Data Science I Course Project Report

Nachiket Hinge

Allan Waweru

Vinny Palermo

**Abstract**

The objective of this project was to build models to review the NBA player's statistics in games and predict what salary should be given to them. We found the two datasets on Kaggle, Player Stats, and Player Salary. Using the pandas library of python, we merged the two datasets on player names. We created three different machine learning models: Random Forest, KNN, and Linear Regression. Our Random Forest model had a test accuracy of 61.4%, KNN had a test accuracy of 61.8% and Linear Regression's test accuracy of 40.4%.

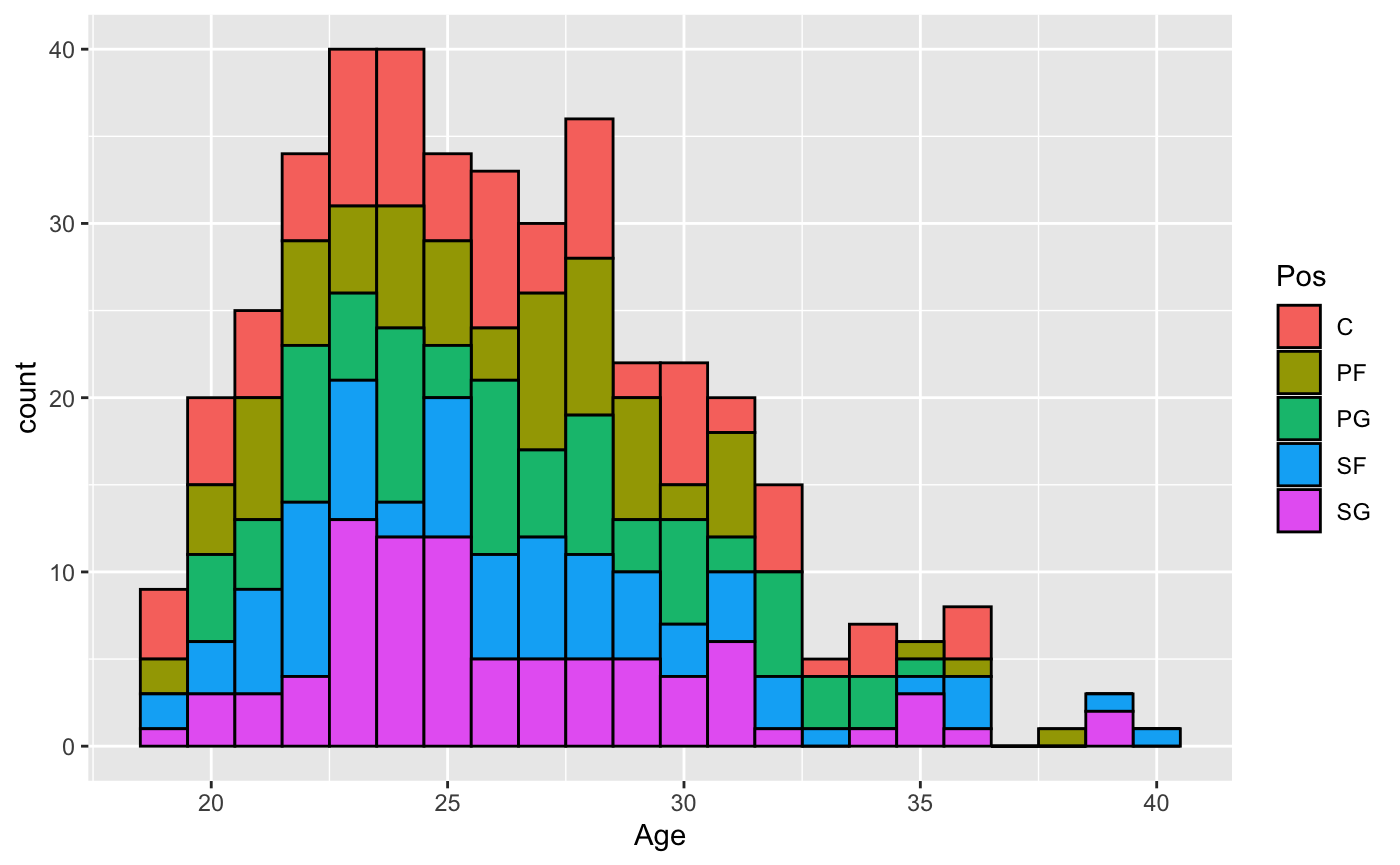
1. **Introduction**

