Capstone 1 Machine Learning In-Depth Analysis

Contents

Introduction

In the previous portions of this project, we had established statistically significant correlations between player salaries and several statistics, including VORP (Value Over Replacement Player), PER (Player Efficiency Rating), and 3P% (three-point percentage), among others. We also noticed smaller correlations between salary and a few other stats. Thus, for the first part of this project, we'll attempt to predict player salary through the best regression model. However, we're not particularly optimistic that we'll be able to develop a highly predictable model given that there are many other factors that help determine salary than just stats, as we'll discuss in more detail later.

We had also established that there is a statistically significant difference between the salaries of different player positions, with point guards historically making less money on average than every other position, with centers and power forwards typically making the most. However, there are other differences between positions as well. For example, point guards tend to have more assists on average than other positions, while centers and power forwards have a higher block rate. Shooting guards and small forwards may have more balanced stats overall, while also averaging higher points per game. As such, for our second analysis, we'll attempt to predict player position from statistical performance.

The Jupyter Notebook associated with this report can be found on Github:
https://github.com/kjd999/Springboard-files/blob/master/Capstone%20Project/nba%20machine%20learning%20analysis.ipynb

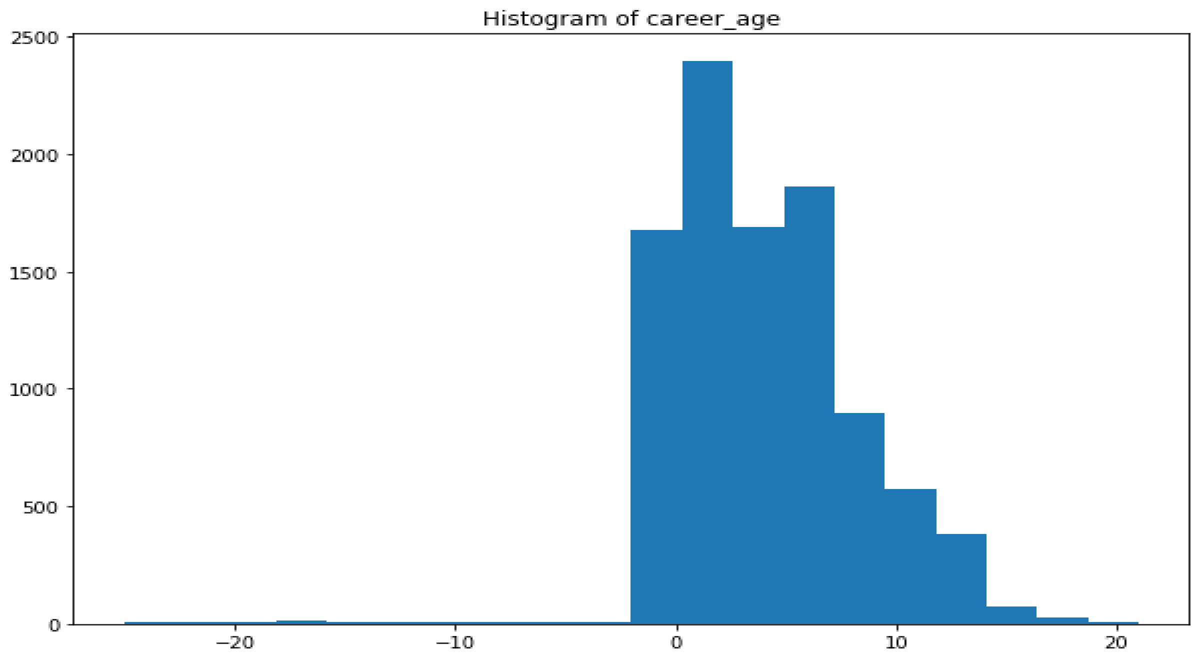
Predicting Player Salary from Statistical Performance

