**Abstract**

The National Basketball Association (NBA) is a men’s professional basketball league comprised of 30 teams. Each of these teams have multi-million dollar budgets that they use in order to build a talented team that will hopefully win enough games to bring home the NBA title. To do this, the teams build their rosters based on talent and budget. While team payrolls are large, they are not unlimited and strategic decisions must be made to develop the most talented team while maintaining budget constraints. The assumption of most teams, and players as well, is that the more money a particular player demands, the better the player. This is not always the case, however. There are numerous instances where players have been drastically overpaid and dramatically underperformed.[1-2] Conversely, there have been players that have performed well above their higher-paid counter-parts.[3] Ideally, there would be a mechanism where player salary could be determined by consistent player performance. It is reasonable to assume that NBA front offices do indeed subscribe to this premise, but some player salaries would suggest otherwise. The purpose behind this course project is to use predictive analytics to predict what a player’s salary should be by looking at player statistics.

**Introduction**

With ever increasing team payrolls, owners can usually afford to pay an underachieving player. The real issue is in determining the opportunity cost of this: What could you have gotten for that money instead? How else could you have improved your roster? It should come as no surprise then that teams are looking for alternative methods for determining appropriate pay levels for their athletes. This is where predictive analytics can hopefully shed some needed light on this very expensive problem as laid out below:

* What should an NBA player be paid based on his performance statistics?
* Which players are drastically overpaid?
* Which players are drastically underpaid?

For this project, we will be utilizing the NBA player statistics dataset that was obtained from Kaggle[4]. The website lists the statistics as originating from 2017-2018 team rosters, but upon further verification, the rosters are more in line with 2016-2017 data[5]. The downloaded data contained 88 missing salary values. All but two of these were obtained from HoopsHype.com5 website of player salaries.

The dataset contains 52 different variables. Not all of these are relevant to predicting player salaries and a detailed description of the variables used will be included in the technical approach section below.

**Methods**

Three distinct methods are used to generate our predictive model. These are broken down into particular phases.

* Phase I: This consists of conducting an exploratory data analysis (EDA) on the NBA dataset. Our dataset contains 52 different variables that could contribute to our predictive model. We realize this is simply too many, so we have developed a process by which we will select the variables of interest. The first step was to look at the summary statistics for each of the variables. The goal of this task was to see if any variables contained missing values, values that were possible outliers, or incorrect values.

The next phase of our EDA was to look at the histograms for each variable. According to our text, *“Applied Predictive Analytics”*[6], many predictive algorithms assume the model variables follow a normal distribution. There are inherent advantages to using normally distributed variables, so our approach will focus on columns that closely follow this distribution. Additionally, we will use the histograms to help determine if any variable has outlier values.

* Phase II: This portion will focus on feature/variable selection from the 52 variables within the dataset. Our research led us to an article[7] that provided a more definitive method for feature selection that is based on various methods, such as a random forest classifier, a recursive feature elimination, an extra trees classifier, a chi squared analysis, and a Lasso regression. These methods generated DataFrames of the top features and a scoring system was added that would keep track of the highest ranking features for each method. The overall feature score was then determined that provided the final feature rankings.

Some features ended up with the same final score and so a correlation matrix was built to further reduce the features that were considered redundant.

* Phase III: This phase takes the results from Phases I and II and provides the final features that will be used in our initial predictive model build. The model will then be run and the summary of results and a path forward will be discussed.

**Results**

*Phase I: Exploratory Data Analysis (EDA)*

**Statistical Summary**: A sample output of this analysis is shown in Figure 1. We took particular note of the **count** values to help verify missing data, as well as the **max** and **min** values to track incorrect or outlier values. The result of these summary statistics showed that there were indeed some missing values that would require further investigation but the max and min numbers appeared to be within the expected ranges of possible values.

