

Assignment 1

Kaggle Competition

Rushab Shah

Student ID: 14351985

https://github.com/rushman95/NBA_draft_classification.git

08/09/2023

36120 - Advanced Machine Learning Application
Master of Data Science and Innovation
University of Technology of Sydney



Table of Contents

1. Business Understanding	2
a. Business Use Cases	2
2. Data Understanding	3
3. Data Preparation	4
4. Modeling	6
a. Approach 1	6
b. Approach 2	6
c. Approach 3	7
5. Evaluation	8
a. Evaluation Metrics	8
b. Results and Analysis	8
c. Business Impact and Benefits	12
d. Data Privacy and Ethical Concerns	13
6. Deployment	14
1. Conclusion	15
2. References	16
3. Appendices	16
Appendix A - Data Dictionary	16



1. Business Understanding

a. Business Use Cases

The NBA (National Basketball Association) Draft is an annual event that serves as the primary gateway for aspiring basketball talents, offering them the opportunity to transition from college or international leagues into the highest echelon of the sport – the NBA. In the highly competitive landscape of the NBA, where every roster spot is fiercely contested, the drafting process becomes a strategic endeavor for teams to identify and secure top-tier talent. With an atmosphere of anticipation and the promise of future stardom, the NBA Draft captivates not only basketball aficionados but also the broader public.

The task of discerning which college or international players possess the attributes and potential to thrive in the NBA is complex, involving the analysis of an extensive array of statistics and historical data to make informed selections. This project makes use of this data and machine learning to predict which college basketball players are to be drafted into the NBA.

b. Key Objectives

The primary aim of this project is to develop a predictive binary classification model that can accurately forecast whether a college basketball player will be drafted into the NBA based on their performance statistics for the current season.

This project is considered to be of interest to 3 key groups of stakeholders:

1. **NBA Teams:** NBA teams who seek to identify and recruit top talent during the draft would be able to use this model to provide insights into a player's potential based on their college performance.
2. **College Players:** College basketball players aspiring to join the NBA can benefit from understanding their likelihood of being drafted and areas for improvement.
3. **Media and Fans:** The media and fans eager to speculate and discuss potential draft picks may gain from accurate drafting predictions.

The best performing model will be deployed to allow open access and prediction on any input data in the same format.





2. Data Understanding

The dataset contains a comprehensive array of performance statistics of college basketball players from 2009 until 2020, including offensive and defensive ratings, shooting efficiency metrics, rebounding percentages, and playmaking statistics. Additionally, it incorporates player-specific attributes such as year of study and height, providing a holistic view of college basketball players' profiles. The data dictionary has been provided in Appendix A.

Some observations found while exploring the data include:

1. The `player_id` column was the key for the dataset.
2. There were 5 columns that contained text: `team`, `conf`, `ht`, `yr` and `type`. All other columns contained numeric values.
3. There was a significant imbalance in the dataset: Only 536 players were drafted from a pool of 56091 players.
4. The `ht` values appeared to be corrupted, as they were provided as dates (for example, 01-Jun).
5. The `type` column contained only one value ('all').
6. The numeric values were provided at multiple different scales.
7. There was a significant presence of missing values in the dataset, with 33 out of the 64 columns containing missing values.
8. The largest number of missing values (54705) occurred in the `pick` column. It was also interesting that no players were drafted outside the pick (when the pick was null).



3. Data Preparation

The data preparation was largely done in accordance with the insights from the cleaning table. Each experiment also included some bespoke feature engineering that has been further described in 5.0 Modeling. The cleaning steps applied to all steps include:

The height values were converted to height in inches. The month was assumed to represent the height in feet so these values were adjusted to reflect height in inches.

Features dropped from the dataset include num, player_id and type, as they were not considered to influence the target variable.



SMOTE and undersampling techniques were performed for different experiments. SMOTE was initially performed due to the large imbalance in data. Undersampling, based on the removal of missing pick values, was applied from the second experiment onwards. This approach was found to produce predictions with similar accuracies to SMOTE, however, this also required the predictions to be manually adjusted to 0 for players that were not picked.

