

# Fantasy Basketball Prediction Using Machine Learning

Ismail Ocak  
University of California  
San Diego  
iocak@ucsd.edu

## Abstract

*The growing power of the internet created many new industries after the 1990s. This phenomenon also created a new area called Fantasy Basketball. On a given night, users draft real basketball players and earn points based on the real performances of these drafts. In this multi-billion-dollar industry, making accurate predictions for player performances or fantasy points is crucial. This project aims to use machine learning for this purpose. In this project, we used different machine learning models in feature extraction, feature selection and, prediction processes. Such models include, XGBoost[13], RandomForest[15], AdaBoost[14], Artificial Neural Networks, Linear Regression and Lasso[11]. Comparing the performances of these different models we found that XGBoost can be successfully used for feature selection and prediction processes in the Fantasy Basketball prediction area. Additionally, we developed a feature extraction method that optimizes the weights of moving average features using Linear Regression.*

## 1. Introduction

Since their inception, Draftkings[1] and Fanduel[2] have changed the fantasy sports landscape. On a given night, millions of dollars of prizes are awarded to users who drafted a team for competitions lasting no longer than one night. Given the vast number of possible lineups and large variance associated with a player's single performance, finding the optimal lineup can be equated to playing the lottery. According to Draftkings, 14% of the player pool can turn a profit, which comes as no surprise considering the website rakes 10% of each competition prize pool. This fact agrees with the research of Smith et al. since their research suggest that decisions of the fantasy players are based on their biases and beliefs rather than being on statistical analyses[3]. Thus, the boom in sporting analytics in a post-Moneyball[4] world offers the opportunity for statisticians and data scientists alike the opportunity to profit on the access to rich data and machine learning methods.

### 1.1. What is fantasy basketball?

Fantasy basketball is an online game where users act like their fantasy teams' general managers. On each game day, fantasy users draft real basketball players and earn points based on their real performances. At the end of each game day, month or season users are ranked based on their total points and earn prizes. As it sounds straightforward, there are certain limitations for the fantasy users. Users need to draft basketball players for each position, they also need to have players as substitutions. In total, users draft 8 players and the required positions are as follows:

Count	Position
1	PG
1	SG
1	SF
1	PF
1	C
1	G (PG, SG)
1	F (SF, PF)
1	Util (PG, SG, SF, PF, C)

Table 1.

Also, each basketball player has an artificial salary that represents his value and the fantasy users have budget limitations. Thus, they cannot simply draft the best player. As a result, this problem turns into an optimization problem and predicting future performances of the real basketball players becomes more important.







LINEUP		Avg. Rem./Player: \$0		Rem. Salary: \$600	
POS	PLAYER	OFF	FPPG	SALARY	
PG	J. Harden 	HOU @ NY	59.9	\$10,500	✗
SG	A. Iguodala 	MIL @ MIA	14.1	\$3,000	✗
SF	B. Bogdanovic 	UTA @ CLE	30.1	\$5,900	✗
PF	Kevin Love 	UTA @ CLE	36.5	\$7,000	✗
C	E. Ilyasova	MIL @ MIA	16.1	\$3,600	✗
G	R. Westbrook 	HOU @ NY	51.7	\$10,000	✗
F	Cedi Osman	UTA @ CLE	21.0	\$3,900	✗
UTIL	G. Dragic 	MIL @ MIA	29.2	\$5,500	✗

Figure 1. An example lineup

## 1.2. How are fantasy points calculated?

Fantasy point is basically an indicator of a player's performance in a given game. A high fantasy point score means the player played well and he was productive in that game. The metric is calculated by using a weighted average of some performance statistics. Below are the weights:

**Fantasy Point Scoring**

Point	+1 Pt
Made 3pt Shot	+0.5 Pts
Rebound	+1.25 Pts
Assist	+1.5 Pts
Steal	+2 Pts
Block	+2 Pts
Turnover	-0.5 pts
Double-Double (Max 1 Per Player: Points, Rebounds, Assists, Blocks, Steals)	+1.5 Pts
Triple-Double (Max 1 Per Player: Points, Rebounds, Assists, Blocks, Steals)	+3 Pts

Table 2.

For example, if a basketball player scores 20 points and makes 10 assists he gets  $(20 \times 1) + (10 \times 1.5) + 1.5 = 36.5$  points.

