



Predictive Modeling with NBA

Player Data

Ross Lauterbach

Table of Contents

1



Introduction

Motivations, Data
Source, Questions

2



Analysis

Exploratory Data
Analysis, Predictive
Modeling, Results

3



Conclusion

Summary of Analysis
and Interpretation of
Results

1



Introduction

Motivations, Data Source, Questions

Motivation

Popularity

We (and many others) love basketball!

Accessibility

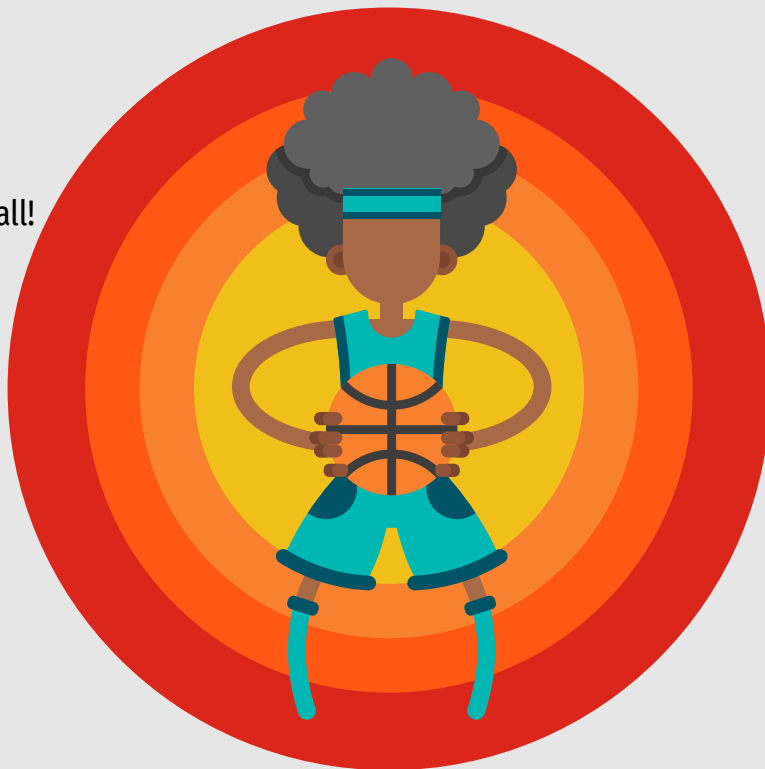
NBA data is publicly available.

Complexity

The relationships between variables are non-trivial.

Fruitful

Insights can be used to make decisions such as drafting.



Data Set



- Biometric, biographic and basic box score stats from 1996 to 2022 season
- The data set is easy to find on Kaggle
- Each row provides statistics for a certain player in a certain season (~13k rows)
- Per player, per season, there are 19 variables to consider

Questions of Interest

If all we know is a player's performance & biometric statistics, can we predict how tall they are?

- Potential usefulness: if one can accurately predict height based solely on performance statistics, it suggests that height should be considered when drafting a college player to achieve specific performance goals.

If all we know is a player's performance & biometric statistics, can we predict how many points they score per game?

- Potential usefulness: points are the whole point (no pun intended) of basketball; understanding where they come from is vital to the success of any team.

