Data Science 2
Final Project Report:
NBA Salary Predictions

For:
Dr. Niloofar Yousefi
Class: Data Science 2
ISC 4242

By:
Alex Chavez, AJ Romaniello, Osvaldo Pelaez, Elizabeth Barnum, and Sophia Kohn

Problem Statement and Motivation:

Professional sports leagues, including the NBA, operate within a complex economic framework where data-driven decision-making is crucial. Teams and management rely on market and player data to inform strategic decisions related to tactics, play style, and player valuation, including transfers, contract negotiations, and salary determinations.

The NBA, in particular, is a multi-billion-dollar industry governed by a salary cap system and a Collective Bargaining Agreement that outlines specific limits and structures for player compensation (*NBA*, 2024; NBA-NBPA, 2023; Teitelbaum, 2024). Under these systems, teams have to make informed decisions as to the allocation of limited salary cap resources, making it essential to identify the key performance metrics that drive player compensation.

Traditional stats like points per game, assists, and field goals are commonly used in media and scouting reports, but overall player impact is best assessed by combining these with more advanced metrics, such as Player Efficiency Rating, Box Plus/Minus, Win Shares, and Value Over Replacement Player. Papadaki & Tsagris (2022), Xiong (2024), and Yang (2024) all explored factors influencing salaries using different approaches, producing a range of conclusions.

While these studies offer valuable insights, their findings vary depending on the datasets, time periods, and statistical methods used. Some emphasize scoring statistics, while others highlight efficiency metrics or team success indicators. However, few studies offer a direct comparison of both traditional and advanced performance metrics using current data, particularly in the context of a fully capped league structure where financial efficiency is crucial.

This project seeks to build on and refine previous work by evaluating a range of in-game statistics within a modern NBA context, aiming to answer the following research question:

Which in-game statistics are the most significant predictors of NBA player salaries, and what is the relative impact of each of these factors on player compensation?

Introduction and Description of Data:

The data for this project is sourced from 'Basketball Reference,' a comprehensive online resource for NBA statistics. This dataset includes player performance metrics from the 2022 NBA playoffs, which will be used to predict salaries. The dataset contains a variety of player attributes, including demographic information (name, age, position, team, and key performance statistics. There were a lot of variables that contained abbreviation,s so to keep things short, let’s highlight the potential relevance in predicting player salaries:

- **MP (Minutes Played): Average minutes a player plays a game**
- **FG (Field Goals per game):** Average number of successful field goal attempts.
- **AST (Assists per game):** Average number of assists contributed.
- **PTS (Points per game):** Average points scored.
- **GS (Games Started):** Total games in which the player was in the starting lineup.
- **USG% (Usage Percentage):** Percentage of team plays involving the player.
- **TOV (Turnovers per game):** Average number of turnovers committed.
- **VORP (Value Over Replacement Player):** An estimate of a player's total contribution compared to a replacement-level player.
- **BPM (Box Plus/Minus):** An estimate of a player's contribution to the team's point differential per 100 possessions.
- **PER (Player Efficiency Rating):** A per-minute performance metric that adjusts for pace.
- **WS (Win Shares):** An estimate of a player's contribution to team wins.

To better understand the dataset in its full potential, let's do a quick Exploratory Data Analysis on it.

First we took a look at the dataframe. We wanted to see what would be useful and what needed to be cleaned up. Most of the data was already cleaned and ready for processing, but there were a few troublesome columns. We took a look at the data types of the columns. With this we also took the head and tail of the dataframe.

Modeling Approach:

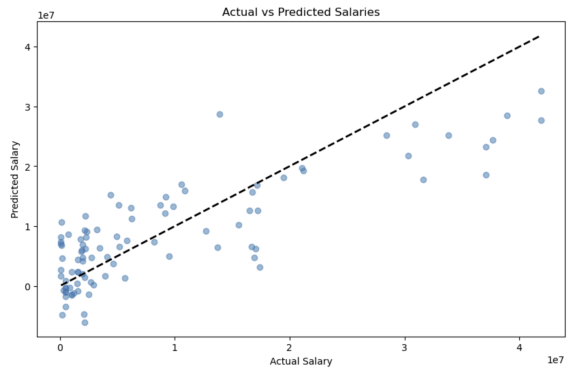
To cover all possible grounds, we decided to use 3 different machine learning models.