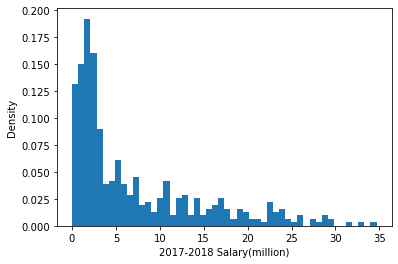
The project is to create a model and predict what salary should be given to the NBA players based on their statistics in the NBA. Our group found it intriguing to work on a model close to a real-world problem.

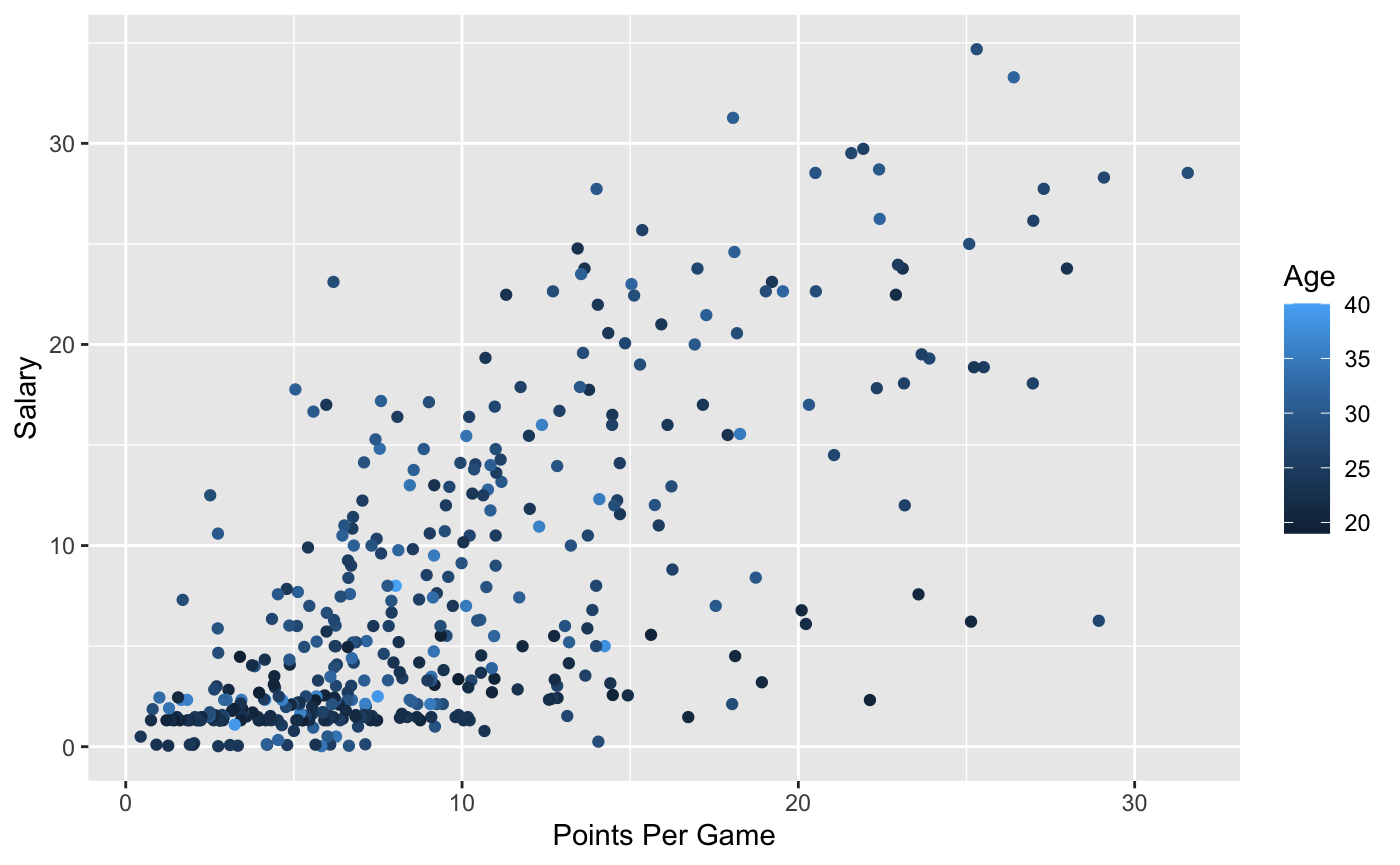
The project's main objective was to conduct an exploratory analysis and develop models that are used in predicting how much salary should be given. The analysis and the models support each other in ensuring that the prediction is nearly accurate. The data sets used in this project are a training data set, which has a model of 353 players, and testing data set with 89 players