2



Analysis

Exploratory Data Analysis, Predictive
Modeling, Results

Numeric Features Explained

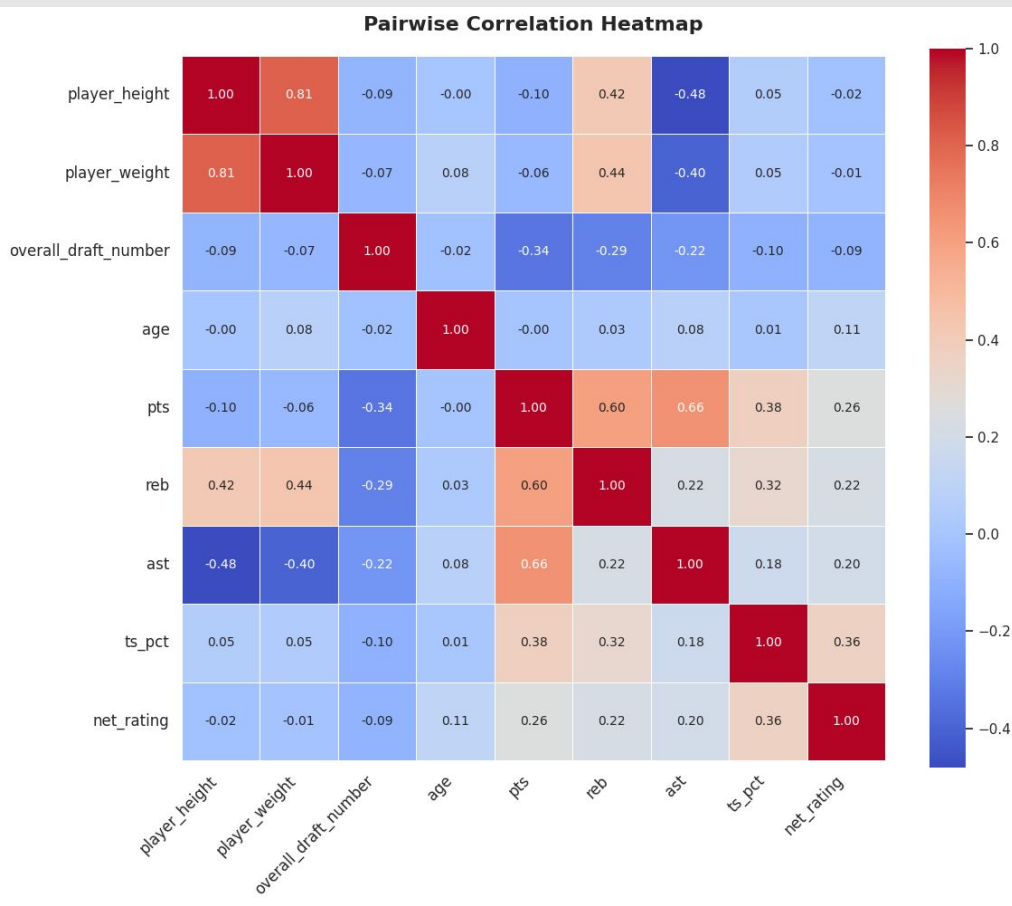
- Draft Number: 2 rounds, 30 picks each round
- Games Played: 82 game season
- Points: 1, 2, and 3 point opportunities available
- Assists: Pass to player who scores
- Rebounds: Grab ball after shot
- DREB and OREB %: Proportion of available rebounds grabbed while on floor
- Assist %: Proportion of total assists made while on the floor
- Usage Rate: How much offense player is responsible for on court
- True Shooting %: Considers overall scoring efficiency
- Net Rating: How much the team improves with player on the court
- And more! → height, weight, age, etc

Data Cleaning

- The data was relatively clean to begin with
- Rows that contained missing or nonsensical values were removed

