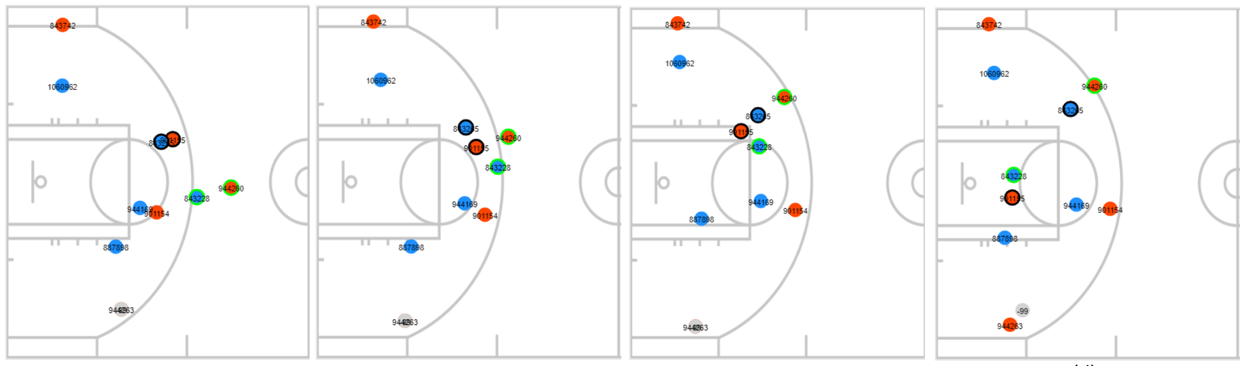
Figure 4: Example showing the defensive matchup using our influence score.

In addition to assigning frame-level defensive matchups using our influence score approach, using the influence score we were also able to assign defender roles for the ball-handler and screener for on-ball screens and the cutter and screener for off-ball screens. Examples are shown in Figure 5 and 6 respectively.



(a) Start of Ball Screen (b) When Ball-Handler/Screener are closest (c) End of Screen (d) End of Ball-Handler Touch

Figure 5: *Shows the various stages of a ball-screen, and the ball-handler, screener, ball-handler defender and screener defender role assignments. Grey: ball/ball handler, red: offense, blue: defense, green border: ball handler/ball handler defender, black border: screener/screener defender. Game is Florida State at Virginia Tech, January 20th, 2018.*



(a) Start of Off-Ball Screen (b) When Cutter/Screener are closest (c) End of Off-Ball Screen (d) 1.5 secs after Off-Ball Screen

Figure 6: *Shows the various stages of an off-ball-screen, and the cutter, screener, cutter defender and screener defender role assignments. Grey: ball, red: offense, blue: defense, green border: cutter/cutter defender, black border: screener/screener defender. Game is Kansas at TCU, January 6th, 2018.*

One of the profound challenges of modeling using potentially only 20-30 games of NCAA data per player is the high variance of low frequency events seen in tracking data. For example, a highly talented one and done player might only attempt 50 isolation shots in a career; this is not enough samples to get a robust mean value for their isolation shooting efficiency. Therefore, in order to improve model performance, we create new player representations by using mean-regression to reduce random noise in the features. Our main mode of doing this is by using what is sometimes known as the "padding" method (Eq. 1), or a simplified version of a beta prior. The method uses a weighted average between the observed values and sample mean. We solve for the optimal weighting constant C which best predicts the next game of a player's career. Since we can apply this to any game-level statistics, every feature in both box-score and AutoStats data was padded, excluding certain player level statistics such as height, weight, and minutes/possessions played.

Eq 1. *Padding Equations Example for 3P%*

$$3P\%_{pred} = (3P\%_{season} * W) + 3P\%_{league\ avg} * (1 - W)$$

$$W = 3PA_{season}/(3PA_{season} + C)$$

Using an NCAA career weighting system developed by Kevin Pelton, we then weight each season of a player's career and combine into one representation of a player [4]. Therefore, we have two datasets at this point, a raw dataset and a padded dataset, both weighted on a player career level in the same manner.

