Predicting the NBA Draft Selection Using College Basketball Player Performance Statistics

Jack Luft
University of Victoria
jack@luft.ca

Otto Willborn
University of Victoria
ottowillborn@uvic.ca

Truman Gao
University of Victoria
trumangao@uvic.ca

Baz Cox
University of Victoria
basilcox847@gmail.com

Chet Avdic
University of Victoria
chetavdic@uvic.ca

Kale Kasdorf
University of Victoria
kale.kasdorf@gmail.com

Project Github Repository: https://github.com/ottowillborn/SENG474Project

Abstract

This paper explores the use of machine learning models to predict NBA draft pick positions based on pre-draft player performance statistics. Unlike prior studies that use draft position to predict future player success, our approach inverts the problem by learning how teams themselves may rank prospects given historical drafting patterns. We collected and preprocessed player statistics from RealGM [3], including compound metrics such as PER, ORtg, and TS%. We trained multiple models including decision trees, linear regression, neural networks, and ensemble methods. Our results show that linear regression achieved the lowest average pick error, while random forest and neural networks provided competitive performance with greater modeling flexibility. This work provides insight into how statistical profiles relate to draft outcomes and highlights the limitations of excluding contextual team-level factors from predictive models.

1. Introduction

The NBA Draft process is relatively simple and is similar to the other three major American sports league drafts. Every year, players in the NCAA, Euroleague, G-league, and others officially declare for the draft, given they are considered eligible. On draft night, 60 of these players are selected by various NBA teams. The order of the teams for draft picks is loosely determined by the team's performance.

An important consideration to make is that the NBA draft is not an ordering of the best eligible players. It is an assignment of eligible players to a predetermined ordering of teams, and these teams consider much more than player statistics. NBA teams do not draft solely based on statistics, but other factors as well such as position or player type needed, player attitudes, ownership sentiment, and even potential ticket sales. Our models are not able to learn from this contextual error and we expected it to add noise in our training data. Despite this assumption, there is much debate on whether NBA teams select on a best-player-available or a best-fit-available basis, as well as where in the draft these paradigms shift. Given that our models are inherently restricted to best-player-available, we expected our results to follow the consensus opinion that the start of the draft is best-player-available, but as better players are selected, teams move to selecting better-fit-available.

Each draft class that our models are trained on consists of on average 113 players. While only 60

are selected, we realized that training our model on the excess undrafted players would avoid selection bias. Without doing so, our models would miss information on what makes a player undrafted and we would miss added results and insight.

Our overall project goal is to investigate multiple machine learning models in order to rank declared NBA prospects based on draft patterns from previous years. We scraped and cleaned NBA prospect data from multiple leagues, trained and compared various models, and were able to gain real information with takeaways on how exactly NBA teams draft.

2. Related Work

Previous studies have primarily focused on using draft pick position to predict future player performance, rather than predicting the draft pick itself. For example, Fredrik Harmén's 2022 thesis [1] investigates how well draft slots predict NBA success, using classification models to group players by win-share tiers, with Random Forest achieving the best performance. Similarly, Nazarii Mamonov's 2023 paper [2] applies machine learning models like XGBoost and logistic regression to predict long-term player outcomes based on college stats and draft position. In contrast, our work reverses this approach by using player performance statistics to predict draft position, aiming to better understand how teams evaluate talent during the draft.

3. Data Collection and Preprocessing

The data for this project came from an accredited NBA/NCAA/International website, RealGM.com [3]. From this site we gathered two main datasets to guide our project. The first was the set of NCAA/International players along with their pre-draft statistics. This included all of the statistical features necessary for our models. The second set was the set of previous draft classes, including the drafted and undrafted players each year, along with their pick number.

Merging these two datasets by player name gave us a full dataset for constructing our models. Each player was stored with their respective statistics, physical features, and actual pick position. The features we chose to focus on were pre-draft career compound statistics, such as player efficiency rating. NCAA players had career averages for their statistics, and we scraped from the advanced stats section of RealGM.com. International players did not, and therefore had to be handled differently; with no career average stats, we took the statistics from the closest year before the player was drafted. This reduced the amount of players with missing features from 12% to 6%. The derivation of each feature can be seen in Table A1 (see Appendix A).

The majority of the selected features are combinations of multiple other features. We chose compound statistics to preemptively counter overfitting as a meaningful compound feature summarizes complex relationships into single interpretable values. Some of the features such as school or position were not numeric values, and were thus encoded using the Sci-kit Learn label encoder. Our target value for the models is the player's pick position. This ranks 1-60 for any given draft year. To avoid any potential selection bias, as well as to include undrafted players into our training data, we considered undrafted players to have a target value of 61.

Some players did not have relevant pre-draft statistics; for example, players drafted straight from high school (e.g: LeBron James), and players heavily scouted without having played in a large-scale league (e.g: Giannis Antetokounmpo). When player statistics were missing, they were estimated instead of being left blank. This was done by calculating the mean value of the feature across all available players. For example, if some players are missing a Turnover Percentage (TOV%), we computed the average TOV% for all players who have data. Then, to make the imputation more accurate, we applied a multiplier to this mean value. The best multiplier was found through cross-validation and the

optimal value was determined to be 0.9. This multiplier made sense as usually worse than average performing players had less coverage, and thus more often had missing values. Finally, the missing value is filled in using the formula:

$$\text{Inserted Value} = 0.9 \times \text{Mean(feature value)}$$

This approach maintains consistency in the dataset while allowing for a degree of adjustment based on empirical tuning, making the imputed values more representative than simply using the raw mean.