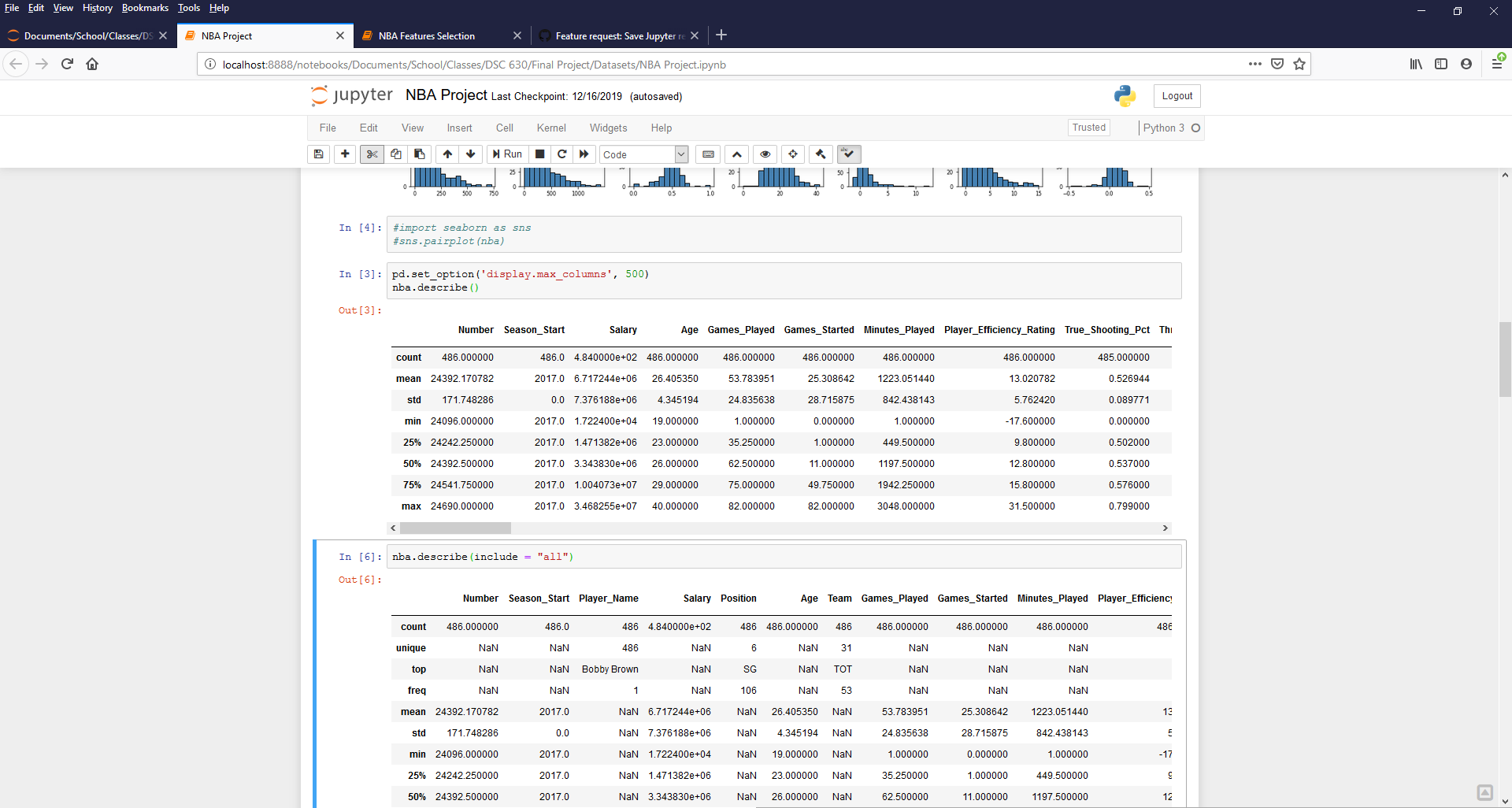


Figure 1. Sample of summary statistics for NBA dataset.

