COMPREHENSIVE ANALYSIS OF THE FUTURE PRICE OF NBA TOP SHOT MOMENTS

By

Miguel A. Esteban Diaz, B.S. in Accounting & Finance

A thesis submitted to the Graduate Committee of

Ramapo College of New Jersey in partial fulfillment

of the requirements for the degree of

Master of Science in Data Science

Spring, 2022

Committee Members:

Osei Tweneboah, Advisor

Scott Frees, Reader

Nikhil Varma, Reader

COPYRIGHT

Table of Contents

List of Tables

List of Figures

Abstract

NBA Top Shot moments are NFTs built on the FLOW blockchain and created by Dapper Labs in collaboration with the NBA. These NFTs, commonly referred to as “moments”, consist of in-game highlights of an NBA or WNBA player. Using the different variables of a moment, like for example: the type of play done by the player appearing in the moment (dunk, assist, block, etc.), the number of listings of that moment in the marketplace, whether the player appearing in the moment is a rookie or the rarity tier of the moment (Common, Fandom, Rare or Legendary). This project aims to provide a statistical analysis that could yield hidden correlations of the characteristics of a moment and its price, and a prediction of the price of moments with the use of machine learning regression models which include linear regression, random forest or neural networks. As NFTs, and especially NBA Top Shot, are a relatively recent area of research, at the moment there is not extensive research performed about this area. This research has an intent to expand the up to date analysis and research performed in this topic and serve as a foundation for any future research in this area, as well as provide helpful and practical information about the valuation of moments, the importance of the diverse characteristics of moments and impact in the pricing of the moments and the future possible application of this information to other similar highlight-oriented sport NFTs like NFL AllDay or UFC Strike, which are designed similarly to NBA Top Shot.

Chapter 1: Introduction

In today's world, the development of technology is one of the most important areas of interest. With the recent growth of the cryptocurrency industry, one of its components has gained more and more popularity, which is the blockchain.

As defined by IBM, "blockchain is a shared, immutable ledger that facilitates the process of recording transactions and tracking assets in a business network". *(IBM, n.d.)* With the establishment of the blockchain appeared the creation of NFTs (Non-Fungible Tokens). NFTs are tokens that we can use to represent ownership of unique items. They let us tokenize things like art, collectibles, even real estate. They can only have one official owner at a time and they are secured by the blockchain – no one can modify the record of ownership or copy/paste a new NFT into existence. *(Ethereum, n.d.*). The first NFT in history was Quantum, which was created on May 2nd in 2014 on the Namecoin blockchain. *(Hyperallergic, 2021)* Quantum initiated the slow development of the NFT environment over the years, with the widely adoption of the Ethereum blockchain most recently to host most of the NFTs, although currently there are other popular blockchains used to create and host NFTs, like the Solana blockchain, the Flow Blockchain, the Binance Smart Chain blockchain or the Wax blockchain. *(Pixelplex, 2021)*.

One of the current most popular NFTs, with a market capitalization close to $600M, is NBA TopShot. NBA TopShot, which is built on the Flow blockchain, is a non-fungible token (NFT)

marketplace in beta phase where fans can buy, sell and trade NBA moments, which are packaged highlight clips that operate like trading cards. *(SportsPro Media, 2021)* These collectibles, known as “Moments”, include a highlight video of a layup, jump-shot, block, assist, steal, handles, 3 pointer or dunk from a player’s NBA or WNBA game and also include a serial number for that specific moment. These collectibles have been referred to as the modern version of trading cards.

The findings provided by this research could benefit more than 700,000 active users of the NBA TopsShot platform, and NFT owners/potential owners interested in learning more about one of the most popular NFT platforms.

Among current users, this information would benefit both collectors and investors due to the knowledge that this research could give them in the scope of moment valuation. Providing this information to every current or potential user is challenging because of the lack of research resources, due to the recent creation of this industry, the periodical influx of new moments into this NFT market and, the fluctuation of moment prices due to the benefits of ownership of some certain moments, as these can give you access to receiving new moments or other rewards awarded by NBA TopShot. The key of this project is being successful at keeping these variables under control to be able to understand the valuation of the moments.

Therefore, the specific deliverable of this research would be the in-depth exploration of the valuation of the NBA TopShot moments. This research would include the exploratory data analysis of the features to find relationships correlated with the price of the moment and

prediction of value of moments in the future. These objectives will be completed with the use of Tableau, Python and packages such as pandas or matplotlib.py to effectively treat, explore and analyze the data, the use of machine learning models to accurately predict future prices of the moments and the application of statistics to perform data analysis.

Regarding the ethical aspects of this research, there are minimal negative effects around this project. Current and potential users would mainly benefit from the outcomes of this research as it will enhance and solidify their knowledge of this NFT collectible. Another stakeholder benefiting from this research would be any individual looking to perform similar research related to a different NFT project, as the results found in this project could help as a foundation for future research. The major harm this research could produce is the misleading interpretation from the readers of the results, which could produce financial losses after acquiring this NFT.

This research will include a background, which includes the objective of this thesis and the previous work performed in the research of this topic, a methodology chapter, where the design and description of the data will be addressed, the analysis and discussion chapter, which will address the results obtained in the methodology and the worked performed in the light of the background provided, and the conclusion chapter, where the final contributions to the body of knowledge are shared and where the future work that could follow up the results of this research will be stated.

Chapter 2: Background

Chapter 2 of the thesis provides a background of previous research performed about this topic, as well as the background of the performance metrics used and the machine learning models applied.

Due to its recent creation, October 2020, and delayed extensive adoption by the consumers until March 2021, this is an area which lacks in-depth research. There are currently no scholarly applications about this topic, and most of the small research done is found in GitHub. This thesis aims to expand the research and analysis done previously about NBA Topshot moments.
One of the analyses found in GitHub was done by Riccardo Del Chin, a Internet of Things, Big Data & Machine Learning student at Universita degli Studi di Udine. This study was performed in June 2021 and the objective of Riccardo Del Chin's analysis was to study if the skill of the player appearing in the moment, the rarity and mint size of moments and the results of NBA games could have a direct impact on the price of a moment.
To study if any of these three variables had an impact on the price of a moment, Riccardo Del Chin started by analyzing a player's efficiency using a dataset that contained the in-game statistics of each player of the NBA during the 2020-2021 season. He calculated the efficiency of a player using a simplified version of John Hollinger's Player Efficiency Rating, also known as PER. This method aims to calculate the productivity per minute of a player, using in-game statistics like 3-pointers, assists, blocks, turnovers, personal fouls, while taking in account a

team's pace and possession time, and disregarding the total amount of minutes played *(ESPN, 2007).* According to Riccardo Del Chin, using the Top 25 players with the highest PER, the results showed for Common moments, even though not consistently, the player's skill did somewhat affect the price of moments, with a tendency of a higher moment's price when the PER was higher and a lower moment's price when the PER was lower.

Next, Riccardo Del Chin studied if the team's standing and the final score's point difference of the moment's game had an impact on the price of the moment. Using the average price of Common moments of each team, the results showed that the position of a team in the standings was not relevant to the price of the moment, and the higher or lower price was derived from the player's playing in that team. His results regarding the score difference of a moment's game also showed that when the point difference of a moment's game was higher, the price of that moment was somewhat lower.

Lastly, Riccardo Del Chin analyzed if the rarity, the number of listings and the total number of serial numbers of a moment had an impact on the price of the moment. Using boxplots he found that Legendary moments tended to be the moments with highest price among the Tiers, Common moments tended to have the lowest price and Rare moments tended to have a value between the Common and Legendary price range. Using scatter plots, the study showed that the moments with higher total serial numbers had a significantly lower price compared to moments with a low total serial numbers. Also, listings had an impact on the price of a moment. Those moments with higher number of listings tended to have a lower price than moments with a lower number of listings.

As a summary of this analysis, Riccardo Del Chin's research found that higher prices for moments are related to a higher in-game efficiency of a player, the point difference of the moment's game did not have a relevant impact in price of the moment, a team's position in the league's standings did not affect the price of the moments of players in that team, and that the rarity, scarcity and listings had an impact in the price of moments.

This research provides a good foundation for future studies but as it was completed in June 2021, there are many variables currently available for analysis that did not exist in June 2021 or that were not included in Riccardo Del Chin's research. This includes the study of the Fandom Tier, the analysis of moments taking in account badges of the moments and the moments belonging to challenge rewards, the type of play in the moment, and the Set and Series a moment belongs to. We believe all these variables missing in the analysis have an important impact on the price of moments.

In order to predict the price of NBA Top Shot moments, predictive modeling was applied to the marketplace data gathered. As the target variable, Price, is a continuous variable, we decided to explore different regression models to predict the price of a moment using the rest of features in the dataset. To measure the success of the models we will use the following metrics:

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

Mean Absolute Error (MAE) is an error metric that calculates the mean of the absolute values of the individual prediction errors on over all instances in the test set where, y_i = individual

prediction of sample i, $\hat{y}_i$ = test value of sample i and N = total number of samples. *(Encyclopedia of Machine Learning, 2011*).

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Mean Square Error (MSE) calculates the mean of the squared difference of the test data and the predicted data where, y_i = individual prediction for sample i, $\hat{y}_i$ = test value of sample i and N = total number of samples (*Encyclopedia of Machine Learning, 2011).*

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

Root Mean Squared Error (RMSE) is an error metric that applies the square root to the Mean Squared Error where, y_i = individual prediction for sample i, $\hat{y}_i$ = test value of sample i and N = total number of samples. *(Encyclopedia of Research Design, 2010)*

$$R^2 = 1 - \frac{SS_{Regression}}{SS_{Total}} = 1 - \frac{\Sigma_i(y_i - \hat{y}_i)^2}{\Sigma_i(y_i - \bar{y})^2}$$

R-Squared (R^2) is used to determine the proportion of variance in the dependent variable that can be explained by the independent variable where, y_i = individual prediction for sample i, $\hat{y}_i$ = test

value for sample i and $\bar{y}$ = is the mean of y *(CFI, n.d.)*.

In this research, a total of nineteen machine learning models were applied to predict the price of moments, which are the following:

Multiple Linear Regression

This model uses the relationship between the dependent variable and independent variables to create a graph with data-points to draw a straight line through all of them, using this line to make a prediction on the target variable (W3Schools, n.d.). This model is a variation of the Linear Regression where, instead of making a prediction using one predictor variable, more than one predictor variable is used to predict the target variable.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p + \epsilon, \qquad (1)$$

Where *Y* represents the predicted value, *X* represents the predictor variables, β is the regression coefficient, *p* is the number of distinct variables and ∈ represents the model's error. The regression coefficient is an unknown value and in order to implement this formula, it is estimated using the method of least squares. This estimation is completed using the regression coefficient to reduce the residual sum of squares. Calculating the regression coefficient in multiple linear is a complex process and therefore software is used to compute it.

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

$$= \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_{i1} - \hat{\beta}_2 x_{i2} - \ldots - \hat{\beta}_p x_{ip})^2 \qquad (2)$$

In this research, the default hyperparameters provided by scikit-learn were used to train the Multiple Linear Regression model.

Different variations of the Multiple Linear Regression were as well explored in this research. The three variations were: Ridge Regression, Lasso Regression and Elastic Net.

Ridge Regression.

This variation of linear regression is a model which performs shrinking to the regression coefficient, with the aim to bring the value of the regression coefficient as close as possible to 0, which makes this model very similar to the residual sum of squares. The method used by ridge regression to minimize the regression coefficient is using the parameter λ. This parameter controls the shrinkage penalty and must be 0 or higher. When λ is equal to 0, there is no shrinkage penalty, but as λ is increased the shrinkage penalty will grow and the regression coefficient will be minimized.