## 2. Description of the Project

### 2.1. Problem and Motivation

In fantasy sports, many users base their lineup drafts on their intuitions. However, this may not always be a good idea since many factors are involved. For example, a player might have performed very well in the recent games but this does not mean that the same player will perform well again in the next game. There are many different factors like opposition's defensive abilities, player's rest and so on. There are infinitely many features that we could consider and handling all of these using human intuition seems impossible.

For example, it is not uncommon that users pick their drafts based on recent basketball player performances. If we choose to use recent performance metrics as predictors following Mean Absolute Value and Root Mean Squared Errors are obtained.

Method	MAE	RMSE
Fantasy Points in the Previous Game	9.394253	12.20495
Avg. Fantasy Points in the Last 5 Games	7.620981	9.854436
Machine Learning Solutions	6.996447	8.959265

Table 3.

As can be seen in Table 3, using the fantasy points of a player from the last game or using the average of his last

5 games' fantasy points results in higher error than machine learning models. Note that, the errors for the machine learning solutions mentioned here were obtained from an intermediate step of this project. These intermediate results prove that machine learning models can be more successful than baseline methods. Thus, we can use machine learning to improve Fantasy Basketball user decisions.

Hence, in this project, our goal is to predict a basketball player's fantasy points in a given NBA game. In other words: Predicting a basketball player's performance

### 2.2. The Data

Basketball is one of the most famous sports both in the world and in the USA. However, it is almost impossible to find a ready to use dataset that satisfies the requirements of this project. On the other hand, the NBA is well known for having properly recorded statistics and information. It is possible to find many data sources on the internet but a system that transforms the data into a usable form was required.

The website basketball-reference.com[5] offers detailed player summaries for every player in the NBA for games dating back to the 1940s. We have scraped the game logs for each player from basketball-reference to gather relevant nightly scoring summaries using the "basketball-reference-web-scraper" Python package[6]. We also scraped betting data from betexplorer.com[16] and obtained daily betting lines that represent the relative powers of the teams playing in the NBA. Also, using rotoguru.net[17] we scraped daily fantasy salary data of the basketball players. Finally, we compiled these sources into a dataset. The raw data consisted of columns such as date, player name, team, opponent, assists, fantasy points, betting odds, fantasy salary, etc. There are many columns that were not mentioned here but all of these columns are indicators of a player's performance on a given game night.



# LeBron James 2018-19 Game Log

LeBron Raymone James • [Twitter: KingJames](#)  
(King James, LBJ, Chosen One, Bron-Bron, The Little Emperor, The Akron Hammer, L-Train)  
Position: Shooting Guard and Point Guard and Small Forward and Power Forward • Shoots: Right  
6-9, 250lb (206cm, 113kg)

More bio, uniform, draft, salary info ▼

2018-19 Regular Season

Share & more ▼

Glossary

Rk	G	Date	Age	Tm	Opp	GS	MP	FG	FGA	FG%	3P	3PA	3P%	FT	FTA	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PTS	
1	1	2018-10-16	33-292	LAL	@ POR	L (-9)	1	37:20	9	16	.563	0	4	.000	8	9	.889	2	10	12	6	1	0	6	3	26
2	2	2018-10-20	33-294	LAL	HOU	L (-9)	1	38:52	9	22	.409	1	7	.143	5	7	.714	0	5	5	5	2	1	1	0	24
3	3	2018-10-22	33-296	LAL	SAS	L (-1)	1	42:52	11	25	.440	2	8	.250	8	11	.727	2	6	8	14	1	1	5	2	32
4	4	2018-10-24	33-298	LAL	@ PHO	W (+18)	1	28:04	7	16	.438	1	5	.200	4	4	1.000	0	7	7	10	3	0	0	1	19
5	5	2018-10-25	33-299	LAL	DET	W (+7)	1	34:40	10	17	.588	1	5	.200	7	10	.700	0	11	11	11	3	0	2	3	28

Figure 2. Example basketball-reference data

After creating the dataset, we conducted a preliminary analysis of our data. The average fantasy points of the basketball players between 2014 and 2020 is 20.64 with a standard deviation of 14.03. The minimum is -1.5 whereas the

maximum is 103.5. As a result of this analysis, some outlier values were found. An outlier elimination process was implemented, the top 0.2% fantasy point rows were eliminated since these cases are thought to be outliers. Similarly, the bottom 0.5% which are negative values were also deleted. Since our dataset is a combination of different data sources some columns include corrupted-missing data. Such records were also imputed.

