CSCI 183 Data Science
NBA Player Salary Predictive Model

Names/Teammates: Wallace Hwang, Aditya Ranade, Austin Wang, Travis Lee
June 3, 2022

All variables and what they represent:
['PTS', 'FG', 'FGA', 'FT', 'TOV', 'FTA', '2PA', '2P', 'MP', 'AST']
=
[Points, Field Goals, Field Goals Attempted, Free Throws, Turnovers, Free Throws Attempted, 2 Pointers Attempted, 2 Pointers, Minutes Played, Assists]

Background:

In this project, we will look at NBA player's stats to try to predict their salaries. In the NBA, salaries can range from anywhere between hundreds of thousands to hundreds of millions. Players are all of different calibers, and we wanted to see if their performance would be a good indicator of how much teams would offer to pay them. In this project, we decided to go with more recent data (2019-2021) so the prediction would be more accurate and less affected by inflation. Using player data from basketball stat websites and salary releases we will train the program to predict salaries based on performance statistics. Our dataset includes the player stats of NBA players from the seasons 2019-2020, 2020-2021, and 2021-2022. Our dataset also includes the salaries for the players from the seasons 2019-2020, and 2020-2021. We will be creating a model that will utilize the NBA player stats as x-values to predict what an NBA player's salary would look like in the 2021-2022 season, which will be our target variable (y-value) for this case.

To obtain the data we needed, we used two separate websites. The first website we used was Hoops Hype (https://hoopshype.com/salaries/players/) that we used to obtain the salaries of the players. Then, we used the website Basketball Reference (https://www.basketball-reference.com/leagues/NBA_2022_per_game.html) to obtain the data points for the statistics of the players. We then used the ConvertCSV website (https://www.convertcsv.com/html-table-to-csv.htm) to put the obtained data into a CSV file. The data that we are working with involves the player's name, age, and their respective stats for each season. The way the data was collected is by the NBA, which is collected throughout each season for each player.

Cleaning and Organizing The Data:

Once we obtained the data for the statistics in the form of CSV files, the first thing we had to do was get rid of duplicate header rows, as after every 20 entries, the headers for all the

columns for all the features were duplicated. After doing this, we realized that for the players who were traded during the season had three total rows that were allotted to them: one for their first team, one for their second team to which they were traded to, and one row that represented the total stats from their stats of both the teams they played on during the season. To remove the duplicates, we decided to keep the total, and remove the two individual rows from each team. Team (Tm) in the dataset was a categorical variable which we were not planning to use in our linear regression since we were only focusing on numerical variables, so 'TOT' as a team which represented total, did not matter. What mattered was we would have the stats for the whole year for that particular player. After we did this, we had the stats for all the players from the 2019-2020, 2020-2021, and 2021-2022 seasons. Now, we retrieved the salaries, but the problem was that these salaries for each of the seasons were not in the same order as the stats. So using pandas in Python, we were able to join the salary to each of the players by using their name as a key and using the "pd.merge" feature. This column becomes a concatenated column at the end of the data frame that we had the stats in. After we joined the salary to the players, we dropped the players who did not have a salary allocated to them or had a missing salary. This way, we cleaned the data and made sure there were no missing values. After this process, we had a dataset that was completely ready to perform multivariate linear regression on.

Design, Process, Implementation, and Results:

Our experimental design would be to first, find all the statistics for each of the players, and figure out which of the statistics have a positive linear relationship with the target variable, which is salary. We will use these variables and the salaries from the first two seasons that we are working with (2019-2020 and 2020-2021) to train our model, and see how well it can predict the salaries for the 2021-2022 season. We will perform single-variable linear regression first with variables that have the highest correlation coefficients, and then we will observe whether or not they have a positive linear relationship with the salary, and then we will use those variables in a multivariate linear regression. If our model does not perform well, we will remove outliers in the target variable and pull the data that is within a certain constraint for salary, and we will also change the number of variables we use in the multivariate linear regression to get a better model.

Our initial model had 16 features: (each of the following features are statistical averages per game, except for Age) Age, Games played, Minutes Played, Field Goals Attempted, Field Goal Percentage, Three-Pointers Attempted, Three-Pointer Percentage, Two-Pointers Attempted, Two-Pointer Percentage, Free Throws Attempted, Free Throw Percentage, Total Rebounds, Assists, Steals, Blocks, and Points. We chose these features because these were the ones we believed were the most important stats to measure a player's performance. However, our 16 feature model was very inaccurate. We believed it was caused by having too many features, thus creating too much noise. So we needed to narrow down our list of features.

