# A Novel Approach to Predicting NBA Team Wins and Player Salaries Using Player Impact Scores

Ali Chouman, Daniele De Luca, Fangzhe Hu, Oluwadamilola Kadiri, and Mark Keran

***Abstract*—In the NBA, clubs depend on the performance of their teams for financial success. The traditional method of scouting for players to increase the performance of a team has fallen to the wayside to advanced statistics. Using machine learning algorithms and artificial intelligence, we propose a novel method to predict the impact a player will have on a team. In this study, two approaches were taken: the first approach creates an average team based on team-level statistics, and using an MLP regressor, uses the player's impact score to predict the number of team wins in given season. The second approach uses player-level statistics and player impact data to predict a player's annual salary using an RNN architecture. Both approaches report similar results when predicting player win shares and predicted salary. Our approach can aid small market NBA teams with strict budgets in decision making. This work constitutes the culminating project as a requirement for the ECE 9603A course offered at Western University by Dr. Katarina Grolinger.**

## I. Introduction

The financial success of an NBA club relies on the performance of their team. Club managers need to make decisions to increase their team's performance in order to continue performing at a high level. These decisions include extending contracts to players to increase their team's performance. Finding suitable players can be a very costly investment as the wrong player can either hurt the team's performance, or cause no change which can result in a steep financial loss. In the modern age, there is a plethora of statistics and advanced statistics that can aid in the management of buying new players if used correctly. Using machine learning techniques and artificial intelligence, we propose a method that can evaluate a player's performance and predict the impact they will have on the team.

Our method consists of two approaches: the first approach uses a multi-layer perceptron (MLP) regressor to evaluate an individual player's impact on team success. We accomplish this by creating an average team using team-level statistics and evaluating how an individual player affects that average team. This assigns wins to players as a modified win share, which is how much the player contributes to team wins in a season. Depending on how much the player impacts the team, we can then predict the salary this player should be earning in that season. Our second approach uses a recurrent neural network (RNN) using player-level statistics and player impact data to predict the annual salary of a player. The predicted salary is then compared with existing contracts to evaluate the model's accuracy.

Ali Chouman, Daniele De Luca, Fangzhe Hu, Oluwadamilola Kadiri, and Mark Keran are students of the ECE 9603A course at Western University e-mail: {achouman, ddeluca8, fhu43, okadiri, mkeran2}@uwo.ca

**The paper is organized as follows:** Section II outlines the algorithms and metrics used for our model. Section III goes over similar works related to our problem. Section IV is our proposed methodologies. Lastly, Section V is our results and evaluation of our models.

## II. Background

### A. Multi-Layer Perceptron (MLP) Regressor

A perceptron is a neural network link that contains computations to track features in the input data [1]. This network is made up of nodes, or artificial neurons, that contain an input, weights, and output. These neurons sum the weights and pass this sum though a nonlinear function to produce an output. A perceptron is a single-layer neural network used for supervised learning of binary classifiers. [2]. A multi-layer perceptron is a fully connected class of feed forward neural networks (FFNN), consisting of at least three layers of perceptions: an input layer, hidden layer, and output layer [3].

### B. Recurrent Neural Network (RNN)

An RNN is a type of artificial neural network which uses sequential or times series data as opposed to traditional ANNs [4]. RNNs have the concept of "memory" that helps them store the states or information of previous inputs to generate the next output sequence [5]. Connections between the nodes create a cycle, allowing the output from some nodes to affect the subsequent input of the same nodes. Unlike traditional FFNNs, RNNs share the same weight parameters across each network layer. These weights are adjusted through back-propagation through time (BPTT) and gradient descent to facilitate reinforcement learning. BPTT differs from traditional back-propagation algorithms by summing the errors at each time step.

### C. Metrics

*1) R-Squared:* R-Squared ($R^2$ or the coefficient of determination) is a statistical measure in a regression model that determines the proportion of variance in the dependent variable that can be explained by the independent variable. In other words, R-squared shows how well the data fits the regression model (the goodness of fit) [6]. The ideal value for R-squared is 1. The closer the value of R-squared is to 1, the better the fit.

*2) Mean Absolute Error (MAE):* In the context of machine learning, absolute error refers to the magnitude of difference between the prediction of an observation and the true value of that observation. Mean absolute error (MAE) takes the average of absolute errors for a group of predictions and observations as a measurement of the magnitude of errors for the entire group. MAE can also be referred as L1 loss function. MAE helps users to formulate learning problems into optimization problems. It also serves as an easy-to-understand quantifiable measurement of errors for regression problems [7].