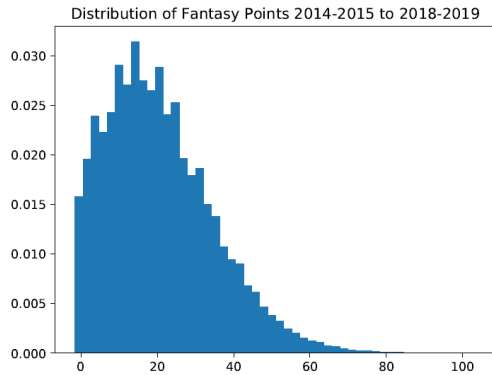


Figure 3. Distribution of Fantasy Points, Raw Data

## 2.3. Methods

In this project, a continuous improvement flow was followed. Trying different approaches at different steps, we followed the outline in Figure 4 and repeated it continuously to achieve better results.

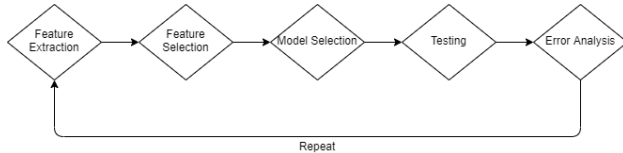


Figure 4. Project Pipeline

Although these steps and methods will be covered in detail in the next section, we will give a general idea in this section.

### 2.3.1 Feature Extraction

As mentioned in the prior sections, lag features (performance statistics from the previous games) are thought to be important in fantasy points prediction. Starting with this assumption we extracted some lag features such as:

A basketball player's fantasy points in the previous game
A basketball player's avg. assists in the last 5 games
Per minute avg. points in a basketball player's career
Total fantasy points of the team in the previous game
Avg. count of blocks of the opponent in the last 3 games

Table 4.

We also had other features like odds from betting websites or the number of days passed since the player's last game. One problem with lag and past moving average features is determining the optimal weights. For example, to predict a basketball player's fantasy points in the upcoming game we can use his fantasy points from the previous game. However, we can use his fantasy points from the game before or the average of his last fantasy points. We can also do exhaustive feature extraction and extract all such features. On the other hand, this might result in multicollinearity and cause our models to perform worse.

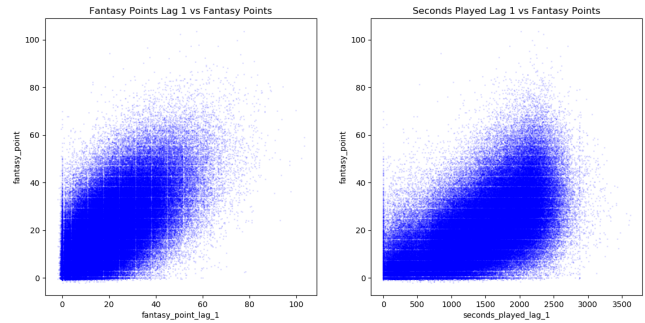


Figure 5. Lag Features vs Fantasy Points

Based on our analyses it was clear that lag features had a relationship with the fantasy points. However, there were infinitely many lag features that we could have used and this led us to the idea of finding an optimal ensemble of lags for each feature.

After running multiple experiments on the dataset and trying many different features, we came up with a simple but novel approach where we used linear regression to optimize the weights in the moving average (rolling mean) features.

### 2.3.2 Feature Selection

After the feature extraction step we ended up with over 500 features. Feeding all of these features to machine learning models could work but we realized that issues such as multicollinearity or curse of dimensionality might increase our error rate. In order to understand the problem we first ran the pipeline in Figure 4 using all of the features. Then, we used Principal Component Analysis (PCA) to reduce the number of features. Since PCA was an unsupervised

method we finally tested a supervised dimensionality reduction method using the feature importance of XGBoost.

### 2.3.3 Model Selection

There were some naive baseline models presented in section 2.1. The main idea of these baseline models is to see this problem as a univariate time-series modeling problem. People generally make their decisions using these types of simple analyses. However, in reality, this problem is more complicated than this.

To solve this problem we used different machine learning models on our data. These models are Linear Regression, LASSO[11], XGBoost[13], Adaboost[14], Random Forest[15] and Artificial Neural Networks.