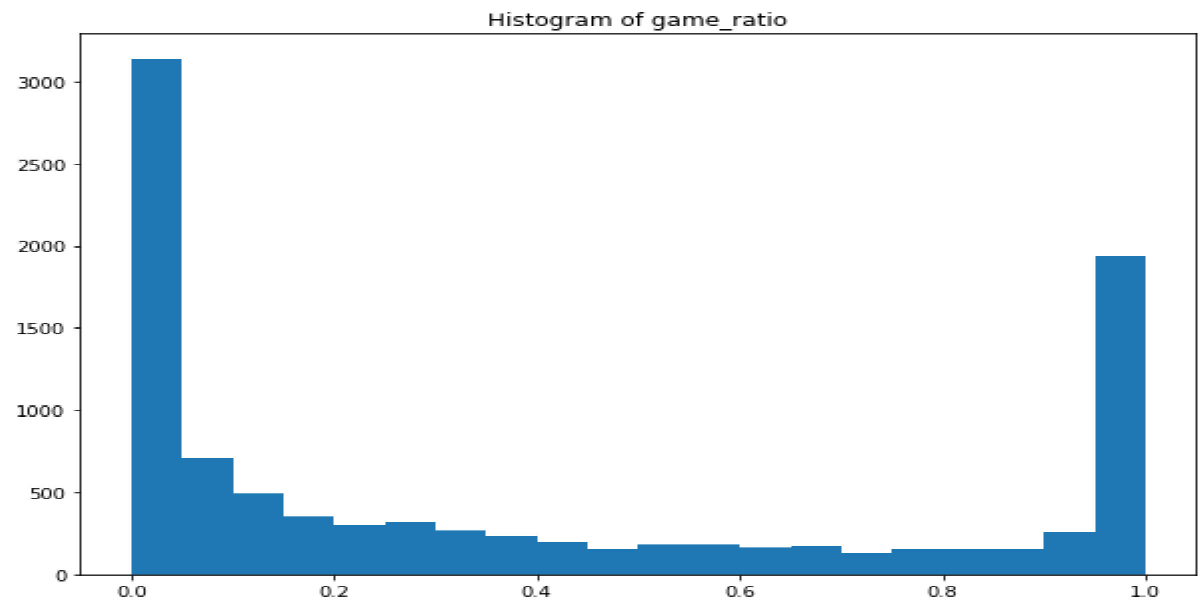
Before we start, it should be noted that a previous attempt at predicting salary using linear regression resulted in 57% accuracy. It was the result of applying linear regression in combination with PCA. In that attempt, we didn't do any feature engineering, nor did we include any categorical features. We're not including the details in this notebook in order to allow for a more streamlined analysis that's solely focused on the steps that resulted in achieving our highest accuracy.

We should also mention that anytime we refer to 'accuracy' during our salary analysis, what were are referring to is actually the coefficient of determination, r-squared. For our purposes, and especially if this analysis is partially aimed toward a client, it is easier to understand the concept of accuracy than r-squared.

We're going to engage in a bit of feature engineering and add a couple of new columns to our dataset in hopes that they may improve our predictive abilities. The first is career_age, which just gives us how long a player has been in the league for a given year. Players who have more experience tend to get paid more. The second column is game_ratio, which is the ratio of games started to games played. Typically, the best paid players start the most games for their respective teams.

Computing the correlations of these features to salary, we find that career_age correlates 45.7% and game_ratio correlates 51.6%. While not strong correlations, they may still help our model.