This model was applied using the default hyperparameters provided by scikit-learn except the value alpha which was set to 415.

Lasso Regression

Lasso Regression was the second variation of Linear regression studied. This model is very similar to Ridge Regression. This model also performs shrinking to the regression coefficient with the help of the λ parameter. In the case of Lasso, when some of the λ values are large

enough, the model will achieve to have some regression coefficients be equal to 0. As a result Lasso regression is able to perform variable selection, and exclude variables.

In this research, Lasso Regression was trained using the default parameters of the library scikit-learn with the exception of the value of alpha which was set to 5.

Elastic Net

This model is the third variation of Linear Regression included in this research. Elastic Net consists of a combination of two models previously discussed, Ridge Regression and Lasso Regression. The aim of this model is to regulate the variable exclusion performed by the Lasso Regression model. The Elastic Net model was implemented in this research using the default parameters of the library scikit-learn with the exception of the hyperparameter random_state which was set to 42.

Decision Trees

This regression model uses a flowchart like a tree structure to show the predictions that result from a series of feature-based splits. It starts with a root node at the top of the tree and ends with a decision made at the terminal nodes, also known as leaves, which represent the target variable's value. Also, all the nodes between the root node and the terminal are known as internal nodes. These internal nodes contain the predictor variables that split into different nodes based on the value of the predictor variable. All the nodes from the decision tree are connected by branches. (Analytics Vidhya, 2021). The Decision Trees model was implemented in this research using the default parameters of the library scikit-learn with the exception of the hyperparameter random_state which was set to 42.

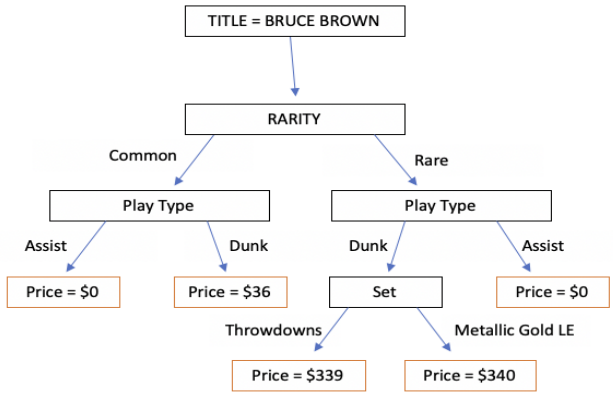


Figure 2.1: Simplified decision tree example of NBA player, Bruce Brown

Bagging

Bagging is a machine learning model that applies the ensemble method. An ensemble method is a technique that consists of using models that would not perform well by themselves, also known as weak learners, and combining them to obtain a stronger model. Bagging is an ensemble method as it combines the Decision Trees model and the Bootstrapping method. Bootstrapping consists of selecting random samples from the training data, which are selected with replacement; every time one sample is chosen, it is possible that the same sample can be chosen again. After all the different groups of data samples are collected they are trained separately, and then all the individual predictions are aggregated. The purpose of Bootstrapping is to decrease the variance of the model which is done through averaging a set of observations. The drawback of this model is that using this model makes it harder to identify the most important predictor variables to predict the target variable. (IBM Cloud Education, n.d.). In this research the Bagging model was applied using the default hyperparameters provided by the scikit-learn library except for the following hyperparameters that were modified: n_estimators = 80 and random_state = 42.

Random Forest

Random Forest is another model that makes use of the ensemble method. As performed in the Bagging model, Random Forest contains different decision trees that had the Bootstrap method applied. The difference of this model is that when the Random Trees model is building its decision trees, when splits happen not all predictor variables are included as a possible node, only a certain group of predictor variables are chosen as possible nodes after a split has been performed. Normally, the amount of possible predictor variables to be used is equal to the root square of the total number of predictor variables. By not selecting all the predictor variables after a split, the strongest predictor variables are not always chosen, as it happens in Bagging. Therefore, as this model is not able to use the strongest predictors at the splits, this creates more uncorrelated trees which helps decrease the variance of the results. (IBM Cloud Education, 2020). The Random Forest model was studied in this research using the default hyperparameters of the scikit-learn library except for the hyperparameters, number_estimators which was set to 285 and random_state which was set to 42.

Boosting

Boosting is another machine learning model that makes use of the ensemble method. Like in the Bagging model, decision trees are used, but the main difference is how these decision trees are created. In Bagging, as previously discussed, decision trees were created using the Bootstrapping method, but in the case of Boosting, each tree is created using the information gathered from the previous tree which leads to using an altered version of the original data.

In this research, two different variations of the Boosting model were used: AdaBoost and

Gradient Boosting Regressor.

AdaBoost

AdaBoost is an ensemble method variation of the Boosting model which makes use of single-level decision trees, also known as decision stumps. In this model, incorrect predictions made by the decision trees are given a high weight, in order to be successfully predicted in the subsequent single-level decision trees. (Brownlee, 2021). The AdaBoost model included in this research used the default hyperparameters included in the scikit-learn library except for n_estimators which was set to 100 and random_state which was set to 42.

Gradient Boosting Regressor.

This is the second type of Boosting model used in this research. Gradient Boosting Regressor is an ensemble model that combines the AdaBoost model and the Gradient Descent model. This model contains decision trees that are larger compared to the decision trees included in AdaBoost. Also, the Gradient Boosting Regressor contains a loss function to minimize the error of the model. The main difference between AdaBoost and Gradient Boosting Regressor is how the subsequent decision trees are added. Gradient Boosting Regressor makes use of the Gradient Descent procedure which helps minimize the errors gradually, for example by updating the weights after each split (Brownlee, 2020). The hyperparameters used for this model were the default hyperparameters set by the library scikit-learn except for n_estimators which was set to 61, the loss which was set to huber and the random_state which was set to 42.

Support Vector Machine

Support Vector Machine model is a supervised machine learning algorithm that predicts the

values of a target variable by using a hyperplane of as many dimensions as features the dataset contains. The values of the predictor variables are represented as data points in each of the dimensions of the hyperplane. In this research, it was analyzed the Regression variation of the Support Vector Machine model. This variation used was Support Vector Regression, also known as SVR. As part of this research, two alterations of the Support Vector Regression model were studied as well, Nu Support Vector Regression (Nu-SVR) and Linear Support Vector Regression (Linear SVR).

Support Vector Regression (SVR)

Support Vector Regression is a Support Vector Machine model used for Regression problems, where the target variable that is predicted is a continuous or discrete value. The objective of using Support Vector Regression is to be able to state how much error is acceptable in the model and adjust accordingly the hyperplane. This maximum error is handled by setting an absolute error smaller or equal to a specified margin, which is referred as ε. By altering the value of ε we will be able to obtain the most optimal error for the model (Sharp, 2020). The Support Vector Regression model included in this research used the default hyperparameters set by the scikit-learn library except for the hyperparameter C which was set to 5.

Nu Support Vector Regression (nu-SVR)

This model is a variation of the Support Vector Regression model that includes a new parameter called nu. The parameter nu is used to control the number of support vectors, by setting the proportion of the number of support vectors kept in the solution with respect to the total number of samples in the dataset. Using parameter nu, the epsilon parameter, which was previously addressed in the Support Vector Regression, is then estimated automatically. (Langhammer &

Česák, 2016). The hyperparameters used in the Nu Support Vector Regression model were the default hyperparameters provided by the scikit-learn library except nu which was set to 0.9 and C which was set 5.

Linear Support Vector Regression (Linear SVR)

A model very similar to SVR with the difference that the parameter used for the kernel is "linear". Using this approach the data points in the hyperplanes are assumed to have a linear relationship and therefore, the regression coefficients are calculated by minimizing the sum of the squared errors.This parameter is implemented using the library liblinear instead of libsvm, in order to have more flexibility choosing penalties and loss functions and known to scale better samples of large size (Paisitkriangkrai, 2012) The Linear SVR model studied included the default hyperparameters of the scikit-learn library except for the hyperparameter epsilon which was set to 3.

Stochastic Gradient Descent (SGD) Regressor.

The Stochastic Gradient Descent model consists of a variation of the Gradient Descent technique. This machine learning model, like Gradient Descent, uses an approximate and iterative method for mathematical optimization used to find a local minimum, but instead of using the whole dataset, the Stochastic Gradient Descent model uses a random sample of the data. (Real Python, 2021). This machine learning model was applied using the default hyperparameters from the scikit-learn library except for the hyperparameter penalty which was set to "elasticnet".

K-Neighbors Regressor

K-Neighbors Regressor. This is a machine learning model consisting of storing all available cases and predicting the target feature based on the similarity of the predictor features of the samples. To calculate the similarity of the predictor features, K-Neighbors Regression uses a distance function. For regression problems, there are two well known distance functions that can be chosen, the Euclidean distance function and the Manhattan distance function.

$$\text{Euclidean } \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2} \tag{3}$$

$$\text{Manhattan } \sum_{i=1}^{k} |x_i - y_i| \tag{4}$$

In Eqs. (3) and (4) are displayed the mathematical formulas for these two methods of calculating the similarity of the predictor variables. x and y represent the data points of the sample's predictor variables, meanwhile i represents the sample, and k represents the number of similar data points calculated. (Analytics Vidhya. (2020). The hyperparameters used in this model were the default hyperparameters provided by the scikit-learn library except for n_neighbors which was set to 30.

Gaussian Process Regressor

Gaussian Process Regressor. This is a non parametric model that uses a Bayesian approach, it transforms the predictor variables into data points and creates different functions that fit the data points and calculates the probability of the functions. As more data points are added, the function

predicted by the model is updated. The update of the function is calculated using the conditional probability function.

$$p(f|y_i) = \frac{p(f,yi)}{p(yi)} \quad (5)$$

Eq.(5) is used after each new data point is added to calculate the probability of each function. In this formula, *f* represents the current function, *y* represents the added point and i represents the sample. After calculating all the possible functions, an average of all the functions created is calculated and a range of the functions where all the data points would be found is given, that will provide the most optimal predictions (Wang, n.d.). This Gaussian Process Regression model was studied using the default parameters of the scikit-learn library except for the hyperparameter random_state which was set to 42.

Partial Least Squares (PLS)

Partial Least Squares (PLS) Regression. This is a supervised machine learning model that applies the dimension reduction method. This model reduces the predictor features to a smaller set of predictor features, and then the smaller set is then applied with least squares regression. Weights are used in those predictor variables that have the strongest relation to the target variable. PLS Regression's purpose is to provide an explanation of the direction of the predictor variables and the target variable (James, Witten, Hastie & Tibshirani, n.d.). The hyperparameters applied to this model were the default hyperparameters of the scikit-learn library except for n_components which was set to 3.

Multi-Layer Perceptron (MLP) Regressor

Multi-Layer Perceptron (MLP) Regressor. This model is a neural networks algorithm which contains an input layer, where the data is introduced to the model, an output layer, where the predictions are given and , in between, hidden layers. These hidden layers contain neurons which use any activation function, although ReLu is the most commonly used activation function. Activation functions are mathematical formulas that are used to decide which neurons are activated in the neural network and which ones are not by performing a weighted-sum of the input. Another element of Multi-layer Perceptrons is backpropagation. This model processes the data from the input layer to the output layer and trains the model through back propagation. Back propagation is a practice commonly used in neural networks which involves tuning the weights of a model based on the error rate of the previous iteration. By using this technique, the Multi-layer Perceptron model calculates a more optimal prediction after every iteration. (Bento, 2021). This model was applied in this research using the default hyperparameters provided by the scikit-learn library except for random_state which was set to 42 and max_iter which was set to 10,000.

Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNN). This neural network algorithm in which the input is a sequence. This model tries to take advantage of the closeness or relationship between the components of the data.