4. Using Tracking Data

4.1. Will You Make the NBA?

One of the challenges associated with draft modeling is the determination of the population of 'NBA possible' players within a given year. Training a model using hundreds of games of low DI or DIII information is not likely to provide useful modeling results when in reality the players covered by those games have nearly impossible odds to make the NBA. Therefore, our first modeling exercise was to predict the chance that player would make the NBA, defined as a) getting drafted or b) receiving any NBA minutes in the first three years of their career. However, we also wanted to demonstrate the value of AutoStats data compared to traditional sources, and built both an AutoStats and a box score only model to compare against each other.

4.1.1. Feature Space

One of the key discoveries of this model was that instead of using simply the padded data, we should create models using the raw data and the padded data and then ensemble the results (these ensembles outperformed the separate constituents across the board). For both datasets, the same process was used to prepare the data for modeling**.** With the high dimensionality and relative similarity between many of the features, we iteratively halved pairs of features that were highly collinear, starting with the most highly correlated. Whichever of each pair was more correlated with remaining features was removed, until no two features had an R^2 of 0.95 or higher.

Given that the colinear features for the raw and padded datasets were different, Table 1 shows the number of features and size of the training and test sets for the two models. The data was split using a random 80% for training and 20% for testing.

Table 1: *Feature space and training split size for raw and padded dataset 'make NBA' models*

Data Type	Total Features	Retained Features	Training Size	Testing Size
Raw	104	101	2,183	485
Padded	109	50	2,183	485

4.1.2. Model Architecture

4.1.2.1. Raw and Padded

Both the raw and padded models used a LightGBM classifier, an open source gradient boosted decision tree developed by Microsoft [5]. The model's hyperparameters were tuned using five-fold cross validation on a random search across a parameter grid [6, 7]. By using a classifier, each model's predictions are a probability of the player making the NBA per our previous definition. The box score only models were also built with this architecture, although with a substantially reduced feature space due to the limited amount of information present in box score data.

4.1.2.2. Ensemble

The ensembling of both the raw and padded models works to include predictive information contained separately in both datasets. The feature space for the ensemble, a random forest classifier, was the raw prediction, the padded prediction, and chances per game, an AutoStats derived feature that is analogous to possessions per game. Again, as a classifier, the results from the ensemble were a probability of making the NBA.

4.1.3. Results

Due to the imbalanced nature of the dataset (few players make the NBA in total), logloss was used as the metric of choice as opposed to a more basic strict accuracy. The AutoStats model averaged approximately 10-20% less error than the box score model depending on the test split, roughly 0.30 and 0.40 respectively. However, while the model must be accurate to be useful, the pure predictions from the ensemble were not the only product of the model.

4.1.3.1. Shapley Values

In order to properly understand why the raw and padded models made their predictions, we used Shapley Values, a game theory approach to interpret results of machine learning models [8]. The Shapley values provide on a per-prediction basis the direction and magnitude of each feature's contribution to the overall prediction. By combining the Shapley values for each of the raw and padded models we can use the result to understand the interplay between the two types of data, and the differing information they provide. For example, the combination Shapley plot for James Wiseman is presented in Figure 7. Only the ten largest magnitude features for each model are shown. Wiseman is a particularly interesting case because he only played a total of three games (69 minutes) in his college career. Looking at the raw data model (orange), features such as PTS/Poss, and BLK/Poss show very strongly as positive indicators of making the NBA, but without their regressed versions (green) which would show up as a stacked bar. Unsurprisingly, the padded data has regressed a three-game sample very heavily and reduced the 'quality' of his raw scoring and block output. Non-regressed features such as Rim Gravity, and Midrange Gravity (both metrics of spatially weighted offensive efficiency and usage) show strongly positive in both the raw and padded datasets. He is also a good example of not blindly adhering to model output. The model does not "know" why he only played three games, but when the padded and strongly regressed data are ensembled, the prediction is a lower probability of making the NBA compared to what would be expected based on known contextual information about his career.