Imputations were performed to handle all other missing values. Simple imputations using the mean and median were both tested, as well as KNN based imputations.

Categorical features for all experiments were defined to be team, conf and yr.

Several feature interactions were also generated and tested for later experiments. The logic for these new features has been described below:

1. pts_per_min: This feature calculates the points scored per minute played, which measures a player's scoring efficiency over the course of a game.
2. pts_created: It represents the total points created by a player, considering both the points they scored and the points they assisted on. This can help assess a player's offensive contribution.
3. second_chance_pts: This metric calculates the total points a player scores, including any second-chance points obtained through offensive rebounds (oreb).
4. disruptive_plays: It quantifies a player's disruptive plays by summing their blocked shots (blk) and steals (stl), which are defensive actions that disrupt the opponent's offense.
5. total_stops: This feature calculates the total defensive stops a player makes, considering both defensive rebounds (dreb) and blocked shots (blk).

- 
6. `total_fg_made`: It represents the total field goals made by a player, including both shots made at or near the rim (`rismade`) and two-point shots that were not made at or near the rim (`midmade`).
 7. `total_fg_attempts`: This metric calculates the total field goal attempts, including both attempts at or near the rim (`rismade_rimmiss`) and two-point shots that were not made at or near the rim (`midmade_midmiss`).
 8. `ast_to_dreb_ratio`: This ratio assesses a player's ability to assist (`ast`) relative to their defensive rebounds (`dreb`), indicating their contribution to playmaking and defensive rebounding.
 9. `defensive_performance`: It represents a composite metric for a player's defensive performance, calculated by summing their defensive rating (`drtg`) and adjusted defensive rating (`adrtg`).
 10. `overall_player_impact`: This composite metric measures a player's overall impact, combining their offensive BPM (`obpm`) and defensive BPM (`dbpm`).
 11. `total_rebounds`: This feature calculates the total rebounds a player collects, including both offensive rebounds (`oreb`) and defensive rebounds (`dreb`).
 12. `total_extended_possessions`: It quantifies a player's involvement in extended possessions, considering offensive rebounds (`oreb`), defensive rebounds (`dreb`), and assists (`ast`).
 13. `pts_per_pick`: This metric assesses a player's scoring efficiency relative to their draft position, by calculating points per pick (`pts/pick`).
 14. `potential_predictor`: It calculates a composite metric that combines a player's recruiting rank (`Rec_Rank`) and the ratio of assists to turnovers (`ast_tov`) as a predictor of their potential impact.
- 

4. Modeling

The modeling strategy involved the use of an auto ML library to fit multiple classifiers on the model, before attempting to refine and improve a single classifier on the model. Pycaret was considered to be an effective choice as it allowed for easy testing of different feature engineering techniques, however, it was difficult to find a version of Pycaret that was accommodated in poetry so SciKit Learn was used in later experiments, as per the assignment requirements.

a. Approach 1

This approach focused on dealing with missing values. Features with over 50% of values missing (including the pick) were dropped.

SMOTE was used to balance the target 'drafted' class as there is a major imbalance, with only approximately 1% of 'drafted' players.

Simple imputations to fill remaining numerical missing values with median and mean value of those features. The mean and median values did not show any significant differences, however, the median was ultimately selected.

The categorical values were specified, as above, and the default train test split from Pycaret was used to plot this model.

Pycaret was used to automatically train several different classifiers to determine the best one. XGboost returned the highest overall metrics but was slow to train, while light GBM was faster to train and returned a high AUC and F1 score, therefore both models were trained, but only light GBM could be tuned due to slow training time for XGboost.

b. Approach 2

The main difference from approach 1 was the sampling technique and inclusion of features like pick and Rec_rank that were dropped. All of the players that were drafted came from players that were picked so this was considered to be an important filter that allowed for undersampling of the data.