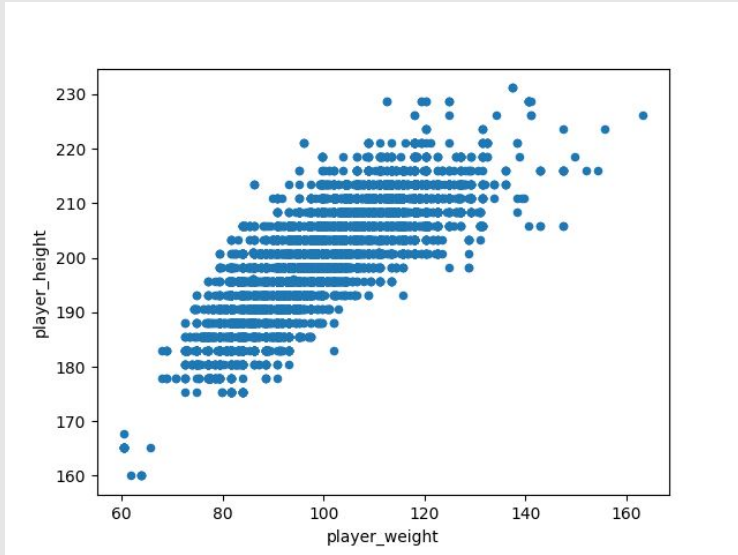


Finding Correlations



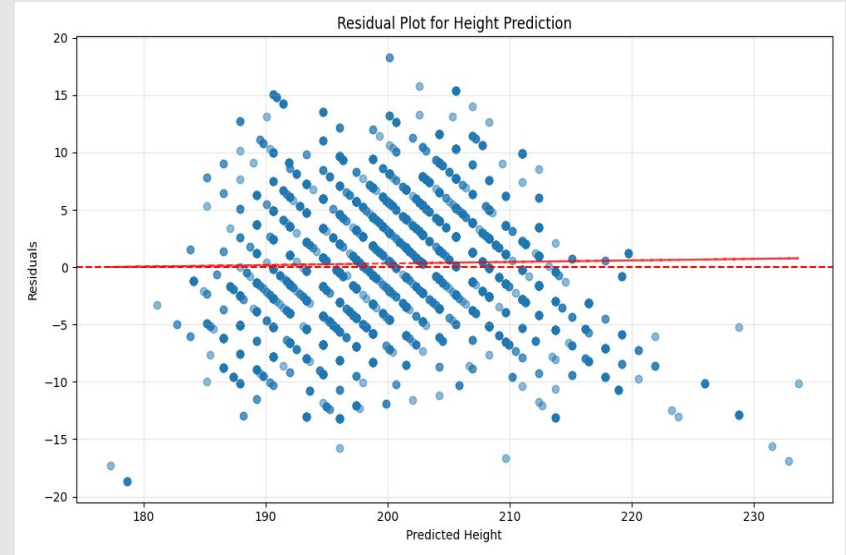
- Height & Weight are correlated, of course
- Height & Rebounds are correlated because the taller you are the easier it is to get a rebound over others
- Points & Rebounds are correlated (for many possible reasons... one is simply the more minutes you play, the more of both you can gain)
- Height & Assists are negatively correlated because point guards (who assist the most) are generally short and agile

Baseline: Height (cm) vs Weight (kg)



Scatter Plot of Height vs Weight:

- Horizontal lines are due to discrete way heights are measured
- See a slight curve?

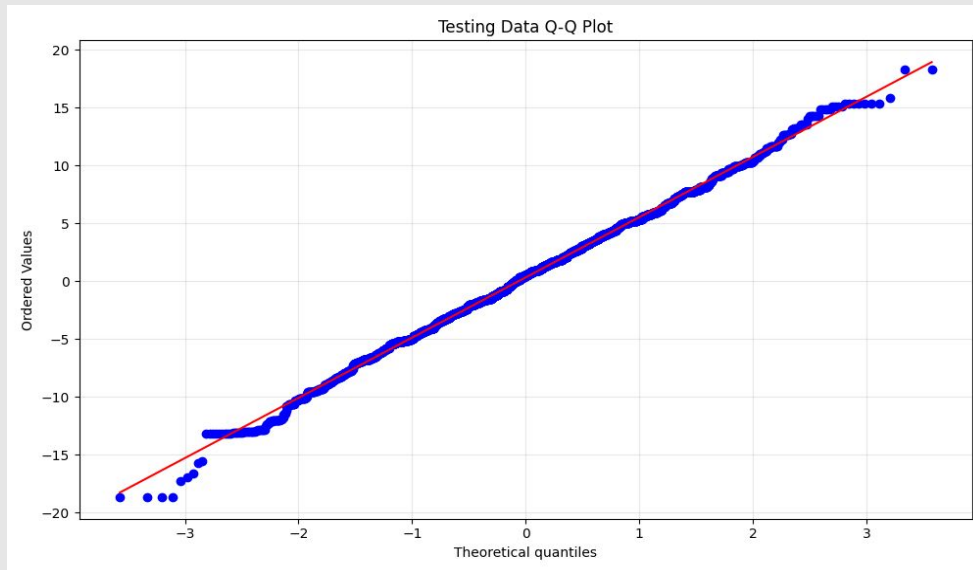


Residuals vs Fitted Height:

- Residuals are on test data (30% of set)
- Mostly random, but extreme predictions tend to be underestimates

Height vs Weight: Are errors normally distributed?