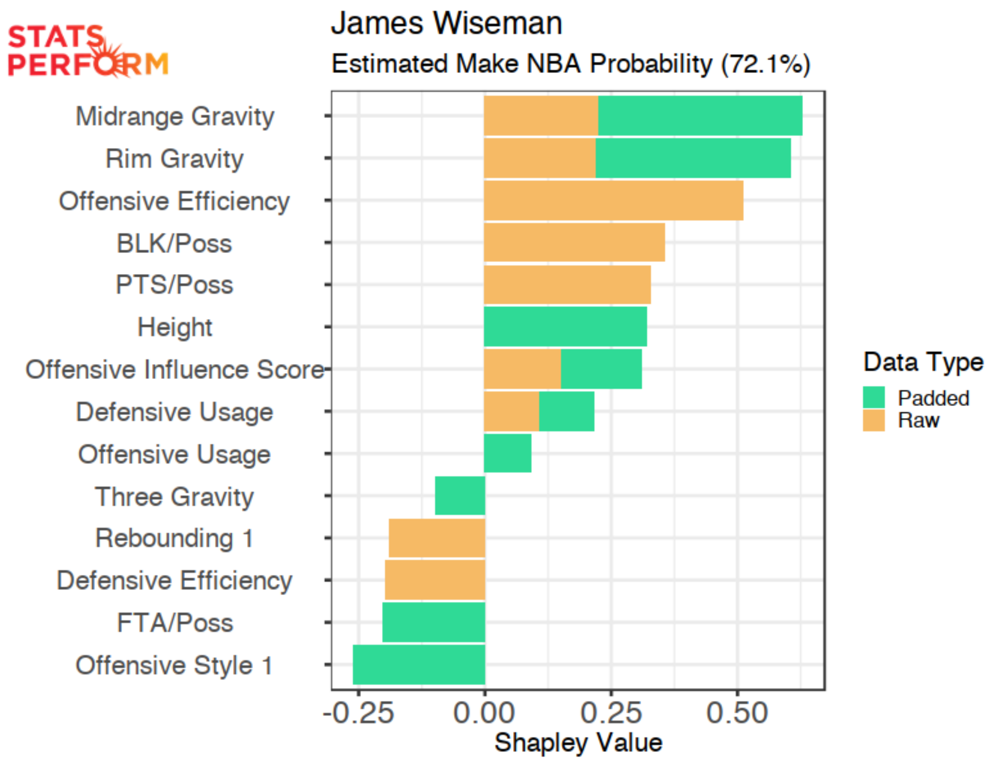


Figure 7: *Combination Shapley values for James Wiseman using raw data and padded data showing only a selection of large magnitude features*

These charts cannot be use in a manner analogous to a linear regression. "If he increased _____ then he would have a higher percentage of making the NBA" is not a valid use case for the Shapley values, and in general we propose using these charts as a high-level overview, not a standalone analytical product. Additionally, these can also be rolled up into a chart that groups features into categories, allowing for an easier visual inspection as seen in Figure 8 for Anthony Edwards. It is important to note again that these are not outputs (except the make NBA probability) from the final ensemble, but are the two primary sub-models of the ensemble.