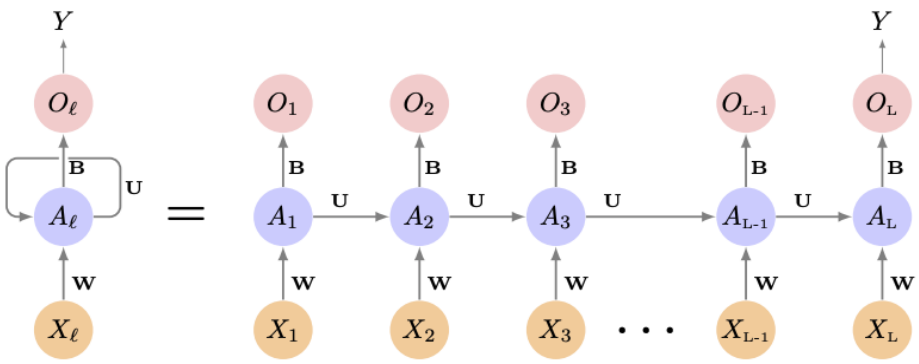


Figure 2.2 - Recurrent Neural Networks Architecture

Figure 2.2 displays a simple example of how Recurrent Neural Networks work. X_L represents the input data, where L is each sample of the dataset. The model processes each data input sequentially, starting with X_1, then X_2, then X_3 and so forth. The X_L is given into the hidden layer which also contains an activation function A_L from each subsequent element of the sequence. The letters B, U and W represent the weights that have been calculated in those elements and that are used in each sequence. Finally, O_L represents the output layers of the recurrent networks model, where the last output is the most relevant as it will be the most optimal prediction made(James, Witten, Hastie & Tibshirani, n.d.). The Keras library was used to applied the Recurrent Neural Networks model in this research, the modified hyperparameters used were: Input Layers =1, Input Layer Neurons = 625, Input Layer Activation = relu, Hidden Layers = 2, Hidden Layer Neurons = 256, Hidden Layer Activation = relu, Output Layers =1, Output Layer Neurons = 1, Output Layer Activation = Linear, Network Compilation Loss = mean_squared_error, Network Compilation Optimizer = adam, Network Compilation metric = mean_squared error, epochs = 500, batch_size = 32.

Voting Regressor

Voting Regressor is an ensemble method model that fits different regression models and averages the predictions made by each model fitted and creates a final prediction (scikit-learn, n.d.). In this research, the models included in the Voting Regressor were the top 4 performing models. The Voting Regressor model was applied using the default hyperparameters provided by the scikit-learn library.

Chapter 3: Methodology

Chapter 3 provides an overview of the data and the variables included in the dataset, as well as a description of the processes applied to perform data cleaning in order to prepare the data for the exploratory data analysis and the predictive modeling analysis.

The data used in this research was obtained by web scraping the marketplace of the NBA Top Shot platform. Upon collection of the data, data cleaning was performed to prepare the data for the exploratory data analysis and the predictive modeling analysis steps. First, the description feature was separated into 5 different features; Play Type, Date, Set, Series and Team. This step was performed in order to clearly analyze the different information that appeared in the Description feature. Next, the extra characters appearing in these new features were eliminated, for example hyphens or parentheses.

After creating the 5 new features, all the temporary features created, as well as the Description feature, the Link feature and the Date feature were eliminated. These last two variables, Link, which provided a link to each sample scraped, and Date, which provided a date of when the highlight appearing in the moment happened, were eliminated as we believed that they would not have a relevant impact in both our exploratory data analysis and our predictive modeling analysis.

Following the elimination of the immaterial features, we continued cleaning the rest of the features by eliminating any extra characters found. After performing this step, we arrived at our final dataset which was used in this research. This dataset includes the following 16 features: title, challenge_reward, rarity, scarcity, badge_rookie_year, badge_top_shot_debut, badge_rookie_mint, bage_rookie_premiere, badge_rookie_three_stars, badge_championship_year, supply, Play_Type, Set, Series, Team and price.

The first feature, “title”, states the NBA or WNBA player appearing in the moment. Another characteristic of NBA Top Shot moments is that they can contain badges & reward icons, indicating that something about that certain moment is special. The following badges/rewards icons which currently exist were included in our features:

- Rookie Mint, which means the moment was created during the rookie season of the player appearing in the moment.
- Rookie Year, which means the play that appears in the moment is from a game during the first season of a player in the NBA.
- Rookie Premiere, for moments that have a play from the first career game of a player.
- Rookie Three Stars, badge given to moments that have a rookie mint, rookie year and rookie premiere badge.
- Top Shot Debut, for moments that contain a player’s first play created in NBA Top Shot.
- Championship Year, badge given to a moment that contains a play from the season that a team won the NBA Championship.

- Challenge Reward, badge given to exclusive moments that are distributed when a user is holding in an account specific moments by a date stated by NBA TopShot.

Another characteristic of these moments is that they have different tiers, which appears in the feature “Tier”: Common, Rare, Legendary or Fandom. Common tier is characterized for having moments with a higher max serial number, above 10,000 mints. Rare tier has moments with a maximum serial number between 500 and 4,999 mints. Legendary tier has moments with a maximum serial number range of 50 to 499 mints *(Topshot101, 2021).* Fandom tier’s maximum serial number size is normally between the Common serial number size and the Rare serial number size, and it is equal to the number of people that want to acquire that specific Fandom moment *(ClutchPoint, 2021)*. The actual mint size of each moment is recorded in the feature “scarcity” in our dataset.

NBA Top Shot moments can also belong to different groups called “Sets”, this feature means that some moments of the same tier that have an attribute in common, can be part of a collection. For example, a set that contains only moments that are steals, a set that contains first-year player moments or a set that contains exclusively moments from the NBA All Star game. Also, each moment has a description of when the moment was released, which appears in the feature “Series” assigned to it. Series 1 are moments released during the 2020 NBA season, Series 2 are moments released during the 2021 season, Series Summer 2021 are moments released during the off-season between the 2021 and 2022 season, and finally, Series 3 are moments released during the 2022 season.

The next feature in the dataset is “supply” which gives information about how many listings are there in the marketplace for that specific moment. The possible values of this feature are each number from 1 to 9 or 10+, which means that there are more than 10 listings for that moment. Next we have the “Play Type” feature that specifies what is the play that is displayed in the highlight, this includes: dunks, assists, steals, jumpshots, blocks, handles, 3 points or layups. The following feature is “Team”, which states what team the player appearing in the moment plays for. Finally, the last feature in the dataset is “price”, which will be the target variable in this research. Price reveals what is the minimum price, also known as floor price, that can be paid for a specific moment.

After processing the initial data and finalizing all the final features that would be included in the dataset, another step was performed in the dataset to prepare it for the exploratory data analysis, which was to set the correct data type for each feature in the dataset. All the variables in the dataset were set as object type, except for price and scarcity, which was set to integer type.

Following the exploratory data analysis, the data needed to be prepared for the predictive modeling analysis. The first step taken was to separate into two different dataframes the target variable and the features. In this research, the feature price was the target variable that we predicted, and the rest of the features were the predictor variables, which helped predict the target variable. Once the features and target variable have been separated, we proceeded to One Hot Encode all the features. One Hot Encoding, creates new features for each possible feature value, using the number 1 if the feature is present, and 0 if the feature is not present. We decided

to One Hot Encode the features in order to avoid any possible weight given to specific values by the machine learning models as none of the features in the dataframe contain ordinal data and then proceeded to convert the target variable and features into two NumPy arrays.

After creating a NumPy array for both the target variable and the features, both arrays were splitted into training and test data, leaving us with training data for the target variable, testing data for the target variable, training data for the features and testing data for the features. In this training/testing split a random seed of 42 was used, and 30% of the data was allocated to the testing (554 samples), with the remaining 70% to the training (1,292 samples). After these modifications, now the data was ready to be deployed into the machine learning models. As previously mentioned, regression models were used in this research. The models applied to our data in this research were: Multiple Linear Regression, Ridge Regression, Lasso Regression, Elastic Net, Decision Trees, Bagging, Random Forest, AdaBoost, Gradient Boosting Regressor, Support Vector Regressor, Nu Support Vector Regressor, Linear Support Vector Regression, Stochastic Gradient Descent Regressor, K-Neighbors Regressor, Gaussian Process Regressor, Least Partial Squares Regressor, Multi-Layer Perceptron Regressor and Recurrent Neural Networks. The performance metric of these models were calculated using the Mean Absolute Error, Mean Squared Error, Root Mean Squared Error and R-Squared. Root Mean Squared Error was used as the main performance metric because it especially penalizes large errors and it is measured in dollars.

Chapter 4: Analysis and Discussion

Chapter 4 provides the relevant results found in the exploratory data analysis and the predictive modeling analysis in the light of the background.

In this chapter, we will discuss the analysis of results when the models discussed in the Background chapter of this thesis were applied to our dataset. First, we address the relevant findings discovered after performing exploratory data analysis.

The first relationship found was the correlation between scarcity and price of moments that contained any of the 3 rookie badges or all 3 rookie badges.

First, we analyzed the moment count for moments with or without a badge rookie mint, a rookie premiere badge, a rookie year badge and rookie three stars. The amount of moments with these rookie badges represent a very low portion of the total moments in the dataset. The rookie mint badge appears in 8.78% of the moments, the rookie premiere badge appears in 0.22% of the moments, the rookie year badge appears in 9.86% of the moments and finally, the rookie three stars badge appears in 2.55% of the moments.

After seeing how scarce rookie moments are in the marketplace, we studied the average price of the moments with different rookie badges to reveal a possible relationship between price and rookie moments scarcity.

The first rookie badge analyzed was Rookie Year. After evaluating the data, the research showed that the average price of moments with the Rookie Year Badge almost doubled the average price of moments without this badge. The average prices were $5,020.8 and $2,753.2 respectively.

The next rookie badge analyzed was Rookie Mint. The average price of Rookie Mint moments are $5,500.14 compared to $2,729.19 for moments without the Rookie Mint badge. This shows that Rookie Mint moments have an average price more than double the price of Non-Rookie Mint moments.

The third rookie badge studied was Rookie Premiere. The results revealed that the average price of moments with a Rookie Premiere Badge are significantly lower compared to the moments without a Rookie Premiere Badge, $574.25 and $2,981.97 respectively.

The last rookie badge analyzed was Rookie Three Stars. After studying this badge, the results showed that the average price of moments that had the Rookie Three Star was $515.47 and the average price of those moments without the Rookie Three Stars badge was $3,041.

After reviewing all four types of rookie badges, the research showed that for moments containing the Rookie Year badge or Rookie Mint badge, the average price of those moments was higher for those moments that did not have either of the badges. Meanwhile, moments that contained the Rookie Premiere badge and the Rookie Three Stars badge, did not have a higher average price than moments without either badges. The results indicated that among the different types of Rookie badges, the Rookie Year/Rookie Mint badges are on average more valuable than the Rookie Premiere/ Rookie Three Stars badges. The visualizations and calculations used in this analysis can be found in Appendix A and Appendix B.

During this research, it was also analyzed the other possible badges a moment can have: Challenge Reward badge, Top Shot Debut badge and Championship Year Badge.

First we analyzed the count for moments that did not contain the Challenge Reward badge, the Top Shot Debut Badge and the Championship year and the count for the moments that did have it. The Challenge Reward badge appears in 139 moments, which is an 7.53% of the moments, the Top Shot Debut badge appears in 826 moments, this number represents 44.75% of the moments and finally, the Championship Badge appears in 118, which represents 6.39% of the moments.