3.1 Preliminary Data Analysis

Once the data was collected, we conducted preliminary data analysis to gain more knowledge to make better decisions when tuning our models. Figure 1 shows the feature correlation matrix.

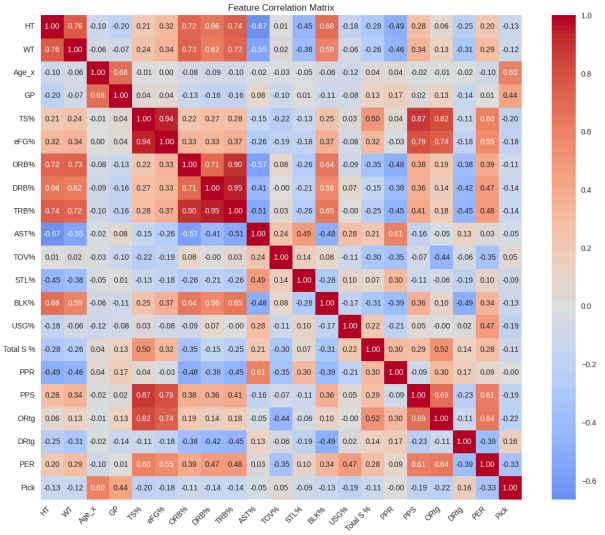

Figure 1: Feature correlation matrix of college player career performance

The correlation analysis reveals that players with higher offensive efficiency, measured by metrics like PER, ORtg, TS%, and PPS, tend to be picked earlier in the draft. High usage rates (USG%) are also associated with earlier selections, suggesting teams value players who contribute significantly on offense. In contrast, older players and those with more games played are more likely to be picked later, indicating a preference for younger prospects with perceived higher potential. Defensive metrics like DRtg and turnover percentage (TOV%) show weaker positive correlations with later picks, suggesting that teams may prioritize offensive output over defensive ability during the draft.

4. Methodology

4.1 Error and Baseline Performance

The error metric across all models is the absolute average pick error, using pick distances. The pick distance refers to the difference between a player's predicted draft pick and their actual draft pick. It is given by:

$$\frac{1}{n}\sum_{i=1}^{n}|\text{actual}_i - \text{predicted}_i|$$

Where n is the number of players in the draft class.

To establish a baseline performance, we compared the 2025 mock drafts from NBAMockDraft.net and Yahoo Sports against the actual 2025 draft results, finding a mean pick distance error of 8.10 and 4.62, respectively. As noted in the introduction, these mock drafts benefit from qualitative insights that our model lacks, such as team needs, player interviews, and organizational sentiment. It's therefore reasonable to expect professional analysts to outperform a purely statistical model.

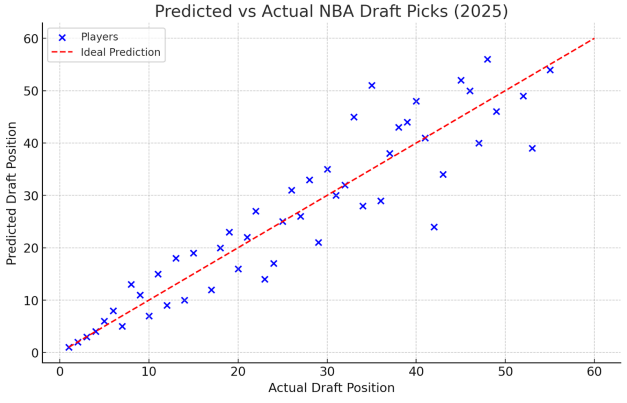

Figure 2: A plot representing the 2025 draft pick error from a mock draft by Yahoo Sports.

It's also important to note that mock drafts made several months before draft day tend to be less accurate, as analysts at that point have limited access to insider information. These early mocks rely more heavily on player performance statistics, similar to the data used in our model. For a fair comparison, we evaluate our model's performance against mock drafts published roughly six months prior to the draft, such as the on previous mentioned by NBAMockDraft.net.

4.2 Train/Test Split

For testing, we chose the years 2010, 2015, 2020 and 2025 as our test set, leaving the rest of the years for our training set. We held these years out during training and validation so that our final reported metrics are based on data that our models had not yet seen. We specifically chose spaced out years throughout our data set for testing to get an accurate understanding on how our model performs. Basketball has changed throughout the years and testing only on recent years or older years would not be an accurate representation of our overall model performances.

4.3 Validation and Model Optimization

For validation, our strategy was based on leave-one-out cross-validation (LOOCV) where each fold is one year of data in our training set. One fold is held out as the validation set, the rest are used for training, and we predict against the validation set to get validation error. We looped through this process for every year in the training data set, using every year as the validation set once, and took the average validation error for each fold. This validation process was used to decide what features to include in each model, tune hyperparameters and decide how to handle missing data in our data set. Learning curves were also created for each model to determine whether the models were underfitting or overfitting, to try to help guide improvements in the models.

5. Models

In the following sections, we go through each different model type, how they were iterated upon and their performances.