Again Pycaret trained multiple classifiers, however, this time the Logistic Regression model proved to be the highest in terms of AUC. The simplicity of the algorithm was considered to be advantageous to further modeling.



c. Approach 3

This approach focused on further optimizing a logistic regression model, this time with a Scikit learn pipeline, allowing for more customizability. A stratified train test split of 0.8 at a seed of 42 was applied.

For categorical values, imputations to fill with mode and one hot encoding. For numerical values, imputations to fill with mean and scaling was performed

Several new features were created for this experiment that aimed to aggregate other features. They have been described in more detail in 4.0 Data Preparation.

In terms of hyperparameter tuning, several different solvers were tested including liblinear, sag, lbfgs and newton-cg to determine any improvements to the classification model. They all returned the same confusion matrix, however, so tuning did not prove to be very effective.





5. Evaluation

a. Evaluation Metrics

The evaluation metrics used to assess the models in this project were the AUROC, Accuracy, Precision, Recall and F1 Score. AUROC was the primary metric used to compare the model's performance. A description of each metric is given below:

- AUROC: Area Under the Receiver Operating Characteristic curve quantifies the model's ability to distinguish between the two classes (positive and negative) by measuring the area under the ROC curve.
- Accuracy: A measure of how many predictions the model got correct out of the total number of predictions.
- Precision: A measure of how many positive predictions made by the model were actually correct.
- Recall: A measure of the model's ability to correctly identify all relevant instances (true positives) out of all actual positive instances.
- F1 Score: It is the harmonic mean of precision and recall, providing a balanced measure that considers both false positives and false negatives.

While the AUROC is the most important measure, for this particular business question, a higher recall than precision is desired, as false negatives would be costly as this would mean that drafted players would be missed.

b. Results and Analysis

Results of Approach 1:

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
xgboost	Extreme Gradient Boosting	0.9906	0.9826	0.4425	0.5149	0.4715	0.4668	0.4704	73.5850
lightgbm	Light Gradient Boosting Machine	0.9904	0.9722	0.4773	0.4982	0.4860	0.4812	0.4821	17.5040
dummy	Dummy Classifier	0.9904	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4610
et	Extra Trees Classifier	0.9903	0.9864	0.4802	0.4901	0.4828	0.4780	0.4792	8.9840
rf	Random Forest Classifier	0.9895	0.9808	0.3814	0.4437	0.4088	0.4036	0.4055	71.6290
dt	Decision Tree Classifier	0.9843	0.6698	0.3491	0.2659	0.2999	0.2922	0.2959	9.8220
gbc	Gradient Boosting Classifier	0.9823	0.9788	0.6826	0.3091	0.4251	0.4174	0.4518	132.5890
ada	Ada Boost Classifier	0.9815	0.9704	0.6239	0.2875	0.3926	0.3845	0.4151	26.4930
lr	Logistic Regression	0.9602	0.9891	0.9331	0.1861	0.3100	0.2988	0.4068	16.3890
knn	K Neighbors Classifier	0.9588	0.8279	0.6297	0.1379	0.2262	0.2138	0.2817	6.3030
svm	SVM - Linear Kernel	0.9427	0.0000	0.9544	0.1544	0.2599	0.2474	0.3649	5.1490
ridge	Ridge Classifier	0.9418	0.0000	0.9707	0.1384	0.2422	0.2293	0.3548	0.6440
lda	Linear Discriminant Analysis	0.9418	0.9891	0.9760	0.1390	0.2432	0.2304	0.3567	2.1980
nb	Naive Bayes	0.8606	0.9558	0.9654	0.0623	0.1170	0.1008	0.2259	1.4480
qda	Quadratic Discriminant Analysis	0.6945	0.8675	0.9760	0.0297	0.0576	0.0398	0.1400	1.1820

The first approach already showed great results, with a very high AUC, however when deployed on the Kaggle test data, the model showed a slightly lower AUC.