From this information, we find that both Challenge Reward and Championship Badges are scarce, meanwhile moments with Top Shot Debut Badges, even though they are a special moment because they are the first moment of that player in NBA TopShot, they are not as scarce as moments with other badges. Next we analyzed the average price for each of these three badges and compared them to moments that did not contain the badge.

The Challenge Reward badge's results showed that the average price of Challenge Reward moments was significantly higher than moments that were not Challenge Reward moments, almost a price 5 times higher. The average price of Challenge Reward moments was $11,334.27 compared to $2,296.20 for non-Reward Challenge moments. This outcome reveals that on average, Challenge rewards are remarkably more valuable than moments that are not Challenge Rewards.

Regarding the analysis of the Top Shot Debut badge, the results showed that the average price of Top Shot Debut was $3,920.54 compared to $2,212.46 for those moments that did not have the

badge, indicating that moments with the Top Shot Debut badge are on average more valuable than moments without the Top Shot Debut badge.

Lastly, the analysis of the feature Championship Year badge revealed that the average price of moments that had this badge had a price more than 4 times higher than the moments that did not have the badge, as moments with the Championship Year badge had an average price of $11,060.45 meanwhile moments that did not have the Championship Year badge had an average price of $2,424.74. These prices indicate that moments with the Championship Year badge have a considerably higher price on average than moments that did not have this moment. The visualizations and calculations of this analysis can be found in Appendix A and Appendix B.

The next feature for which relevant results were found was Rarity. As seen below in Figure 4.1, the graph visualizes the count of the different types of Rarity. Common moments were the most found in the data with 899 moments, followed by Rare with 645 moments, then Legendary with 272 moments and finally, Fandom with 30 moments.

```
Common        899
Rare          645
Legendary     272
Fandom         30
Name: rarity, dtype: int64
```

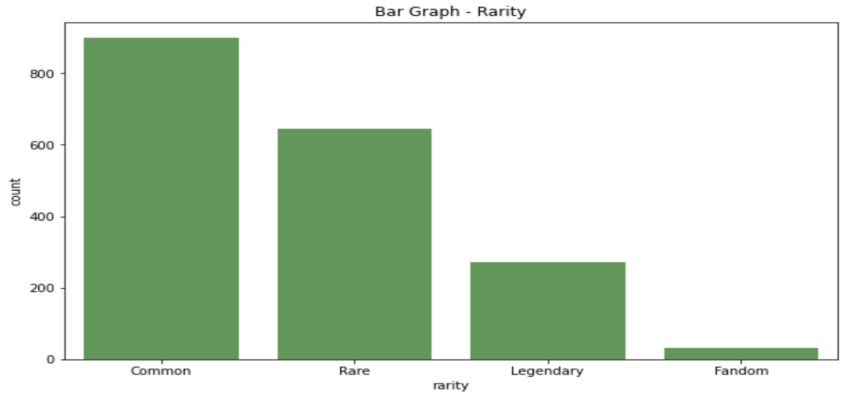


Figure 4.1: Bar graph of the count of the values of the feature “Rarity”

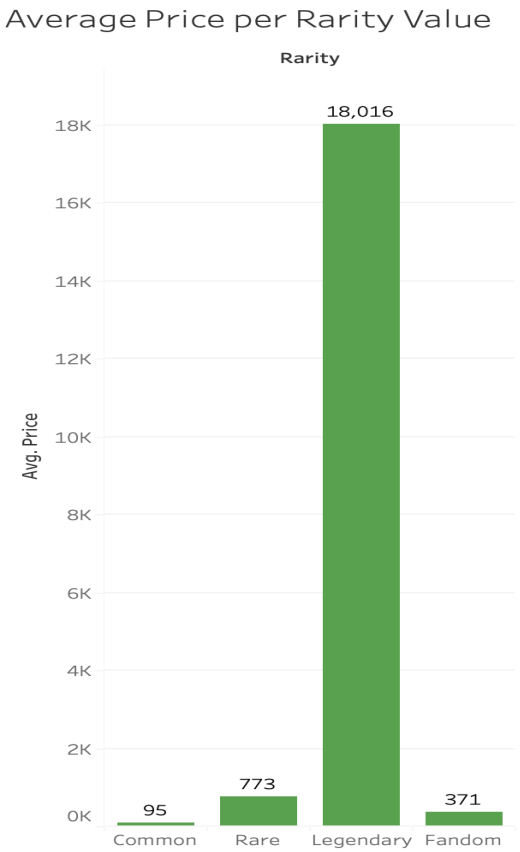


Figure 4.2: Bar graph average price of the values of the feature “Rarity”

The results of the analysis of feature Rarity showed that Legendary moments had the highest average price by a large margin with an average of $18,016 per moment. The next highest average price was $773 which belonged to Rare moments, then $371 for Fandom moments and finally, the Rarity value with the lowest average price was Common with $95. These results reveal that Legendary moments are distinctively the Rarity most valuable on average, and how Fandom values are not as valuable as Rare moments but more valuable than Common moments on average.

As part of this research, it was as well analyzed which NBA players had the highest average price.

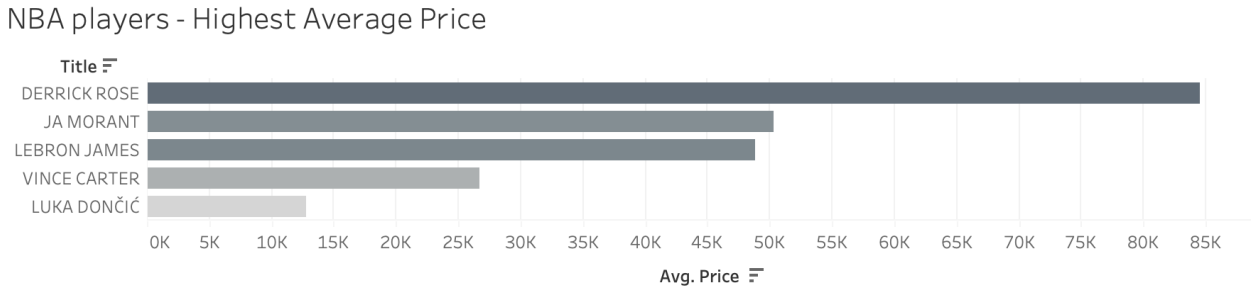


Figure 4.3: Horizontal Bar Graph of Top 5 NBA players with the highest average price

The players found to have the five highest average were: Derrick Rose, Ja Morant, Lebron James, Vince Carter and Luka Doncic. Derrick's Rose, which had the highest average price out of all the players, had an average price of $84,542, Ja Morant followed him with an average price of $50,298. In 3rd position is Lebron James with an average price of $48,805, next is Vince

Carter with $26,743 and finally, Luka Doncic with $12,806 as his average price. These results align with the opinion of the majority of basketball fans, as the 5 players that were found to be the most valuable on average, are players beloved and respected by the basketball community or players that have demonstrated an elite level of basketball skills in their career.

Another finding observed in this research was the relationship between the Series when a moment was released and the average price of the moment. This finding was analyzed comparing the average price of each Series and the count of moments in each Series.

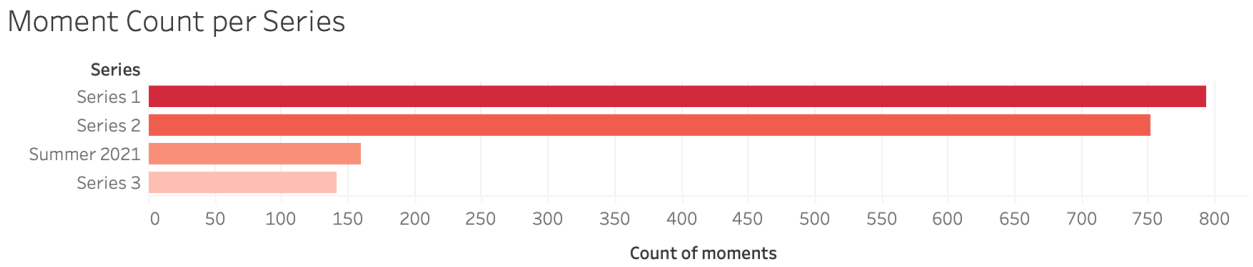


Figure 4.4: Moment Count Per Series

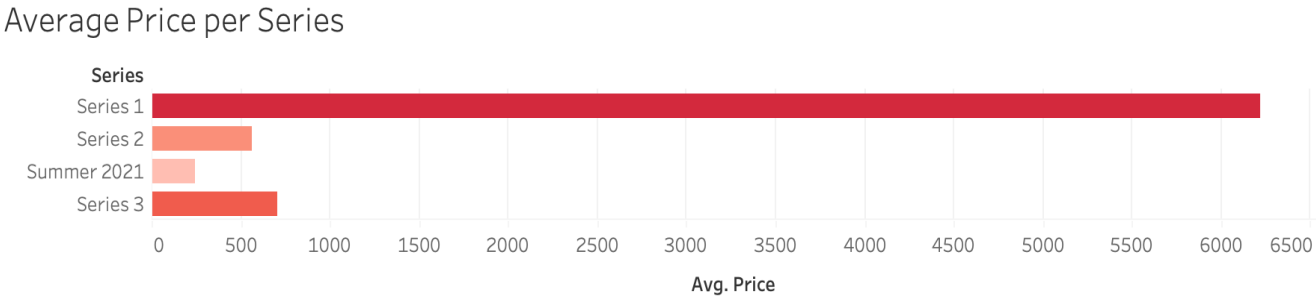


Figure 4.5: Moment Average Price Per Series

As observed in Figure 4.4, Series 1 had the highest number of moments with close to 800 moments, followed by Series 2 with around 750, and then Series Summer 2021 and Series 3 with 160 and 141 moments respectively. Then in Figure 4.5, we can observe that Series 1 had a much higher average price than any other Series. The average price for Series 1 was $6,219, meanwhile the average price for Series 2, Series Summer 2021 and Series 3 were $566, $239 and $709, respectively. The outcome of this analysis revealed that Series 1, even though it has the highest count of moments, it's the Series with the highest priced moments on average by a large margin compared to the other 3 Series.

After finding the average price of moments in each Series, we analyzed the average price of moments with each Play Type grouped by the Series when the moment was created, as visualized below in Figure 4.6.