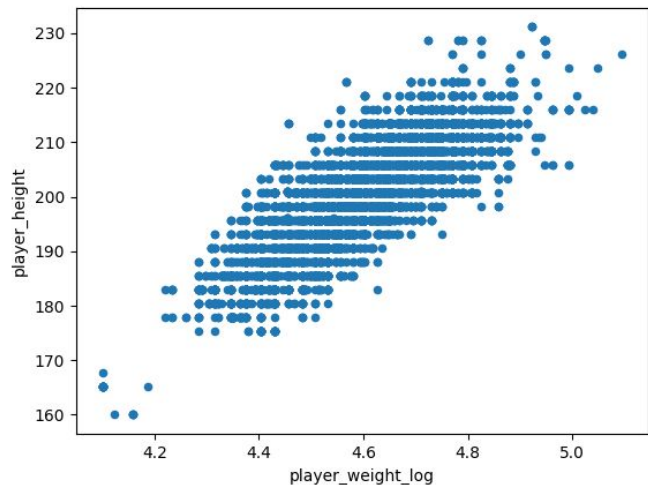
- Due to the slight curve in the data, we may benefit from transforming x (weight)
- Now we can consider transforming y (height)
- From residual plot, we see generally constant variance (except edges)
- Are the residuals normally distributed? Let's see:



→ Relatively normal (except edges)

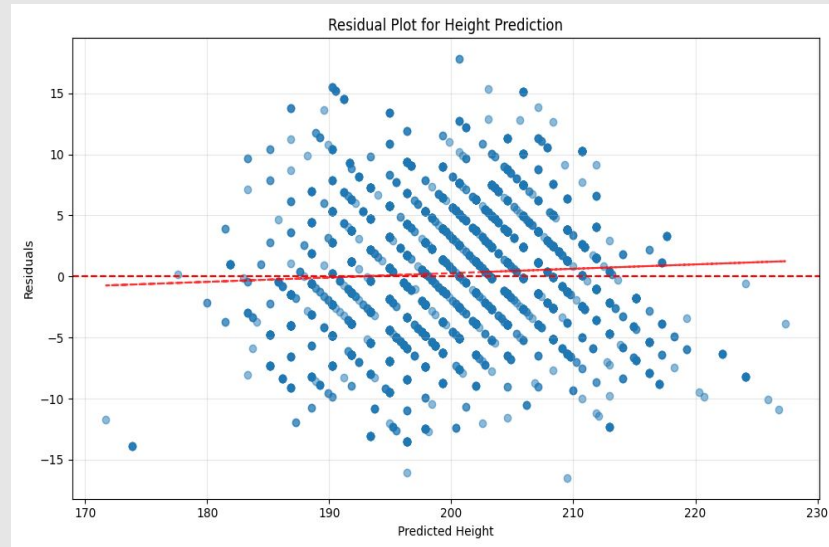
No need to transform height.

Better Baseline: Height vs log(Weight)



R^2 :