To choose the particular variables from the most important statistics that we would use for the multivariate linear regression, we used .corr from the pandas library to figure out the

correlation coefficients of each of the numerical variables with the target variable, which is the salary. Out of all the correlation coefficients, we chose to use the variables which have a correlation coefficient of 0.6 or higher. Doing this, we still ended up with 10 variables to conduct multivariate linear regression which was quite high. So instead, out of all the correlation coefficients, we chose the 5 highest ones, which were PTS, FG, FGA, FT, and TOV. Below is the table of correlation coefficients for each of the variables in the dataset:

Figure 1: All Features and their Correlation Coefficients in regards to the Feature Variable of Salary

```
Age       0.403313
G         0.296024
GS        0.572729
MP        0.625788
FG        0.693492
FGA       0.686245
FG%       0.145855
3P        0.473993
3PA       0.481965
3P%       0.106283
2P        0.634592
2PA       0.645994
2P%       0.063536
eFG%      0.142452
FT        0.668325
FTA       0.659611
FT%       0.174445
ORB       0.243537
DRB       0.591799
TRB       0.529237
AST       0.630365
STL       0.495369
BLK       0.293308
TOV       0.659684
PF        0.380870
PTS       0.708797
Salary    1.000000
Name: Salary, dtype: float64
```

After we chose the five variables with the highest correlation coefficients, we decided to see if they had a positive linear relationship with the salary, to determine and confirm if they would be good choices while doing multivariate linear regression. Below are 5 graphs that represent the data and the linear regression results for each of the five variables chosen:

Figure 2: PTS vs. Salary

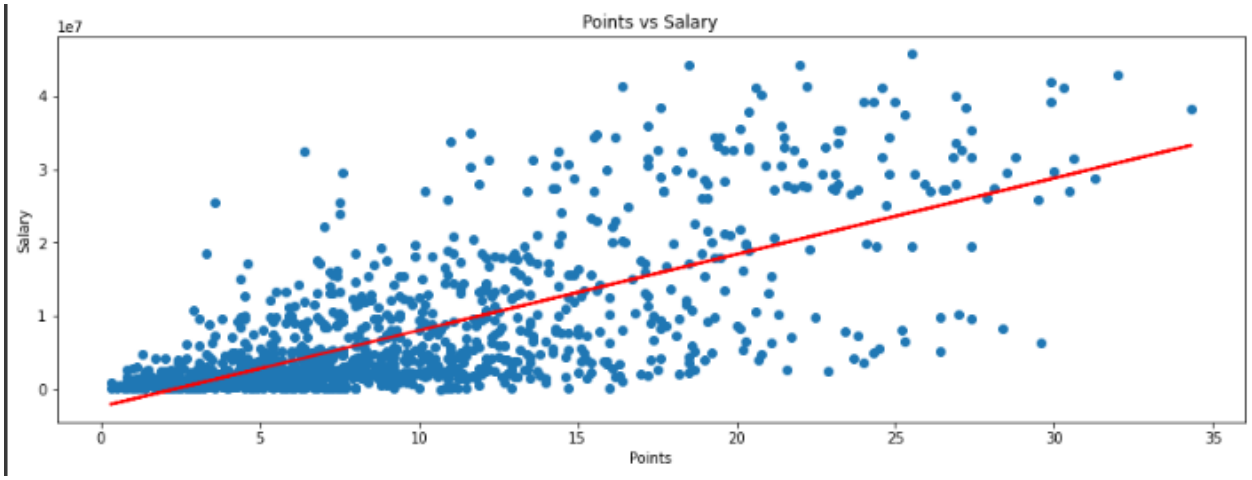


Figure 3: FG vs. Salary

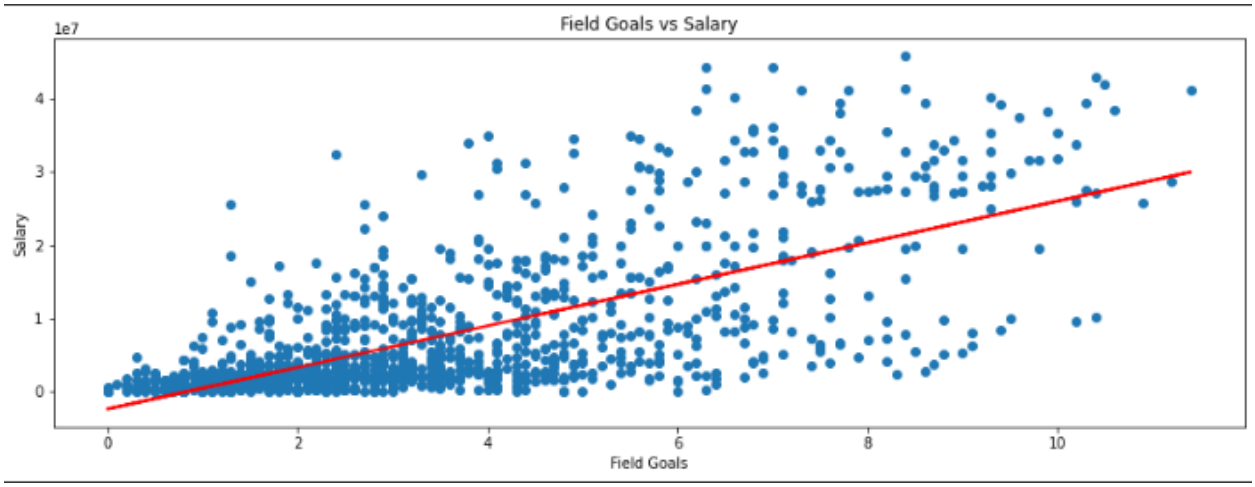


Figure 4: FGA vs. Salary

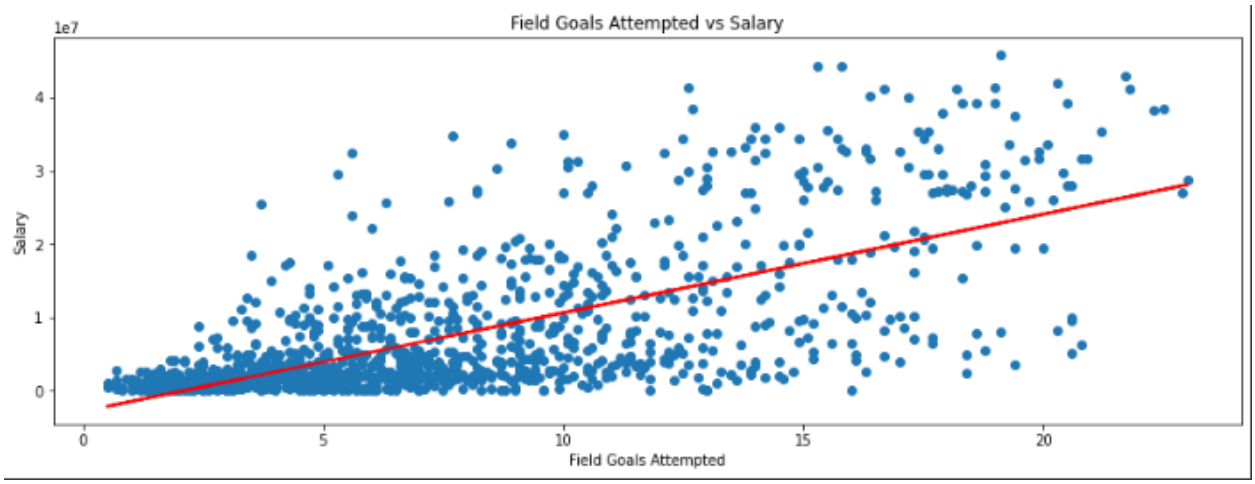


Figure 5: FT vs. Salary

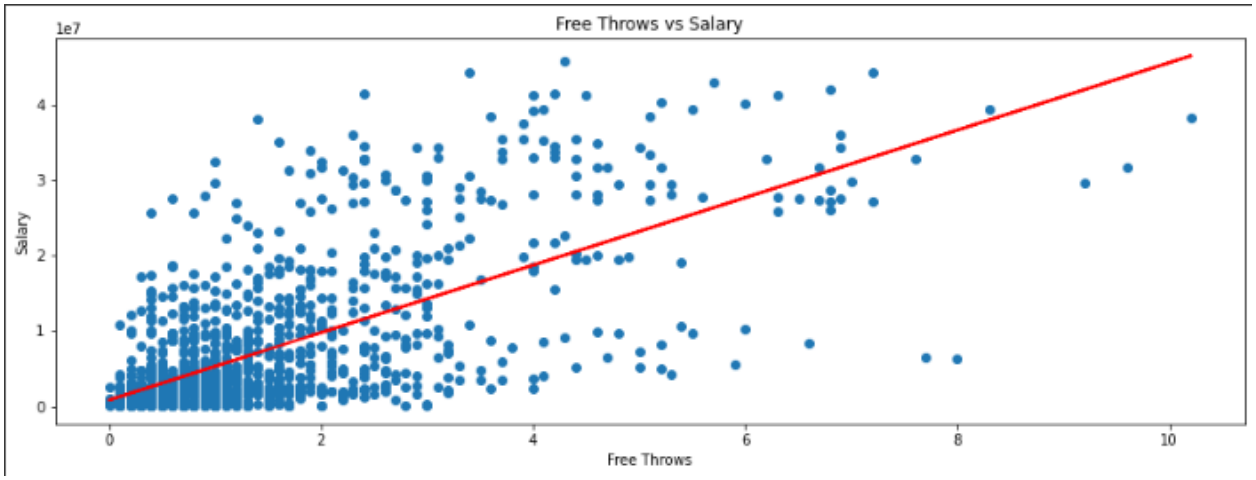
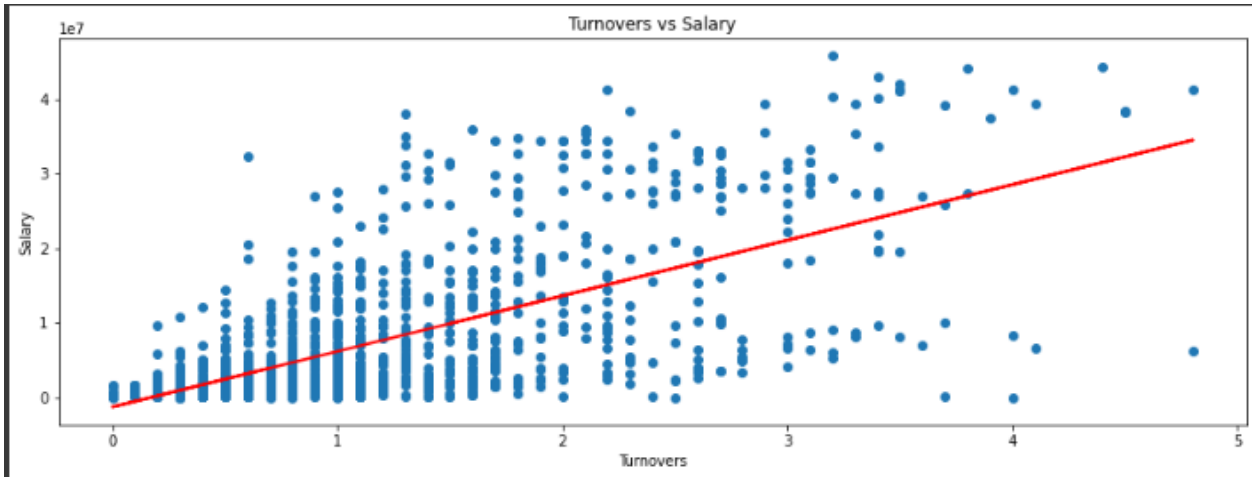


Figure 6: TOV vs. Salary



After graphing all 5 variables individually and observing that all of them had a positive linear relationship with salary, we decided to proceed with our multivariate regression model that would take in all five of these statistics as features and the salary as the feature variable, and would train on the first two seasons of data (2019-2020 and 2020-2021), and then would try to predict the salaries for the 2021-2022 season. We would then use Mean Square Error (MSE) and Mean Absolute Error (MAE) to see how our multivariate linear regression model performed.

After we got the multivariate linear regression model, our predicted values were nowhere close to the actual values and we got the following model with the following errors:

Figure 7: Hypothesis Function of Model 1: Multivariate Linear Regression with PTS, FG, FGA, FT, and TOV

Intercept and Coefficients for Hypothesis Function

```
[ ] print("Intercept: ", model.intercept_)
    print("Coefficients:")
    list(zip(X, model.coef_))
```

```