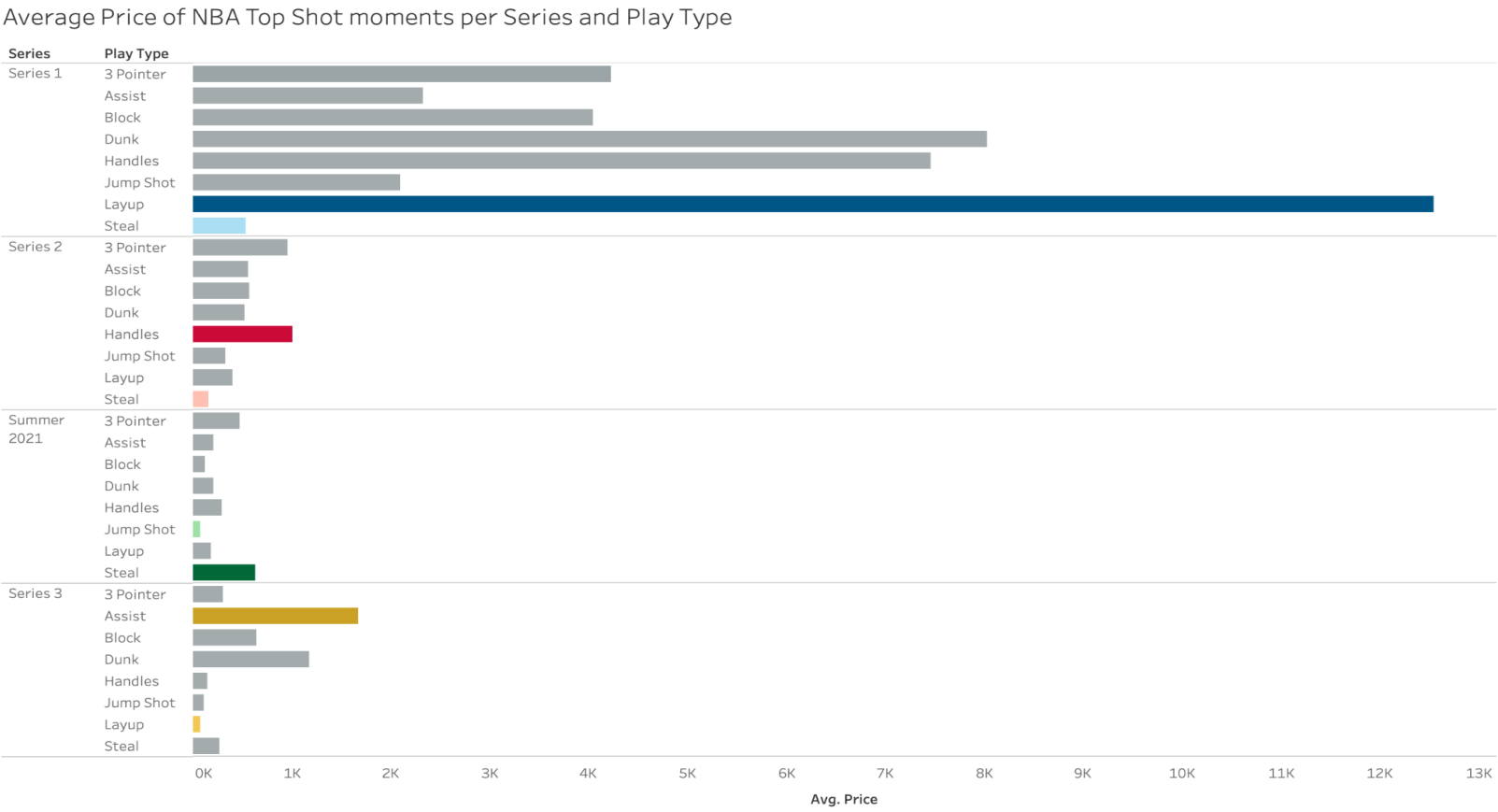


Figure 4.6: Average Price of NBA Top Shot moments per Series and Play Type

Figure 4.6 shows a horizontal bar chart created using the average price of moments grouped by the Play Type appearing in the moment and the Series when the moment was released. The results showed that Layup was the play type with the highest average price for Series 1 and also the highest average price among all the Series, meanwhile, Steal was the play type with the lowest average price in Series 1. For Series 2, Handles, followed closely by 3 Pointer, had the highest average price and Steal had the lowest average price. The Series Summer 2021 showed that Steal had the highest average price and Jump Shot had the lowest average price. Finally,

Series 3 had Assist as the Play Type with the highest average price and Layup as the Play Type with the lowest average price. The results also showed that, with the exception of Steal in Series 1, the rest of the Play Types in Series 1 have a higher average price than any other Play Type in the rest of the Series.

After analyzing the results obtained by performing exploratory data analysis to the NBA Top Shot dataset, next will be presented the results obtained after conducting predictive modeling analysis.

Below in Table 4.1, it is represented how each machine learning model performed predicting the price of moments using the 4 performance metrics discussed in the Background chapter of this research. The four performance metrics are Mean Absolute Error (MAE), R-Squared, Mean Squared Error (MSE) and Mean Root Square Error (RMSE). The initial outcome obtained from analyzing Figure 10, was the extremely poor performance of the models SGD Regressor and Multiple Linear Regression, therefore these models were not considered for further analysis of performance.

	Metrics			
Models	MAE	R-Squared	MSE	RMSE
Multiple Linear Regression	$17,187,669,336.72	-$89,909,312,409,256.00	$17,879,103,578,396,200,000,000.00	$133,712,765,203.61
Ridge Regression	$3,502.76	$0.17	$165,001,064.94	$12,845.27
Lasso Regression	$10,077.46	-$4.72	$1,137,465,000.48	$33,726.33
Elastic Net	$3,491.62	$0.14	$170,501,707.62	$13,057.63
Decision Trees	$2,781.43	-$2.34	$663,198,153.40	$25,752.63
Random Forest	$2,601.31	-$0.59	$316,032,819.41	$17,777.31
AdaBoost	$11,462.49	-$2.86	$766,771,670.67	$27,690.64
Gradient Boosting Regressor	$1,394.27	$0.18	$162,275,503.63	$12,738.74
SVR	$2,145.43	-$0.02	$202,504,850.87	$14,230.42
Nu SVR	$2,151.03	-$0.02	$202,650,748.89	$14,235.55
Linear SVR	$2,228.00	-$0.01	$200,628,639.96	$14,164.34
Bagging	$3,106.05	-$0.99	$394,755,821.90	$19,868.46
SGD Regressor	$225,584,631,764,580,000.00	-$1,047,226,085,839,020,000,000,000,000.00	$208,248,324,417,022,000,000,000,000,000,000,000.00	$456,342,332,484,093,000.00
K-Neighbors Regressor	$2,066.63	$0.12	$175,146,975.09	$13,234.31
Gaussian Process Regressor	$2,055.42	-$0.11	$220,929,062.07	$14,863.68
PLS Regression	$9,323.07	-$4.08	$1,009,578,583.94	$31,773.87
Voting Regressor	$2,687.39	$0.19	$161,580,823.74	$12,711.44
MLP Regressor	$2,895.02	$0.19	$160,471,336.27	$12,667.73
Sequential - RNN	$4,772.39	-$5.40	$1,272,552,158.70	$35,672.85

Table 4.1: Table of the performance of each machine learning model per Metric used

The next step performed was to analyze the performance of the models under each metric.

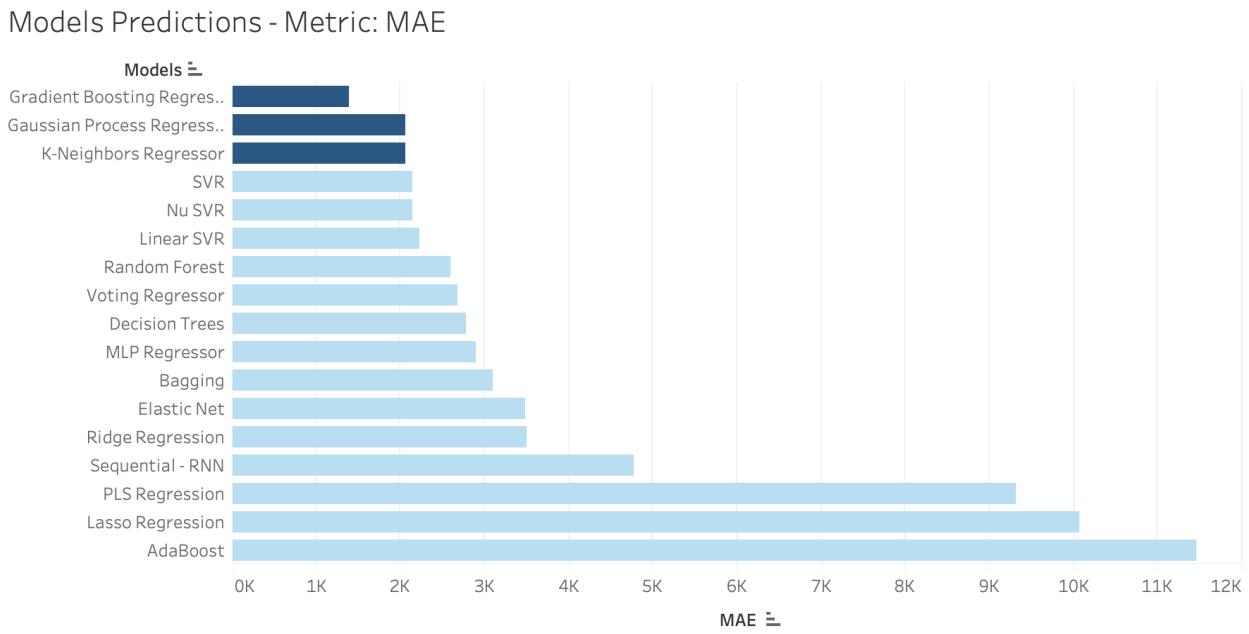


Figure 4.7: Machine learning models performance using the metric MAE

As it can be observed in Figure 4.7, this visualization shows the sorted performance of each model, with the most successful models at the top and the least successful models at the bottom. The results showed that the most accurate models predicting the price of a moment using MAE as the performance metric were Gradient Boosting Regressor, with an error of $1,394, followed by Gaussian Process Regressor, with an error of $2,055 and finally K-Neighbors Regressor with an error of $2,067.

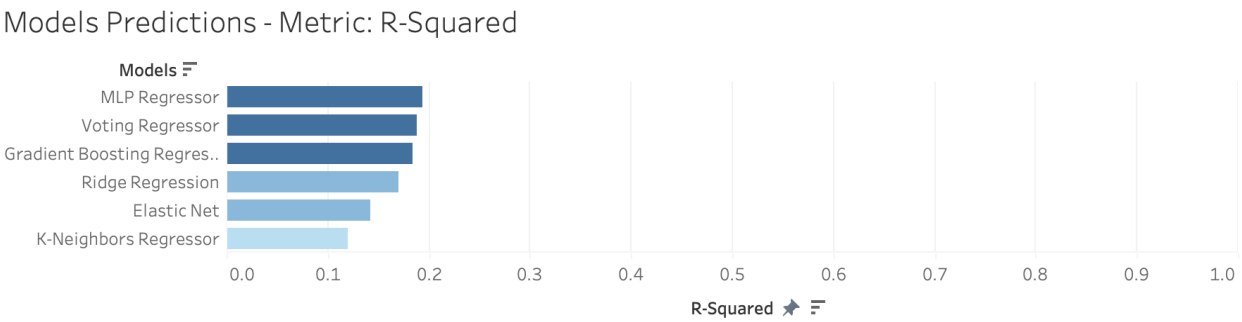


Figure 4.8: Machine learning models performance using the metric R-Squared

The next performance metric analyzed was R-Squared. Figure 24 includes the performance of all the models that had a score between 0 and 1, which is the score scale used for this performance metric. All the models that had a negative R-Squared error were not included in the visualization. The best performers according to this metric were MLP Regressor, Voting Regressor and Gradient Boosting Regressor, with a R-Squared score of 0.193, 0.187 and 0.184 respectively.