5.1 Decision Trees

From the perspective of a team doing a draft pick, the process is akin to making several strategic decisions until the best player to pick for the team is decided. Thus for our initial attempts, we tried several decision tree based models to see whether this model type performs well in predicting draft picks. As a baseline model, we first tried a simple Scikit-Learn decision tree to compare with other decision tree models.

5.1.1 Sklearn Decision Tree

In this model, we developed a machine learning model to predict NBA draft picks from the years 2010, 2015, 2020, and 2025. This model was built using a Decision Tree Regressor using Scikit-Learn with a max depth of 6 nodes and trained using NBA draft data from the years 2006 to 2025 (excluding the years 2010,2015,2020, 2025). Below are the predictions for the year 2025.

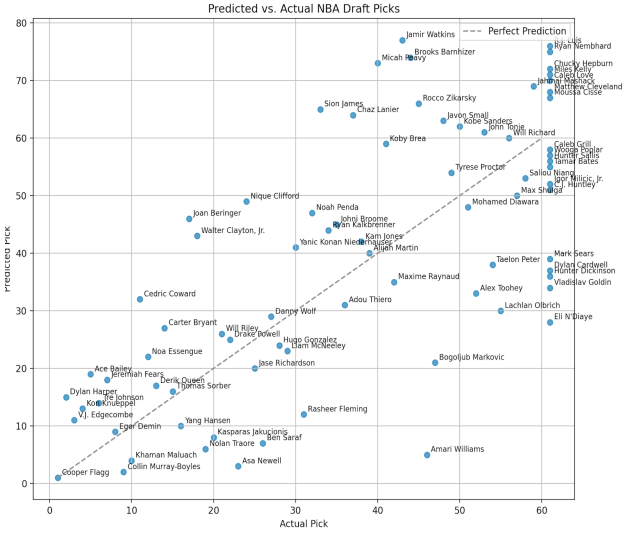

Figure 3: Predicted vs Actual NBA Draft Pick, Test Year 2025

The Sklearn decision tree resulted in a final performance of 25.77 ± 7.64 average pick

distance error across the test set's years. This basic decision tree model serves as a baseline model to compare performance.

5.1.2 Gradient Boosted Trees

After implementing the SciKit-Learn Decision tree model, we decided to attempt both the bagging and boosting methods for decision trees. For the boosting, we used XGBoost's Learning To Rank library which came with robust built-in tools for analysis. Learning To Rank works by training gradient-boosted decision trees to minimize a ranking-specific loss function. For our purposes, we set the objective function to rank:pairwise, which optimizes the model by comparing pairs of players within the same draft class and encouraging the model to assign higher scores to players picked earlier in the draft. While we found that XGBoost did well earlier in the draft (illustrated below via top-10 accuracy), its error increased significantly as actual pick values rose. An interesting insight we noticed at this point was that the model was heavily valuing the age feature as shown in Figure 5. We were satisfied to see this as intuitively, it makes sense that given two players with similar features, an NBA team would select the younger player. After performing our leave-one-out cross-validation, we found the optimal number of boosting rounds to be 50, and optimal maximum depth to be 6 splits.

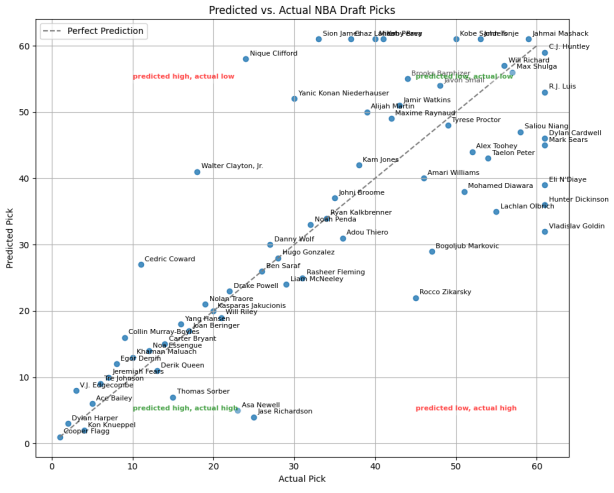

Figure 4: Predicted vs Actual NBA Draft Pick, Test Year 2025

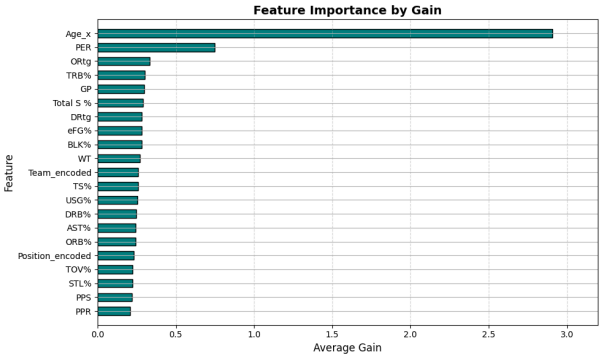

Figure 5: XGBoost Feature Importance

Table 1: XGBoost model performance on test set

Test Year	Avg. Pick Error	Top-10 Accuracy	Top-5 Avg. Pick Error
2010	17.08	70%	3.8
2015	15.44	50%	3.8
2020	16.40	40%	13.4
2025	10.32	50%	2.2
Avg.	14.8± 3. 1	52.5± 12. 5	5.8± 5. 1

5.1.3 Random Forest

The Random Forest model was built with the RandomForestRegressor from the sklearn.ensemble package. Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the mean prediction of the individual trees. It is an aggregation of numerous trees to reduce overfitting and improve the generalizability of the model, which was a clearly explained weakness of individual decision tree models such as the Scikit-learn Decision Tree Regressor (Section 6.1.1).