**Histograms**: The output from the histogram plots are shown in Figure 2.

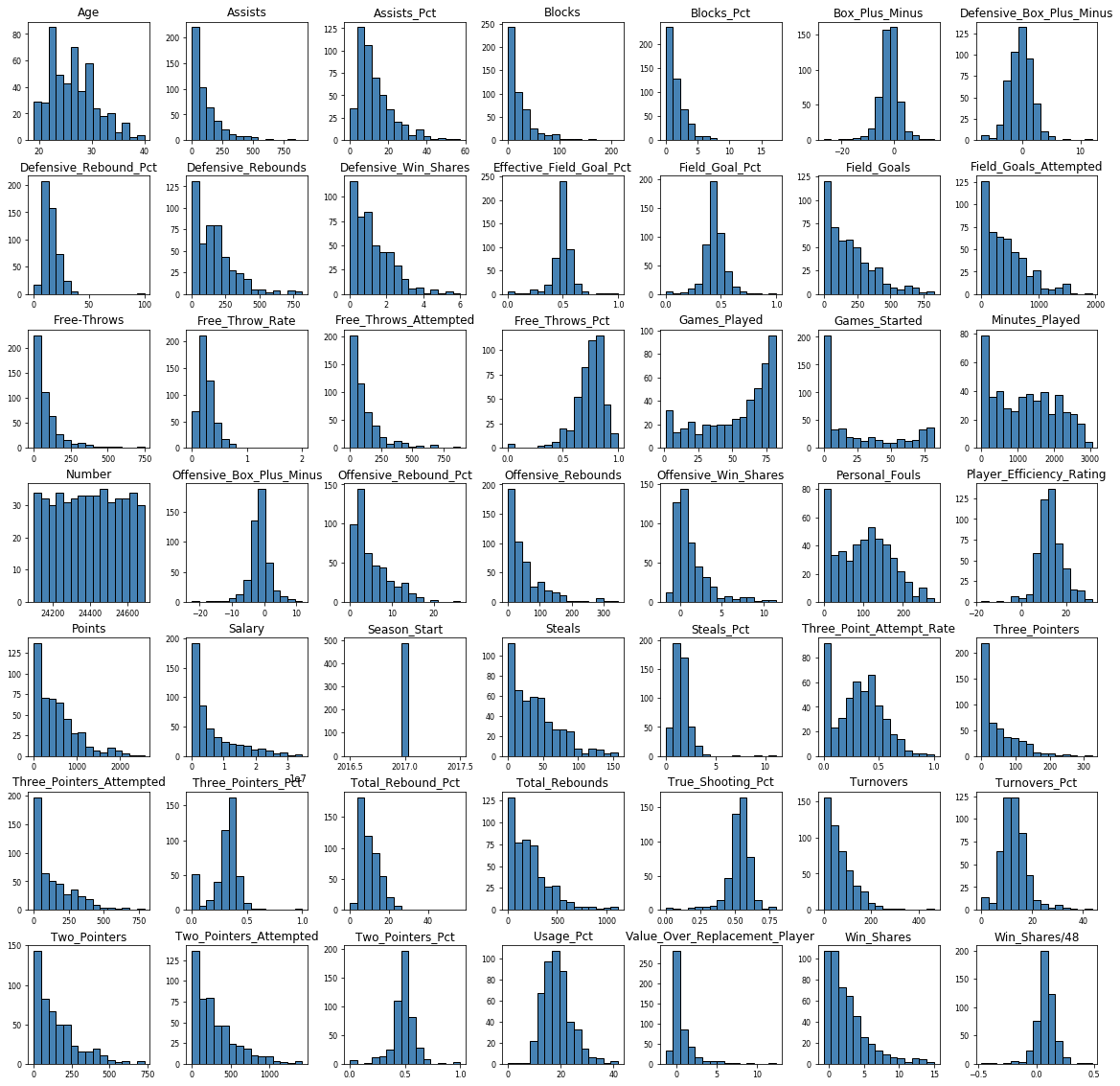


Figure 2. Histogram plots of all NBA dataset variables.

The histogram plots do indeed indicate those variables with normal distributions, such as Player\_Efficiency\_Rating. The plots also show that there do not appear to be any outlying data points that we need to be concerned about.

While both of these two techniques helped us get a better view of our variable distributions, we still were not comfortable selecting our model variables from these two EDA techniques alone.

*Phase II: Feature Selection*

* **Variable Importance: Random Forest Classifier**—this method utilized the RandomForestClassifier from the sklearn library. Random forests are one the most popular machine learning algorithms. They are so successful because they provide in general a good predictive performance, low overfitting, and easy interpretability. This interpretability is given by the fact that it is straightforward to derive the importance of each variable on the tree decision. In other words, it is easy to compute how much each variable is contributing to the decision.[8] Each of our variables were subjected to this algorithm and then ranked on importance to the model. The output of this process is shown in Figure 3. The first column is the DataFrame column number, the **index**column is the name of the dataset variable, and the **RF** column is the Random Forest value. Due to the length of the output, only the first 10 values are shown.