## III. Related Work

Feature selection in deep learning models were considered for the objective of predicting team outcomes (wins or losses) in the NBA. It involves engineering new features in NBA analytics, such as the Elo rating, that are typically used in other competitive ranked sports, such as chess [8].

Other features considered include the "Four Factors" for individual player statistics, which are the effective field goal percentage (EFG), turnovers per possession (TPP), free throw rate (FTR), and offensive rebounding percentage (ORP). Baghal considered the Four Factors as reasonable features for predicting outputs such as team success or individual salary through a novel offensive-defensive feature framework [9].

Advanced metrics to evaluate player performance has had rapid integration into sports after its successful implementation into baseball [10]. A commonly used metric in basketball is a player's win share which is a numerical calculation which incorporates basic statistics to evaluate the impact an individual has on winning [11]. By summing up the win shares of all players on a team, this provides an estimate of the total number of wins that the team is expected to have in that season [11]. This methodology is easy to implement, accurate, and has a direct connection to team success by evaluating wins. Its limitations come from approximations of offensive and defensive production. Difficulties arise in producing a numerical value of how productive a player is on the defensive end due to the subjectivity of the matter. If the opposition does not score, there is no definite answer as to whether to place the blame on the offensive player or to credit the defensive player.

Two methodologies [12],[13] focused on soccer player evaluation using both player performance and player popularity using machine learning algorithms. Numerous different metrics for both performance and personality were evaluated to see their correlation to player valuations [13]. An approach outlined in a paper by Deshpande and Jensen [14] estimates an NBA players effect on team wins after accounting for the contributions of his teammates and opponents. It attempts to contextualize the players impact by assigning a low weight to the importance of performance in so called "garbage time" [14] and a high weight to performances in high leveraged points of the game. By combining their method with context agnostic methods like player efficiency rating (PER) they can ascertain if a player who puts up high box score numbers actually improved his team's chances of winning a game. This model [14] uses win probability and Bayesian linear regression to estimate each player's contribution. The method for estimating win probability makes use of the team's lead and the time remaining in the game and utilizes a play-by-play dataset obtained from ESPN to perform their analysis. It is noted that unit changes in the lead affected win probability more than unit changes in time, which introduced a bias against players who are subbed in for defensive possessions and taken out for offensive possessions. They were unable to mitigate this bias due to their inability to determine which team has possession on a second-by-second basis from the play-by-play dataset they use. This approach to estimating player impact does not make use of machine learning techniques and instead uses probabilistic mathematical models, however it highlights an issue in estimating player impact that arises from the difficulty in obtaining relevant data from NBA games.

Alternatively, we attempt to evaluate a player's performance in our first approach through their contribution to expected wins rather than incorporating outside factors which do not correlate to true impact, such as popularity. In addition, instead of looking at how a player affects win probability in a game, we evaluate how a player affects total wins over a season, a which is a better indicator of success. In our second approach we use previous performance and salary figures to provide an update to salary.

## IV. Methodology

### A. Multi-Layer Perceptron Utilizing Average Team

*1) General Overview:* The first approach determines how the success of an NBA team is influenced by team statistics over a season, and then determining how individual players contribute to those stats. The value of a player, expressed in wins, can then be used to determine their worth to the team. An overview of the architecture of this model can be seen in Fig. 1. The next sections will go more in depth about each aspect of this methodology.

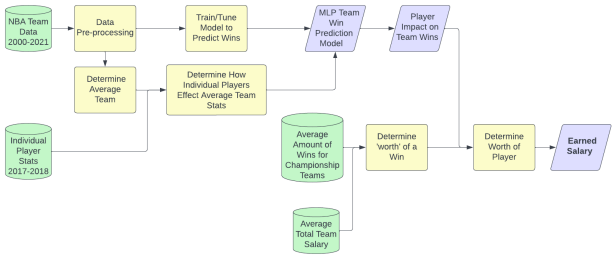


Fig. 1: General architecture of evaluating player performance using an average team