- Height vs log(Weight): 0.69
- Height vs Weight: 0.67

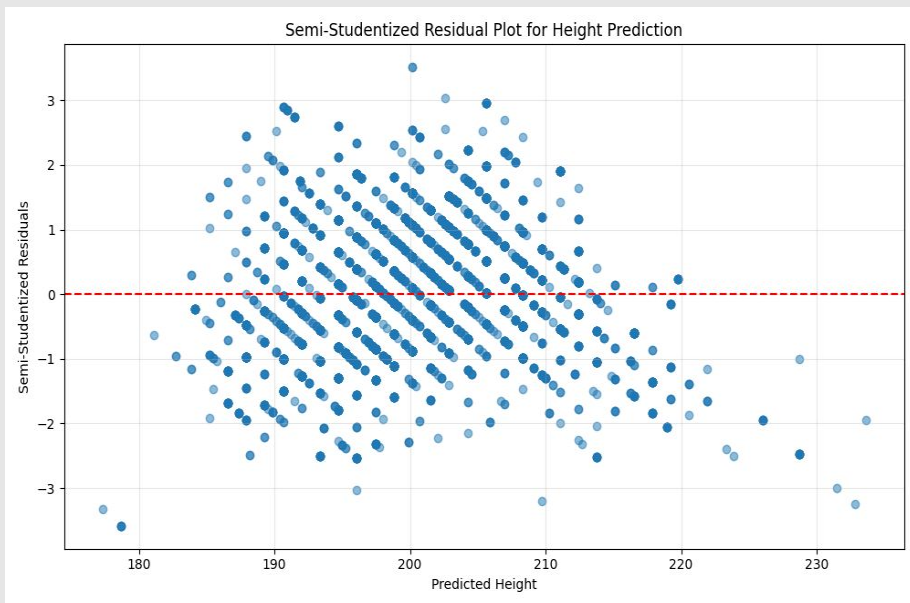


Test RMSE:

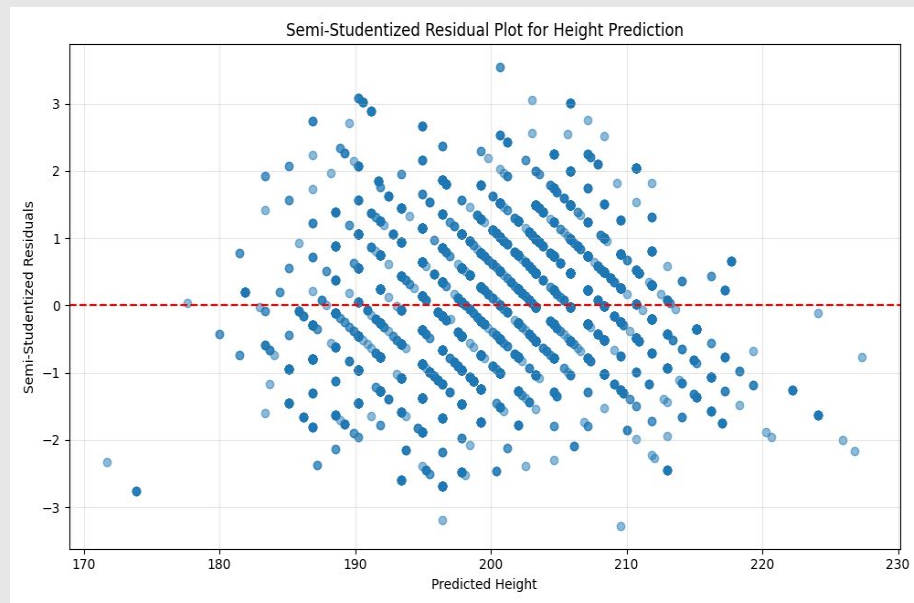
- Height vs log(Weight): 5.04
- Height vs Weight: 5.21

Benefit of Transformation: Semi-Studentized Residuals

- Semi-studentized residuals lets us easily identify outliers based on test data (< -3 or > 3)
- $\log(\text{weight})$ is a clear improvement \rightarrow now outliers exist only near average NBA heights



For: Height vs Weight



For: Height vs $\log(\text{Weight})$

Can performance statistics predict height better than weight can?

- The correlation between weight and height is typically measured between 0.4 and 0.8 → here it is 0.81 → why?
- It is unsurprising that weight can predict height, since both are biometric.
- What would really be impressive is if you can more accurately predict how tall a player is based on how they play rather than how much they weigh → can we do this?
- Here are the candidate performance statistics:
 - ['games played', 'pts', 'reb', 'asts', 'net rating', 'off reb pct', 'def reb pct', 'usg pct', 'true shooting pct', 'ast pct']



To predict height, which performance stats matter?