Intercept:  -2158086.247566011
Coefficients:
[('PTS', 1578649.6478291133),
 ('FG', -1322086.1018171408),
 ('FGA', -522761.7241906703),
 ('FT', 210264.0590319276),
 ('TOV', 2492441.0513005797)]
```

Figure 8: Error of Model 1: Multivariate Linear Regression with PTS, FG, FGA, FT, and TOV

Error for {'PTS', 'FG', 'FGA', 'FT', 'TOV'} vs 'Salary'

```
[ ] meanAbErr = metrics.mean_absolute_error(y, pred_y)
    meanSqErr = metrics.mean_squared_error(y, pred_y)

    print('Mean Absolute Error:', meanAbErr)
    print('Mean Square Error:', meanSqErr)
```

```
Mean Absolute Error: 4507209.247131441
Mean Square Error: 42265091519745.8
```

We collectively decided that this model was not a good representation of predicting salaries, so we then decided to use more features that we had initially ruled out, which had a correlation coefficient of 0.6 or higher. However, we decided to rule out using Turnovers (TOV) as a feature, because of the reason given on page 12. After we chose the other variables with the highest correlation coefficients, we decided to see if they had a positive linear relationship with the salary, to determine and confirm if they would be good choices while doing multivariate linear regression. Below are the remaining graphs that represent the data and the linear regression results for each of the other variables chosen:

Figure 9: FTA vs. Salary

Figure 10: 2PA vs. Salary

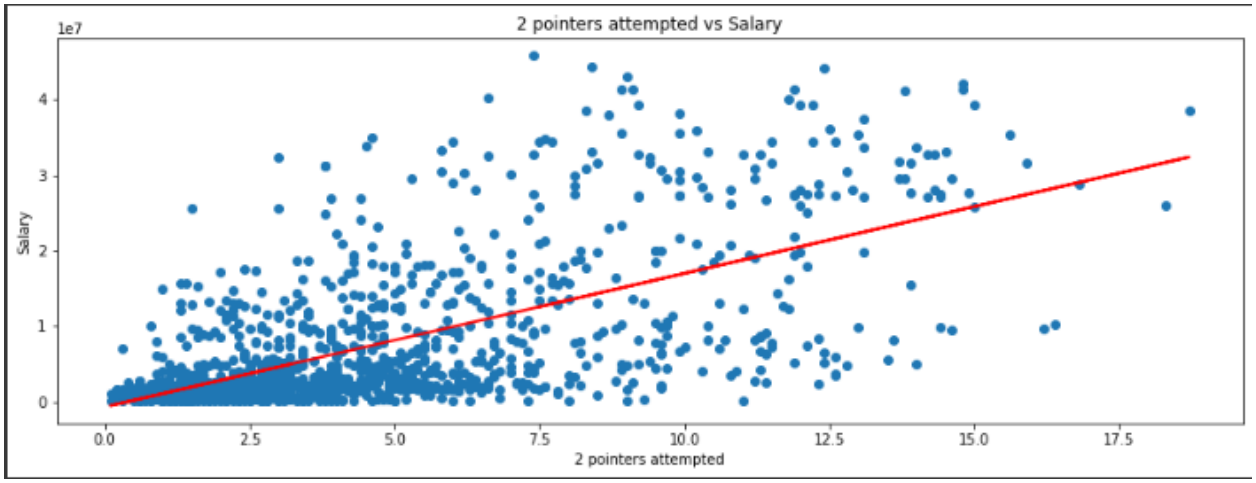


Figure 11: 2P vs. Salary

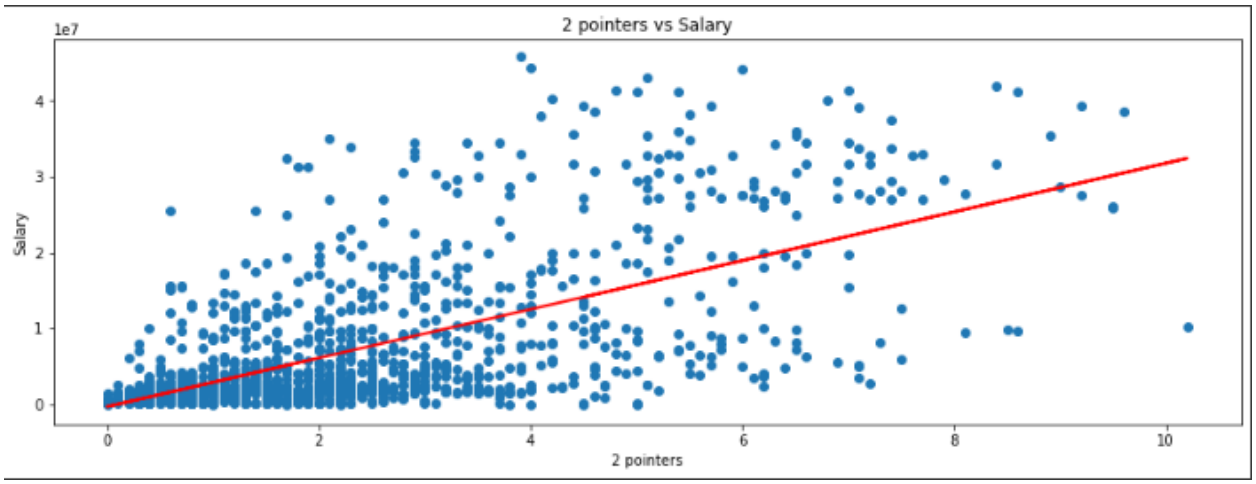


Figure 12: MP vs. Salary

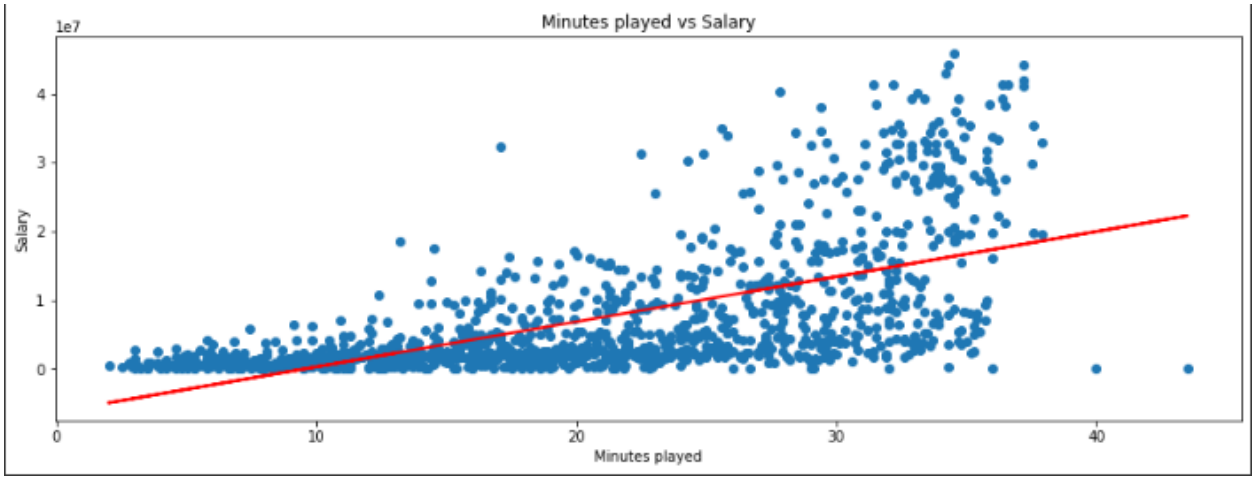
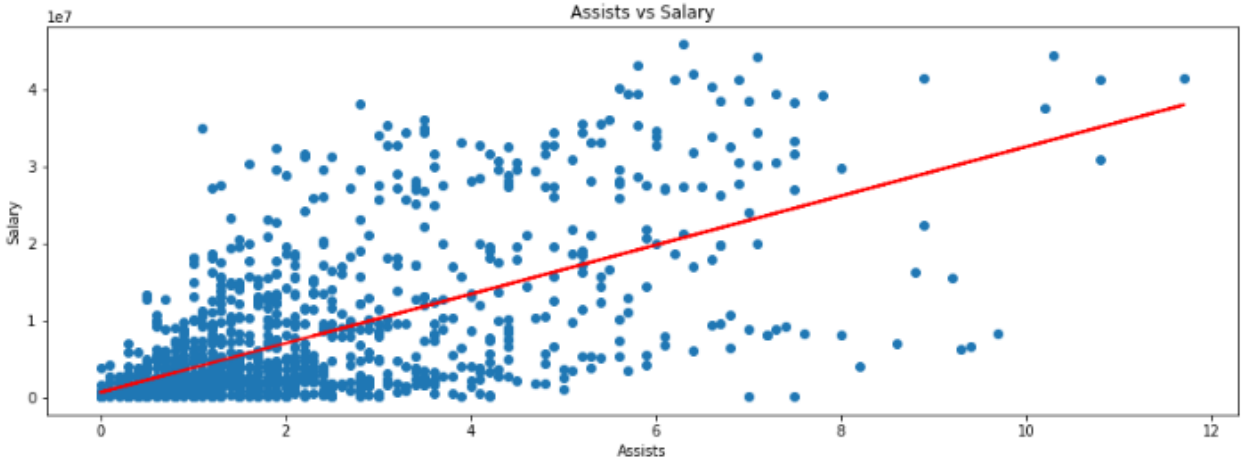


Figure 13: AST vs. Salary



After graphing all the other 5 variables individually and observing that all of them had a positive linear relationship with salary, we decided to proceed with our multivariate regression model that would take in all 9 of these statistics (PTS, FG, FGA, FT, FTA, 2PA, 2P, MP, AST) as features and the salary as the feature variable, and would train on the first two seasons of data (2019-2020 and 2020-2021), and then would try to predict the salaries for the 2021-2022 season. We would then use Mean Square Error (MSE) and Mean Absolute Error (MAE) to see how our multivariate linear regression model performed.