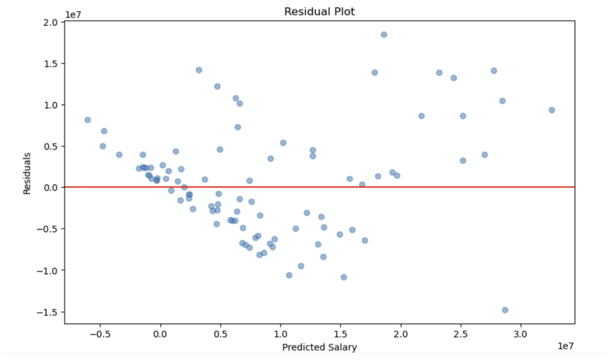
1) Linear Regression - Baseline Model
2) Gradient Descent Linear Regression
3) Feedforward Neural Network
4) Random Forest

<u>Gradient descent Linear Regression</u>:

First, we built a baseline linear regression model using the key features. Giving us a baseline in feature performance and showing us what we had to work with.

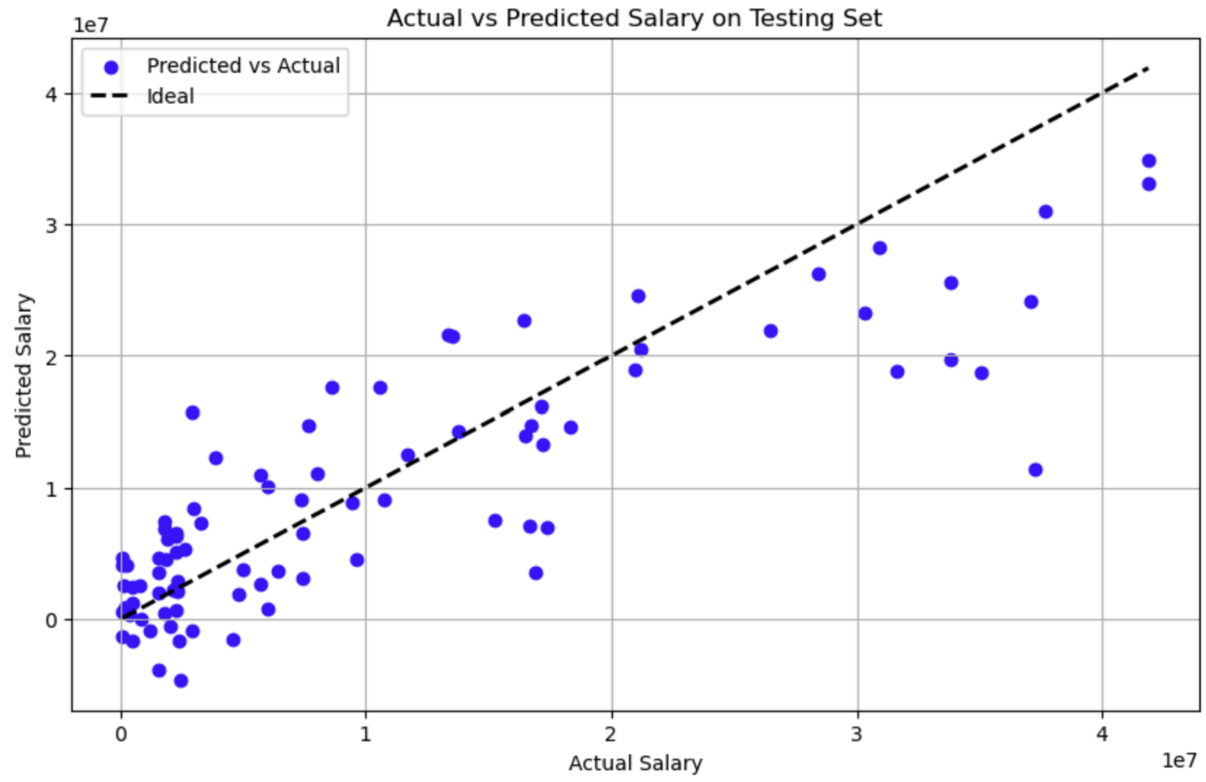
Baseline Model Performance:
RMSE: 6,527,466.26
R-squared: 0.6791





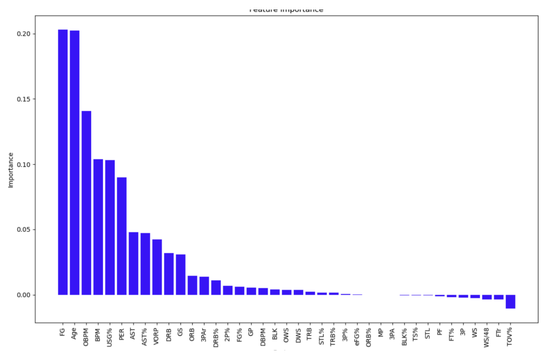
Next, We ran the Gradient Descent Linear Regression model using the BL model as a baseline.