*2) Creating MLP Based Team Model:* An MLP is a robust neural network which can be used for this non-time series and non-image regression application which has multiple inputs and one output. The NBA team data consisted of a multitude of basic statistics which can be utilized directly or incorporated in various advanced statistics. These statistics act as inputs to the MLP to produce a predicted winning percentage. To save on computation and to prevent overfitting, only the statistics that were expected to produce the largest effect on winning were

utilizing. These were chosen to be offensive rating efficiency, defensive rating efficiency, points, true shooting percentage, a modified Player Efficiency Rating (PER), and Pythagorean win expectation [15]. Basketball, like many other major sports, is centred around, and thus values, both offensive and defensive production. Offensive production can be more easily quantified than defensive production and was represented as points scored, offensive rating efficiency, and true shooting percentage. Issues arise with defensive production as evaluating how many points a player is responsible for has some subjectivity to it, as mentioned previously. Defensive rating efficiency is utilized to measure defensive output. In addition, a modified PER which is extended to team metrics, as well as Pythagorean win expectation was utilized to incorporate both offensive and defensive metrics. The offensive ($ORE$) and defensive ($DRE$) efficiencies can be seen in the following equations

$$ORE = \frac{Points_{Scored}}{Possessions} \times 100 \tag{1}$$

$$DRE = \frac{Points_{Allowed}}{Possessions} \times 100 \tag{2}$$

where the amount of team possessions is approximated by

$$Possessions = 0.96 \times (FGA - ORb + TO + 0.44FTA). \tag{3}$$

In this equation *FGA* represents field goal attempts, *ORb* is offensive rebounds, *TO* is turnovers, and *FTA* is free throw attempts. True shooting percentage attempts to approximate the shooting efficiency of a player or team through the following equation

$$TS\% = \frac{Points}{2 \times (FGA + 0.44FTA)} \times 100. \tag{4}$$

PER is a player-based formula with numerous different factors that attempts to evaluate a players overall efficiency on both offense and defense. By replacing individual statistics with team statistics, PER can be extended to a team. Pythagorean win expectation incorporates the total number of points scored for a team and points scored against them to approximate the winning percentage they will have, as seen in the equation below.

$$Win\%_E = \frac{(Points_{Scored})^{\alpha}}{(Points_{Scored})^{\alpha} + (Points_{Allowed})^{\alpha}} \times 100 \tag{5}$$

The value of $\alpha$ is found numerically and does not have a consensus value but is often approximated as 13.9. Although this prediction is fairly accurate and easy to implement, it is difficult to see a player's direct contribution, and thus was not used as the only metric in predicting wins, but instead acting as one of several inputs. The MLP can then use backpropagation to train on the data set to predict winning percentages. All values for both input and output were normalized between 0 and 1. Using a grid search and 5-fold cross validation allows for tuning to identify the hyperparameters needed for a more accurate model. Once the average team is created, the team statistics can be put into the tuned MLP to determine how many wins they are projected to have, which should be around 41 (50% of 82 game season).

*3) Evaluating Player Impact on Average Team:* The total amount of available minutes for players on a team to play in a season is

$$T_{min} = \frac{82games}{season} \times \frac{48minutes_{game}}{game} \times \frac{5minutes_{player}}{minutes_{game}}. \tag{6}$$

since each team will always have 5 players on the court at once. All basic stats can then be represented per player minute. A new team is created for each player which will be evaluated. The team statistic totals will then be used to find the advanced statistics found in (1)-(5) so it can be used to predict total wins as per the initial algorithm. As an example, the total number of points for the created average team after the impact of the evaluated player is taken into consideration is shown in the following equations

$$Points_{Team} = Points_{av} + Points_{Player} \tag{7}$$

where $Points_{Player}$ is the number of points scored by the evaluated player throughout the season. The contribution of the remaining average players is shown below

$$Points_{av} = (T_{min} - Minutes_{Player}) \times \frac{AveragePoints}{playerminute} \tag{8}$$

Statistics such as points, rebounds, assists, steals, and blocks can be evaluated in a similar form as the equations seen above. As mentioned previously, to produce an accurate team-based win prediction model, it was deemed necessary to incorporate metrics which include the number of points allowed by a team. It is not a trivial task to determine how individual players affect this, so various approaches were tested, and the following produced the most accurate results. The number of net points allowed by a player is expressed as follows

$$Points_{net} = Points_{Player} - (PointsAllowed)_{Player} + Points_{Poss.Change} \tag{9}$$