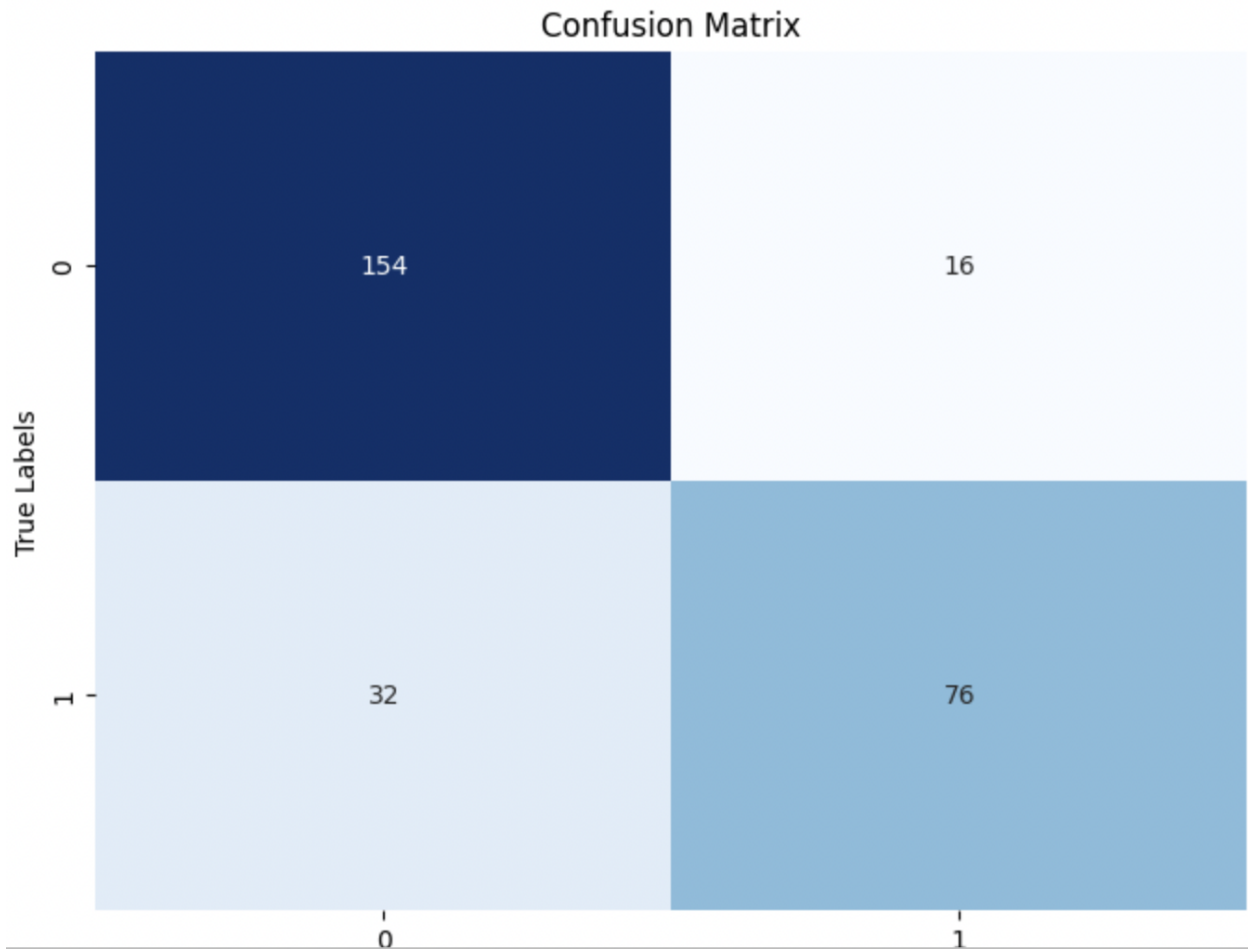
Results of Approach 2:

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lr	Logistic Regression	0.8144	0.8903	0.7279	0.7817	0.7502	0.6034	0.6076	1.6660
ridge	Ridge Classifier	0.8093	0.0000	0.6749	0.8067	0.7306	0.5856	0.5946	0.4790
lda	Linear Discriminant Analysis	0.7938	0.8641	0.6536	0.7849	0.7077	0.5516	0.5613	0.8570
et	Extra Trees Classifier	0.7876	0.8588	0.6610	0.7611	0.7039	0.5404	0.5463	1.7860
rf	Random Forest Classifier	0.7680	0.8494	0.6774	0.7113	0.6899	0.5059	0.5094	1.3880
gbc	Gradient Boosting Classifier	0.7629	0.8486	0.6534	0.7190	0.6770	0.4919	0.4988	2.1490
xgboost	Extreme Gradient Boosting	0.7526	0.8364	0.6371	0.7048	0.6624	0.4693	0.4760	1.3150
lightgbm	Light Gradient Boosting Machine	0.7505	0.8411	0.6243	0.7048	0.6553	0.4626	0.4696	2.4640
nb	Naive Bayes	0.7474	0.8586	0.8396	0.6307	0.7193	0.4982	0.5171	0.5770
qda	Quadratic Discriminant Analysis	0.7412	0.8354	0.8074	0.6278	0.7057	0.4809	0.4945	0.7490
ada	Ada Boost Classifier	0.7402	0.8211	0.6106	0.6972	0.6449	0.4423	0.4491	0.8870
svm	SVM - Linear Kernel	0.7041	0.0000	0.6389	0.6581	0.5903	0.3738	0.4162	0.3890
dt	Decision Tree Classifier	0.6990	0.6749	0.5681	0.6258	0.5915	0.3551	0.3591	0.6490
knn	K Neighbors Classifier	0.6814	0.7156	0.5361	0.5988	0.5644	0.3150	0.3170	0.4110
dummy	Dummy Classifier	0.6134	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.3910

Despite the lower scores, the second approach showed an improved result to what was achieved in approach 1, highlighting that the pick is a very important feature to predict if a player will be drafted. A SMOTE sampling technique was also trialed with the inclusion of pick but returned very similar results so it was decided that it may be better to work with less data.

Results of Approach 3:

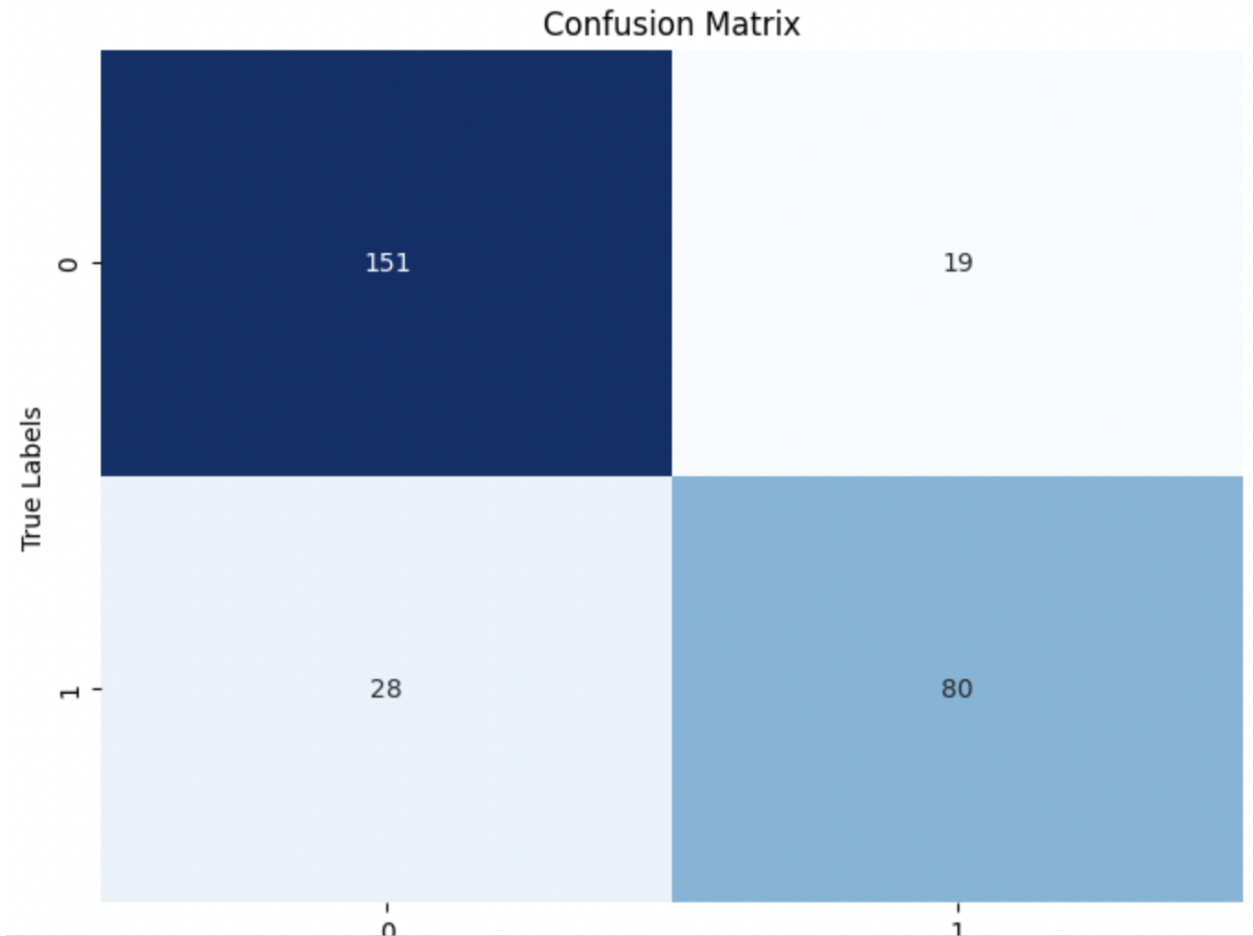
Accuracy: 0.8273381294964028
AUC: 0.8651960784313725
Precision: 0.8260869565217391
Recall: 0.7037037037037037