After we got the multivariate linear regression model, our predicted values were closer than before to the actual values and we got the following model with the following errors:

Figure 14: Hypothesis Function of Model 2: Multivariate Linear Regression with PTS, FG, FGA, FT, FTA, 2PA, 2P, MP, AST

Intercept and Coefficients for Hypothesis Function

```
[ ] print("Intercept: ", model.intercept_)
    print("Coefficients:")
    list(zip(X, model.coef_))
```

```
Intercept:  -2241751.26966572
Coefficients:
[('PTS', 2831052.6808110727),
 ('FG', -5369346.695504102),
 ('FGA', -492016.01055842824),
 ('FT', -1927228.1458506198),
 ('FTA', 810579.1857891668),
 ('2PA', -307042.3758407396),
 ('2P', 1957288.2011341886),
 ('MP', 21065.01337166503),
 ('AST', 1315986.5282734726)]
```

Figure 15: Error of Model 2: Multivariate Linear Regression with PTS, FG, FGA, FT, FTA, 2PA, 2P, MP, AST

Error for {'PTS', 'FG', 'FGA', 'FT', 'FTA', '2PA', '2P', 'MP', 'AST'} vs 'Salary'

```
[ ] meanAbErr = metrics.mean_absolute_error(y, pred_y)
    meanSqErr = metrics.mean_squared_error(y, pred_y)

    print('Mean Absolute Error:', meanAbErr)
    print('Mean Square Error:', meanSqErr)
```

```
Mean Absolute Error: 4444666.059727842
Mean Square Error: 40675826105362.26
```

After doing this, we were still not satisfied with the model, so to make our model better, we decided to get rid of the outliers in the salary by implementing a salary cap. For our salary cap, we chose those specific bounds, $6 million to $25 million, to eliminate the outliers in our data. We excluded the highest paid players and the majority of players on their rookie contracts. We removed the highest paid players because the difference between their performance and other players is not proportional to the difference between their salaries and the rest of the league. We removed the players below $6 million because the majority of players on rookie contracts have a salary below $6 million. We wanted to remove players on their rookie contracts because players on their rookie contracts are usually underpaid compared to their performance on the court. The $6 Million boundary also excludes the players who get paid very little because they don't play very much.

After successfully implementing a salary cap on the dataframe, we were able to perform multivariate linear regression with both of our previous models, giving uis the following models and errors:

Figure 16: Hypothesis Function of Model 3: Multivariate Linear Regression with PTS, FG, FGA, FT, and TOV (With Salary Cap)

Intercept and Coefficients for Hypothesis Function

```
[ ] print("Intercept: ", model.intercept_)
    print("Coefficients:")
    list(zip(X, model.coef_))
```

```
Intercept:  9818199.788246553
Coefficients:
[('PTS', 882827.9139905305),
 ('FG', -2000685.2150654392),
 ('FGA', 177640.9441547594),
 ('FT', -642984.7980569535),
 ('TOV', 82928.5850706227)]
```

Figure 17: Error of Model 3: Multivariate Linear Regression with PTS, FG, FGA, FT, and TOV (With Salary Cap)

Error for {'PTS', 'FG', 'FGA', 'FT', 'TOV'} vs 'Salary' (With Salary Cap)

```
[ ] meanAbErr = metrics.mean_absolute_error(y, pred_y)
    meanSqErr = metrics.mean_squared_error(y, pred_y)

    print('Mean Absolute Error:', meanAbErr)
    print('Mean Square Error:', meanSqErr)
```

```
Mean Absolute Error: 3603197.715044754
Mean Square Error: 19217224663874.113
```

Figure 18: Hypothesis Function of Model 4: Multivariate Linear Regression with PTS, FG, FGA, FT, FTA, 2PA, 2P, MP, AST (With Salary Cap)

Intercept and Coefficients for Hypothesis Function

```
[ ] print("Intercept: ", model.intercept_)
    print("Coefficients:")
    list(zip(X, model.coef_))
```

```
Intercept:  5967047.153327018
Coefficients:
[('PTS', 3735352.95553833),
 ('FG', -11253081.786891654),
 ('FGA', 27361.186071593314),
 ('FT', -3301934.9492468415),
 ('FTA', -127750.20681665791),
 ('2PA', 136036.51717200596),
 ('2P', 3082855.832418477),
 ('MP', 271912.4854335785),
 ('AST', -71913.13384342496)]
```

Figure 19: Error of Model 4: Multivariate Linear Regression with PTS, FG, FGA, FT, FTA, 2PA, 2P, MP, AST (With Salary Cap)

Error for {'PTS', 'FG', 'FGA', 'FT', 'FTA', '2PA', '2P', 'MP', 'AST'} vs 'Salary' (With Salary Cap)

```
[ ] meanAbErr = metrics.mean_absolute_error(y, pred_y)
    meanSqErr = metrics.mean_squared_error(y, pred_y)

    print('Mean Absolute Error:', meanAbErr)
    print('Mean Square Error:', meanSqErr)
```

```
Mean Absolute Error: 3552173.3745546606
Mean Square Error: 18317821109558.156
```

After comparing the errors from each of the models, we found that Model 4 (Multivariate Linear Regression with PTS, FG, FGA, FT, FTA, 2PA, 2P, MP, AST (With Salary Cap)) had the lowest error, and this was also the model where the predicted values were much closer to the actual value than before, in comparison to the other models we had tried. However, the predicted values were still not close to the actual values, and our error was still considerably high.