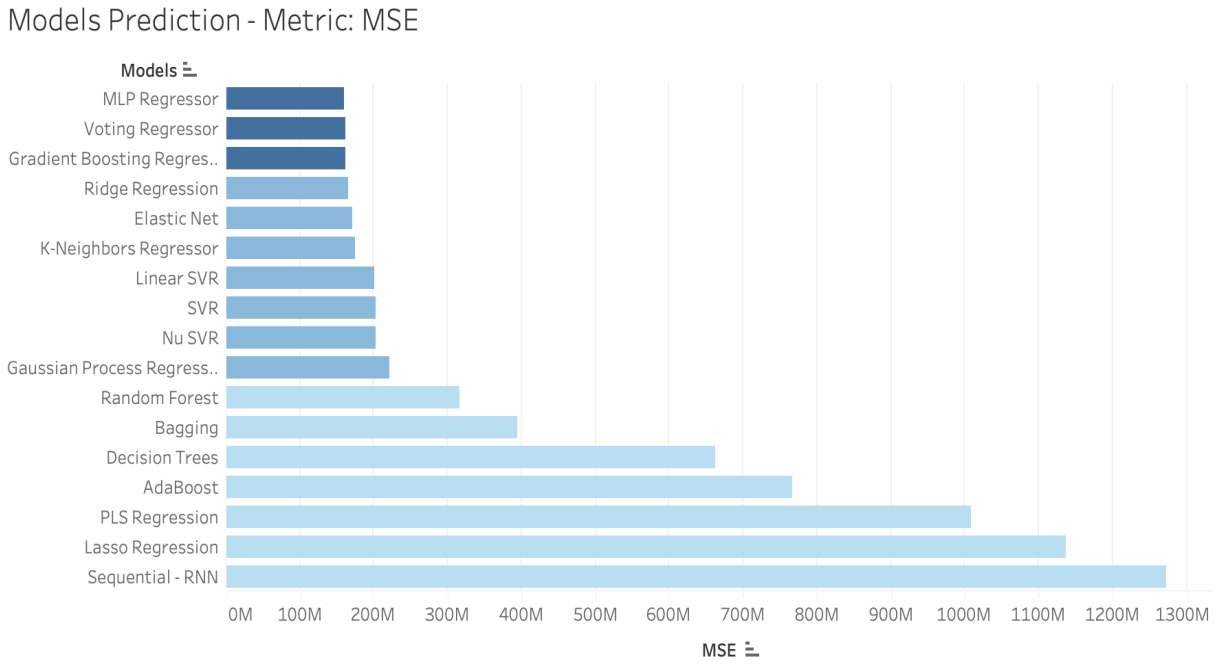


Figure 4.9: Machine learning models performance using the metric MSE

The third performance metric evaluated was MSE. The results provided by Figure 4.9 revealed that the top three performers of under this metric were MLP Regressor, which had an error of $^2160,471,336 followed by the Voting Regressor model, which had an error of $161,580,824, and finally, followed by the Gradient Boosting Regressor model, which had an error of $^2162,275,504.

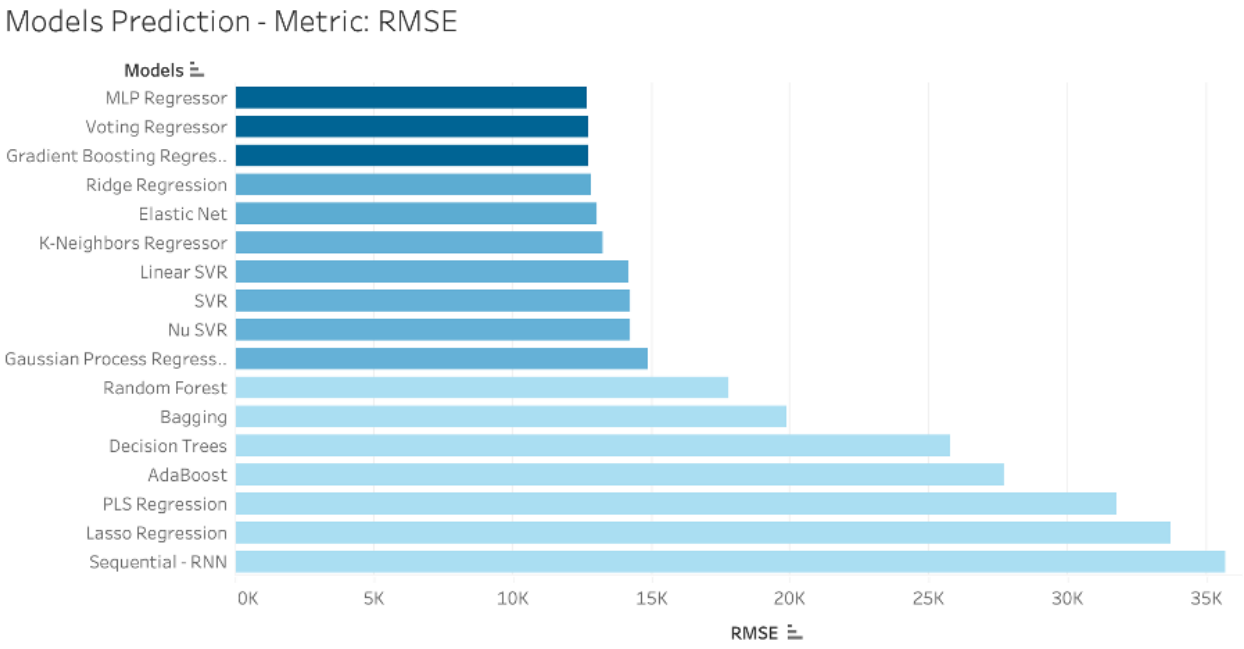


Figure 4.10: Machine learning models performance using the metric RMSE

The last metric used to analyze the performance of the machine learning models was RMSE. These performances can be observed in Figure 4.10. The outcome of the models evaluated by the RMSE metric revealed that best performing model was MLP Regressor, which had an error of $12,668 followed closely by Voting Regressor and Gradient Boosting Regressor, which had an error of $12,711 and $12,739 respectively.

After evaluating the performance of each model using different performance metrics, it was decided that the most optimal metric performance to calculate the error of the models was RMSE. This metric is the most suitable due to the fact that the data contains moments which have a very high price, and metrics like the MAE and R-Squared, could show positive results but there is the possibility that some of the moments highly priced were severely underpriced or overpriced. Therefore, using MSE or RMSE, the errors are squared which means that the bigger the error, the higher it's penalized. MSE was discarded as the most optimal metric performance because its performance is measured in Squared dollars. Therefore, RMSE, which is measured in dollars, was found to be the most optimal performance metric.

Therefore, using this metric it was found that the best performing model was MLP Regressor, which is also the best performing model under R-Squared and MSE. The error of this model, as stated previously, was $12,668.

In the predictive modeling analysis, it was also studied the variable importance. This analysis provided us with an insight of the most important and least important features in the dataset used in the predictive modeling to predict the price of a moment. Although now there are methods to apply the variable importance on "black box" models, like neural network models, we decided to not explore this option in this research, and applied the variable importance analysis to the third best performing model, Gradient Boosting Regressor. As shown below in Figure 4.11, the most important feature found by the Gradient Boosting Regressor, was Scarcity, followed by Title and

Supply. On the other hand, the least significant feature was Rarity. With Rookie Mint and Rookie Year as the next 2 least significant features.

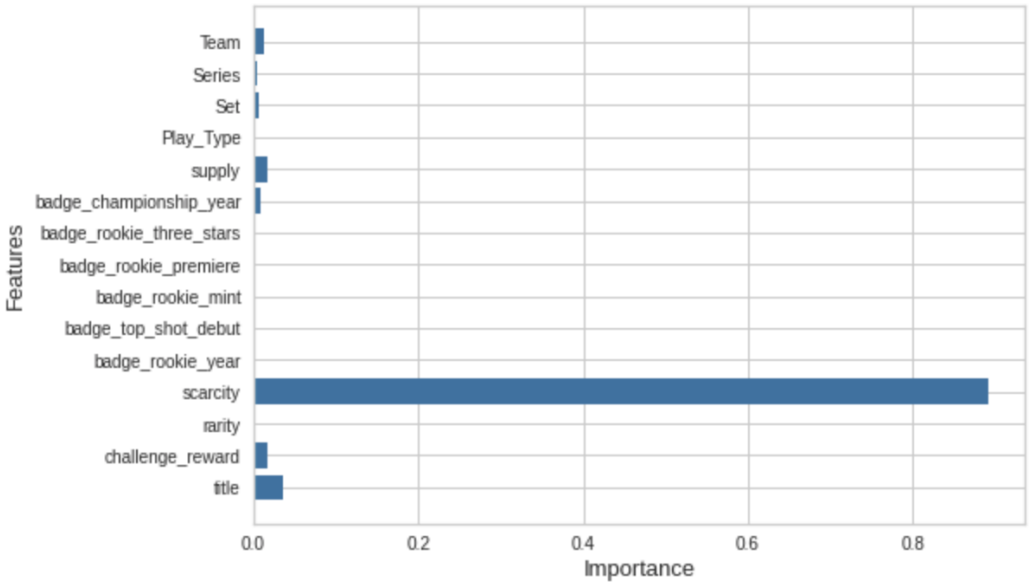


Figure 4.11: Variable importance plot of the Gradient Boosting Regressor

Chapter 5: Conclusions

As stated in the Introduction chapter of this thesis, the specific deliverable of this research was the in-depth exploration of the valuation of the NBA TopShot moments. This research included the exploratory data analysis of the features to find relationships correlated with the price of the moment and prediction of value of moments in the future, which will be used as a foundation for further future research in this topic.

In the Analysis chapter, we first provided the different results found in regards to the exploratory data analysis. This section of the analysis provided relevant results regarding the relationships discovered between the features found in the dataset.

One of the findings in this section was learning that even though all 4 types of rookie badges, Rookie Year, Rookie Mint, Rookie Premiere and Rookie Three Stars, were scarce compared to the rest of the moments in the marketplace, not all of them were more valuable on average than moments not containing that specific badge. This proved that in regard to all rookie badges, Rookie Year and Rookie Mint moments are on average significantly more valuable rookie moments than rookie moments with the Rookie Premiere or Rookie Three Stars badge, as well as more valuable on average than moments that did not contain the Rookie Year or Rookie Mint badges.

This research also provided valuable information in regards to the rest of badges that a moment can have; Challenge Rewards badge, Top Shot Debut badge and Championship Year badge.

The Challenge Rewards and Championship Year badges were found to be scarce, meanwhile the Top Shot debut badge, even though it is less common than moments without the badge but not as scarce as the other two badges, its scarcity will grow overtime as more and more moments are minted by NBA Top Shot. All three badges had a higher average price than moments without each badge, especially moments with the Challenge Reward badge and the Championship Year badge. This proves that these 3 badges are on average more valuable than moments without any of these three badges.

The results obtained for all seven available badges showed that the NBA Top Shot marketplace follows the supply and demand economic relationship in some areas but not to a full extent.

The outcome of the exploration of the Rarity feature showed that Legendary moments have a remarkably higher average price than any other Rarity feature; 23 times higher than the following rarity. This analysis also showed that the Fandom Rarity has an average value between Common Rarity and Rare rarity, which means that Fandom moments are valued more than a Common moment but not as much as a Rare moment.

Another feature that provided relevant results was Title. The analysis of this feature confirmed that the top most expensive players, on average, were players that are widely loved by the basketball community or players that have performed at an elite level in their career. This proves that players who are appreciated by the basketball community or players that play basketball exceptionally have a high value on average.

Another meaningful result found in this research was the difference in value between different Series. Series 1 was proved to be the Series with the highest average value, this result determines

that moments that have been minted in the past do have a higher price than moments that have been created more recently. Also the results confirmed that Summer 2021 Series, was the least expensive on average which could prove that moments released during the off-season period between two Series, are less valuable than moments released during the NBA season.

This feature also provided relevant results when it was analyzed in depth with the feature Play Type. The outcome of the analysis demonstrated that most of the Play Types of Series 1 are on average more valuable than any other Play Type in any other Series, except for the Play Type Steals which was significantly less valuable on average compared to the rest of the Play Types of Series 1.

The other area of this research studied was the use of predictive modeling to predict the price of moments

In regards to this area of our research, the machine learning model Multi-layer Perceptron (MLP) was the best performing model out of the 19 models applied to this data in predicting the price of a moment, with an error of $12,668.