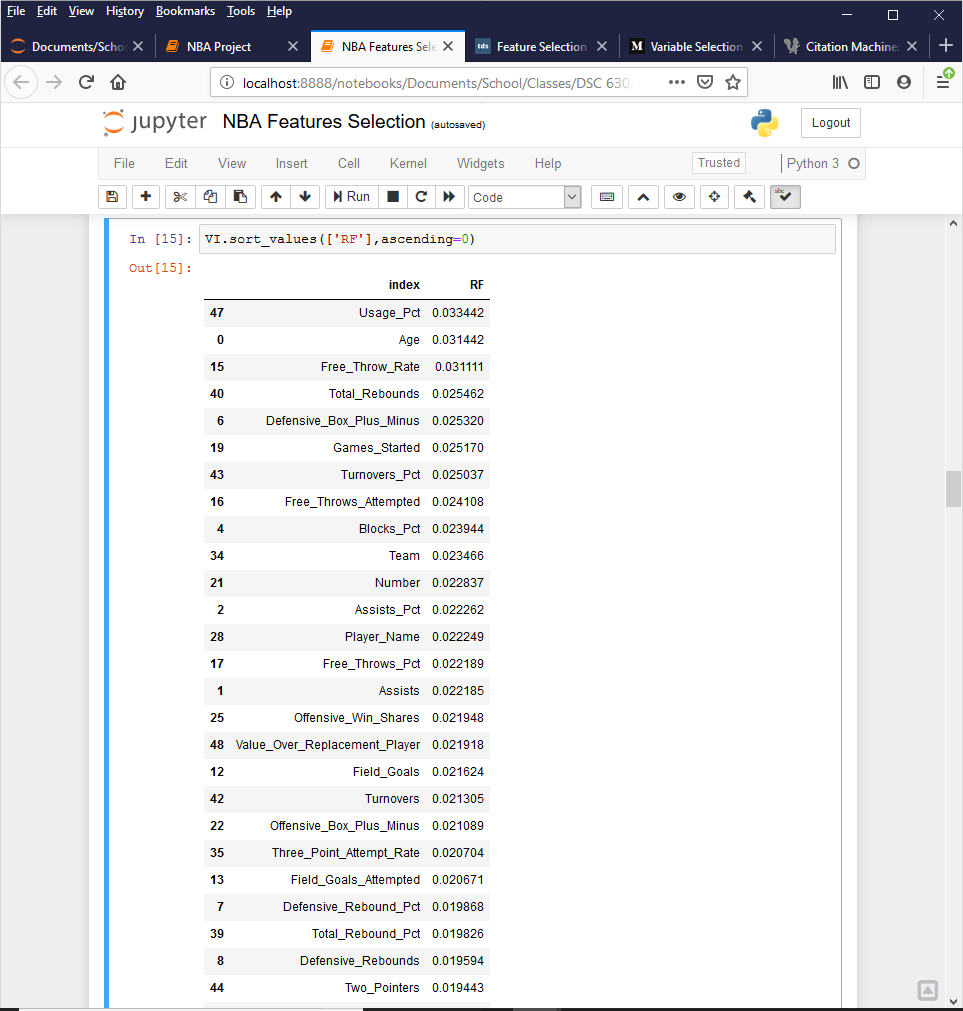


Figure 3. Output from Random Forest variable importance calculation.

* **Recursive Feature Elimination (RFE)**—this technique is also from the sklearn library. RFE is basically a backward selection of the predictors. This technique begins by building a model on the entire set of predictors and computing an importance score for each predictor. The least important predictor(s) are then removed, the model is re-built, and importance scores are computed again, hence the recursive nature of the process.[9] The output from this process is shown in Figure 4. Like before, the first column is the DataFrame column number, the **index** column is the name of the dataset variable, and the **RFE** column is the Recursive Feature Elimination result. The values of the RFE column are shown as True because they have not been eliminated in the RFE selection process. In other words, these remaining variables are considered important to the overall predictive model. The output below (in alphabetical order) shows all of the variables that returned a True value.

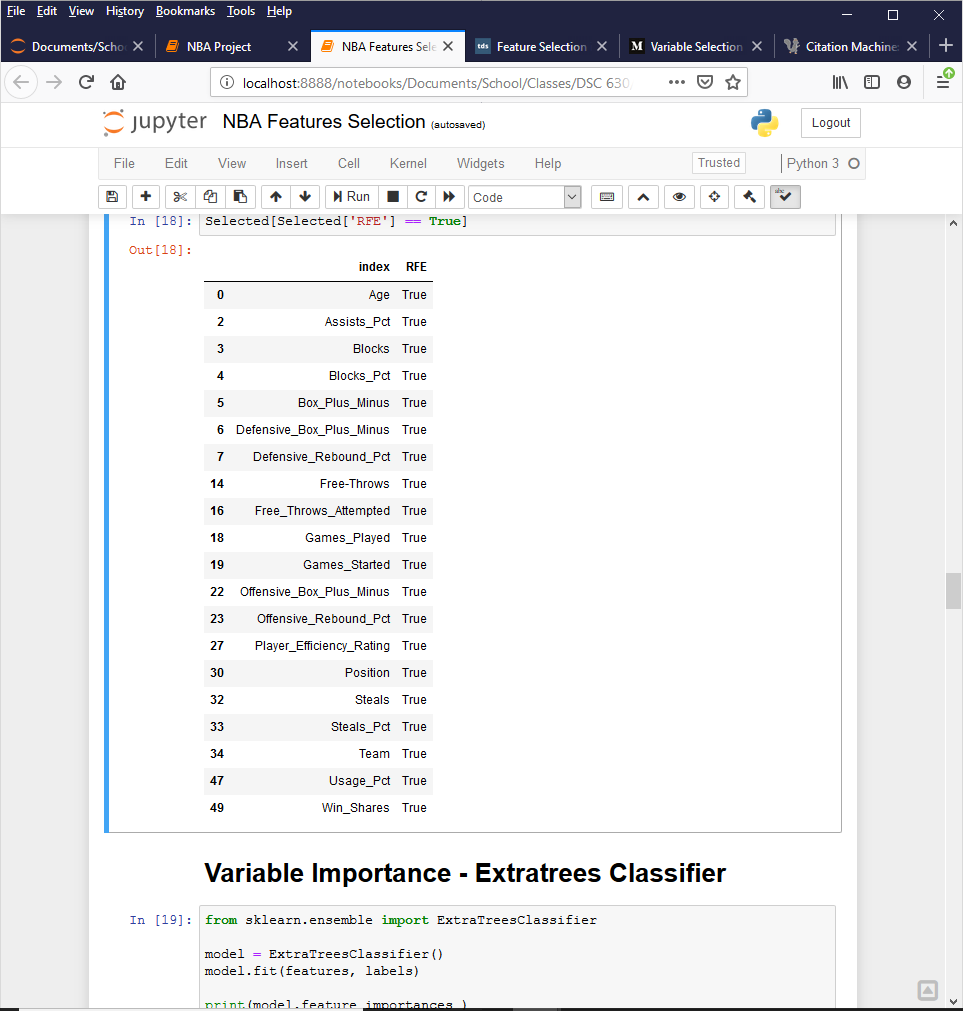


Figure 4. Output from RFE calculation.