The $Points_{Player}$ is the same statistic which was utilized in (7) and simply refers to the total number of points scored in a season by that individual player. The $PointsAllowed_{Player}$ initially approximates all players as an average defender as per the following equation

$$(PointsAllowed)_{Player} = \frac{AveragePoints}{playerminute} \times Minutes_{Player}. \tag{10}$$

This represents the number of points an average player would score if they played the same number of minutes as the player being evaluated. Since, there is a 1:1 ratio of offensive players to defensive players since both teams will always have five players on the court, this is also the number of points an average defender will allow in the specified time frame. Since it is not accurate to assume all players can be represented as an average defender, the $Points_{Poss.Change}$ term attempts to remedy this.

$$Points_{Poss.Change} = (\frac{Points}{Basket})_{league} \times (Blocks_{Player} + Steals_{Player} - Turnovers_{Player} - \frac{Fouls_{Player}}{2}) \times (TrueShooting\%)_{league} \tag{11}$$

Equation 11 approximates the player's defensive impact through the number of possession changes. Each block and steal by the player results in a loss of possession by the opponent which consequently leads to an offensive possession for your team. On the other hand, each turnover and foul will lead to a possession for your opponent. Since, there is approximately a 1:1 ratio of free throws to fouls, and each free throw is worth a single point, the number of fouls is divided by 2. The positive attributes of blocks and steals subtracted by turnover and fouls represents the number of net possession changes that the evaluated player directly causes. The remaining portion of (11) attempts to see how many points for your team (positive value) or for the other team (negative value) these possessions contribute to. The net possessions multiplied by league average trueshooting percentage is approximately how many of those possessions will results in a successful possession where any player scores. The league average points per basket is a weighted average of the number of two-point and three-point field goals to approximate the number of points per successful possession. This is a simplified approach since blocks, steals, turnover, and fouls are not the only ways a player can affect a defensive possession. Additional metrics such as how often they contest the shot of the opponent, the level of difficulty that they make the opponent's shot, and how often they let the opposition run past them are all important metrics which affect how productive a player is on the defensive end. This data is difficult to quantify as it has both intra-rater and inter-rater reliability issues and it is also difficult to find data sets which contain this information.

*4) Practical Meaning of Player Impact Value:* Once the new team statistics are created for each player which is to be evaluated, the MLP can be used to predict the new number of wins for each created team. By subtracting the win total from the number of wins expected for an average team (approximately 41), the number of more, or less, wins a team would have if they used the evaluated player rather than an average player is produced. Thus, if the player impact scores determined by this algorithm are summed for all players on a team and this value is added to the average number of wins (approximately 41), it should approximate the total number of wins the team had that season as seen in the equation below.

$$(PredictedWins)_i = \sum_{j=1}^{N} (PlayerImpact)_j + (AverageWins) \quad (12)$$

Where $i$ represents the index for each team in the NBA and $j$ is for each player on a team with $N$ different players. By comparing this data to the actual team win data in the 2017-2018 season allows for accuracy comparisons. To ensure the average predicted winning percentage is around 50% for the 30 NBA teams, the following equation is used.

$$(\hat{PredictedWins})_i = (PredictedWins)_i - \frac{\sum_{p=1}^{30} (PredictedWins)_p}{30} + (AverageWins) \quad (13)$$

*5) Interpreting Player Impact in Terms of Salary:* The NBA championship teams from 2005-2019 had an average regular season winning percentage of approximately 73.7% or just over 60 wins [16]. The average salary of an NBA team in the 2017-2018 season was approximately $111 million [16]. Thus, the average team can spend $1.85 million for each win to be considered a championship caliber team. The salary earned by each NBA player can then be expressed as

$$(EarnedSalary)_i = (AverageSalary)_{league} + (PlayerImpact)_i \times \frac{Cost}{Win} \quad (14)$$

In other words, in the 2017-2018 season, if a player contributes 1 extra win when compared to an average player, they should be paid $1.85 million more than the average salary. There are additional restrictions to salary, such as minimum and maximum salaries available; however, the results of (14) can show the true worth of an individual player to their team.

### B. Recurrent Neural Network

The second approach uses individual statistics to update salary figures based off of past and present performance and contracts. The idea of Recurrent Neural Networks (RNNs) using memory in its recurrent sequencing inspired and justified their use in this scenario, where player statistics across multiple seasons should impact the prediction of the player's salary beyond the current season (2017-2018 in this study).