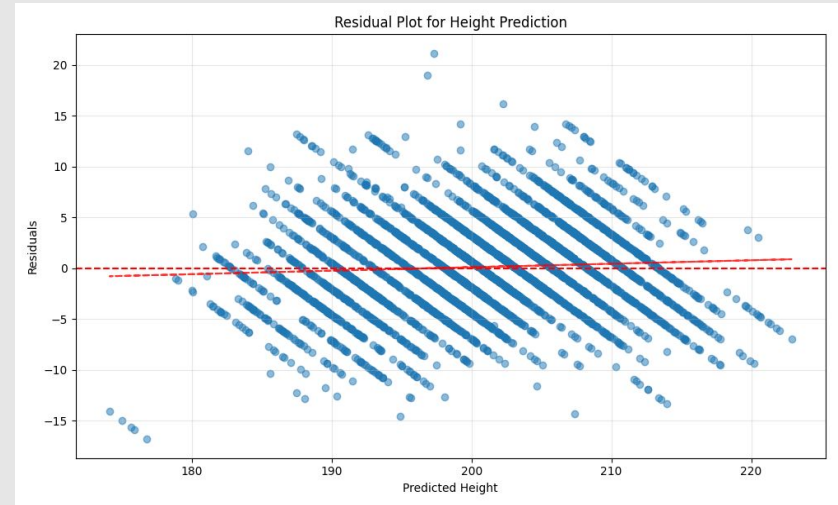
We can start with a full model (all 10 features), and use backward selection on training data to keep the most important ones. We sequentially remove features with the highest p-value, until all that remain are below 0.01.

- Remove 'points' $\rightarrow p = 0.54$
- Remove 'true shooting pct' $\rightarrow p = 0.05$
- Remove 'games played' $\rightarrow p = 0.09$
- Left with 7 features

- Resulting **$R^2 = 0.63$** and **Test RMSE = 5.52**
 - This is “worse” than just using weight, but still comparable

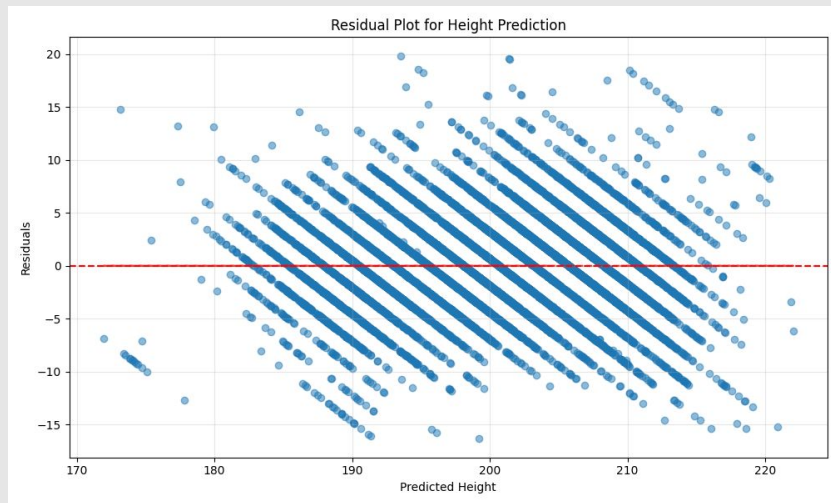
Final Linear Model for Height: Candidate #1

- Instead of letting biometric and performance stats compete to predict height, we can let them work together
- Let's re-do the backward selection but this time add $\log(\text{weight})$ to our full model at the start
- Remove 'points' $\rightarrow p = 0.68$
- Remove 'net rating' $\rightarrow p = 0.06$
- Remove 'true shooting pct' $\rightarrow p = 0.02$
- Remove 'games played' $\rightarrow p = 0.05$
- Left with 7 features
- Resulting $R^2 = 0.76$ and Test RMSE = 4.44