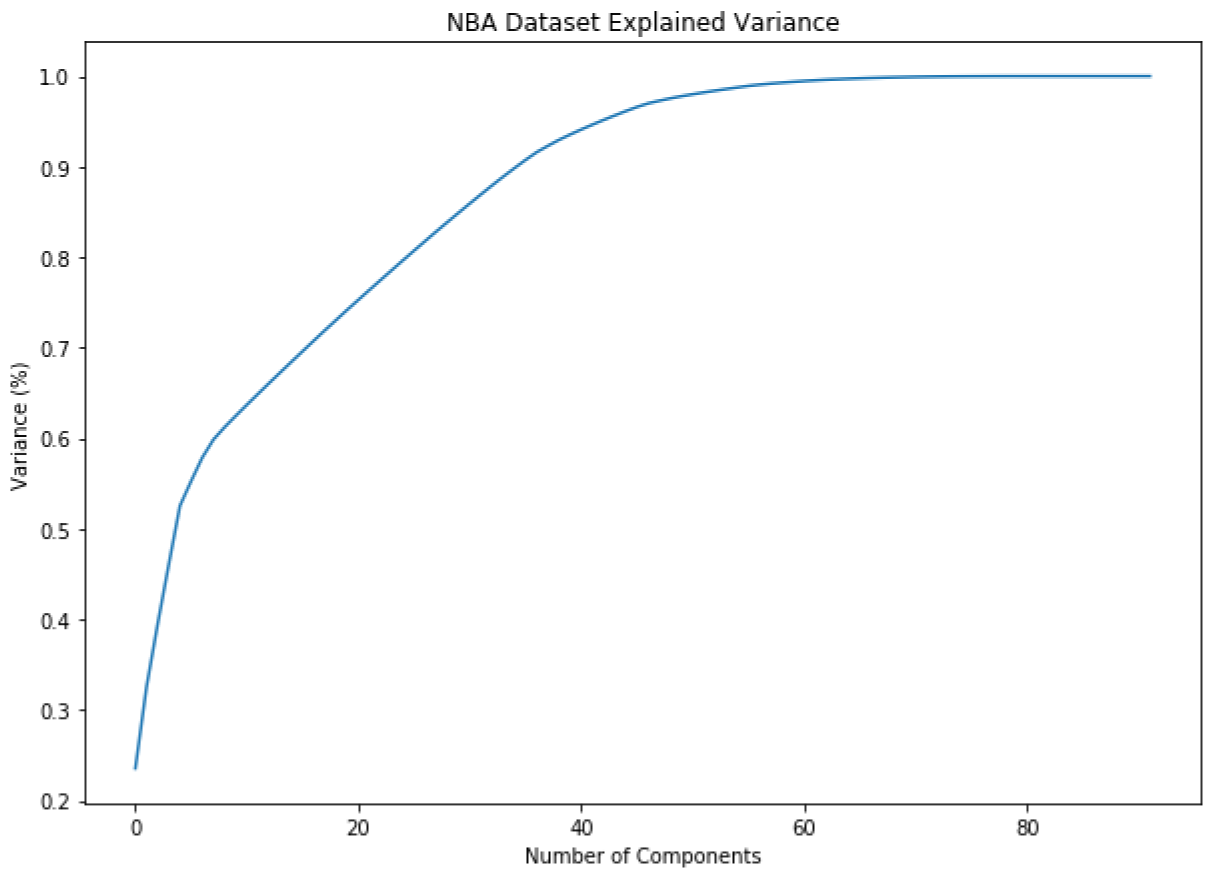


It should be noted that there are a small number of values in the career age histogram. This is due to a slight error in our data in which a very small number of players had an incorrect starting-year value. It should make a negligible difference to our results. Both histograms display an uneven distribution of the histograms suggesting that there is a good deal of variance in the data.

We have two categorical features that we would like to include in our analysis - Team and Pos. However, in order to include them, we need to innumerate them using OneHotEncoder and ColumnTransformer. We also pass our numeric variables through ColumnTransformer and then develop a pipeline that concatenates both sets and applies linear regression to training data in order to predict test data. We use a similar pipeline for several other regression models to determine which seem the most promising. The additional regression models that we test are: random forest, ridge, lasso, and SVR. What we find is that random forest is the most successful, with a test accuracy of around 65%. Yet, it also has a training accuracy of 93%, which suggests overfitting. As a result, not only do we need to increase our test accuracy, but decrease overfitting at the same time.

Applying PCA

Due to applying ColumnTransformer, our number of components more than doubled from around 40 to more than 80. We decide to apply PCA to our model in order to reduce dimensionality, in hopes that it will cut some of the noise from our data and allow for greater accuracy.



As the graph suggests, close to 100% of our variance can be explained by about 70 components. As such, we decide to pass PCA into our pipeline, along with MinMaxScaler, while specifying 70 components. However, while we have reduced overfitting, our test accuracy drops by 5 percentage points compared to our initial calculation, suggesting that PCA is actually hurting our model. PCA is usually more successful for linear models. Since we're using a random forest model, PCA may not be as effective.

We decide to drop PCA altogether, and through trial and error, we tune our random forest parameters and find that 50 n_estimators, a max depth of 15, and 10 minimum leaf samples provide us with 68% accuracy. Yet, because we used trial and error, there may be other combinations of parameters that allow for greater accuracy.