Complex models require a process called hyperparameter tuning since they have many different parameters. The key idea of tuning is to find the best parameters using the training data. After we carefully tune the parameters the models will be ready to use in the testing phase.

A common approach in tuning is to use k-fold cross-validation. Many machine learning studies use k-fold cross-validation to apply a grid search on training data. For example, Random Forest model has a parameter called ‘ntree’ which stands for the number of trees in the model. Using ‘ntree’ = [100, 200, 300], a 5-fold cross-validation fits 5 models for each value, takes the average error and picks the best model. However, using the k-fold cross-validation approach is not suitable for our problem since the dataset has the date dimension. Using a randomized k-fold cross validation will result in data leakage which might potentially harm the validity of our experiments. As a matter of fact, we should not use the data from the future in the training set or validation set. One potential problem in tuning, we may end up with undesired parameters since the data leakage spoiled the tuning process.

As an alternative to k-fold cross-validation, we used Walk-Forward validation to prevent data leakage issues. Similar to k-fold, Walk-Forward validation splits the data into folds. However, it follows a time-series splitting approach. The illustration below explains the logic:

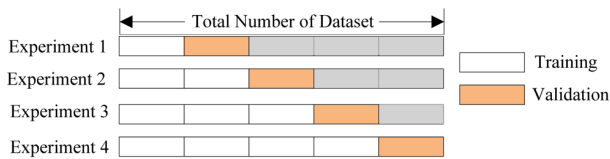


Figure 6. Walk Forward Cross Validation

In this project, the best model was found to be XGBoost in terms of error metrics. The methodology will be covered in the following sections. As stated by Chen et al. in their original XGBoost paper[13], XGBoost uses the following

loss function where  $l(\cdot)$  is a differentiable convex loss function. Also, predictions are made using an ensemble of  $K$  regression trees. The formulas used in the model are as follows:

$$\bullet \mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k)$$

$$\text{where } \Omega(f) = \gamma T + \frac{1}{2} \lambda \|w\|^2$$

$$\bullet \hat{y}_i = \Phi(x_i) = \sum_{k=1}^K f_k(x_i), f_k \in F$$

### 2.3.4 Testing

After finding the best hyperparameters for each model, we tested our models using a walk-forward approach again. Basketball seasons are often defined as [starting year, ending year]. For example, when we talk about the 2015-2016 season, it is the season ending in 2016. While testing different models we first trained the models using the data from seasons ending in 2015 and 2016 and tested in 2017. Then, using the season ending in 2015, 2016, 2017 we trained our models and tested the models in 2018. Finally, we tested our models in 2019. The results are explained in the Experiments section.

### 2.3.5 Error Analysis

Using our testing outcomes (predictions) we calculated different error metrics and compared the error metrics of different models. In the existing literature Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) are heavily used as performance metrics. Thus, we used the same metrics as well.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2}$$

$$MAE = \frac{1}{n} \sum_{i=1}^n |\hat{y}_i - y_i|$$

## 3. Experiments

All of the experiments in this project were conducted in accordance with the 2.3 Methods section. If a potential improvement was made at a particular step of the project pipeline, then the pipeline steps were completed until the last step. Finally, the error metrics were compared to the previous findings.

### 3.1. Modeling with All Features

After completing the feature extraction that was presented in section 2.3.1, we trained models using all features. The aim was to find an initial best model. The feature importance information of this best model could be used in feature selection. Since it constitutes a baseline model and creates a starting point for model-feature selection, this step was crucial for our project.

The candidate models were Linear Regression, Lasso, XGBoost, Adaboost, Random Forest and Artificial Neural Networks. Except for Linear Regression, all the other models have model parameters that require hyperparameter tuning. We carefully conducted hyperparameter tuning on the training data. The training data consisted of fantasy basketball data from the seasons ending 2015 and 2016. As stated before, a walking forward validation approach was followed in accordance with time series splitting. After tuning the models and determining the best parameters for the models, we tested the models in 2017 data. Then, we trained the models in 2015, 2016, 2017 and tested them on 2018 data. Similarly, we also tested the model on 2019 data as well. After calculating MAE and RMSE for every year, we calculated the averages for each model. Based on our result XGBoost was the best model both in terms of MAE and RMSE.