## V. Results and Evaluation

A set of Python libraries were imported into the project source file of a *condas* distribution, and an equivalent Jupyter notebook was also generated to display all imports. *Seaborn* and *pandas* were used for the visualizations and plots that are mentioned and shown in exploratory data analysis. *Sklearn* provided the metrics and measures of accuracy and loss, along with dataset splitting, cross validation, principle component analysis, and feature scaling.

### A. Multi-Layer Perceptron Regression

*1) Tuning the Model:* Hyperparameter tuning for this approach was done using GridSearchCV. GridSearchCV is a function available in Scikit-Learn's model_selection package. The function loops through predefined hyperparameter sets and fits the model on the training set based on the optimal set of hyperparameter values.

To pass predefined values to the grid search, a dictionary was used, where the key:value pairs are the hyperparameter names for the keys and the values they can take. Figure 2 shows the dictionary definition which highlights the hyperparameters tuned in the model. These were the number of neurons and hidden layers, learning rate, and activation function.

Five-Fold cross-validation was used to evaluate each hyperparameter set using the *cv* parameter on GridSearchCV. The evaluation metrics selected for the model were the coefficient of determination or $R^2$ and the mean absolute error, MAE. The

```
search_space = {
    "hidden_layer_sizes" : [(100, 75, 50, 25, 13, 5,),
      (50, 40, 30, 20, 10, 5,), (100, 50, 25,), (30, 20, 10,), (100,), (50,)],
    "learning_rate_init" : [0.001, 0.01, 0.1, 1],
    "activation" : ['logistic', 'tanh', 'relu']
}
```

Fig. 2: Dictionary of hyperparameters tuned in the MLP model

optimal set of hyperparameter values were determined by the set with the highest $R^2$ value. After tuning, the grid search found the optimal hyperparameter values to be as follows: Number of layers = 3, Number of neurons in each layer = (100, 50, 25), Learning rate = 0.01, Max iterations = 100, Activation function = Rectified Linear Unit (ReLU). The MLP was trained and tuned using data team data except from the 2017-2018 season.

The MAE or Mean Absolute Error is calculated by taking the sum of the absolute difference between the actual values and predicted values. The MAE was used to plot a loss curve for the tuned model over time for training and validation sets. Figure 3 displays the loss function produced below. From the loss function we observe no overfitting or underfitting of the model. The model also displays a good learning rate with the optimal set of hyperparameters. We observe the model converges after 32 iterations.

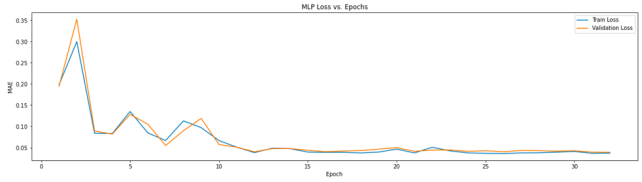


Fig. 3: Loss vs Epochs of MLP Model with optimal hyperparameter set

To access the MLP model's ability to predict wins, The $R^2$ value of the tuned MLP model was compared to an $R^2$ value of the Pythagorean win expectation model [17]. Table I shows the results of this comparison. The MLP model outperforms the Pythagorean win expectation model in terms of $R^2$.

TABLE I: $R^2$ values for different approaches to predicting team wins in a season

| Model | Coefficient of Determination |
|---|---|
| MLP model (untuned) | 0.8707 |
| MLP model (tuned hyperparameters) | 0.9301 |
| Pythagorean Win Expectation model | 0.9180 |

*2) Evaluating Player Impact Score:* The model was then used to predict the number of wins that a team constructed of 15 average players would accumulate in a regular season. Since every NBA game has a winner and loser and a regular season consists of an 82-game schedule, the average number of wins in a season is 41 games. Thus, it is expected that a team comprising of 15 average players will be 41 games. The tuned model accurately predicts that the fictional average team constructed will win 41 games.