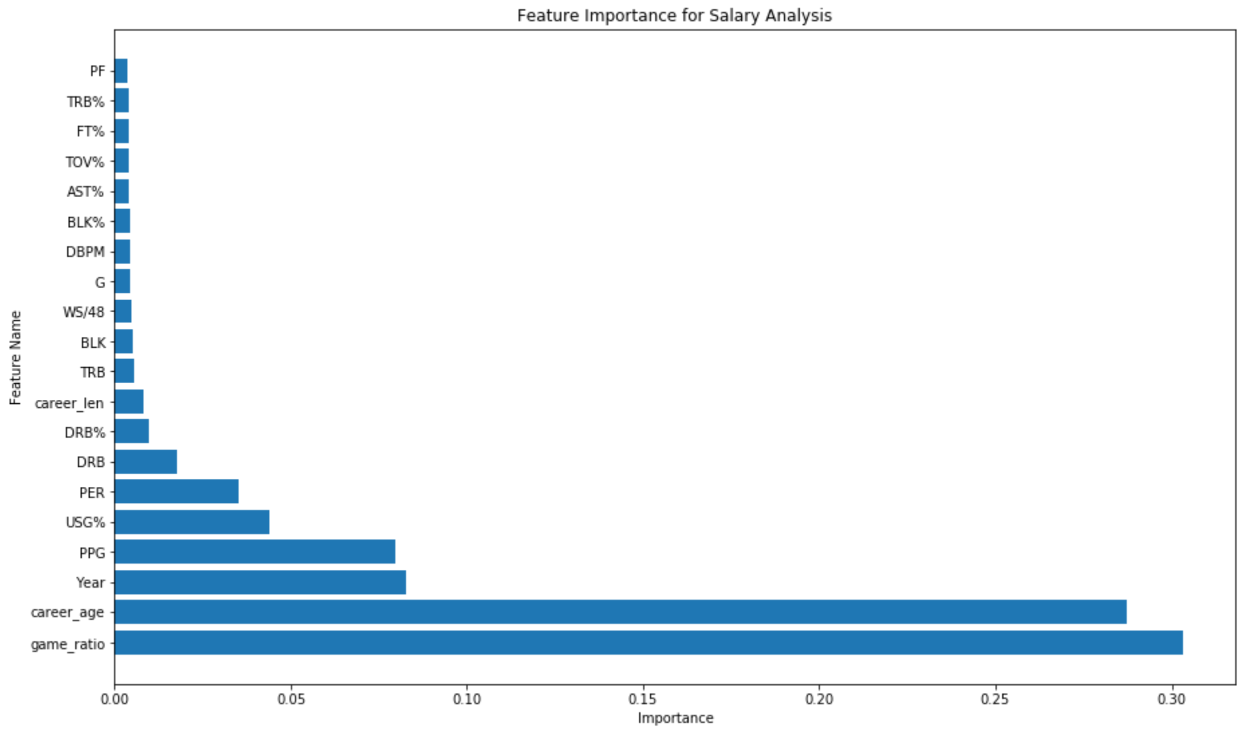
Applying GridSearchCV

In an attempt to find the optimal mix of parameters, we decide to apply GridSearchCV. After running GridsearchCV, we find that our trial and error approach was rather accurate. The only parameter we need to add is a min_samples_split value of 6. Upon fitting our random forest pipeline with our new parameters, our accuracy increases to 71%. Given where we initially started at 57%, this 14 percentile point increase through the use of feature engineering and parameter fitting is significant. And yet, 71% is still not accurate enough to predict salaries consistently.

Feature Importance

While it was great to see a nice increase in accuracy, we don't really have a good sense of how the model managed to do what it did. That is, it would be great if we could figure out which features the model found to be most important in making successful predictions. As such, we'll develop a feature importance table and graph the most important features for our model's accuracy.

Since our categorical and numerical names and feature values are part of a pipeline that involves OneHotEncoder and Columntransformer we simply can't call a feature_improtance_ command on or model. We need to extract the transformer array values and combine them with the feature names into a new dataframe. From there, we can sort the values and create the following graph:



Interestingly, the two variables that we created, game_ratio and career_age, had far greater importance for our model than our preexisting variables.It suggests that the amount of time that a player spends in the league and how many games a player starts are fairly strong indicators of how well he'll be paid. Some of the other variables that we found important in our exploratory data analysis, such as PPG (another variable that we created in a previous part of this project), Year, USG%, and PER also had a noticeable difference. While we had previously found that 3PA and 3P% both had a statistically significant correlation with salary, they both barely cracked the top 30 in terms of importance.

Conclusions

While we do get an increase in accuracy, this is still well below the ability to predict a salary accurately. There may be several reasons why we may never achieve high accuracy values given this dataset:

1. Recall from our previous work on this dataset that there is some bias. Namely, the dataset only includes players whose careers began in 1991 and after. So, many players who played in their early '90s but began their careers in the '80s were not included. This may slightly decrease our accuracy. However, this is probably the least likely reason for low accuracy.

2. Stats haven't increased at the same rate as salaries. In other words, while average salaries have increased substantially since the 1990s, many statistical values have not increased at the same rate. As we saw in a heatmap in a previous portion of this project, most correlations with salary were fairly low, rarely being greater than 60%. As such the model would have a difficult time projecting stats to salary values because players are being paid substantially higher wages for only slightly better statistical output.