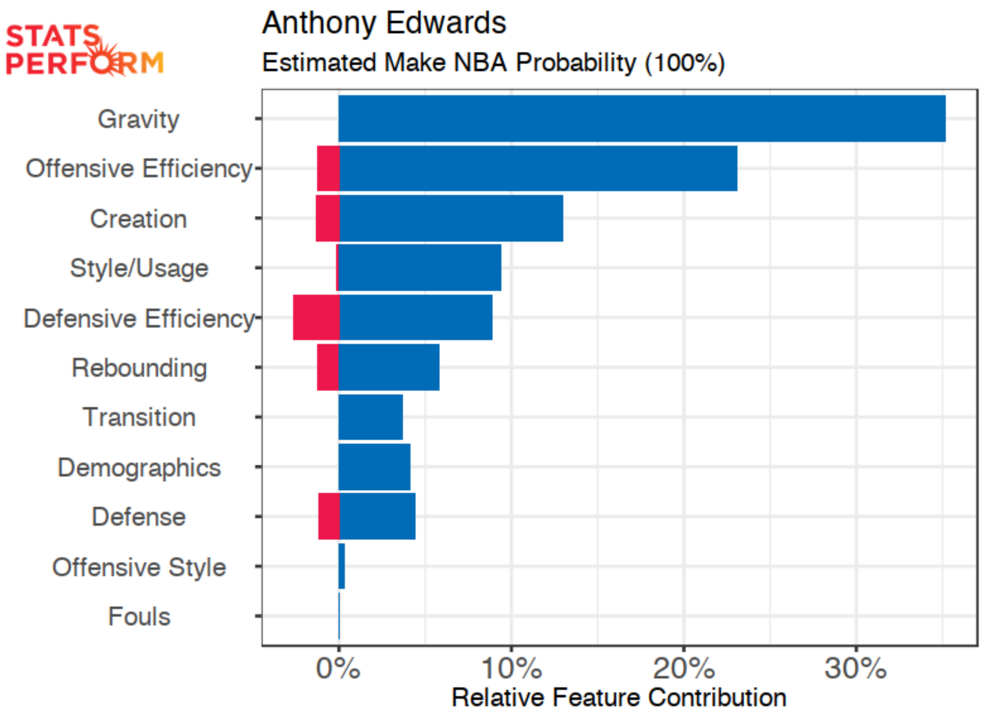


Figure 8: *Combination Shapley values for Anthony Edwards using raw data and padded data with features rolled up into pre-defined categories*

While these graphics are not a particularly detailed tool for quantitative investigation, they can be used to understand how the models function as well as assist with model interpretation for non-technical stakeholders. With a reasonable estimate for a player's probability to make the NBA, we can now use the probabilities to trim the dataset to plausible NBA players and begin the actual draft modeling.

4.2. What Kind of Draft Pick Are You?

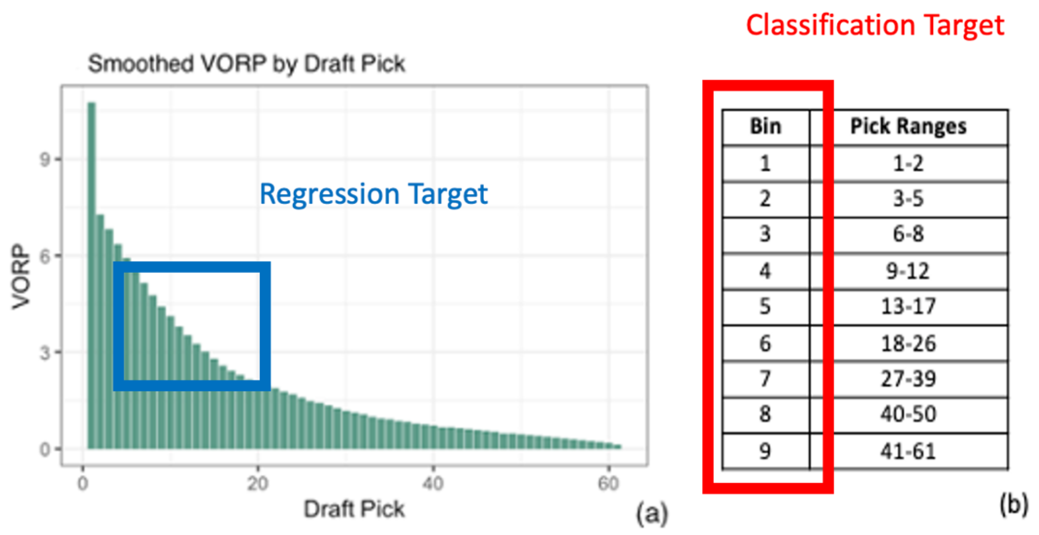
In lieu of immediately building a model that attempts to directly predict NBA RAPM or NBA SPM from college data, we elected to build an intermediary model that characterizes a player's talent profile in comparison to previous draft picks by attempting to predict a player's actual draft pick from 1-61 (61 being undrafted). One of the primary challenges with a direct quality model is properly accounting for the context of the drafted player's NBA situation. Therefore, our talent profile model explicitly does not take NBA context into account and serves as an NCAA scouting model for talent evaluation.