Since our model was quite inaccurate, we came up with a few reasons as to why we thought this was happening:

Reasons the model could be inaccurate:

1. A player's position greatly changes the way their performance is evaluated. The expectations for a point guard or a shooting guard are different from those for a power forward or center. For example, a guard is expected to pass well, which is reflected in the assists per game stat, but a center is not. There are 64 guards in the NBA that averaged 4 or more assists per game, while there were only 4 centers who also achieved that. This is

why picking features that would apply to all positions is hard. All stats are technically positionless, but a player's position or role on the team does affect their opportunity to record certain stats. It would make sense to create separate models for each position, as each position weighs each stat differently.

2. We used the TOV feature in our model, which is the turnovers per game statistic. At first we were confused because having a high turnovers per game is a negative. However, if we rank the players by their TOV, the players with the highest TOV are generally the best and highest paid players in the league. Their TOV is so high because they are the ones handling the ball the most, so their opportunity for turnovers is very high. However, there comes a point where it turns from the best players in the league to bad players, who turnover the ball a lot. This drop off is why we are concerned about using TOV as a feature in our model
3. The fact that a player's salary is not determined year to year, but by contract could also have been a factor. Generally a player's contract can last 1 year to as long as 6 years, so their salary is mainly set for the duration of their contract (disregarding incentives, bonuses, etc.). During that time, their performance could warrant a different salary than what they are receiving, higher or lower. Our model has no concept of contracts and predicts a salary based on the payer's performance last year, so this could be a reason for inaccuracy. It would make sense to create a model that could take into account a player's contract instead of just year to year.
4. A special case of player contracts is the rookie contract. Rookie contracts can range from one to four years long, and pay much lower than a regular contract. These young players on their rookie contracts can be a great example of players not being paid for their performance. For example, Luka Doncic was paid 10.1 million dollars this year (the fourth and final year of his rookie contract), but in his first four years he was a 3 time all-star and 3 time all NBA selection. This past year, Luka averaged 28.4 points per game, 9.1 total rebounds per game, and 8.7 assists per game, while Tomas Satoransky, a back-up point guard for the Wizards, averaged 3.6 points per game, 2.3 rebounds per game, and 3.3 assists per game, but got paid 10 Million dollars this year, almost the same as Luka.

NOTE: since we were performing multivariate linear regression, we had difficulty in visualizing the data, since to accurately do so, we would need a multi-dimensional space to represent the data points and the regression performed, therefore, we have just provided the error and the hypothesis function with coefficients for each model we iterated through

Conclusions:

After iterating through four different models to predict NBA players' salaries based on their statistics, we determined that using statistics as a way to predict salary was not a good

measure as our predictive model had a large error, and our predicted values for salary were quite far from the actual values of the salaries of players. As mentioned before in the reasons our model could be inaccurate, possible future iterations that could improve the model are making a contract based model and a position based model.

Resources Used:

Software and Tools we used and for what purpose they served:

- Jupyter Notebook/Google Colab
    - For writing the code in Python
- Site referenced for how to perform multivariate linear regression
    - https://medium.com/machine-learning-with-python/multiple-linear-regression-implementation-in-python-2de9b303fc0c#:~:text=Multiple%20Linear%20Regression%20is%20an,more%20than%20one%20independent%20variable.
- Excel/Google Sheets
    - For organizing and cleaning the data before exporting in form of a CSV file
- Python Libraries (pandas, scikit-learn, matplotlib)
    - To conduct data analysis (regression, error, etc.)
    - Documentation for libraries for referencing and better understanding of built-in functions
- Websites used to collect data:
    - Hoops Hype (Salaries): https://hoopshype.com/salaries/players/
        - Used to obtain the salary information for each of the players
    - Basketball Reference (Stats): https://www.basketball-reference.com/leagues/NBA_2022_per_game.html
        - Used to obtain the statistics information for each of the players
    - HTML to CSV File: https://www.convertcsv.com/html-table-to-csv.htm
        - Used to convert the data from both of the above two websites into CSV files that we would import into Google Sheets to clean and organize