1. **Data Analysis/Preprocessing**

For the data analysis and preprocessing, we visualized our data first to understand the trends we would be working with and make conclusive observations. We started by creating histograms to know the distributions of salary and age distribution. Looking at the salary distribution, we realized that much of our response variable would be near 5 to 10 million. Regarding age, we observed that age is not evenly distributed in the NBA, and there are more players aged 20 to 25 than in the 30s. We then plotted some correlation plots to understand individual variables and how they determine a player's salary. After plotting various independent variables with the salary variable, the Points Per Game variable stood out to correlate the most with salary. The correlation was a strong positive correlation of 0.70. In the chart, we also included age as a factor shown by the color of the points on the chart. As mentioned before, more players between the ages of 20 to 25 had fewer points per game and less salary than the others. This could be explained as the young players will not have enough NBA experience as the other older players.







We decided to split the data in training and testing set for preprocessing to build the models. We used the 70 to 30 split method for all the models and trained and tested on 353 and 89 datasets.

1. **Machine Learning Models**

* 1. **Random Forest Model**

The first model created was the Random Forest model. This was created using the sklearn.ensemble library. The model was fit using the training set x and y train. The following code sample shows the implementation of Random Forest.

→ clf = RandomForestRegressor(random\_state=42)

→ dtree = clf.fit(x\_train, y\_train)

→ yp = clf.predict(x\_test)

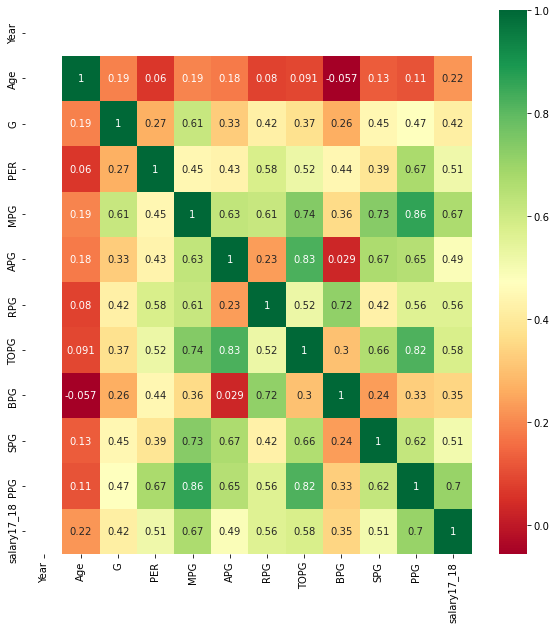
The prediction values for the testing set were determined based on the training values. This was done by using the predict() method. Additionally, an R-squared value was obtained to check the efficiency and usefulness of the model. The following code demonstrates the prediction of the values and the creation of the regression report.