* **Variable Importance: Extra trees Classifier**—this technique uses the ExtraTreesClassifier module from the sklearn library. In concept, the Extra Trees Classifier is very similar to a Random Forest Classifier and only differs from it in the manner of construction of the decision trees in the forest. Each Decision Tree in the Extra Trees Forest is constructed from the original training sample. Then, at each test node, each tree is provided with a random sample of k features from the feature-set. From this, each decision tree must select the best feature to split the data. This random sample of features leads to the creation of multiple de-correlated decision trees.[10] The output from this process is shown in Figure 5. The first column is the DataFrame column number, the **index** column is the name of the dataset variable, and the **Extratrees** column is the algorithm result. Due to the length of the output, only the first 10 values are shown.

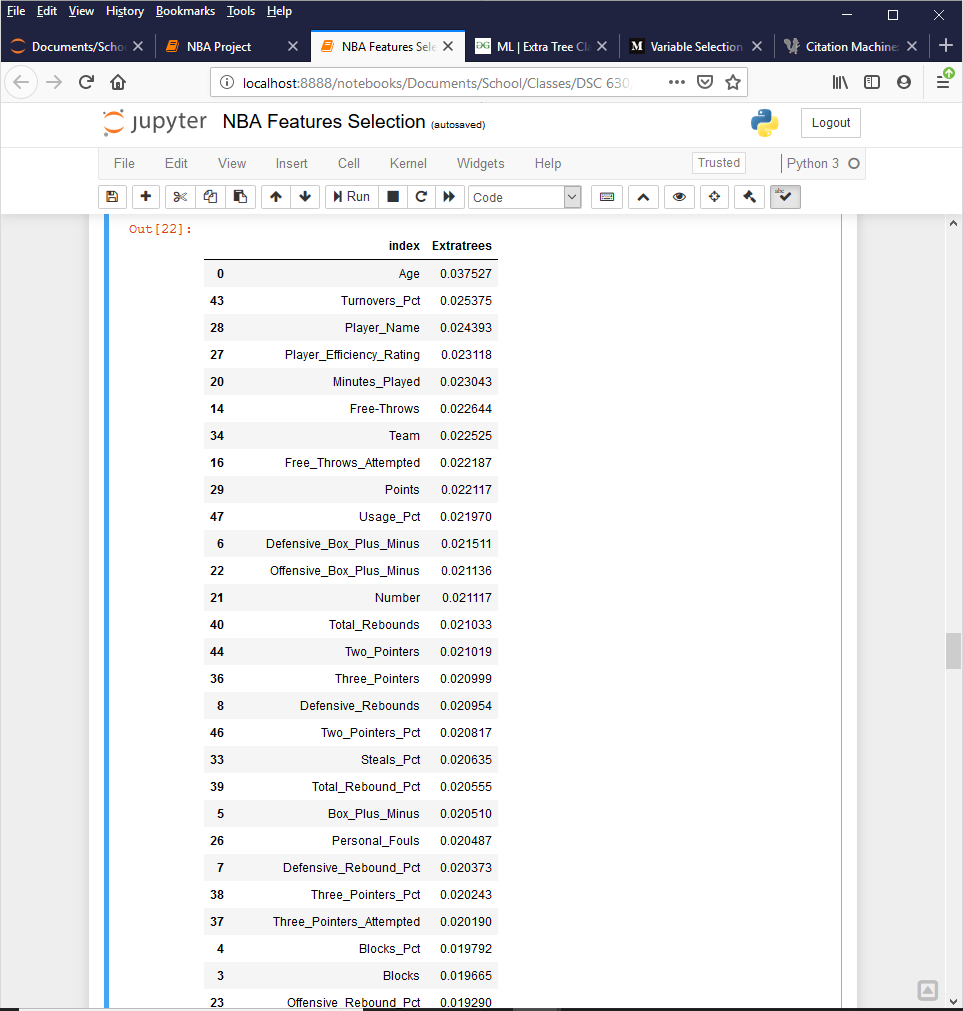


Figure 5. Output from Extra Trees Classifier.

* **Chi Square**—this method is also from the sklearn library. The Chi Square Test is used in statistics to test the independence of two events. In feature selection, the two events are occurrence of the feature and occurrence of the class. In other words, we want to test whether the occurrence of a specific feature and the occurrence of a specific class are independent. When the two events are independent, the observed count is close to the expected count, thus a small chi square score. So a high chi square value indicates that the hypothesis of independence is incorrect. In other words, the higher value of the chi square score, the more likelihood the feature is correlated with the class, thus it should be selected for the model.[11] The output of the chi square test is shown in Figure 6. The first column is the DataFrame column number, the **index**column is the name of the dataset variable, and the **Chi\_Square** column is the calculated chi square value. Due to the length of the output, only the first 10 values are shown.