To compare the Random Forest model, we trained it on each draft class between 2006 and 2025, excluding the target years 2010, 2015, 2020, and 2025. This ensured that the results were indicative of generalization performance rather than memorization of past data. For each test year, we predicted the draft pick ranking of all eligible players using their pre–draft statistics.

Below is a summary of the model's performance across the test set.

Table 2: Random forest performance on test set

Test Year	Mean Absolute Error (MAE)	RMSE	% Exact Match	% Within 5 Picks
2010	11.87	15.60	4.92%	28.69%
2015	10.10	13.33	7.58%	36.36%
2020	10.13	13.31	9.35%	35.97%
2025	9.64	12.01	2.60%	31.17%
Average	10.94	13.61	6.61%	33.05%

As seen in the table above, the model achieved a mean absolute error of approximately 10.94 pick position and an RMSE of 13.61 across the four test years. While the exact match accuracy was modest, the model successfully predicted over 33% of picks within five positions of their true draft spot. This level of performance highlights the model's ability to rank players with reasonable accuracy, especially given the inherent noise in the draft process where non-statistical factors (e.g., team fit, personality, injuries) play a significant role.

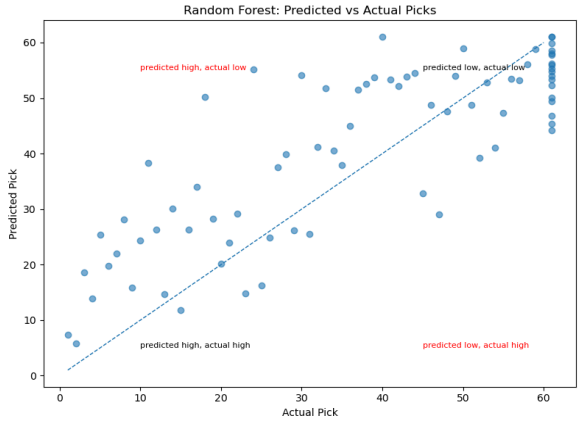

Figure 6:Predicted vs Actual NBA Draft Pick, Test Year 2025

The 2025 performance graph (Figure 6) displays predicted versus actual picks, with points near the diagonal being more accurate predictions. We notice that while top-ranked players at times have high prediction error (for instance, Cooper Flagg predicated at 7.38, chosen at 1), numerous mid and later round predictions congregate tightly near their real values.

Hyperparameters for the Random Forest model were first set at default. Learning curve analysis (not included) informed selecting the number of trees and max depth as a tradeoff between complexity and performance. In practice, it was possible to set the number of estimator at 200 and enable deeper trees with good generalization without severe overfitting,

Overall, the Random Forest model performed notably better than individual decision tree models, validating the merits of ensemble methods at identifying more intricate, nonlinear interaction within our dataset. An important reference illustrating the merits of Random Forests in sport analytics is by Bunker and Thabtah (2019), who demonstrate similar ensemble techniques in predicting athlete performance outcomes [5].

5.4 Linear Regression

Linear regression was implemented to model our draft prediction problem using the normal equation. A linear model is sensible because preliminary analysis of our feature correlation matrix revealed moderate linear relationships between several efficiency statistics, such as PER, ORtg, and TS%, and draft pick position, indicating that a linear approach could effectively capture key predictive signals in the data. After developing a functional model, feature importance was calculated and used to determine how many features provided optimal performance from this model. Although feature scaling is not crucial to normal equation performance, features were scaled to allow parameters to be directly comparable to help determine feature importance. The resulting parameters from training on scaled features acted as a "feature importance score". Based on the

feature importance scores, analysis was done using leave-one-out cross-validation to determine how many of the top features should be included in the model. It was determined that 12 features provided optimal performance and can be seen in Figure 7.

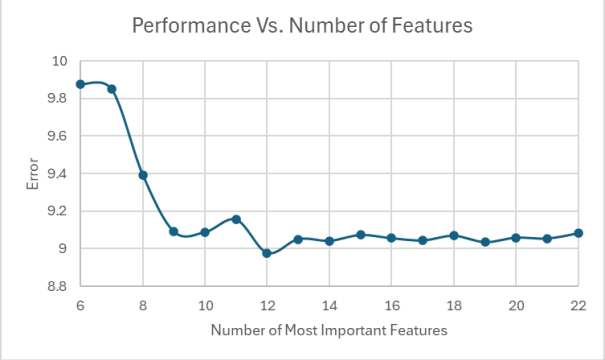

Figure 7: Linear Regression Feature Importance Analysis

Once optimal features were determined, learning curve analysis was implemented to determine if this model is overfitting or underfitting. The learning curves show that this model does not seem to have high bias or high variance as seen in Figure 8. Training error starts low, increasing with more data and validation error starts high, decreasing with more data. They both converge to a similar level of error, indicating that this model learns well from the data, predicts well on unseen data and has a good balance of bias and variance.