Model	MAE	RMSE
adaboost	7.525	9.446
neural_networks	7.325	9.246
random_forest	7.143	9.105
linear_regression	7.117	9.037
lasso	7.085	9.033
xgboost	7.075	9.022

Table 5. Average Test Errors, All Features

### 3.2. Modeling with PCA Features

As mentioned in Section 2.3.2, we tested the Principal Component Analysis method in feature extraction. We started with more than 500 features and some of them were correlated with each other. In order to reduce the number of features, we applied PCA on the data and chose first 100 features which explained 90% of the total variation.

After getting the principal components, we followed the steps in our pipeline again. We tuned the models, trained them with the final parameters, made predictions on test data and compared the errors. The errors were higher than the previous experiment step where we used all features. Although PCA reduces the number of dimensions, it is an unsupervised method and it may lose some important information which is related to the predicted variable. Since PCA resulted in higher errors we decided to look for a supervised dimensionality reduction technique.

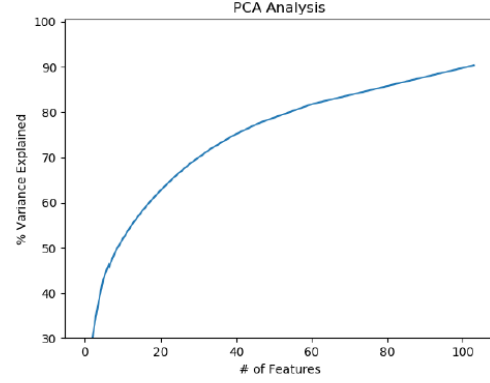


Figure 7. No. Features vs Variance Explained with PCA

Model	MAE	RMSE
adaboost	7.829	9.832
neural_networks	7.629	9.632
random_forest	7.279	9.325
linear_regression	7.111	9.095
lasso	7.088	9.042
xgboost	7.216	9.231

Table 6. Average Test Errors, PCA Features

### 3.3. Modeling with XGBoost Features

XGBoost was proven to perform better than other models when we used all features. Thus, we decided to use XGBoost model in feature selection as well and test the performance. In order to do this, we have taken the XGBoost model that performed best in section 3.1 and inspected its feature importance. XGBoost model has a method called 'f-score'. This score basically stands for the feature importance and it shows how many times each feature is split on. Thus, a higher 'f-score' means that particular feature is important. 160 out of 500 features consisted 90% of the total 'f-score'. Thus we chose to use these 160 features.

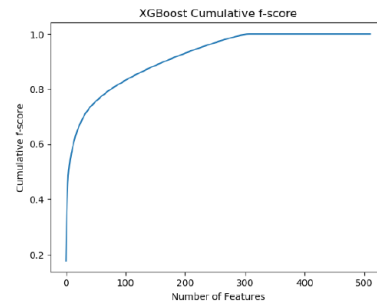


Figure 8. XGBoost f-score

After following the steps in the project pipeline we calculated average test errors again. Feature selection with XGBoost decreased the average error compared to using all features.



index	col_name	f_score	f_score_cumsum
0	fantasy_point_lag_1_rollmean_4	0.177106	0.17710562
1	fantasy_point_lag_1_rollmean_5	0.126957	0.30406213
2	fantasy_point_lag_1_rollmean_3	0.083938	0.388
3	seconds_played_lag_1_rollmean_5	0.065712	0.45371187
4	attempted_field_goals_lag_1_rollmean_4	0.034229	0.48794082
5	seconds_played_lag_1_rollmean_4	0.020798	0.50873905
6	seconds_played_lag_1_rollmean_3	0.01632	0.52505857
7	attempted_field_goals_lag_1_rollmean_5	0.016266	0.5413246
8	player_ly_total_attempted_free_throws	0.012505	0.55382925
9	player_ly_total_turnovers	0.012405	0.5662339

Figure 9. Top 10 important features

Model	MAE	RMSE
adaboost	7.512	9.425
neural_networks	7.312	9.225
random_forest	7.156	9.137
linear_regression	7.083	9.002
lasso	7.022	8.977
xgboost	6.997	8.933

Table 7. Average Test Errors, XGBoost Features

After these 3 steps, XGBoost was found to be the best approach for dimensionality reduction and the best model for prediction. As a result, in the next steps of this project, we used XGBoost both for feature selection and prediction.