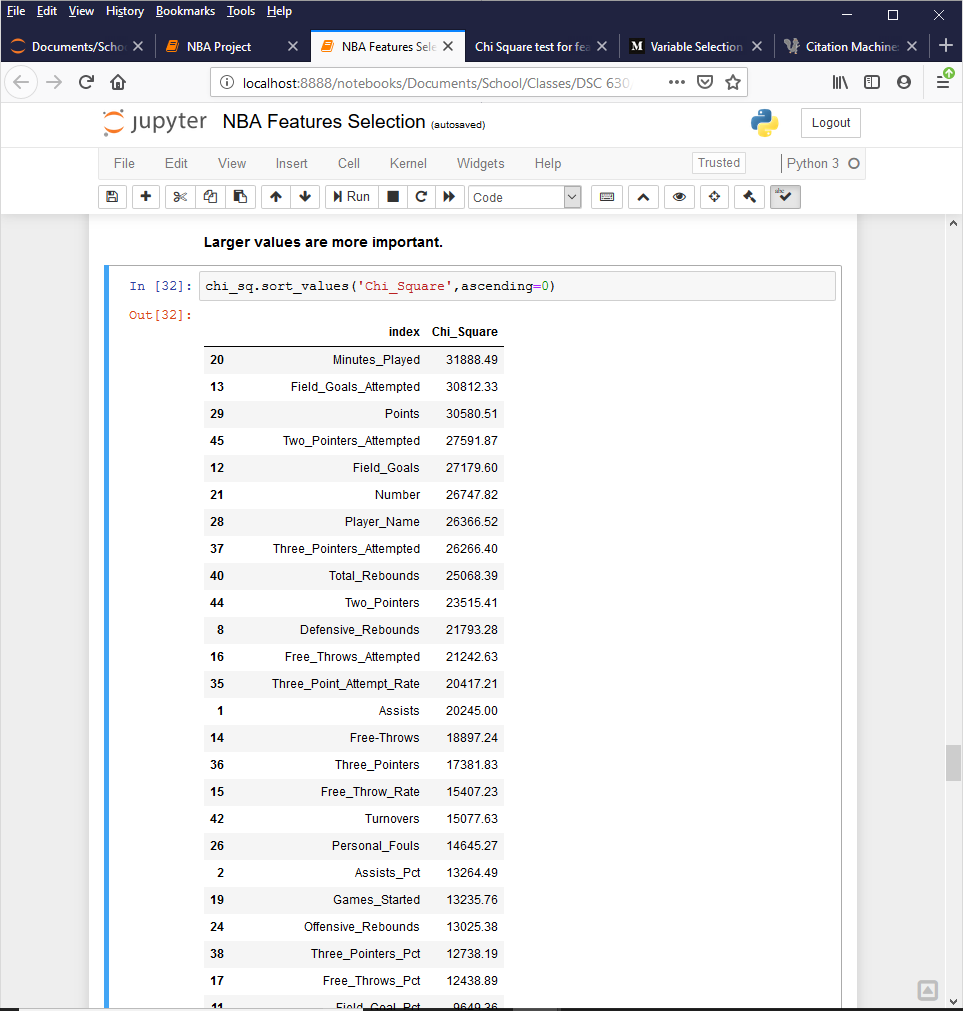


Figure 6. Output from the Chi Square calculation.

* **Lasso Regression (L1)**—this method was also used from the sklearn library. Regularisation consists in adding a penalty to the different parameters of the machine learning model to reduce the freedom of the model and in other words to avoid overfitting. In linear model regularisation, the penalty is applied over the coefficients that multiply each of the predictors. From the different types of regularisation, Lasso or L1 has the property that is able to shrink some of the coefficients to zero. Therefore, that feature can be removed from the model.[12] The output from our Lasso Regression is shown in Figure 7. The first column is the DataFrame column number, the **index** column is the name of the dataset variable, and the **L1** column is the Lasso Regression result. Like with the RFE output, the values of the L1 column are shown as True because they have not been eliminated in the Lasso Regression selection process. In other words, these remaining variables are considered important to the overall predictive model. The output below (in alphabetical order) shows all of the variables that returned a True value.