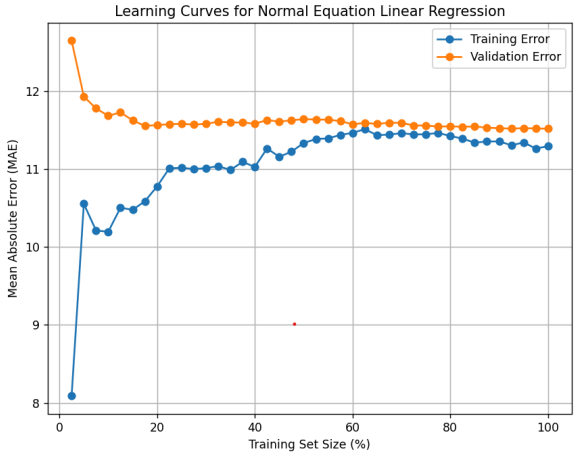

Figure 8: Linear Regression Learning Curves

As a final performance by the test set, the linear regression model achieved an average pick error of 9.57 ± 0.41 and a top-10 hit rate of 52.5% ± 13.0% across the four test years. Pick prediction performance for 2025 can be seen in Figure 9. All test year prediction performance can be found in Appendix D.

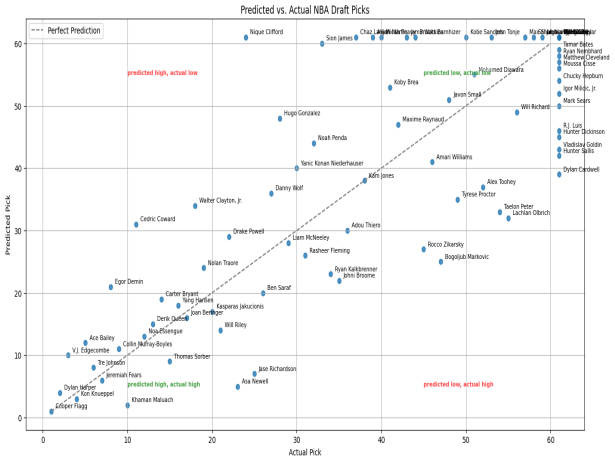

Figure 9: Linear Regression 2025 Performance

Table 3: Linear regression model performance on test set

Test Year	Mean Absolute Pick Distance Error	Top 10 hit rate
2010	10.71	60%
2015	9.39	40%,
2020	8.83	40%
2025	9.64	70%
Average	9.57 ± 0.41	52.5% ± 13.0%

5.5 Neural Network

We explored the use of a neural network with the hope that it could learn the key characteristics necessary for a college basketball player to be drafted into the NBA based on their college statistics. The initial network architecture was designed to capture potential characteristics. As shown in Figure 10, the first layer consists of 16 neurons and is intended to learn both the raw features and any meaningful combinations of those features that could aid in making accurate predictions. Layer 2 consists of 4 nodes, each responsible for a distinct aspect of the player's profile. The final layer outputs the predicted draft ranking for the player. Each neuron in the

nerwork applies a linear transformation followed by ReLU activation.

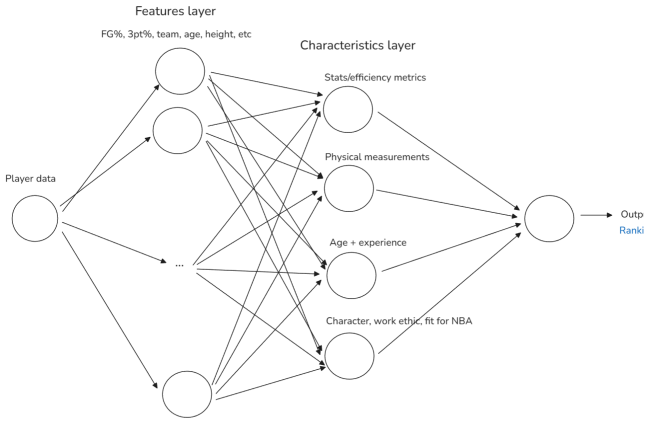

Figure 10: An initial architecture for the neural network model.

After training and testing, our initial model revealed significant underfitting in its learning curve, as there were not enough neurons to capture the complexity of the data. To address this, we refined the model by increasing the number of neurons to 128 in the first layer, and added four parallel branches with 16 neurons each for the second layer. We also included a third layer with 32 neurons to combine the output of the second layer before producing the final prediction. With the additional neurons and layers, dropout regularization is added to combat overfitting. This updated structure yielded better performance, and the model demonstrated a much more fitting learning curve.

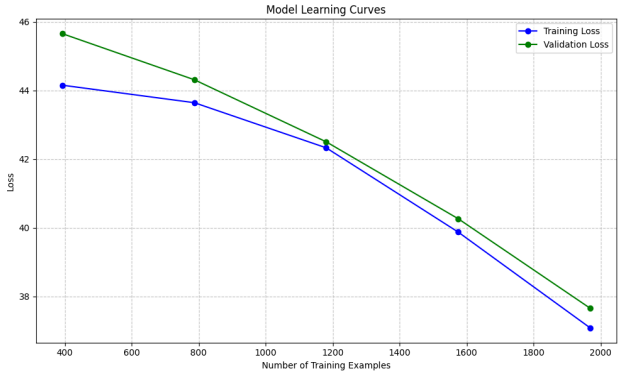

Figure 11: Learning curve for the initial neural network model showing underfitting.