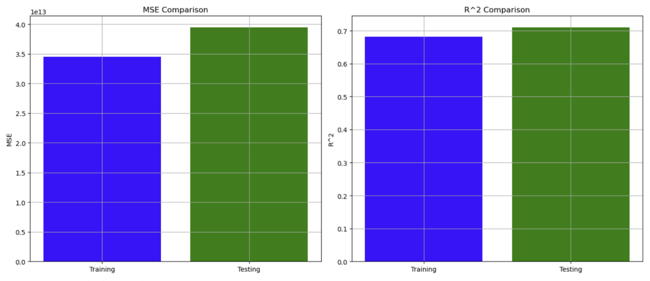
We split the data into a testing set and a training set and visualize the MSE and R^2 metrics for each of the sets.



```
Mean Squared Error (MSE) on Testing set: 39770862860054.95
R-squared (R2) on Testing set: 0.7070854228809903
```

Next we calculated the performance of each of the statistics to see which had more impact on predicting salary using this model.





```
Mean Squared Error (MSE) on Testing set: 39435987936986.37
R-squared (R2) on Testing set: 0.7095517949791657
```

Using these visualizations we can see that this new Gradient descent linear regression model is slightly better at predicting salary than the Baseline model. From the performance metrics we can see that Statistics such as FG and Age have a big impact using this model. This makes sense because the more points or field goals you score the higher likely that you have a higher salary. Age also makes sense because players that are NBA veterans usually have proven their value and make more money because of that.

Feedforward Neural Network

Neural networks are a more sophisticated learning algorithm. They have the ability to learn non-linear trends and relationships that linear regression can't do. We chose to do a FNN along with XGBoost to get a deeper understanding of any nonlinear trends in the data. XGBoost can help get a deeper understanding of the complexity of the data through decision trees while FNN learns trends through gradient descent and backpropagation. While these 2 models can find non-linear trends, they implement it differently and we were interested to see how they both perform with our data.

For the FNN we transformed the salary data (dependent variable) using a logarithmic transformation. The salary data was severely right skewed and the logarithmic transformation helped correct for the skewness. Along with the transformation on salary, we scaled the rest of the data because neural networks can be sensitive to unscaled data. Now our data is ready for our neural network.

Throughout the process of refining my network to find the optimal structure and run time, I used metrics like MAE, R^2, and MSLE. MAE tells us, on average, how far off our prediction is from the actual salary. R^2 tells us how well the model predicts the variation in salaries based on our predictors. MSLE is a variation of MSE where we are measuring the difference between the predicted and actual salaries, but penalizing the higher differences more. We want as small of an MAE as possible but aiming for \$1-3 million. R^2 should be above 0.7. MSLE should be as close to 0 as possible, but anything under 0.5 is acceptable.

The first network I created had 3 hidden layers. Here is a description of each layer:

1. The first hidden layer had 128 neurons and ReLU as the activation function.
2. The second hidden layer had 64 neurons and also used ReLU for activation.
3. Our last hidden layer had 32 neurons and used ReLU activation.

With this structure, I compiled the network and updated the weights through backpropagation with Adam optimization. The values were MAE = 31,558,749.75, R^2 = -68.0513, and MSLE: 2.1294. None of these metrics are good. This is indicating that our model is not predicting our actual values closely. On average we are \$31 million away from the actual value with our prediction. Having a negative R^2 means our model isn't any better at predicting salaries than someone randomly guessing. Having a high MSLE is another indicator that our model is not predicting well.

Since our first model didn't have good metrics, we have to make some adjustments. This could be due to overfitting/underfitting, outliers, or model architecture. To check for overfitting/underfitting I ran a training loss vs. validation loss graph and saw that they were almost identical and smooth curves. Which is good news and pointing towards my model being a good fit. So our model isn't predicting well due to one of the other factors.

In the next 2 models we decided to change the structure of the network. I increased and decreased the number of hidden layers and increased the number of epochs to 1000. I had 5 hidden layers with the number of neurons being 128, 64, 32, 16, and 8 respectively. My results got worse. For my increased number of hidden layers model, the metrics were MAE = 64,470,632.34, R^2 = -552.3727, MSLE = 1.9696. All of these metrics got worse showing me that making the network more intricate did not help. So I went to a simpler network. Here I only had 2 hidden layers with 64 neurons and 32 neurons. My results were MAE = 37,3576,346.75, R^2 = -148.3985, MSLE = 1.9635. These results are worse than my increased hidden layers model.

