# **Predicting NBA Player Career Longevity**

# 

# **Introduction**

The purpose of this project is to generate a machine learning (ML) model on a dataset of new NBA basketball players, with the aim of predicting the probability of these basketball players to continue to play after 5 years. This report outlines the model design requirements and performance outcomes along with challenges faced and their resolutions. This analysis is conducted on behalf of the NBA league’s management team to enable them to identify and invest in rookie players with future potential, while reducing the risk of investing in players who are less likely to succeed and deliver the required Return on Investment (ROI) for the league.

# **Business Understanding**

The purpose of this project is to enable the NBA league management team to make data-based business decisions by predicting the future potential of their rookie players. The league needs to make million dollar investments when attracting new players and training them, along with many on-the-job benefits that are provided to retain the players. Currently, a rookie NBA player is paid nearly USD1million per season (Kennedy, 2021). Considering the huge financial investment this requires, it’s critical for the league to identify players who will last long enough in the game for them to deliver sufficient return on the league’s investment (ROI). This project is designed to enable the league management to identify rookie players who will be worth the long-term investment that the league will be making in these players.

# **Data Understanding**

The dataset is pre-divided into a training dataset of 8000 observations against 21 variables describing different performance statistics of the players, and a test dataset of 3799 counts and 20 variables. The test dataset does not contain the dependent variable column of ‘Target\_5Yrs’, therefore this dataset is only used to produce the prediction for Kaggle submission. The other variables are the same between the training and the test datasets.

Out of the 21 variables, the independent variable ‘Id’ is a player description variable, and not related to the player performance. Therefore this variable is not used for ML model training. Instead, this variable is only used for the test dataset as part of the Kaggle submission.

***Figure 1: training and test data sets***



Figure 1 shows that the dependent variable ‘Target\_5Yrs’ of the dataset shows 6669 counts of positive values i.e. the player lasted longer than 5 years in the league, and 1331 counts of negative values i.e. the player failed to stay in the league longer than 5 years. This means that the dataset is not very balanced which will have implications for the model accuracy since the model won’t be able to learn properly leading to issues like overfitting.

The list below is the provided data glossary highlighting the definitions of the 22 features. The variable “Id\_Old” does not exist on both the train and test datasets and no. 22 “TARGET\_5Yrs” exists in the train dataset only. Hence there are only 21 variables in the training dataset.

***Table 1: Data glossary of the 22 variables of the dataset.***

| **Data Glossary** | | |
| --- | --- | --- |
| **No** | **Column Names** | **Comments** |
| 1 | Id\_old | Previous Player Identifier (Not on datasets) |
| 2 | Id | Player Identifier |
| 3 | GP | Games Played |
| 4 | MIN | Minutes Played |
| 5 | PTS | Points Per Game |
| 6 | FGM | Field Goals Made |
| 7 | FGA | Field Goals Attempts |
| 8 | FG% | Field Goals Percent |
| 9 | 3P Made | 3-Points Made |
| 10 | 3PA | 3-Points Attempts |
| 11 | 3P% | 3-Points Percent |
| 12 | FTM | Free Throw Made |
| 13 | FTA | Free Throw Attempts |
| 14 | FT% | Free Throw Percent |
| 15 | OREB | Offensive Rebounds |
| 16 | DREB | Defensive Rebounds |
| 17 | REB | Rebounds |
| 18 | AST | Assists |
| 19 | STL | Steals |
| 20 | BLK | Blocks |
| 21 | TOV | Turnovers |
| 22 | TARGET\_5Yrs | Outcome: 1 if career length >= 5 years, 0 otherwise |

The individual variables were examined for their individual statistical distributions. Given the range of values for each variable is different, a separate plot is generated for each variable. These boxplots show most variables are biased to the lower end. This is a reasonable feature in this dataset, as these variables describe in-game performance of new players. It is likely that most players will not get much in-game play time in their initial games. Therefore most players will have lower variable values - see Figures 2 and 3 below.



***Figure 2: Set of 20 variables in the training dataset.***



***Figure 3: Set of 19 variables in the test dataset.***

# 

# **Data Preparation**

A few steps were taken to prepare the dataset for modelling. Firstly, since the test set is reserved for kaggle testing, the training dataset needed to be split further into training and validations sets for testing model performance. To facilitate this, the target variable “TARGET\_5Yrs” was extracted from the independent variables in the training dataset first and then the dataset was split as mentioned above. This split the dataset into 4 sub-sets - X\_train, Y\_train, X\_test and Y\_test.