Despite a lower AUC, approach 3 still returned the same AUROC score on the Kaggle test data. This indicates that the newly added features did not show any notable improvement in the model's performance. Due to this, they were excluded from the final model. A K nearest neighbor's imputation also failed to show any improvement, perhaps indicating that the features with missing values were of low importance to the model.

The results for the final model have been shown below:


Accuracy: 0.8309352517985612
AUC: 0.903267973856209
Precision: 0.8080808080808081
Recall: 0.7407407407407407



The final model did not show much improvement from Approach 2 and 3 on the Kaggle data, however it used less data and fewer features to obtain the same results.

c. Business Impact and Benefits

The analysis showed that the pick is the most important feature when predicting if college basketball players will get drafted for the NBA. All of the drafted players were from the pick. The team, conf and height also proved to be important features in determining whether a player would be drafted. There appears to be a high bias for teams, with a large number of drafted college players coming from a small number of teams.



The final model would be useful to all of the stakeholders mentioned, allowing them to predict the likelihood of getting drafted. Since the accuracy is at 82%, the results of the model should be taken with caution, but there is a high probability of correctly predicting if a player will be drafted.

- NBA teams will be able to use this model to identify talent early to give them a competitive advantage going into the draft.
- Players would be able to understand how their own stats compare against what is required to join the draft.
- Media and Fans: There are additional incentives for both, the media and fans to identify who will be drafted. The results of this model would be of interest to both as they eagerly wait to see who will be drafted.

A deployment of the model has also been demonstrated in section 7.

d. Data Privacy and Ethical Concerns

Data Privacy Implications:

College NBA players have privacy rights regarding their performance and personal information. The dataset was obtained from an open source, with the players names anonymized to protect player privacy. For this purpose, it means that player details cannot be traced back to them. It is unclear whether players consent was obtained before compiling this dataset. For match and league performance related metrics, players are unlikely to own this data.

Ethical Concerns:

The main ethical concerns pertain to the deployment of the model. The model may inadvertently introduce biases, favoring certain types of players while disadvantaging others, leading to unfair treatment in the draft process. A lack of transparency in the prediction process can also be unethical. Players and stakeholders should understand how decisions are made based on their data, highlighting the importance of explainable models.