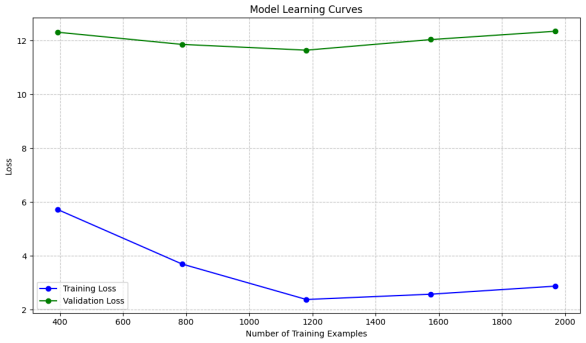

Figure 12: Learning curve for the second neural network model showing a good fit.

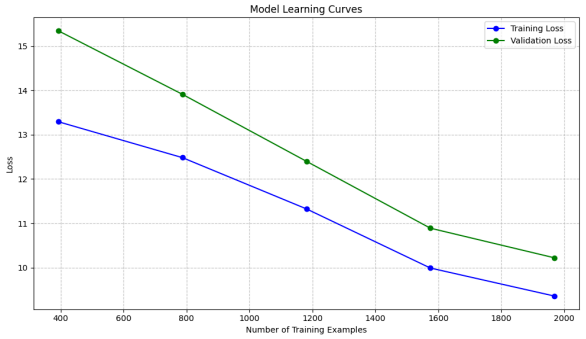

Figure 13: Learning curve for final initial neural network model showing overfitting.

To further assess if increasing the number of neurons would improve predictions, we experimented with doubling the number of neurons in each layer to 256, 128, and 64 neurons, respectively. However, when training the model with these configurations, the resulting learning curve showed high variance and overfitting, as expected.

To further tune the model we tried excluding features and adding different derived features, such as BMI from weight and height, in hopes that it would reduce noise and improve error. After several rounds of iteration with more/less features as well as different features, we found that the model's performance did not significantly differ, hovering around a pick distance of 11. A possible explanation is that with all the features, the neural network was already able to learn from our training data about any impactful features and relationships in the data. Thus, the manual

feature engineering was not impactful for this particular model.

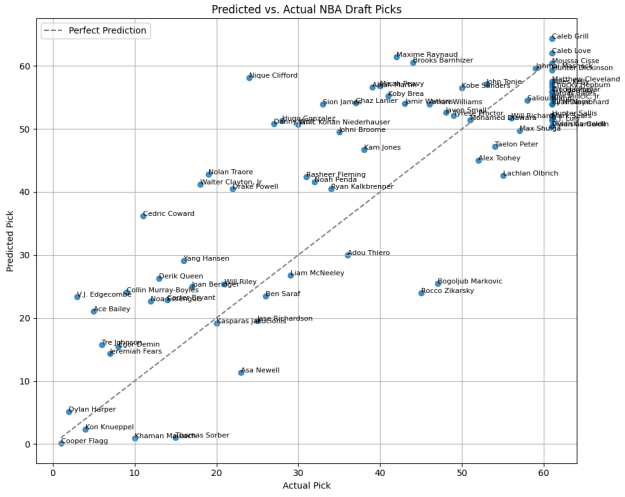

Figure 14: Neural network 2025 performance

As a final report for the best tuned neural network's performance, the model achieved an average pick distance error of 10.99 ± 0.64 with a top 10 hit rate of 60.0% ± 15.8% across all years from the test set.

Table 4: Neural network model performance on test set

Test Year	Mean Absolute Pick Distance Error	Top 10 hit rate
2010	11.62	70%
2015	11.59	50%
2020	10.58	40%
2025	10.16	80%
Average	10.99 ± 0.64	60.0% ± 15.8%

5.6 Ensemble Model

XGBoost, Neural Network, and a Normal Equation model were combined in an ensemble to provide a more stable and accurate prediction of draft outcomes. Each model's contribution to the ensemble was determined by its performance against the validation set, with higher-performing models receiving greater weight. The workflow for this approach involved merging the three sets of predicted draft orders, calculating a weighted average of each player's projected pick, and then reordering the results based on these averages. This process produced the final predicted draft order, balancing the strengths of all three models while reducing the impact of individual model biases or weaknesses.

The ensemble performed poorly because the models were not sufficiently complementary, often sharing the same weaknesses and amplifying errors instead of correcting them. The weighting scheme, based on validation performance, may have been misaligned with the test data, while the inclusion of the simpler Normal Equation model likely introduced noise. Additionally, averaging draft pick predictions blurred important distinctions, and a lack of proper calibration between models further reduced accuracy.

6. Evaluation and Results

To evaluate model performance, we compared each model's average draft pick prediction error and top-10 hit rate across four test years: 2010, 2015, 2020, and 2025. Each model was trained on all other years and tuned using leave-one-out cross-validation. The results are summarized below in the table below.

Table 5: Summary of Model Performance

Model	Avg. Pick Error	Best Year (error)	Worst Year (error)
Linear Regression	9.57 ± 0.41	2020 (8.83)	2010 (10.71)
Neural Network	10.99 ± 0.64	2025 (10.16)	2010 (11.62)
Random Forest	10.94 ± 0.59	2025 (9.64)	2010 (11.87)
XGBoost (Rank:Pairwise)	14.81 ± 3.01	2025 (10.32)	2020 (16.40)
Sklearn Decision Tree	25.77 ± 7.64	2025 (19.50)	2010 (35.40)

6.1 Model Accuracy

As shown in the predicted vs. actual draft position plots earlier in the report, models such as linear regression and the neural network exhibited tighter clustering along the diagonal line, especially in the top half of the draft, indicating stronger predictive power for early picks. In contrast, models like the Sklearn Decision Tree showed a much wider scatter, reflecting poor generalization and higher variance in their predictions.