This is making me believe that either a FNN won't be the most effective way to describe these trends or there are severe outliers that are messing up the model. FNN models are very susceptible to outliers, so this could be throwing off the model. Next, I decided to remove all of the outliers using the IQR range and rerun my original FNN. After removing the outliers I was left with 247 observations. A little less than I wanted, but it might help the model.

I reused my original model because it gave us the "best" testing metrics. There are 3 hidden layers using ReLU activation. MAE = 37,336,312.94, R^2 = -370.0508, MSLE = 4.8149. These metrics are much worse than my previous models. This doesn't surprise me a lot, because FNN are made for larger datasets. Decreasing the sample size by almost half can impact the training of the FNN model. Due to this I think that an FNN is not the best model to use for our data. After trying to fine tune the parameters, modify the structure, and remove the outliers with no better performance on the model, we can say that the FNN is not a good choice and may want to try a new model. Due to this, we decided to try Random Forests and compare that with XGBoost.

Random Forests

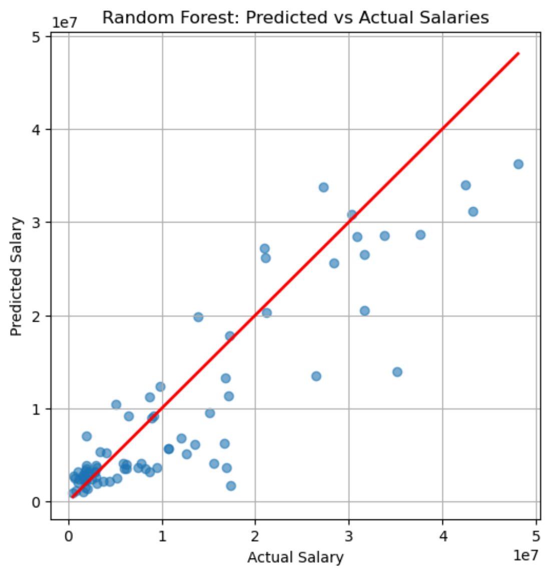
Since my FNN models weren't predicting well, I decided to go to a Random Forest (RF) model. Random Forests are very similar to XGBoost models, but are slightly simpler. The FNN model might be too complex for the data we have and a simpler model, like Random Forests, would be more effective. To test this I ran a couple

Random Forests models to find the optimal one. The metrics I used to measure my model were the same as FNN, MAE, R^2, and MSLE.

For my first RF model had a logarithmic transformation on y, with all numerical features in X. I split my training and testing sets into 80% and 20% respectively. Running my RF with 100 estimators and a maximum depth of 10 seemed to give good results. I converted my predictions back into normal numbers for ease of interpretations and got MAE = 3,836,162.08, R^2 = 0.7718, MSLE = 0.4164. These results are much better than the FNN. our MAE is a little higher than I want. We would really like it to be less than $3 million, but it is a lot better than the FNN model.Our R^2 is good now. Anything over 0.7 tells us that our model is predicting well and fairly accurately. The closer to 1 I can get the better the model will perform. Having an MSLE of 0.4164 is much better. That is an acceptable metric, but if I can get it closer to 0, then the model will be getting better.

The RF model is much better than the FNN model. I went and ran a few more RF to see if I could make it even better and more efficient. I ran feature importance with my RF model to see which features I should keep and which ones would be better to discard. There were a couple features that had high importance like MP, Age, FG, GS, TOV, and a couple more. Due to this I decided to take the top 10, because the rest were all pretty close to 0.01 for importance. I then retrained my RF model with the 10 top most important features. The top 10 features were MP, Age, FG, TOV, PTS, GS, Total Minutes, GP, AST, and fTr. With these features my new RF had an MAE = 3,681,159.55, R^2 = 0.7840, and MSLE = 0.4164. These metrics are telling us that our model got slightly better, but probably isn't very significantly better. Our MSLE is exactly the same and our other 2 metrics got marginally better.

To see if there is a better RF model, I ran a GridSearch method to see what the best hyperparameters are for the RF. I wanted to run this to see if there was a significantly better model than my original RF model. After running the Grid Search, it found that with the best hyperparameters the MAE = 3,733,983.14, R^2 = 0.7634, and MSLE = 0.4164. As we can see, these numbers aren't significantly better than our first model. Any RF model we make will probably have good results, which is very good.