6. Deployment

The deployment process is described in the table below.

Action	Command
Poetry first needs to be installed.	<code>pip install poetry</code>
Clone the github repo.	<code>git clone https://github.com/rushman95/NBA_draft_classification.git</code>
Change the directory to the cloned folder	<code>cd ../NBA_draft_classification/</code>
Launch a poetry shell	<code>poetry shell</code>
Finally, predict by specifying your filepath	<code>python main.py --input_data "{input_filepath}"</code>

Poetry provides a useful way to manage all required packages, however, conflicting dependencies may arise that may affect deployment.

The poetry shell has also been built with python 3.9.13, which is not the latest version of python. This may have its own limitations in terms of newer packages being installed. However, updating the poetry versions should take care of this.





7. Conclusion

The project was an overall success with a highly accurate model achieved. The model is able to effectively predict the likelihood of whether a college player will be drafted to the NBA based on their current season. It is also advantageous that the algorithm used is logistic regression, which has great interpretability.

However, there is still room for improvement. The modeling process did not apply a consistent split for all the experiments with Pycaret and Scikit-learn. Despite the use of 2 different libraries, a consistent split would have allowed for a fairer comparison. Many, if not all of the newly generated features were not useful to the model. The imputations also did not have a major effect on the model suggesting



Appendices

Appendix A - Data Dictionary

name	description
team	Name of team
conf	Name of conference
GP	Games played
Min_per	Player's percentage of available team minutes played
ORtg	ORtg - Offensive Rating (available since the 1977-78 season in the NBA); for players it is points produced per 100 possessions, while for teams it is points scored per 100 possessions. This rating was developed by Dean Oliver, author of Basketball on Paper. Please see the article Calculating Individual Offensive and Defensive Ratings for more information.
usg	Usg% - Usage Percentage (available since the 1977-78 season in the NBA); the formula is $100 * ((FGA + 0.44 * FTA + TOV) * (Tm MP / 5)) / (MP * (Tm FGA + 0.44 * Tm FTA + Tm TOV))$. Usage percentage is an estimate of the percentage of team plays used by a player while he was on the floor.
eFG	eFG% - Effective Field Goal Percentage; the formula is $(FG + 0.5 * 3P) / FGA$. This statistic adjusts for the fact that a 3-point field goal is worth one more point than a 2-point field goal. For example, suppose Player A goes 4 for 10 with 2 threes, while Player B goes 5 for 10 with 0 threes. Each player would have 10 points from field goals, and thus would have the same effective field goal percentage (50%).
TS_per	TS% - True Shooting Percentage; the formula is $PTS / (2 * TSA)$. True shooting percentage is a measure of shooting efficiency that takes into account field goals, 3-point field goals, and free throws.
ORB_per	ORB% - Offensive Rebound Percentage (available since the 1970-71 season in the NBA); the formula is $100 * (ORB * (Tm MP / 5)) / (MP * (Tm ORB + Opp DRB))$. Offensive rebound percentage is an estimate of the percentage of available offensive rebounds a player grabbed while he was on the floor.
DRB_per	DRB% - Defensive Rebound Percentage (available since the 1970-71 season in the NBA); the formula is $100 * (DRB * (Tm MP / 5)) / (MP * (Tm DRB + Opp ORB))$. Defensive rebound percentage is an estimate of the percentage of available defensive rebounds a player grabbed while he was on the floor.
AST_per	AST% - Assist Percentage (available since the 1964-65 season in the NBA); the formula is