→ print(clf.score(x\_test, y\_test))

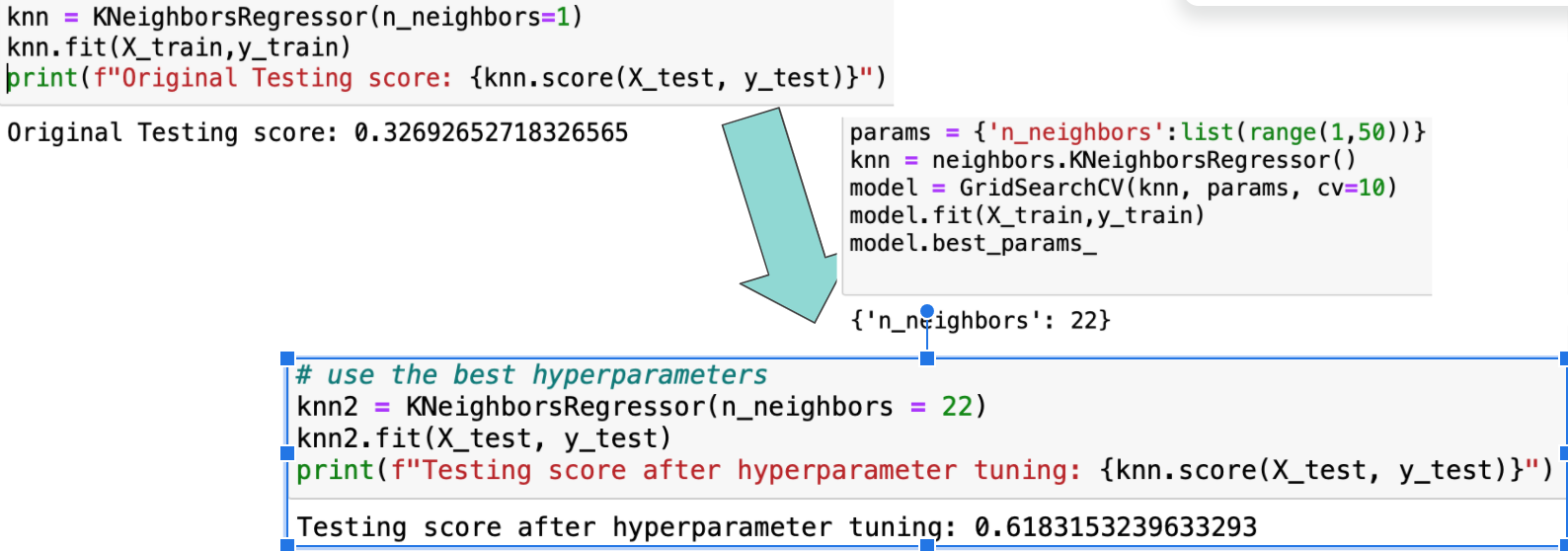
→ print(clf.score(x\_train, y\_train))

* 1. **KNN Model**

KNN is perhaps the most straightforward model to implement compared to the other two in this project. Instead, it is known as a 'lazy learner' because when we supply training data to this algorithm, it does not train itself at all; instead, all of the data is kept and used at run-time for prediction. We can still test the model to assess the quality of the predicted NBA salary. For KNN, it helps to keep the test set small because if you predict unknown values by averaging a few points closest to it, it helps if you have points close to the one you wish to predict. This is why I specifically opted for a train/test split of 80/20.

KNN is very susceptible to the curse of dimensionality, meaning it is better suited for lower dimensional data. This means that the algorithm's accuracy greatly benefits from feature selection. I used a correlation matrix plot and an Extra trees classifier to find the most important features related to player salary.

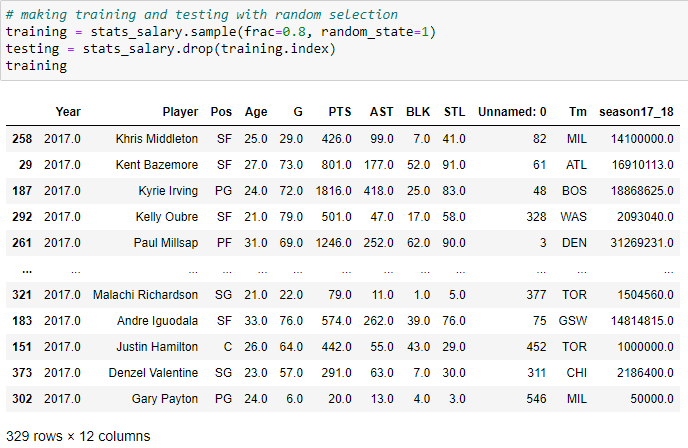
As we know, 'K'in the KNN algorithm is based on feature similarity; therefore, finding the optimal value of k, a process called parameter tuning, is critical for better model accuracy. This is a hard process, but for this project, I chose to use GridSearchCV, which exhaustively considers all parameter combinations to find optimal 'k'.