3. One may suggest that another possible reason may be inflation. The inflation rate in the U.S. has varied throughout the years, but salaries in the 1990s would be larger if the inflation rate were taken into consideration. However, this approach still may not make our model very accurate. In the U.S., salaries have never kept pace with the rate of inflation. Even as inflation has increased, salaries have remained largely stagnant. Yet, there's little reason to believe that NBA salaries have much to do with inflation at all. As suggested below, the NBA often lives inside it's own little bubble where salaries get dictated by a number of factors not necessarily related to current economic climates.

4. There may also be a variety of factors related to players being underpaid or overpaid. For example, teams competing with one another over particular players in a given situation drive up the price of a player in such a way that such an increase may not be warranted outside of that context (such as playoff contenders needing to pick up last minute help, a team looking to rebuild during the offseason with very few strong free agents to choose from, etc). There are also considerations of labor union influenced league minimums, which have lead to average salary increases, regardless of player skill. Player popularity may also play a role as teams may be eager to cash in on advertising and ticket sales from a popular player, often resulting in a player getting paid more that his statistical output may warrant. Additionally, as the NBA has become a global brand over the last decade, more money has flowed into the organization, resulting in much of that money being distributed to players.

All of these various factors make it difficult to predict salary simply from stats. In fact, without datasets that took into account many of the factors mentioned above, we likely couldn't ever make an accurate model (anything greater than 95% accuracy) for this particular question.

Takeaway for Players

While our model was not able to accurately predict player salaries, this project is still useful for players to consider when evaluating their financial worth in the NBA. What our model did allow us to determine is that statistical performance is not the sole factor in determining a player's salary. As mentioned above, there are a whole host of factors that may not directly be connected to statistical output. Having that insight can prove useful for players. They, for example, can learn to better market themselves. Even if they happen to have mediocre stats, by marketing themselves properly, they can make themselves more financially lucrative to a team. This could be in the form of advertising deals, both for the team and player. The increased popularity can lead to increased ticket sales and, of course, a greater salary for that player than his on-court performance warrants.

Predicting Player Position from Statistical Output

Just to note, the main goal of our machine learning analysis was to determine how well we could predict salary, which was demonstrated above. This next section on predicting player position is an interesting side analysis, but we won't go into as much depth as with the salary analysis due to time constraints.

There are five different positions in the NBA - center, power forward, small forward, shooting guard, and point guard. There are noticeable differences between positions. For instance, point guards average more assists, while centers and power forwards have higher numbers of blocks. Also, as we found in a previous project, point guards average a lower salary than any other position, particularly centers and guards. However, we are also conducting a multi-classification analysis rather than a binary classification. As a result, matching five different positions will be more difficult than simply confirming whether a given player is a particular position.

The overall procedure will be very similar to that used during our salary analysis. We'll add career_age and game_ratio variables to our dataset. We'll also get rid of any unnecessary variables and eliminate the position column since that's what we'll be trying to predict.

We begin by transforming our categorical data through OneHotEncoder and concatenating both the numerical and categorical columns with ColumnTransformer. We then import several classification models - logistic regression, decision trees, random forest, linear discriminant, Gaussian NB, KNN, and SVC. We'll create a pipeline for each model along with our transformed dataset.

Running each model analysis results in random forest, logistic regression, and SVC being our most promising models, each with around 65% accuracy. We then decide to tune the hyperparameters of each model in hopes of increasing their respective accuracy. We do this by implementing GridSearchCV.

Applying GridsSearchCV

For both logistic regression and SVC, due to time constraints, we'll only be tuning C. Gridsearch finds that for both models, 10 is the best value for C. When we apply Gridsearch to the random forest model; it recommends: n_estimators=50, min_samples_split=6, and min_samples_leaf=8. After fitting our models with our new parameters, we get following accuracy scores:

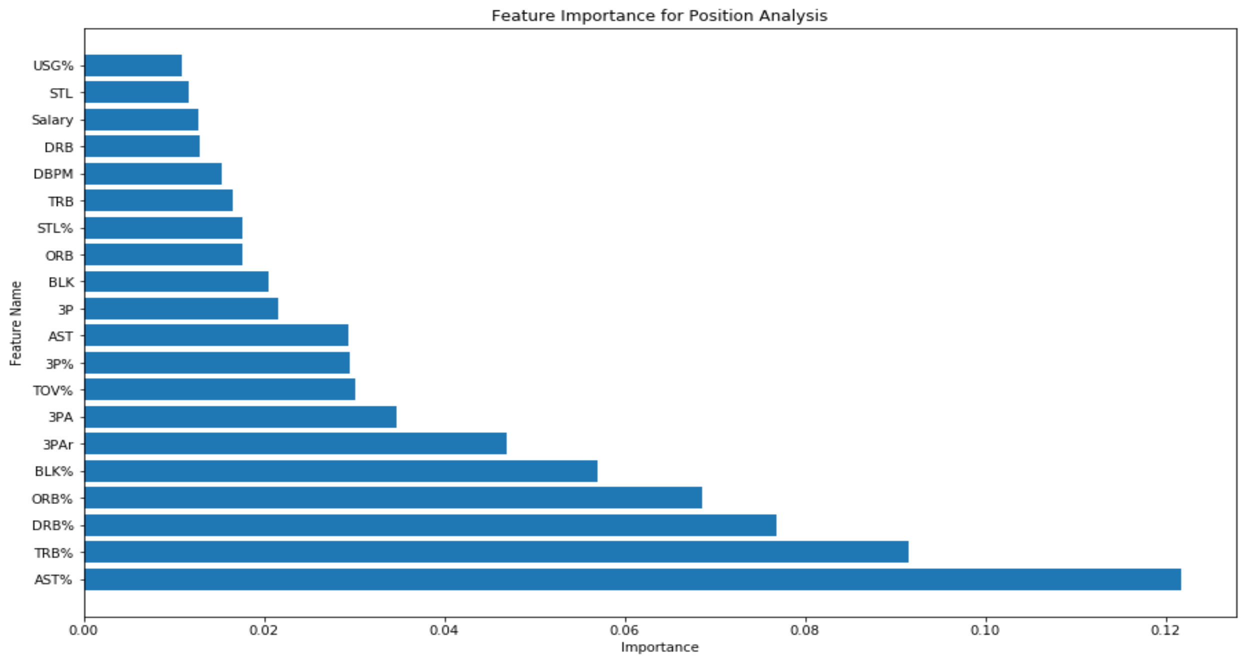
Logistic regression - 67%
Random forest - 67%
SVC - 69%

SVC performs the best, but not by much. Our new random forest model also allowed us to decrease overfitting, with a 14 percentile point decrease (99 to 85) in the training score. While the test accuracies are still relatively low, the increases are promising given that we only did a small amount of parameter tuning.

Feature Importance

Out of sheer curiosity, we'll also find the feature importances for the random forest classifier. While SVC was our most successful classifier, we already have a template for finding feature importances of random forest models. Given our time constraints, this is more efficient than developing a method to extract SVC feature importances.

Following a similar procedure to that described above for our salary model, we acquire the following feature importance graph:



It appears that many of the stats that help distinguish players of different positions are the ones with the highest importance levels. AST%, which measures the percentage of plays that a player assists o when on the court, is likely to be higher among point guards. The next three stats - TRB%, DRB%, and ORB% (all rebounding stats) - are all likely to be higher among centers and power forwards. 3-point related stats are more likely to be higher for point and shooting guards. Salary, which we had previously established as making a statistically significant difference among positions, is also one of the top 20 factors.

Conclusions

While we could tune our models further, it's doubtful that their accuracies would be significantly better. We could likely get our accuracies into the 70% range, but it's unlikely that we could get into the 80% range. There are several reasons why:

1. Our models are looking for accuracies for all positions rather than running a binary analysis to confirm whether a given player is a center or point guard or any of the other three positions. If we were testing for just one position, the accuracy would be much higher, likely in the 80% range. We'll elaborate further on this point below.

2. As we mentioned at the start of this analysis, while there are considerable differences between positions such as centers and power forwards averaging higher salaries and blocks or point guards averaging the lowest salaries and assists, the small forward and shooting guard positions muddle things up a bit. These last two positions tend to have more well-rounded stats than the other three positions. However, if we were using a binary model, we could likely get high accuracy for the three most distinguishable positions. But since our search isn’t binary, the model is probably getting confused by, for example, small forwards with high assist values or shooting guards with a high number of blocks. In addition, there may be certain outliers, such as point guards with high salaries or centers with below average block values. All of these various factors, combined with a multi-position analysis result in lower accuracy levels.

3. Additionally, there’s one other factor that we weren’t able to take into account - height. Height varies greatly between positions, with point guards typically being the shortest players and centers and power forwards being the tallest. Had we had access to that data, our model may have been able to better distinguish between positions.