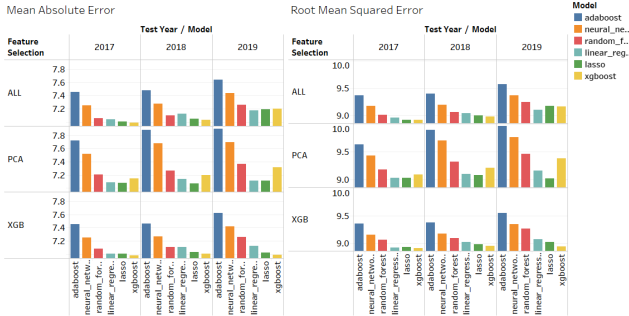


Figure 10. Errors Summary

### 3.4. Additional Features

The existing literature[7, 8, 9, 12] used some additional features like a player's fantasy salary or a game's line-up information. We also decided to use these features as well. Repeating the project pipeline with only XGBoost we have seen a significant decrease in error metrics. Using these features average test MAE dropped from 6.997 to 6.917, RMSE dropped from 8.933 to 8.861.

### 3.5. Optimized Moving Average Features

As mentioned in section 2.3.1 Feature Extraction we used the lag and moving average features from the past. In our experiments, the data from the previous games was named as lag1, lag2, lag3 features. Similarly, we also used simple rolling mean functions that stand for the moving average of lag features. For example, if a player earned 10,

15, 35 fantasy points in the last 3 games then 'fantasy point lag 1' is 35, 'lag 2' is 15, 'lag 3' is 10. Similarly, the feature 'fantasy point lag 1 roll mean 3' is  $(10 + 15 + 35)/3 = 20$ . We also created these features for other statistics such as assists, points scored, free throws scored, blocks, steals, etc.

As can be seen, one can produce numerous features using this approach. There is no limitation to use the average of the last 3 or 10 games. However, as can be seen in Figure 5 there is a clear positive correlation with lag 1 features and this correlation decreases as we move to higher lag features. Also, as can be seen in Figure 9 which shows the top 10 important features in XGBoost with initial features, fantasy point rolling mean (moving average) features are the most important features. Based on f-scores 'fantasy point lag 1 rollmean 4' is more important than 'rollmean 5' and 3. However, they are all on the top of the list. This fact gives an important clue about the most important features but it also points out a possible problem point. 8 out of 10 the most important features are 'rolling mean' features but most of them are correlated with each other. This problem of multicollinearity might affect the models negatively and increase error rates.

Simple moving average basically takes the average of last  $n$  records and calculates their average. In this process all the weights are equal. However, based on our preliminary analyses it is was clear that recent records have more predictive power than the older records. In order to establish more powerful features and to get rid of multicollinearity, we have decided to implement a simple but effective solution.

In order to come up with 'super-features' we decided to use linear regression as a weight-optimized predictor. In this process, we treated each feature separately and independently. Using every single column from our raw data one-by-one, we considered the last 30 records as different features and modeled them to predict the next fantasy points. As a result of this analysis, we found the weights and later used these weights to create 'super-features'. In order to prevent data leakage, we only used the data before 2015 to determine these weights. Otherwise, we would be inserting some future testing information to our training data and this might have disrupted the experiments. In order to give a better idea of the steps followed in this method we can use the following example:

1. Use the data before 2015 as the training data
2. Filter a single statistic and the target variable (fantasy points)
3. Create lag features from 1 to 30 for the filtered statistic.
4. Fit a linear model to these 30 lag features (predicted variable is still fantasy points)

- Find the weights for the 30 lag features
- Apply these weights to the data after 2015, for a single statistic create one 'super-feature' using the past 30 records.
- Repeat these steps separately for every statistic in our data. (assists, rebounds, blocks, etc.)

We used this approach to every single statistic in our dataset. Some of these statistics are 'assists', 'attempted field goals', 'attempted free throws', 'attempted three point field goals' and 'fantasy point' itself. For example, the weights of the first 5 features for the lagged fantasy points were 0.202, 0.137, 0.099, 0.0786, 0.060. As can be seen, lag 1 features have the largest weight and the weights decrease as we go the older records. This also intuitively makes sense since the recent performance of a player is a better predictor for his future performance.

After applying this method to our dataset, we created 19 new features but these new features had more predictive power than the initial features. Also, these features were the outcomes of a prediction process. Thus, they can be seen as predictions as well and using all of these predictions in a new model can be considered as an ensemble of different models.