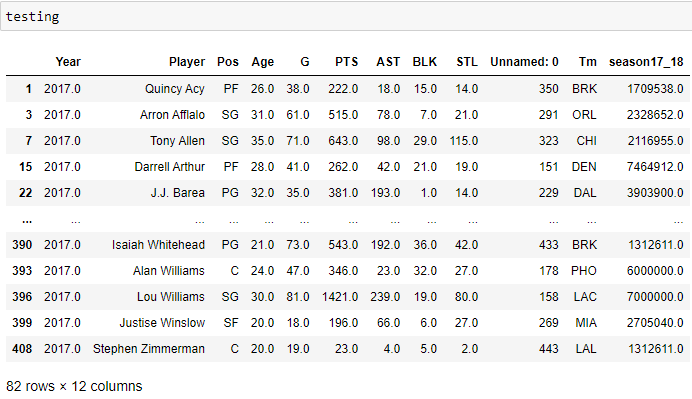


As shown, after implementing the GridSearchCV, I found the optimal k to be 22, which increased the testing score from a mere 32.7% to 61.8%.

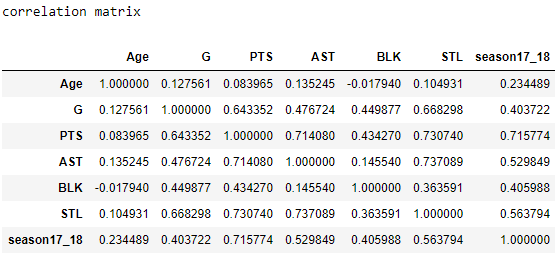
* 1. **Multiple Linear Regression Model**

The last model we created was the multiple linear regression model, also imported from the sklearn library. The dataset was first combined to include all players for the 2017 season, then was split using an 80 to 20 split to separate the training and testing sets, as seen below.

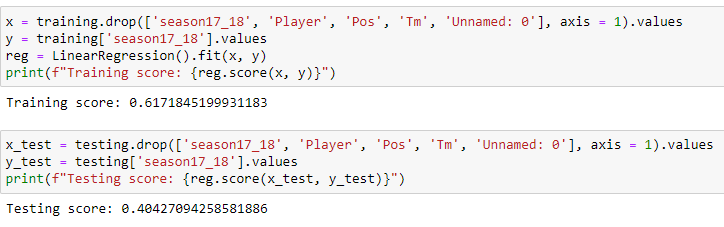




The correlation matrix below shows that the three with the highest correlation to salary were in order, points, steals, and assists out of the six statistics we were focusing on.



After separating training and testing, we used the following code to retrieve the training and testing scores 0.617and 0.404, respectively.



The testing score for the multiple linear regression model is only 40.4%, so this is not the best model for predicting NBA player salary based on the given statistics.

1. **Model Comparison and Conclusion**

Overall, the results we obtained are significant because this was the most important part of our analysis. To understand the efficiency of the models, we used the R-Squared method to understand how much of the data is explained by the model built. The R-Squared values we got for each of the models were:

* Random Forest Regressor: 61.4%
* K-Nearest Neighbor: 61.8%
* Linear Regression: 40.4%

The results may be misleading due to the data, how they are partitioned, the different machine learning models we used, and how the models are configured. The data was imported from a raw file which is not always the best form for analysis. Before any regression or analysis was done, we trained the models, used the specified datasets, and cleaned them to remove any null values. As for the most preventable part of the analysis, we separated it into X and Y for training and testing sets to partition our data. More information will be found in this report's Data Analysis/Preprocessing section. Overall, the models we build we reasonably reasonable to be used in an actual world situation but to improve it, we would need more data instead of just the 2017-18 season data.