4.2.1. Feature Space

The predicted probabilities to make the NBA from the prior model were used to trim the number of players used in the models from all players (n = 2,911) to any player with greater than 40% chance to make the NBA (n = 393). The inclusion threshold was chosen somewhat casually based on a review of the prediction results and could be tuned more rigorously in future implementations. Similar to the make NBA model, a variety of model outputs will be ensembled together to create the final predictions.

4.2.2. Target

When predicting the draft pick for a given player, there is an inherent non-linearity in draft pick value that must be accounted for. Figure 9 shows an approximate curve of value over replacement (VORP) for picks 1-60 that highlights the enormous difference between value for #1 vs. #5 whereas #40 vs. #45 has minimal difference, despite both ranges being five picks.



Bin	Pick Ranges
1	1-2
2	3-5
3	6-8
4	9-12
5	13-17
6	18-26
7	27-39
8	40-50
9	41-61

Figure 9: *(a) Smoothed VORP by draft pick with pick 61 corresponding to undrafted (b) corresponding Jenks natural breaks classifier bins and pick ranges*

Therefore, to ensure that the VORP curve is modeled correctly, we binned the VORP values using a Jenks natural breaks classifier with nine bins. The bins can now be used as model classification targets that take into account the value variation across classes.

4.2.3. Model Architecture

The talent bin ensemble model was composed of results from six constituent models, three based on the make NBA model, and three new models specifically for this application. The six constituents were then ensembled using a LightGBM classifier with the bins from Figure 3b as the target.

4.2.3.1. Existing Components

The raw dataset make NBA prediction, padded make NBA prediction, and the ensembled make NBA prediction in probability form were used as three of the six constituents for the ensembled talent bin model.

4.2.3.2. New Components

The new components for the talent bin ensemble model reused the framework where both the decorrelated raw and decorrelated padded data will be used in separate models and then ensembled to create three sets of predictions that will be carried forwards. Each of the raw and padded models were random forest regressors using the VORP pick value at each draft pick as target. The predictions from these were then ensembled with additional information from the make NBA models using NGBoost to create regression predictions with independently modeled means and variances (Duan et al. 2019). Lastly, the outputs from all existing and new components were ensembled using a random forest multiclass classifier (bins 1-9).

4.2.4. Results

An example of the graphical output for Aaron Nesmith (created pre-draft) is provided in Figure 10. The model gave him an approximately 62% chance of having the statistical profile of a player picked in the 18-26 range historically. As this does not include any NBA or pre-draft rankings, it is explicitly not predicting where a player will be taken, only what range of player they are like.