Using these new features, we repeated the processes in our pipeline again. Since XGBoost was found to be the best model initially, we only repeated these processes for XGBoost. In the hyper-parameter tuning phase, we conducted a grid search on the candidate parameters using walk-forward cross validation approach. These parameters are:

Name	Candidates	Winner
learning_rate	[0.005,0.01,0.05,0.3,0.5,0.9]	[0.005]
max_depth	[3,5,7,10]	[5]
min_child_weight	[1,5,10]	[10]
subsample	[0.5,0.7,1]	[0.7]
colsample_bytree	[0.5,0.7,1]	[0.7]
n_estimators	[100,200,500,1000]	[1000]
reg_lambda	[0.3,0.5,0.9]	[0.3]
alpha	[0.3,0.5,0.9]	[0.9]
eval_metric	[rmse,mae]	[rmse]
booster	[gblinear,gbtree]	[gbtree]

Table 8. Final Tuning of XGBoost Model

As can be seen from the top 10 important features, 'weighted moving average predictor features' (wma features) have become the most important features and the importance of the old features decreased. Prior to these new features 'fantasy point lag 1 roll mean 4' was the most important feature with an 'f-score' 0.177, the new score of this feature is below 0.01 and related wma feature is the second most important feature.

	col_name	f_score	f_score_cumsum
0	predictor_double_count_wma_30	0.23516862	0.23516862
1	predictor_fantasy_point_wma_30	0.098937996	0.33410662
2	predictor_seconds_played_wma_30	0.078694135	0.41280076
3	salary_edited	0.062688604	0.47548938
4	predictor_points_scored_wma_30	0.031943016	0.5074324
5	starter_yes	0.024839206	0.5322716
6	player_ly_total_turnovers	0.018901298	0.5511729
7	starter_no	0.01741007	0.568583
8	predictor_made_field_goals_wma_30	0.014991085	0.5835741
9	fantasy_point_lag_1_rollmean_4	0.008324629	0.59189874
10	predictor_attempted_free_throws_wma_30	0.008153693	0.6000524

Figure 11. Feature Importance, XGBoost

As a result, using 'Optimized Moving Average Features' we managed to decrease the test MAE from 6.917 to 6.852, and RMSE from 8.861 to 8.761. Also, using the feature importance, we can conclude that we prevented multicollinearity and decreased the mutual importance of correlated features. Using this method, we created smart and optimized lag features that have more predictive power.

## 4. Conclusion

In this project, we aimed to predict fantasy points of the real basketball players on a given game and presented an effective solution for this. From different candidate models, XGBoost was found to be the best one both for Feature Selection and Modeling-Prediction. Also, for the smart feature extraction part, we presented a new method called 'Optimized Moving Average Features'. Below is the progress of MAE and RMSE related to the steps we completed in this project:

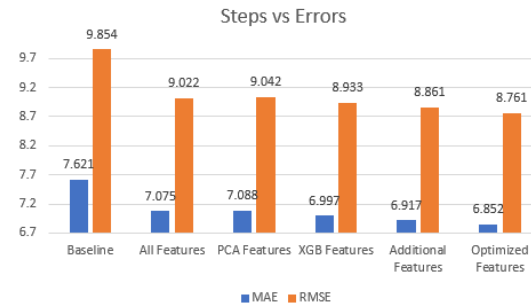


Figure 12. Steps

### 4.1. Findings

- Based on the experiments, XGBoost is the best model for both feature selection and prediction.
- Since PCA is an unsupervised technique it may lose some important information. It is not desirable in the context of feature selection.
- The existing literature does not have systematic dimensionality reduction techniques in this field. We have shown that XGBoost is a desirable method for this.

- MAE improvement from 7.621 to 6.852 is significant with a p-value less than 0.01 based on the t-statistic:

$$t = \frac{(MAE_{XGB} - MAE_{Baseline}) - (0 - 0)}{\sqrt{\sigma_{XGB}^2/n + \sigma_{Baseline}^2/n}} = 22.24$$

We have similar results in RMSE as well. This concludes that our improvements compared to the baseline model are statistically significant.

- Based on the error metrics they provided in their papers, our final MAE and RMSE are higher than the errors (6.603, 8.541) provided in the study of Barry et al. [7]. Also, the errors we achieved in this paper are lower than the errors in Arao’s study (6.8486, 8.9581). However, such comparisons are not completely true since these studies do not reveal their testing data and its size. We tested our models in 3 different seasons using 71200 records. Thus, our testing procedure uses a high number of records.
- The errors of LASSO and Linear regression are worse than XGBoost. However, they still perform better than other machine learning models. Also, their errors are close to XGBoost errors. As we presented in section 2.3.1 there is a linear relation and positive correlation between some features and the predicted variable. Based on the results, linear models such as LASSO are able to catch these kinds of relationships and tend to perform satisfactorily. Since these models are easy to interpret and fast, they can be used in the Fantasy Basketball area as well.