Results:

As we can see from the analysis above, the Feedforward Neural Network didn’t perform as well as we anticipated. This is probably due to the inherent sophistication and complexity of neural networks. Neural networks work better with larger dataset, and our dataset was probably slightly smaller than what a neural network is usually used for. All of our metrics for the FNN were underwhelming and signified a poor model for the data. Therefore we couldn’t make any interpretations or predictions about the salary of NBA players from this model. Due to this model not fitting well, we decided to try a Random Forest instead. A Random Forest is simpler than a FNN and doesn’t require a large dataset to work. They are very similar to XGBoost models, but are slightly less complicated. XGBoost uses a lot of decision trees like Random Forests, but regularizes terms and the boosts are sequential.

Our Random Forest models were more insightful. We got good metrics back that signified that the models were predicting quite accurately. We tested a bunch of different parameters, but they all seemed to give similar evaluation metrics. From the Random Forests we determined the top 10 most important features were MP, Age, FG, TOV, PTS, GS, Total Minutes, GP, AST, and fTr in that order. So we can use the Random Forests to accurately predict a player's salary based on a few key stats that are given. We can see that our model predicts salary well because the points in this graph huddle closely to the prediction line. We can see that the model tends to slightly underpredict the salary. We can see this because there appears to be more points under the prediction line than over the line.

<u>Conclusion and Future Work:</u>

All models used have their own separate advantages and disadvantages that need to be looked at before we choose a particular one:

• **FNN**

Pros: Can model non-linear relationships and interactions between features. Potentially more accurate than linear models if tuned properly.

Cons: Requires careful tuning of parameters, more data to train effectively, and is less interpretable than linear models.

• **Gradient Descent Linear Regression**

Pros: Simple, interpretable, and easy to implement. Good for understanding the direct relationship between features and the target variable.

Cons: Assumes a linear relationship between variables, which might not capture more complex patterns in the data.

• **Random Forests**

Pros: Provide more accurate feature importance scores, making it easier to understand which variables influence predictions. RFs work directly on raw data (since they use decision trees). RFs generalize decently even with limited data.

Cons: RFs do not inherently capture temporal dependencies, struggle with raw image, audio, or text data, and they make predictions based on observed splits and may fail on unseen patterns.

Using all these pros and cons we can see exactly what model would be best for the main prompt of this paper, and that is to predict NBA players' salaries using their in-game statistics. To answer this question, we need simple and interpretable results to give an accurate answer to the questions. With that being said we will use the Random Forests model over the FNN and Gradient descent Linear Regression model because of the R^2 being relatively high and the pros being exactly what we need.

In Conclusion, statistics that measure longevity like MP (Minutes Played), Age and GP have a good correlation on how much a player is paid but by far the most important statistics is FG which is how many shots a player is shooting a game. This makes sense because a player that is being paid the most generally will shoot a lot more shots than a player who is not.

More variables need to be added into the dataset like a team's salary cap, how much the player's teammates are being paid, and what team they play for, to get a better overall model performance. One thing that really surprised me was that advanced statistics such as VORP, OBPM and WS in general were not as effective at predicting an NBA player's salary. Overall, more research needs to be done on the subject on player statistics predicting salary.

<u>References:</u>

NBA-NBPA. (2023, July). *Collective Bargaining Agreement*. https://imgix.cosmicjs.com/25da5eb0-15eb-11ee-b5b3-fbd321202bdf-Final-2023-NBA-Collective-Bargaining-Agreement-6-28-23.pdf

NBA salary cap for 2024-25 season set at $140.588 million. (2024, July 1). NBA. https://www.nba.com/news/nba-salary-cap-set-2024-25-season

Papadaki, I., & Tsagris, M. (2022). Are NBA Players' Salaries in Accordance with Their Performance on Court? In M. K. Terzioğlu (Ed.), *Advances in Econometrics, Operational Research, Data Science and Actuarial Studies: Techniques and Theories* (pp. 405-428). Springer International Publishing. https://link.springer.com/chapter/10.1007/978-3-030-85254-2_25#citeas

Teitelbaum, J. (2024, October 24). *Forbes Most Valuable NBA Teams 2024 List — What All 30 Franchises Are Worth*. Forbes. http://www.forbes.com/lists/nba-valuations/

Xiong, A. (2024, November 1). Analysis of NBA player salary based on multiple linear regression model. *Theoretical and Natural Science*, *51*, 206-213. https://doi.org/10.54254/2753-8818/51/2024CH0205

Yang, Z. (2024, September 27). Analysis of the Relationship between NBA Player Salary and Their On-Court Performances. *Theoretical and Natural Science*, *41*, 51-58. https://doi.org/10.54254/2753-8818/41/2024CH0147