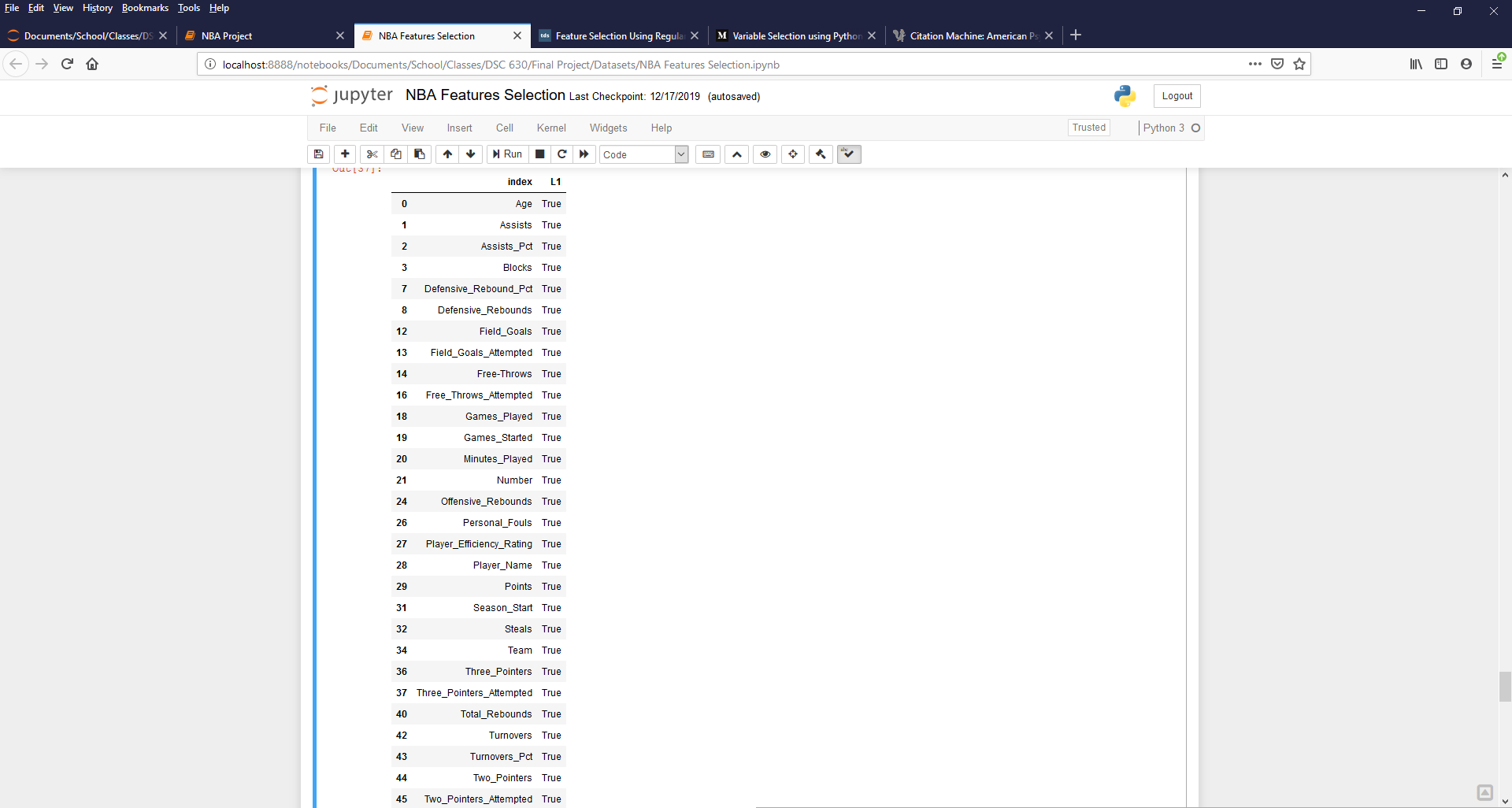


Figure 7. Output from Lasso Regression (L1) calculation.

* **Scoring Table of Final Results**—a scoring table was constructed that tallied all of the different feature selection methods. From this table, we were able to determine which variables should be included in our model. The scoring table, listing only values with a final\_score above 2, is shown in Figure 8.