## 4.2. Novelty

- The existing literature [7, 8, 9, 12] in the Fantasy Basketball area is limited and the majority of the studies are projects that are available online. The use of XGBoost was not a new approach in Fantasy basketball area. According to Kengo Arao’s studies [12] XGBoost can be used in the Fantasy Basketball Prediction area successfully. However, feature selection with XGBoost is a novel approach in the area. Our studies prove that XGBoost can be successfully used in dimensionality reduction as well.
- The method ‘Optimized Moving Average Features’ is a totally new and unique approach in Fantasy Basketball. Using this approach we optimized and automated the feature extraction process. Instead of a feature extraction process that relies solely on industry knowledge, we proposed a simple but effective machine learning approach. In their studies, Barry et al. suggested using weighted features as well. However, the weights they used are based on intuitions and industry knowledge.

- The importance of cross-validation in tuning is an undisputed fact. However, the existing literature does not focus on Walk-Forward validation using time-series splitting. Walk-forward validation is an approach where we could apply to sports forecasting, demand forecasting, and the areas where the time dimension is important. In this project, we successfully used this approach to tune the model hyperparameters.

## 4.3. Further Studies

As mentioned in earlier sections, the aim of the Fantasy Basketball users is to pick the best players that will earn the most fantasy players. However, since every player has a salary and there is a budget limitation for the fantasy line-up one cannot choose all of the best players. As a result, an optimization process is required to pick the optimal line-up based on the predicted points and player salaries. We can optimize this problem using a simple solver and a knapsack-like problem approach.

## References

- [1] “Daily Fantasy Sports For Cash.” *DraftKings*, draftkings.com.
- [2] “Daily Fantasy Football, MLB, NBA, NHL Leagues for Cash.” *FanDuel*, fanduel.com.
- [3] Smith, Brian, et al. “Decision Making in Online Fantasy Sports Communities.” *Interactive Technology and Smart Education*, vol. 3, no. 4, 2006, pp. 347–360., DOI:10.1108/17415650680000072.
- [4] Miller, Bennett, director. *Moneyball*. Universal, 2011.
- [5] “Basketball Statistics and History.” *Basketball*, www.basketball-reference.com/.
- [6] “Basketball-Reference-Web-Scraper.” *PyPI*, pypi.org/project/basketball-reference-web-scraper/.
- [7] Barry, et al. *Beating DraftKings at Daily Fantasy Sports*.
- [8] Hermann, Eric. “Machine Learning Applications in Fantasy Basketball.” (2015).
- [9] Shivakumar, Shreyas. *Learning to Turn Fantasy Basketball Into Real Money Introduction to Machine Learning*. shreyasskandan.github.io/.
- [10] Hoerl, Arthur E., and Robert W. Kennard. “Ridge Regression: Biased Estimation for Nonorthogonal Problems.” *Technometrics* 42 (2000): 80-86.



- [11] Tibshirani, Robert. “Regression Shrinkage and Selection Via the Lasso.” *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, 1996, pp. 267–288., doi:10.1111/j.2517-6161.1996.tb02080.x.
- [12] Arao, Kengo. NBA Player Performance Prediction and Lineup Optimization. [github.com/KengoA/fantasy-basketball](https://github.com/KengoA/fantasy-basketball).
- [13] Chen, Tianqi, and Carlos Guestrin. “XGBoost.” *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD 16*, 2016, doi:10.1145/2939672.2939785.
- [14] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *Machine Learning: Proceedings of the Thirteenth International Conference*, pages 148–156, 1996
- [15] Breiman, Leo (2001). “Random Forests”. *Machine Learning*. 45 (1): 5–32. doi:10.1023/A:1010933404324.
- [16] “Betexplorer.” *BetExplorer Soccer Stats - Results, Tables, Soccer Stats and Odds*, [www.betexplorer.com/](http://www.betexplorer.com/).
- [17] “Daily Blurbs.” *RotoGuru*, [www.rotoguru.net/](http://www.rotoguru.net/).