In this research, we were also able to find the model that produced the best price prediction and interpretability. The variable importance results obtained in the Gradient Boosting Regressor model, helped to provide a better understanding about the importance of the Scarcity feature to predict the price of a moment, as well as the importance of the Title feature and the Supply

feature. Also, these results provided an understanding of the least significant features which included the Rarity, Rookie Mint and Rookie Year features.

In any future work, we believe there are different steps that could possibly help improve the results obtained in this research. Some of the possible changes would include eliminating those moments which were clearly listed by users with a price that does not correspond to the moment listed. In some cases, because of the low amount of listings available for a moment, sellers can choose to set the floor at an unrealistic price. An example of this case could be Kentavious Caldwell-Pope's legendary moment which is listed for $19,995 and only has 2 listings. Even though this is a legendary moment and from the Series 1, Kentavious Caldwell-Pope is not known for being an elite basketball player and there are other players that are more recognized and have a legendary moment from Series 1, that have an equivalent moment with a significantly lower price, like Russell Westbrook or LaMarcus Alridge.

Another possibility to improve the results of the predictive modeling analysis would be to divide the data to help the machine learning model identify the prices of moments based on their features. This would include for example only applying the model to a specific rarity or a specific badge. Making this change would help exclude some possible outliers and narrow the data according to the type of moments desired to be predicted.

Lastly, we believe that another step to improve the results obtained would be to not include those features found to have the least importance to predict the price of a moment. Not using those

features could provide better results as the noise could be reduced, and the model could make better predicting decisions.

We believe that results obtained in the exploratory data analysis and the predictive modeling were successful. This research expanded the previous analysis performed by Riccardo Del Chin and provided relevant results about this new research area and will serve as a solid foundation for future research in this topic, as well as research for other sports-oriented NFTs that are structured similarly to NBA Top Shot.

Chapter 6: References

1. IBM (Accessed 20 December 2021). *What is blockchain technology?* https://www.ibm.com/topics/what-is-blockchain
2. Ethereum (Accessed 22 December 2021). *Non-fungible tokens (NFT)* https://ethereum.org/en/nft/
3. SportsPro Media (Accessed 17 December 2021) (2021, May 4). *What is NBA Top Shot? Dapper Labs' Caty Tedman explains the NFT platform everyone is talking about* https://www.sportspromedia.com/analysis/nba-top-shot-dapper-labs-nfts-digital-collectibles-caty-tedman/
4. EvaluateMarket (Accessed 22 December 2021) https://evaluate.market/nbatopshot
5. Hyperallergic (Accessed 10 February 2022) (2021, June 10). *"First Ever NFT" Sells for $1.4 Million* https://hyperallergic.com/652671/kevin-mccoy-quantum-first-nft-created-sells-at-sothebys-for-over-one-million/
6. Pixelplex (Accessed 10 February 2022) (2021, November 24). *Which Blockchain to Choose for Your NFT Project?* *https://pixelplex.io/blog/top-ten-blockchains-for-nft-development/*
7. *NBATopShot/presentazione_progetto.pdf at main - github.com. (n.d.). Retrieved May 2, 2022, from* *https://github.com/Rikydc01/NBATopShot/blob/main/Presentazione_Progetto.pdf*
8. Topshot101 (Accessed 11 February 2022) (2021, May 6). *Common Vs Rare Vs Legendary Moments in NBA Top Shot.* https://topshot101.com/common-rare-legendary/
9. ClutchPoints (Accessed 11 February 2022) (2021, 4 August). *NBA Top Shot's all-new Fandom Tier explained.* https://clutchpoints.com/nba-top-shot-news-the-all-new-fandom-tier-explained/
10. ESPN (Accessed 19 February 2022) (2007, 26 April). *What is PER?* https://www.espn.com/nba/columns/story?columnist=hollinger_john&id=2850240
11. Encyclopedia of Machine Learning. (Accessed 29 March 2022) (2011) *Mean Absolute Error.* https://doi.org/10.1007/978-0-387-30164-8_525
12. Encyclopedia of Machine Learning. (Accessed 1 April 2022)(2011) *Mean Squared Error.* https://doi.org/10.1007/978-0-387-30164-8_528
13. Encyclopedia of Machine Learning. (Accessed 1 April 2022) (2000) *Mean Absolute Percentage Error.*

https://doi.org/10.1007/1-4020-0612-8_580
14. Encyclopedia of Research Design (Accessed 2 April 2022)(2010) *Root Mean Squared Error.* https://methods.sagepub.com/reference/encyc-of-research-design/n392.xml
15. IBM Cloud Education (Accessed 6 April 2022) (2020, 7 December) *Random Forest.* https://www.ibm.com/cloud/learn/random-forest
16. Analytics Vidhya. (Accessed 6 April 2022) (2021, 29 August) *Decision Tree Algorithm A Complete Guide* *https://www.analyticsvidhya.com/blog/2021/08/decision-tree-algorithm/*
17. W3Schools (Accessed 8 April 2022) *Machine Learning - Linear Regression* https://www.w3schools.com/python/python_ml_linear_regression.asp
18. Bevans, R. (2020, October 26). *An introduction to multiple linear regression*. Scribbr. Retrieved May 2, 2022, from https://www.scribbr.com/statistics/multiple-linear-regression/
19. Wu, S. (2021, June 5). *What are the best metrics to evaluate your regression model?* Medium. Retrieved April 10, 2022, from https://towardsdatascience.com/what-are-the-best-metrics-to-evaluate-your-regression-model-418ca481755b
20. Brownlee, J. (2021, April 26). *How to develop an ADABOOST ensemble in python*. Machine Learning Mastery. Retrieved April 16, 2022, from https://machinelearningmastery.com/adaboost-ensemble-in-python/
21. Brownlee, J. (2020, August 14). *A gentle introduction to the gradient boosting algorithm for machine learning*. Machine Learning Mastery. Retrieved April 24, 2022, from https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/
22. Choudhury, A. (2021, October 17). *AdaBoost vs gradient boosting: A comparison of leading boosting algorithms*. Analytics India Magazine. Retrieved April 16, 2022, from https://analyticsindiamag.com/adaboost-vs-gradient-boosting-a-comparison-of-leading-boosting-algorithms/
23. Gandhi, R. (2018, July 5). *Support Vector Machine - introduction to machine learning algorithms*. Medium. Retrieved April 24, 2022, from https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47
24. Sharp, T. (2020, May 6). *An introduction to support vector regression (SVR)*. Medium. Retrieved April 24, 2022, from https://towardsdatascience.com/an-introduction-to-support-vector-regression-svr-a3ebc1672c2
25. Langhammer, J., & Česák, J. (2016, November 30). *Applicability of a nu-support vector regression model for the completion of missing data in hydrological time series*. MDPI. Retrieved April 16, 2022, from https://www.mdpi.com/2073-4441/8/12/560

26. Paisitkriangkrai, P. (2012, October 24). *Linear regression and support vector regression*. Retrieved April 24, 2022, from https://cs.adelaide.edu.au/~chhshen/teaching/ML_SVR.pdf
27. *sklearn.svm.linearSVR*. scikit-learn. (n.d.). Retrieved April 16, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVR.html
28. IBM Cloud Education. (n.d.). *What is bagging?* IBM. Retrieved April 16, 2022, from https://www.ibm.com/cloud/learn/bagging
29. Real Python. (2021, January 19). *Stochastic gradient descent algorithm with python and NumPy*. Real Python. Retrieved April 16, 2022, from https://realpython.com/gradient-descent-algorithm-python/
30. Great Learning Team. (2022, March 22). *What is ridge regression?* GreatLearning Blog: Free Resources that Matters to shape your Career! Retrieved April 17, 2022, from https://www.mygreatlearning.com/blog/what-is-ridge-regression/
31. Glen, S. (2021, April 27). *Lasso regression: Simple definition*. Statistics How To. Retrieved April 17, 2022, from https://www.statisticshowto.com/lasso-regression/#:~:text=Lasso%20regression%20is%20a%20type,i.e.%20models%20with%20fewer%20parameters).
32. Brownlee, J. (2020, June 11). *How to develop elastic net regression models in Python*. Machine Learning Mastery. Retrieved April 17, 2022, from https://machinelearningmastery.com/elastic-net-regression-in-python/
33. *Regularization Tutorial: Ridge, Lasso & Elastic Net Regression*. DataCamp Community. (n.d.). Retrieved April 23, 2022, from https://www.datacamp.com/community/tutorials/tutorial-ridge-lasso-elastic-net
34. Muhajir, I. (2020, February 8). *K-neighbors regression analysis in Python*. Medium. Retrieved April 17, 2022, from https://medium.com/analytics-vidhya/k-neighbors-regression-analysis-in-python-61532d56d8e4
35. *K-Nearest Neighbors Algorithm: KNN Regression Python*. Analytics Vidhya. (2020, May 25). Retrieved April 26, 2022, from https://www.analyticsvidhya.com/blog/2018/08/k-nearest-neighbor-introduction-regression-python/
36. Wang, J. (n.d.). An Intuitive Tutorial to Gaussian Processes. Retrieved April 26, 2022, from http://www.gaussianprocess.org/gpml/chapters///RW.pdf
37. Sit, H. (2021, October 13). *Quick start to gaussian process regression*. Medium. Retrieved April 17, 2022, from https://towardsdatascience.com/quick-start-to-gaussian-process-regression-36d838810319#:~:text=Gaussian%20process%20regression%20(GPR)%20is,uncertainty%20measurements%20on%20the%20predictions.
38. James, G., Witten, D., Hastie, T., & Tibshirani, R. (n.d.). *An introduction to statistical learning: With applications in R* (Second).
39. *What is partial least squares regression?* Minitab. (n.d.). Retrieved April 17, 2022, from https://support.minitab.com/en-us/minitab/18/help-and-how-to/modeling-statistics/regression/supporting-topics/partial-least-squares-regression/what-is-partial-least-squares-regression/

40. *Sklearn.ensemble.votingregressor*. scikit-learn. (n.d.). Retrieved April 17, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingRegressor.html
41. Bento, C. (2021, September 30). *Multilayer Perceptron explained with a real-life example and python code: Sentiment Analysis*. Medium. Retrieved April 17, 2022, from https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141
42. *Multilayer Perceptron*. Multilayer Perceptron - an overview | ScienceDirect Topics. (n.d.). Retrieved April 26, 2022, from https://www.sciencedirect.com/topics/computer-science/multilayer-perceptron