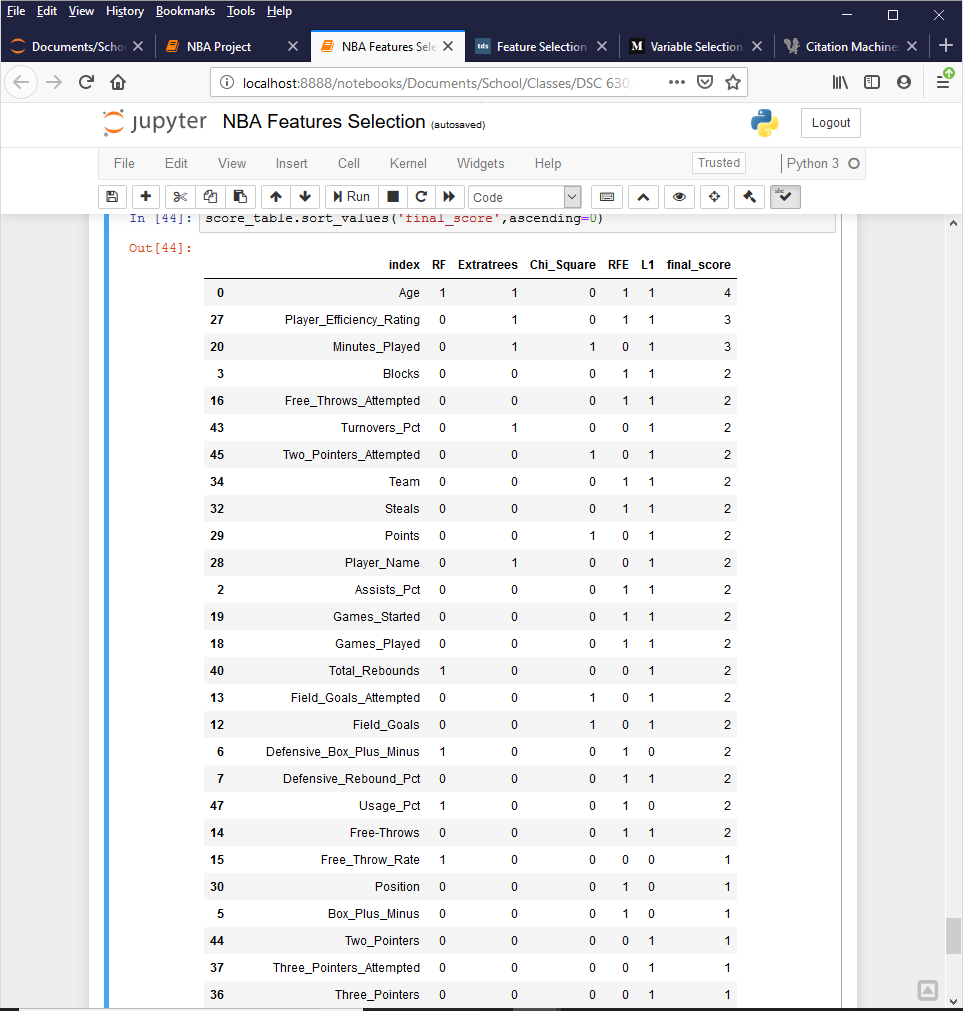


Figure 8. Scoring table of all feature selection results.

We are obviously going to include Age, Player\_Efficiency\_Rating, and Minutes\_Played, since they scored the highest. After that, however, we have many other variables that all returned a sum score of 2. We eliminated some of these features by looking at the correlation matrix.

* **Correlation Matrix**—(Taylor should probably do this section)

*Phase III: Feature Finalization and Model Build*

**Discussion/Conclusion**

**Acknowledgments**

**References**

[1] Gleeson, S. (2019, June 19). NBA draft: Ranking the five biggest busts as the No. 1 pick. Retrieved from https://www.usatoday.com/story/sports/nba/draft/2019/06/19/nba-draft-ranking-five-biggest-busts-no-1-pick/1488407001/.

[2] Davis, S. (2019, June 19). WHERE ARE THEY NOW? The biggest NBA Draft busts of all time. Retrieved from https://www.businessinsider.com/nba-draft-busts-2015-6.

[3] Ye, D. (2019, July 4). Top 10 Most Underrated NBA Players of the 2018-19 Season. Retrieved from https://howtheyplay.com/team-sports/Top-10-Most-Underrated-NBA-Players-of-the-2018-19-Season.

[4] Casalan, A. (2018, October 20). NBA player stats 2017-18. Retrieved from https://www.kaggle.com/acasalan/nba-player-stats-201718.

[5] These are the 2016/17 salaries of all NBA teams. (n.d.). Retrieved from https://hoopshype.com/salaries/2016-2017/.

[6] Abbott, D. (2014). *Applied predictive analytics: principles and techniques for the professional data analyst*. Indianapolis, IN: Wiley.

[7] Krishnan, S. (2019, December 20). Variable Selection using Python - Vote based approach. Retrieved from https://medium.com/@sundarstyles89/variable-selection-using-python-vote-based-approach-faa42da960f0.

[8] Dubey, A. (2018, December 15). Feature Selection Using Random forest. Retrieved from https://towardsdatascience.com/feature-selection-using-random-forest-26d7b747597f.

[9] Rade, D. (2019, September 2). Feature Selection in Python - Recursive Feature Elimination. Retrieved from https://towardsdatascience.com/feature-selection-in-python-recursive-feature-elimination-19f1c39b8d15.

[10] Gupta, A. (2019, July 26). ML: Extra Tree Classifier for Feature Selection. Retrieved from https://www.geeksforgeeks.org/ml-extra-tree-classifier-for-feature-selection/.

[11] Chi Square test for feature selection. (2018, July 5). Retrieved from http://www.learn4master.com/machine-learning/chi-square-test-for-feature-selection.

[12] Dubey, A. (2019, February 4). Feature Selection Using Regularisation. Retrieved from https://towardsdatascience.com/feature-selection-using-regularisation-a3678b71e499.