Each NBA player in the 2017/18 season is evaluated and assigned a player impact score. Their impact score is calculated by substituting an average player from the average team with the player to be evaluated. The model is used again to predict a new number of wins for the average team. The difference between the number of wins the average team has with our evaluated player on the roster and the average number of team wins will be the player impact score or 'modified win-share'. Summing up the player impact score values of players on one NBA team and adding the average number of wins should add up to the total number of wins the team had that season. To assess the accuracy of the player impact score, we compare the wins predicted to those predicted when you sum up the already established 'Win-share' [11] values of each player on an NBA team. This is performed for each NBA team and the $R^2$ values from each approach are compared. Table II shows the results of this comparison. Win-share metric outperforms the Player Impact Score Metric in terms of $R^2$. This is expected due to the difficulties in estimating a players' defensive impact on the game. These difficulties were highlighted in Section IV-A3 of this report. To visualize the Player impact score predictions, a plot of the predicted wins, actual wins, and absolute difference between predicted and actual is shown in Fig. 4. We see from the figure that the absolute difference mostly stays below 10 wins. Although it is currently not as accurate as the Win-share metric , these results are promising for future improvements. This approach should still accurately show player impact of more offensively oriented players.

TABLE II: Player Impact Score vs Win Shares accuracy comparison

| Model | Coefficient of Determination |
|---|---|
| Actual Win Shares | 0.9132 |
| Proposed Player Impact (modified win share) | 0.6012 |

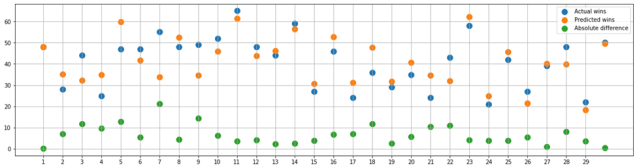


Fig. 4: Absolute Difference of Actual Wins and Predicted Wins using sum of Player Impact Score for all 30 NBA teams in 2017/18 season

Another useful metric for analyzing the result of this approach is the MVP (Most Valuable Player) rankings for an NBA season. The MVP for an NBA season is selected by a panel of sportswriters, experts and broadcasters throughout the United States and Canada. The MVP votes are based on different factors, the stats of the player and their perceived impact on team success throughout the year. Since the introduction of media into MVP voting the narrative of the player's season also plays a factor in MVP voting. The top 15 MVP ranking for the 2017-18 season [16] is compared to the top ten performers using the player impact score in Table III. We see a strong correlation between the two. Nine out of ten players appear in both rankings.

TABLE III: Comparing the top 10 projected earners with the NBA MVP rankings

| Player | MLP Predicted Salary (million USD) | MVP ranking |
|---|---|---|
| Anthony Davis | 51.77 | 3 |
| Giannis Antetokounmpo | 41.47 | 6 |
| LaMarcus Aldridge | 39.73 | 9 |
| Kevin Durant | 39.14 | 7 |
| James Harden | 38.46 | 1 |
| Lebron James | 36.7 | 2 |
| Damian Lillard | 33.48 | 4 |
| Stephen Curry | 33.45 | 10 |
| Victor Oladipo | 33.23 | 13 |
| Kyrie Irving | 31.2 | N/A |

### *B. Recurrent Neural Network*

*Kerastuner* was used to provide a hyperparameter tuning method with a defined search space and objective. The objective is set to measure the mean absolute value metric for loss and the search algorithm employs a maximum number of three trials. The hyperparameters tuned in the model were the following: the number of units/neurons in each layer, the number of layers, the dropout rate for added LSTM layers, the choice of activation function, and the learning rate of the optimizer. Multiple search algorithms, such as randomized search and Hyperband, were implemented, however the comparison of all results show the use of the grid search algorithm. Figure 5 displays the results of the search for the indicated hyperparameters. The tuning process can be demonstrated

```
"Search space summary Default search space size: 5
num_layers (Int) {'default': None, 'conditions': [],
'min_value': 1, 'max_value': 3, 'step': 1,
'sampling': None} units_0 (Int) {'default': None,
'conditions': [], 'min_value': 32, 'max_value': 512,
'step': 32, 'sampling': None} activation (Choice)
{'default': 'relu', 'conditions': [], 'values':
['relu', 'tanh'], 'ordered': False} dropout (Boolean)
{'default': False, 'conditions': []} lr (Float)
{'default': 0.0001, 'conditions': [], 'min_value':
0.0001, 'max_value': 0.01, 'step': None, 'sampling':
'log'} "
```

Fig. 5: Hyperparameter Search Space Summary

either through the summary results provided in the source code, which shows the change in model loss based on the searched learning rate within the tuner. The selected learning rate coincides with the minimal loss that the figure shows for the increasing values of the learning rate.