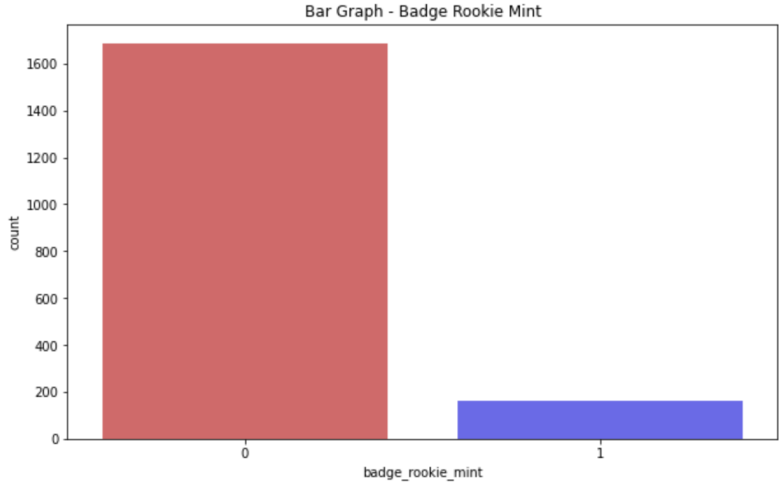
Appendices

Appendix A - Badge/ Non-Badge comparison visualizations

Appendix B - Badge/Non badge average price comparison

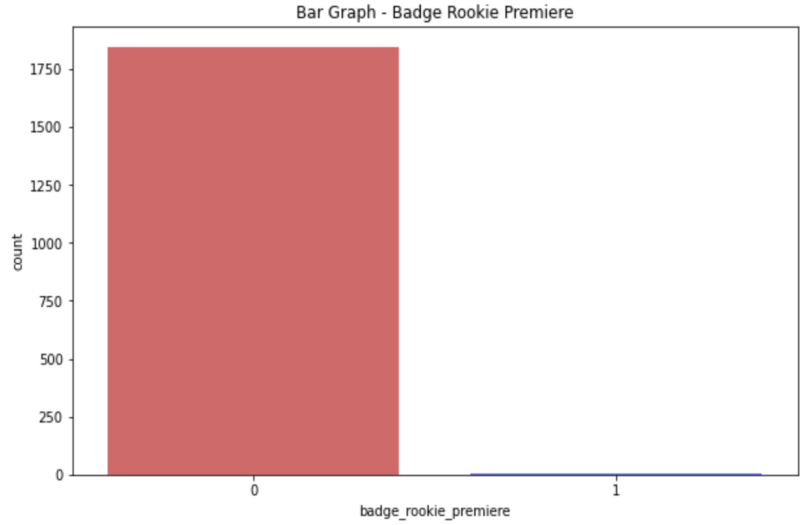
Appendix A - Badge/Non badge comparison visualizations

```
0    1684
1     162
Name: badge_rookie_mint, dtype: int64
```



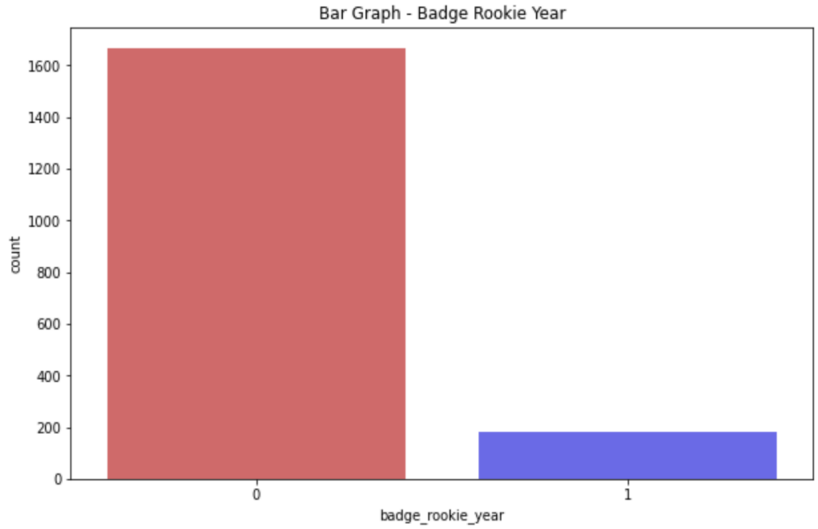
Bar graph of the count of Moments with/without the Rookie Mint Badge

```
0    1842
1       4
Name: badge_rookie_premiere, dtype: int64
```



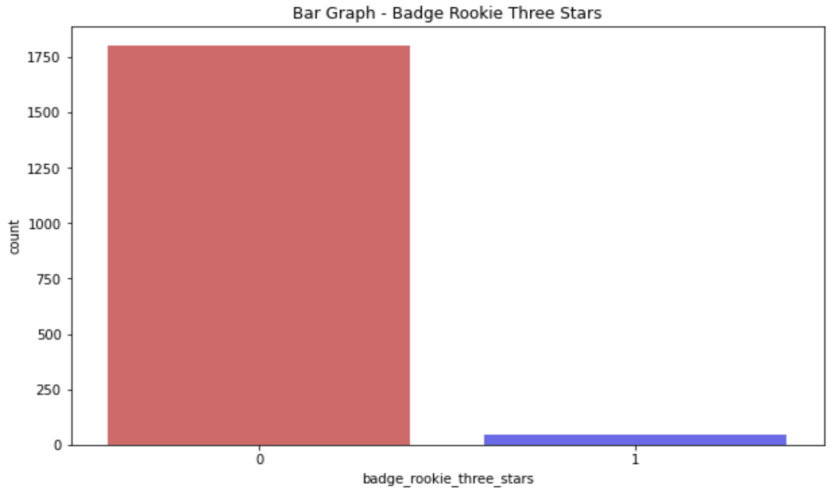
Bar graph of the count of Moments with/without the Rookie Premiere Badge

```
0    1664
1     182
Name: badge_rookie_year, dtype: int64
```



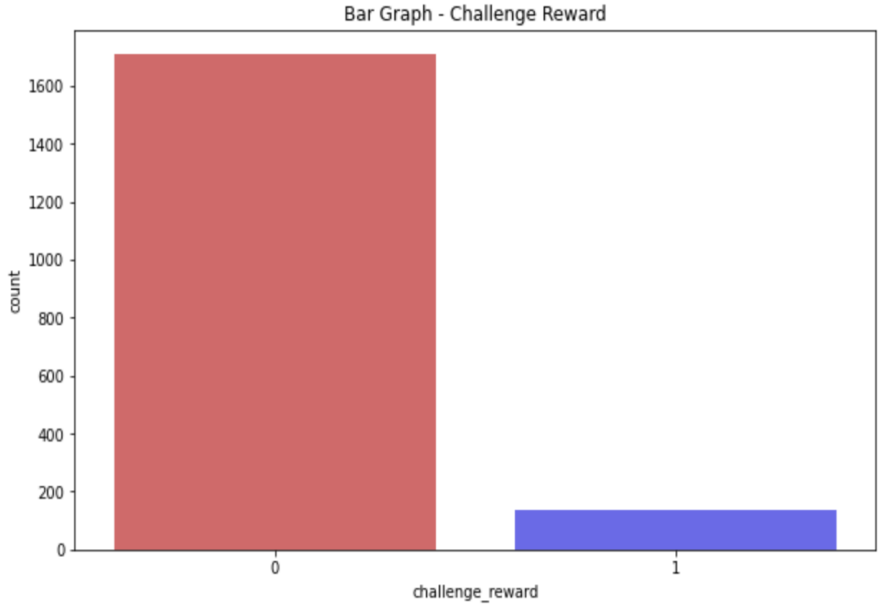
Bar graph of the count of Moments with/without the Rookie Year Badge

```
0    1799
1      47
Name: badge_rookie_three_stars, dtype: int64
```



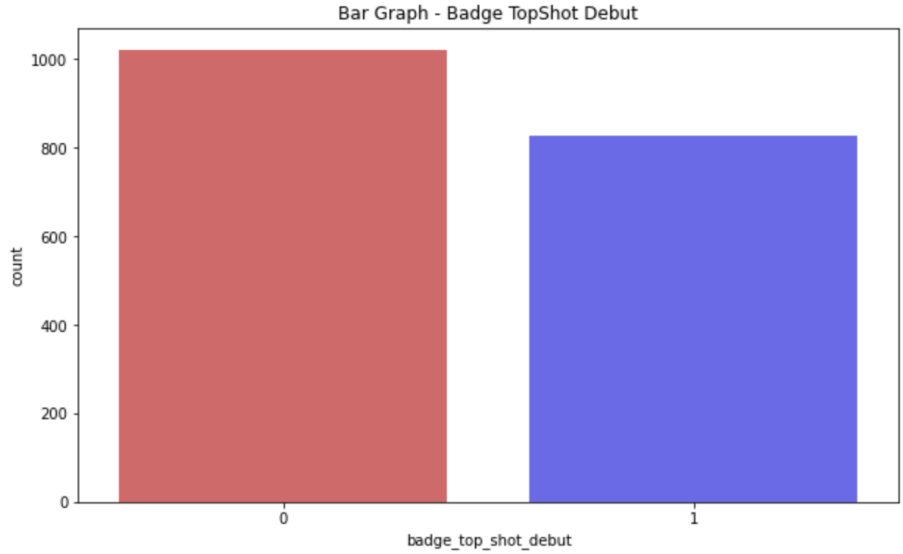
Bar graph of the count of Moments with/without the Rookie Three Stars Badge

```
0    1707
1     139
Name: challenge_reward, dtype: int64
```



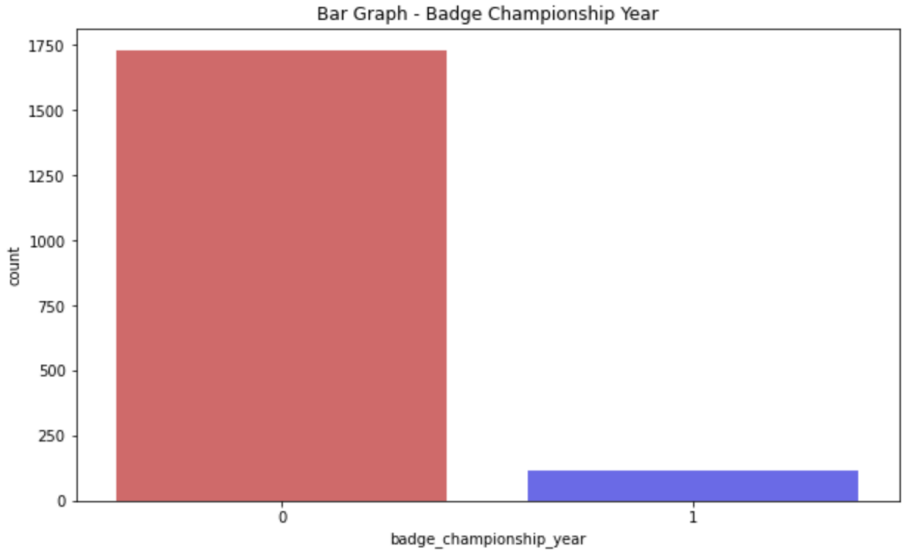
Bar graph of the count of Moments with/without the Challenge Reward Badge

```
0    1020
1     826
Name: badge_top_shot_debut, dtype: int64
```



Bar graph of the count of Moments with/without the Top Shot Badge

```
  FutureWarning
0     1728
1      118
Name: badge_championship_year, dtype: int64
```



Bar graph of the count of Moments with/without the Championship Year Badge

Appendix B - Badge/Non badge average price comparison

```
Average price of Non-Rookie Year and Rookie Year moments.
badge_rookie_year
0    2753.180288
1    5020.796703
```

Average price of Non-Rookie Year and Rookie Year moments

Average Price of Non-Rookie Mint and Rookie Mint moments.

```
Average price of Non-Rookie Mint and Rookie Mint moments.
badge_rookie_mint
0    2729.188836
1    5550.141975
```

```
Average price of Non-Rookie Premiere and Rookie Premiere moments.
badge_rookie_premiere
0    2981.965255
1     574.250000
```

Average Price of Non-Rookie Premiere and Rookie Premiere moments.

```
Average price of Non-Rookie Three Stars and Rookie Three Stars moments.
badge_rookie_three_stars
0    3041.051140
1     515.446809
```

Average Price of Non-Rookie Three Stars and Rookie Three Stars moments.

```
Average price of Non-Challenge Reward and Challenge Reward moments.
challenge_reward
0     2296.200351
1    11334.266187
```

Average Price of Non-Challenge Reward and Challenge Reward moments

```
Average price of Non-Top Shot Debut and Top Shot Debut moments.
badge_top_shot_debut
0    2212.459804
1    3920.542373
```

Average Price of Non-Top Shot Debut and Top Shot Debut moments

```
Average price of Non-Championship Year and Championship Year moments.
badge_championship_year
0     2424.736111
1    11060.449153
```

Average Price of Non-Championship Year and Championship Year moments