6.2 Interpretation of Results

Across all metrics, the linear regression model performed the most consistently. This suggests that, despite the nonlinearities of the draft process, much of the predictive signal lies in compound linear relationships among player statistics such as PER, offensive rating, and age. The model also benefitted from strong feature selection and low variance, as shown in the learning curves.

The neural network model performed slightly worse in terms of average error but had the highest top-10 hit rate. This implies it learned richer representations of high-draft-value players, likely those with standout statistical profiles, but was less consistent in middle and later rounds.

Random Forest offered a balance between model flexibility and interpretability. It had slightly higher error than linear regression but outperformed other tree-based models by a significant margin. Its strength lies in reducing overfitting through averaging many deep trees.

XGBoost performed relatively well in top-10 accuracy but suffered from inconsistency in later picks. While its pairwise ranking objective helped with high-value players, it appears sensitive to draft class composition and may overemphasize features like age, resulting in drift during the mid-to-late draft.

The Sklearn Decision Tree performed the worst, confirming that shallow trees fail to generalize on such a complex task, likely overfitting to year-specific patterns in training data.

Lastly, the ensemble model did not offer the expected improvement. Instead, it suffered from misalignment between validation and test performance, as well as error averaging that diluted strong individual predictions.

Table 6: Top-10 Hit Rate Breakdown

Model	**2010**	**2015**	**2020**	**2025**	**Avg.**
Linear Regression	60%	40%	40%	70%	52.5%
Neural Network	70%	50%	40%	80%	60.0%
XGBoost	70%	50%	40%	50%	52.5%
Random Forest	~29%	~36%	~36%	~31%	33.1%
Ensemble	~40%	~50%	~45%	~45%	~45%

7. Discussion

7.1 Why Linear Regression Worked Best

Based on the results of our models, the linear regression model consistently showed the lowest error across all test years. This suggests that the relationship between player statistics and draft pick was more linear than we initially expected, particularly for higher draft picks, where selections appeared to rely more heavily on measurable player performance. This trend became even more pronounced when we used only the most impactful features for prediction. Another factor is that linear regression was easier to tune the hyper parameters efficiently. It's possible with other models that the same prediction accuracy can be achieved, but much more tuning would be needed.

7.2 Why the Ensemble Method Performed Poorly

Despite our initial hypothesis that an ensemble method combining the linear regression, XGBoost, and neural network models would cancel out individual errors and produce a more accurate prediction by leveraging the strengths of each model, we found that it actually performed worse than using any single model on its own. Our best explanation for this outcome is that the errors and noise from each model did not offset one another; instead, the ensemble effectively averaged the predictions from the three models, leading to a diluted result rather than an improved one.

7.3 Comparing Decision Tree Models

From our results, we saw that the Random Forest model performs substantially better than the scikit-learn Decision Tree and XGBoost decision tree in predicting players being drafted in the NBA. This is likely due to Random Forest models ability to reduce overfitting by aggregating predictions from multiple decision trees, leading to more stable and generalizable results. In contrast, single decision trees like the Sklearn decision tree and the XGBoost decision tree tend to capture noise in the training data, resulting in less reliable predictions. However, XGBooost differs from a basic decision tree by iteratively minimizing prediction errors through gradient boosting. Unlike a single decision tree that can easily overfit to noise or fail to capture complex patterns, XGBoost builds an ensemble of trees where each new tree learns from the errors of the previous one. This learning approach helps control overfitting and improves generalization. XGBoost also has the ability to handle missing values and split decisions more effectively, which is particularly useful when working with real world data that may be noisy or missing data. As a result, XGBoost generally produces more accurate and reliable predictions compared to the Sklearn decision tree. This difference in model performance is clearly shown in Figure 15.

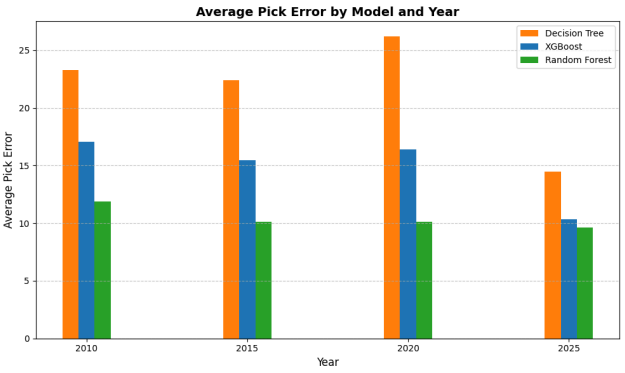

Figure 15: Difference in performance between Sklearn decision tree, XGBoost decision tree and Random Forest

7.4 Limitations of Our Models

One of the major limitations of our model is its inability to capture all the considerations that go into a draft pick solely from statistical performance data. From the perspective of an NBA team selecting a player from the draft pool, the decision is not based solely on college statistics but also on factors like the team's needs, how well the player fits within the team, and the player's attitude and work ethic. These attributes are just as important as the statistical data and are likely a key reason why we were unable to reduce the average pick distance error below around 10 from any model.