Table IV demonstrates the obtained results of each model type by comparing predicted salary to actual salary. The initial model with the preset configured values and Simple RNN architecture demonstrates a mediocre fit with a hefty loss. These values were obtained despite the grid search algorithm being employed to exhaustively search all possible values in the hyperparameter search space. The source code was run using the randomized search algorithm, although, for the purpose of value convergence with faster performance. The tuned RNN model shows an improvement in the model's fit while minimizing the loss and this is acceptable within the problem's bounds. One important note is that the model's

TABLE IV: RNN Model Fit and Loss Values by Model Type

| Model Type | Correlation Coefficient, $R^2$ | Mean Absolute Error (MAE) |
|---|---|---|
| Initial RNN Model | 0.5203 | 5714.4062 |
| Tuned RNN Model | 0.6496 | 5065.0655 |
| Tuned RNN Model with Dimensionality Reduction | 0.6678 | 4963.2553 |

correlation coefficient should be compared to an optimization model for salary prediction with the salary cap bound for each NBA team. This proposes an excellent avenue of future considerations for this project work. The tuned RNN model using the PCA components, or dimensionality reduction on the features, produced a slightly better fit with minimal loss and the improvement can also be seen in performance time for training the neural network.

## VI. Conclusion

By combining advanced statistics in basketball and modern machine learning methods together, we analyzed NBA stats from multiple dimensions to attempt to relate a player's salary to their true impact. Evaluating a modified win share to determine a new salary and creating a new salary based off of past performance and contracts were successfully implemented. This can provide quantitative advice to NBA teams about the salary of a player, and aid small-market NBA teams with strict budgets.

## References


[1] M. Banoula, "What is perceptron? a beginners guide [updated]: Simplilearn," Nov 2022. [Online]. Available: https://www.simplilearn.com/tutorials/deep-learning-tutorial/perceptron

[2] DeepAI, "Perceptron," May 2019. [Online]. Available: https://deepai.org/machine-learning-glossary-and-terms/perceptron

[3] "Multilayer perceptron," Sep 2022. [Online]. Available: https://en.wikipedia.org/wiki/Multilayer_perceptron

[4] I. C. Education, "What are recurrent neural networks?" [Online]. Available: https://www.ibm.com/cloud/learn/recurrent-neural-networks

[5] M. Saeed, "An introduction to recurrent neural networks and the math that powers them," Nov 2022. [Online]. Available: https://machinelearningmastery.com/an-introduction-to-recurrent-neural-networks-and-the-math-that-powers-them/

[6] "R-squared," Nov 2022. [Online]. Available: https://corporatefinanceinstitute.com/resources/data-science/r-squared/

[7] "Mean absolute error," Mar 2022. [Online]. Available: https://c3.ai/glossary/data-science/mean-absolute-error/

[8] M. Migliorati, "Features selection in nba outcome prediction through deep learning," *arXiv preprint arXiv:2111.09695*, 2021.

[9] T. Baghal, "Are the" four factors" indicators of one factor? an application of structural equation modeling methodology to nba data in prediction of winning percentage," *Journal of Quantitative Analysis in Sports*, vol. 8, no. 1, 2012.

[10] C. R. Link and S. L. Rubin, "Moneyball after 10 years," *Journal of Sports Economics*, vol. 18, no. 8, pp. 771–786, 2015.

[11] Basketball, "Nba win shares," accessed 5-December-2022. [Online]. Available: https://www.basketball-reference.com/about/ws.html

[12] M. A. Al-Asadi and S. Tasdemir, "Predict the value of football players using fifa video game data and machine learning techniques," *IEEE Access*, vol. 10, pp. 22 631–22 645, 2022.

[13] J. Hofmann, O. Schnittka, M. Johnen, and P. Kottemann, "Talent or popularity: What drives market value and brand image for human brands?" *Journal of Business Research*, vol. 124, pp. 748–758, 2021.

[14] S. K. Deshpande and S. T. Jensen, "Estimating an nba player's impact on his team's chances of winning," *Journal of Quantitative Analysis in Sports*, vol. 12, no. 2, pp. 51–72, 2016.

[15] J. Hollinger, "Pro basketball prospectus," dulles, VA: Brassey's, 2004.

[16] Basketball, ""basketball statistics & history of every team & nba and wnba players"," accessed 5-December-2022. [Online]. Available: https://www.basketball-reference.com/.

[17] E. H. Kaplan and C. Rich, "Decomposing pythagoras," *Journal of Quantitative Analysis in Sports*, vol. 13, no. 4, 2017.