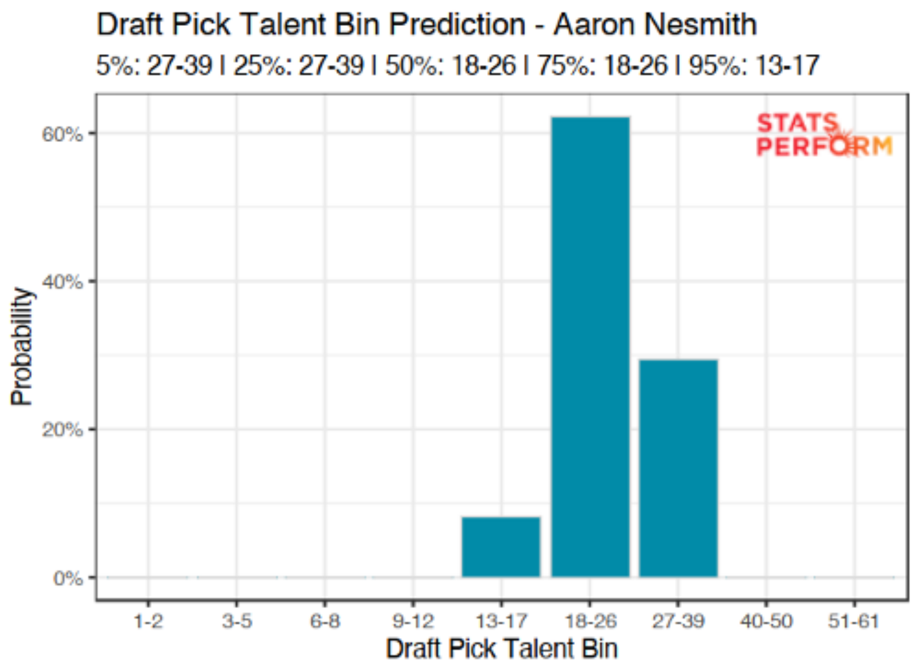


Figure 10: *Talent bin predictions for Aaron Nesmith*

The model captures a fair amount of uncertainty in some players like Shabazz Muhammad (Figure 11) and is extremely confident in others such as Jayson Tatum (Figure 12). And while this model does not actually attempt to answer the "how good will Player X be" question, there is some semblance of a quality gradient under the assumption that early picks are usually better NBA players than later picks.