	$100 * AST / (((MP / (Tm MP / 5)) * Tm FG) - FG)$. Assist percentage is an estimate of the percentage of teammate field goals a player assisted while he was on the floor.
TO_per	TOV% - Turnover Percentage (available since the 1977-78 season in the NBA); the formula is $100 * TOV / (FGA + 0.44 * FTA + TOV)$. Turnover percentage is an estimate of turnovers per 100 plays.
FTM	Free Throws
FTA	Free Throw Attempts
FT_per	Free Throw Percentage; the formula is FTM / FTA .
twoPM	2P - 2-Point Field Goals
twoPA	2PA - 2-Point Field Goal Attempts
twoP_per	2P% - 2-Point Field Goal Percentage; the formula is $2P / 2PA$.
TPM	3P - 3-Point Field Goals (available since the 1979-80 season in the NBA)
TPA	3PA - 3-Point Field Goal Attempts (available since the 1979-80 season in the NBA)
TP_per	3P% - 3-Point Field Goal Percentage (available since the 1979-80 season in the NBA); the formula is $3P / 3PA$.
blk_per	BLK% - Block Percentage (available since the 1973-74 season in the NBA); the formula is $100 * (BLK * (Tm MP / 5)) / (MP * (Opp FGA - Opp 3PA))$. Block percentage is an estimate of the percentage of opponent two-point field goal attempts blocked by the player while he was on the floor.
stl_per	STL% - Steal Percentage (available since the 1973-74 season in the NBA); the formula is $100 * (STL * (Tm MP / 5)) / (MP * Opp Poss)$. Steal Percentage is an estimate of the percentage of opponent possessions that end with a steal by the player while he was on the floor.
ftr	
yr	Student's year of study: `Fr` for freshmen, `So` for sophomores, `Jr` for juniors, `Sr` for seniors
ht	Height of student
num	Player's number
porpag	Points Over Replacement Per Adjusted Game
adjoe	AdjO – Adjusted offensive efficiency – An estimate of the offensive efficiency (points scored per 100 possessions) a team would have against the average D-I defense.
pfr	
year	Season's year
type	Type of metrics displayed: `All` for all types, `C` for conference, `NC` for non-conference, `PC` for pre-conference tour, `R` for regular season, `P` for post-season, `T` for NCAA
Rec_Rank	Recruiting rank i.e. what the player was ranked as a recruit coming out of high school
ast_tov	Ratio Assists against Turnovers

rimmade	Shots made at or near the rim
rimmade_ri mmiss	Sum of Shots made at or near the rim and Shots missed
midmade	Two point shots that were not made at or near the rim
midmade_ midmiss	Sum of Two point shots that were not made at or near the rim and Shots missed
rim_ratio	Ratio between Shots made at or near the rim against Shots missed
mid_ratio	Ratio between Two point shots that were not made at or near the rim and Shots missed
dunksmade	Dunks made
dunksmiss_ dunksmade	Sum of Dunks made and Dunks missed
dunks_ratio	Ratio between Dunks made and Dunks missed
pick	Order of NBA draft
drtg	DRtg - Defensive Rating (available since the 1973-74 season in the NBA); for players and teams it is points allowed per 100 possessions. This rating was developed by Dean Oliver, author of Basketball on Paper. Please see the article Calculating Individual Offensive and Defensive Ratings for more information.
adrtg	Adjusted DRtg
dporpag	Asadjusted porpag
stops	Stops - Stops; Dean Oliver's measure of individual defensive stops. Please see the article Calculating Individual Offensive and Defensive Ratings for more information.
bpm	BPM - Estimate the player's contribution in points above league average per 100 possessions played
obpm	Offensive BPM
dbpm	Defensive BPM
gbpm	BPM 2.0
mp	MP - Minutes Played (available since the 1951-52 season)
ogbpm	Offensive BPM 2.0
dgbpm	Defensive BPM 2.0
oreb	ORB - Offensive Rebounds (available since the 1973-74 season in the NBA)
dreb	DRB - Defensive Rebounds (available since the 1973-74 season in the NBA)
treb	TRB - Total Rebounds (available since the 1950-51 season)
ast	AST - Assists
stl	STL - Steals (available since the 1973-74 season in the NBA)
blk	BLK - Blocks (available since the 1973-74 season in the NBA)
pts	PTS - Points



player_id	Unique identifier of player
drafted	Target - Was the player drafted at the end of the season