Secondly, a correlation plot was created as shown in Figure 4 to better understand the strength of each feature, particularly against the “TARGET\_5Yrs” variable, to identify any strong association with the predictor variables. This was critical because while correlation between predictors and target variables is useful, correlation between predictor variables can negatively influence the model performance and accuracy.

**

***Figure 4: Correlation graph of 21 variables in the training dataset, including the primary key variable Id.***

The “Id” variable was removed from the training features as it is a primary key identifier for the rookie players and hence has no impact on the model performance itself. When comparing features against “TARGET\_5Yrs” the predictor variables: “3P Made”, “3PA, 3P%”, “FT%”, “AST” and “BLK” had very poor association with the target variable. For other variables, a relatively low correlation value was observed. This suggests that the target variable has a relatively weak relationship to all independent variables in this dataset.

Additionally, all of the variables were numerical and no numerical conversion was required. Considering the bias in the values of these variables, as well as the relatively different range of these values, they were scaled as normal distributions.

Also there are negative values in some of the variables. This can be explained in terms of how these variables are recorded. In other words, for these negative values, the score by the opposite team counts as negative values, or reducing the score contributed by the player. This means that these are real values in terms of the data recording. While removing might have had some impact on the model performance, for the purposes of data preparation, these observations with negative values were not removed from the dataset since that could have reduced the number of observations the model is trained on.

Finally, the relatively large ratio of majority class and minority class required that some form of data balance should be considered. The following approaches are considered in this study:

* No data balance. The difference in the majority class and minority class is handled as stratification in dataset splitting.
* Upsampling the minority class. This duplicates the existing minority class data to bring it to the same size as the majority class so the model can learn and predict better.
* SMOTE. While a simple oversampling just duplicates the existing data which doesn;t any new information for the model to learn, Synthetic Minority Oversampling Technique or SMOTE synthesises new examples for the minority class based on the existing data (Brownlee, 2020).

However, both these techniques (SMOTE and Upsampling) were not implemented in this project since there was a risk that they could inflate the player statistics leading to unreliable model outcomes.

# **Modelling**

## **ML Model Training Pipeline**

The pipeline of ML model training is as follows:

1. Drop the variables Id and Target\_5Yrs from training and test datasets.
2. Normalise independent variables.
3. If needed, generate second order variables.
4. Perform data balance.
5. Perform training dataset splitting to create a validation set.
6. Perform cross validation training, with grid search or randomised search.
7. Evaluate the base ML model against other models.

## **ML Models and Algorithms Considered**

### Logistic Regression Model

As the intention for this study is to predict the probability of whether a player will continue to play after 5 years, this requires a classification model. Logistic regression is used as a first model since it’s the simplest and most common type of binary classification model and it requires a small number of parameters and hyperparameters.

### Logistic Regression Model Feature Importance

Since a correlation map was unable to identify the most relevant independent variables to the target variable, to enhance the outcomes of this logistic regression model, feature importance to the target variable was calculated to better predict the outcomes by focusing on the variable coefficient. This will help identify the predictor variables most relevant to the target variable. Based on the feature importance scores in Figure 5, any variable above 0.000 threshold indicates relevance to the model and hence was included in model training for future iterations, like ‘GP’, ‘Min’ and ‘PTS’. This technique paid off since the best performing model with AUROC score of 0.715 was based on the features selected using their importance scores.

For this model, the coefficients were prioritised, where prediction outcome is based on the weighted sum of the predictor variables. For this to work properly, this technique assumes that the predictor variables have the same scale, hence standard scaler was used to scale the dataset.



***Figure 5: feature importance scores for the logistic regression model***

| **No** | **Selected Predictor Variables chosen** | **ROC\_ROC\_AUC validation score** |
| --- | --- | --- |
| 1 | 4 Features  ['GP','MIN','PTS','FG%']  Kaggle score: 0.70339 | **0.7158555417468881** |
| 2 | 7 Features  ['GP','MIN','PTS','FGA','FG%','REB','OREB']  Kaggle score: 0.70636 | 0.7151900714100861 |
| 3 | 9Features  ['GP','MIN','PTS','FGA','FGM','FG%','REB','FTA','OREB']  Kaggle score: 0.70612 | 0.7135235516777532 |
| 4 | 11 Features  ['DREB','BLK','GP','MIN','PTS','FGA','FGM','FG%','REB','FTA','OREB']  Kaggle score: 0.70591 | 0.7133898888323271 |
| 5 | 10 Features  ['Id','GP','MIN','PTS','FGA','FGM','FG%','REB','FTA','OREB']  Kaggle score: 0.70443 | 0.7127955157537305 |
| 6 | 13 Features  [Id','GP','MIN','FGM','FG%','3PMade','3P%','FTA','FT%','OREB','DREB','AST','BLK'] | 0.7116835546353991 |
| 7 | 3 Features  ['GP','MIN','FGM'] | 0.711289675824 |
| 8 | 5 Features  ['GP','FGM','3P Made','OREB','DREB']  **Best test score in Kaggle 0.71077** | 0.7093259695533102 |