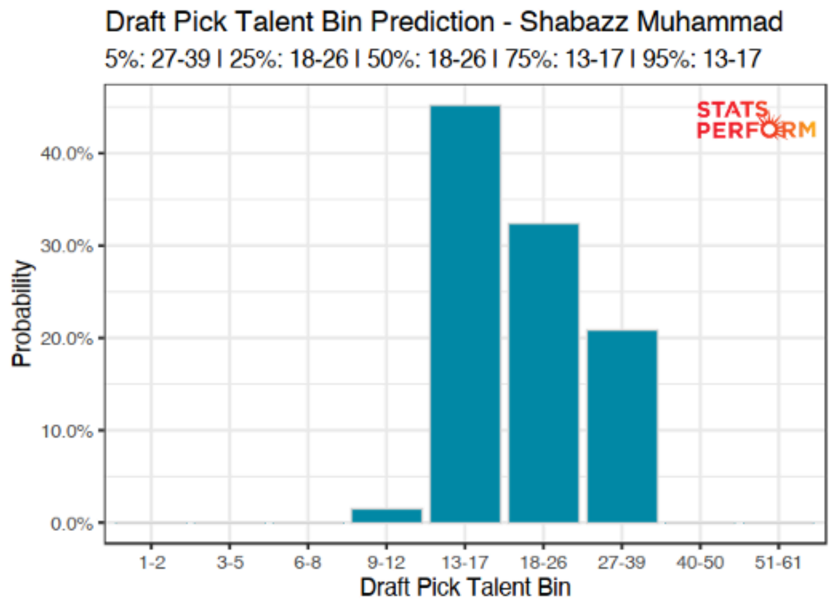


Figure 11: *Talent bin predictions for Shabazz Muhammad*

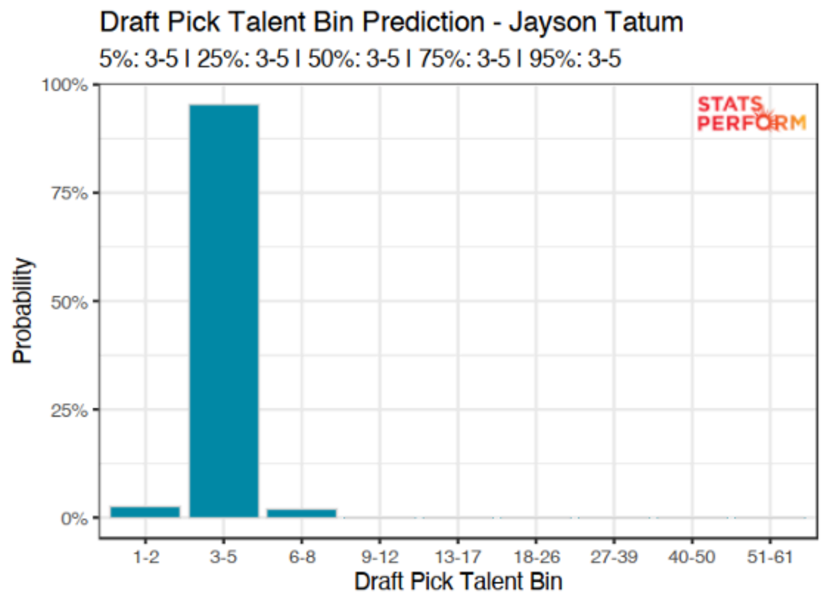


Figure 12: *Talent bin predictions for Jayson Tatum*

We believe that one of the primary use cases for this model is draft decision making. If a team knows the players they want, they can trade up or down to enter a pick range that player is representative of, or they can perform additional scouting and due diligence if the player they target with their first-round pick profiles far below expectations.

5. Conclusion

The process of creating NCAA to NBA projections from AutoStats data is predominantly broken into three parts. The first is the acquisition of the raw tracking data based on coordinates of players and the ball as well as timestamps for all frames. The second is the enrichment of the data to create features that are meaningful for analysis such as number of transition chances, influence scores, points per post up, etc. These intermediate features (based on raw coordinates) are limited in scope only by the use case for the data. There are certainly potentially useful metrics that have not yet been created, and there are many currently under development. The third step is the actual modeling itself. We provided two example models that were developed and found to have a high degree of utility for characterizing a player into the NBA Draft. Models that are currently in progress include projecting individual component skills (TS%, Contested Rebound Rate, Block %, etc.) based on padded and filtered NBA skill targets, identification of NCAA player roles and corresponding prediction into NBA player roles, and many others. The universe of possible predictive tasks with this data is essentially limited by creativity and technical ability.

References

[1] Johnson N. 2020. Extracting Player Tracking Data from Video Using Non- Stationary Cameras and a Combination of Computer Vision Techniques. MIT Sloan Sport Anal Conf.

[2] Sha L, Hobbs J, Felsen P, Wei X, Lucey P, Ganguly S. 2020. End-to-end camera calibration for broadcast videos. Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit 13624–13633; doi:10.1109/CVPR42600.2020.01364.

[3] Cao Z, Hidalgo G, Simon T, Wei S-E, Sheikh Y. 2018. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. arXiv 2017-Janua: 1302–1310.

[4] Pelton K. 2020. How my NBA draft projections work. ESPN. Available: https://www.espn.com/nba/insider/story/_/id/16235135/explaining-kevin-pelton-nba-draft-projection-system [accessed 9 December 2020].

[5] Guolin K, Meng Q, Finley T, Wang T, Chen W, Ma W, et al. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. 31st Conf Neural Inf Process Syst.

[6] Bergstra J, Bengio Y. 2012. Random Search for Hyper-Parameter Optimization. J Mach Learn Res 13: 281–305.

[7] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. 2011. Scikit-learn: Machine Learning in Python. J Mach Learn Res 12: 2825–2830.

[8] Duan T, Avati A, Ding DY, Thai KK, Basu S, Ng AY, et al. 2019. NGBoost: Natural Gradient Boosting for Probabilistic Prediction.

[9] Lundberg S, Lee S. 2017. A Unified Approach to Interpreting Model Predictions. In: *Advances in Neural Information Processing Systems 30* (I. Guyon, U. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, et al., eds). Curran Associates, Inc. 4765--4774.