In addition to these conceptual limitations, our model also faced technical constraints. The dataset had some missing values, especially for players with fewer games or incomplete stat lines. Only using the average value for missing data likely lead to worse predictions. Moreover, our model focused exclusively on numerical performance metrics and did not incorporate crucial contextual factors such as team dynamics, player interviews, or scouting reports, data that NBA teams heavily rely on. These omissions has no doubt limited the models' ability to fully replicate real-world draft decision-making.

8. Future Work

Future improvements to our model should focus on incorporating team-specific context and qualitative data. Currently, all teams are treated equally in our models, but in reality, each team has their specific positional needs and drafting strategies. Integrating team-level data and modeling the draft as a sequential process, where one team's pick affects the next, would better reflect how real drafts unfold. Additionally, incorporating sentiment from scouting reports and analyst commentary using natural language processing could capture non-statistical factors such as character and perceived potential. By combining these human and strategic elements with statistical inputs, we can build a more realistic and accurate draft prediction model. Further refinement in modeling undrafted players would also reduce noise and improve lower-tier prediction accuracy.

9. Conclusion

Our project set out to predict NBA draft positions using machine learning models trained on pre-draft player statistics. While we achieved reasonable performance, especially with linear regression and neural networks, all models plateaued around 9–11 average pick error, falling short of analyst mock drafts.

One of our biggest takeaways was that we spent too much time tuning many different models, including some that offered minimal improvement over simpler baselines. In hindsight, that time would have been better spent improving our data, whether by incorporating team context, positional needs, or qualitative scouting information. The limitations of our models weren't just about architecture or hyperparameters; they stemmed from the lack of information that real teams rely on during the draft.

Our strongest results came from models that emphasized simplicity and interpretability. Linear regression, with carefully selected features, consistently outperformed more complex methods. The neural network showed strength in identifying top picks but lacked consistency further down the draft.

Going forward, we would shift our focus from trying many models to enriching the dataset and targeting modeling approaches that capture ranking dynamics more effectively. This experience showed that in real-world problems, better data often beats more model complexity.

References

[1] F. Harmén, "Predicting basketball performance based on draft pick: A classification analysis," DIVA. Accessed: Aug. 01, 2025. [Online]. Available: https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1685098&dswid=2799

[2] N. Mamonov, Accessed: Aug. 01, 2025. [Online]. Available: https://er.ucu.edu.ua/server/api/core/bitstreams/010f4f2e-c176-4a79-9c2f-b997dff43f77/content

[3] "RealGM," NBA Basketball News, Rumors, Scores, Stats, Analysis, Depth Charts, Forums. Accessed: Aug. 01, 2025. [Online]. Available: https://basketball.realgm.com/nba/

[4] "Learning to Rank, xgboost 3.1.0-dev documentation." Accessed: Aug. 01, 2025. [Online]. Available: https://xgboost.readthedocs.io/en/latest/tutorials/learning_to_rank.html

[5] R. P. Bunker and F. Thabtah, "A machine learning framework for sport result prediction," *Applied Computing and Informatics*, vol. 15, no. 1, pp. 27–33, Jan. 2019, doi: 10.1016/j.aci.2017.09.005.

[6] A. Fromal, "NBA Draft: Why Age Should Be a Big Factor When Evaluating Prospects," Bleacher Report. Accessed: Aug. 01, 2025. [Online]. Available: https://bleacherreport.com/articles/964894-nba-draft-why-age-should-be-a-big-factor-when-evaluating-prospects

Appendix A (Tables)

Table A1: The list of features and compound features from our data.

Statistic	**Derived from**
Age	Age
Height	Height
Weight	Weight
School	School
Position	Position
Games played	Games played
True Shooting Percentage	PTS ÷ [2 × (FGA + 0.44 × FTA)]
Field Goal Percentage	FGM ÷ FGA
Offensive Rebound Percentage	100 × [ORB × (TmMP ÷ 5)] ÷ [MP × (TmORB + OppDRB)]
Defensive Rebound Percentage	100 × [DRB × (TmMP ÷ 5)] ÷ [MP × (TmDRB + OppORB)]
Assist Percentage	100 × AST ÷ (FGA by teammates while player is on court)
Total Rebound Percentage	100 × [TRB × (TmMP ÷ 5)] ÷ [MP × (TmTRB + OppTRB)]
Steal Percentage	100 × [STL × (TmMP ÷ 5)] ÷ [MP × OppPoss]
Block Percentage	100 × [BLK × (TmMP ÷ 5)] ÷ [MP × Opp2PA]
Usage Percentage	100 × [(FGA + 0.44 × FTA + TOV) × (TmMP ÷ 5)] ÷ [MP × (TmFGA + 0.44 × TmFTA + TmTOV)]
Pure Point Rating	100 × (AST − TOV) ÷ MP
Points Per Shot	PTS ÷ FGA
Offensive Rating	100 × (Points Produced ÷ Possessions Used)
Defensive Rating	100 × (Points Allowed ÷ Possessions Faced)
Player Efficiency Rating	PTS + REB + AST + STL + BLK - Missed FG -Missed FT - TO) / GP
Target	Player draft position