***Table 2: Different tested models with their validation and test performance AUROC scores***

### Random Forest Model

Random forest model is used as a model to provide an alternative way of interpreting the relationship between the independent variables and the target variable. This can consider some interactions between the variables given the use of decision trees. As there are more hyperparameters to train compared to logistic regression models, this is considered after the logistic regression models are compared and interpreted.

### Stochastic Gradient Descent (SGD)

Stochastic gradient descent is an algorithm to optimise the weak classifiers to generate a model by the use of a gradient descent algorithm that implements an stochastic nature. This provides another decision tree based model to consider any interaction between the independent variables.

### XGBoost

Another algorithm based on the weak classifiers to generate a model by the use of gradient boosting method. As this algorithm has the largest number of hyperparameters, this is used after all algorithms have been applied to the dataset.

## Regularisation

All of the above mentioned ML models can consider regularisation as part of the ML model training. This prevents the model from overfitting when the L1 and/or L2 parameter value is set at the optimal value, depending on which regularisation term is included. The regularisation term included in a model training is explained in the corresponding topic in the Scikit learn model class. For most models outlined above, only the L2 term is used by the Scikit learn class.

## Grid Search and Randomised Search

For all of the ML models being considered, grid search strategy was implemented to obtain an optimal set of hyperparameter settings to provide the best ML model. In addition, randomised search was performed using a hyperparameter space that is similar to that used for the grid search as another way of obtaining the optimal set of hyperparameter settings.

# **Evaluation and Results**

## Metric for Model Evaluation

The only metric used for this study is area under receiver operator curve (AUROC). This is because in the case of class imbalance data, while accuracy metrics can be unreliable, AUROC is the most reliable metric of performance since it works with the predicted probabilities instead of the predictions directly. In other words, it focuses on how much the model is able to distinguish between classes.

The highest performer based on Kaggle ROC AUC score is logistic regression with coefficient importance at 0.71077 using these five selected features (GP, FGM, 3P Made, OREB, DREB). This was surprising since this particular model with 5 variables was not the best performing model when tested on the validation dataset.

## ML Model Comparison

When comparing all trained ML models, Logistic Regression models performed consistently better than other ML models. Interesting, all the different models’ performance ranged between an AUC score of 0.70 and 0.72. This suggests that the independent variables in this dataset do not have strong interaction relationships.

Another salient feature is that, in terms of the test dataset performance, the models tend to overfit even with optimised hyperparameters. This is observed when most of the ML models produce a higher AUC score on the validation dataset compared to the score reported by Kaggle using the test dataset. This makes the model selection challenging, as the best model based on the training/validation dataset may not necessarily be the best model for prediction on the test dataset.

# **Conclusion and Recommendations**

To conclude, while a simplified logistic regression model with select features performed the best, most of the trained models have similar AUC scores. Also, given this score range is between 0.70 to 0.72, these ML models don’t seem to be the best prediction models for this data project. As the model hyperparameter optimisation to all algorithms have been implemented in this study, this suggests that this dataset itself is unlikely to produce ML models that provide stronger prediction power.

One aspect that can be considered in a future work is to request professional input from domain experts in the model training process. This can inform on the feature selection with the dataset to suggest the important features of player performance to be considered for the model training. This can also inform on whether new data with different variables to be measured.

While none of the data balancing techniques were leveraged in this project for the reasons stated above, it could be beneficial to apply them in the next iterations for better model performance. These techniques would include SMOTE, ensemble methods and k-fold cross validation.

Furthermore, due to the small size of the data received, it was critical not to lose the observations if possible, and hence some negative values were not removed, as highlighted above. In the next iterations, this will also need to be considered at the risk of losing some data observations if it helps improve model accuracy beyond AUROC score of 0.72. Access to both additional observations and variables could also help improve model performance in the future.

**Reference**

Kennedy, A., 2021, ‘Examining minimum-contract bargains from 2021 NBA offseason’, *BasketballNews,* <<https://www.basketballnews.com/stories/examining-minimum-contract-bargains-from-2021-nba-offseason>>

Brownlee, J. 2020, SMOTE for Imbalanced Classification with Python, <<https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/>>