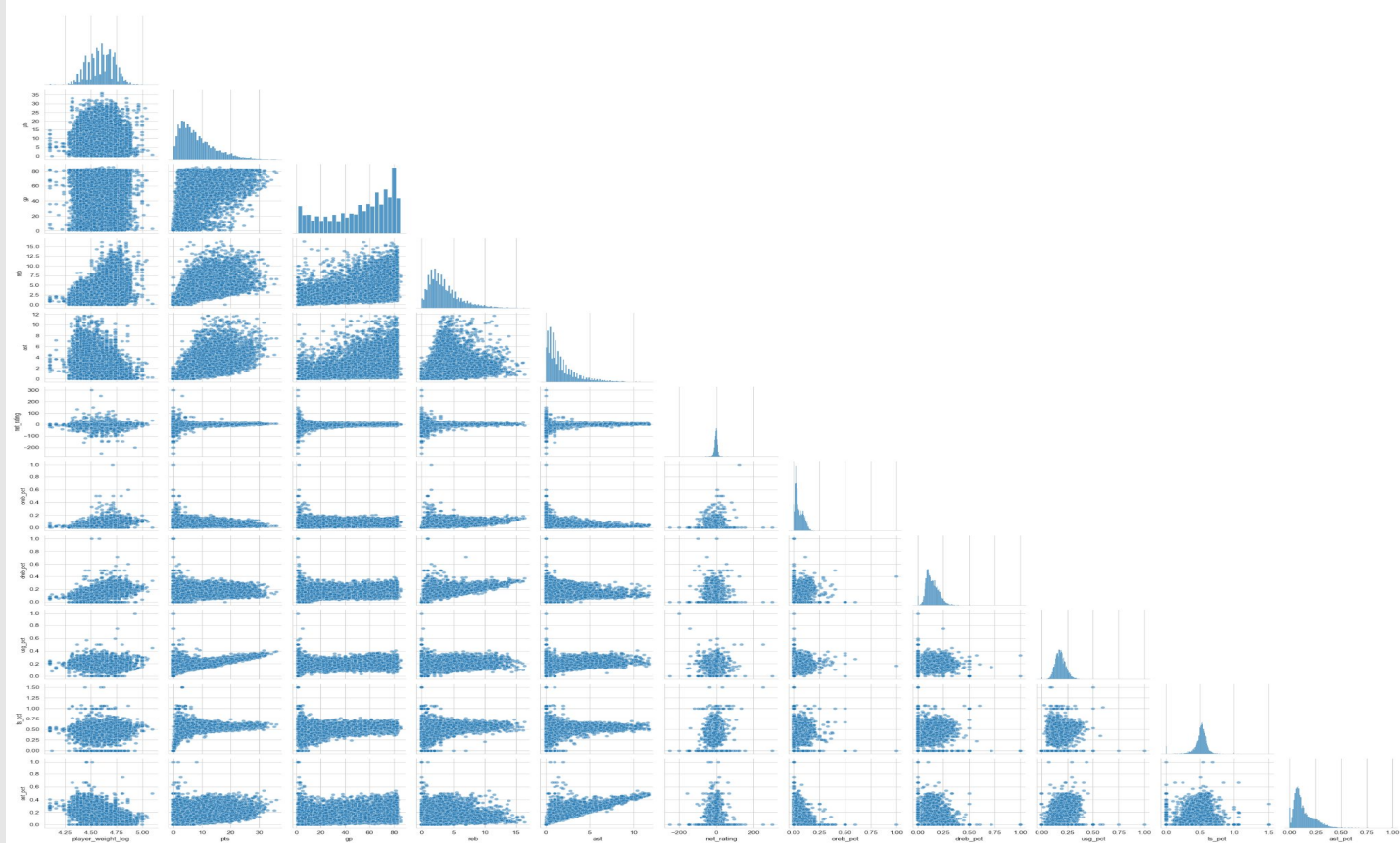
Final Linear Model for Height: Candidate #2

- In the last slide, we did backward selection with p-values
- But... with just 10 variables to consider, we can fit all subsets
- This time we use AIC as the criterion
- Best model on training data:
 - Includes 10 features
 - Only excludes 'points'
- Resulting **$R^2 = 0.76$** and **Test RMSE = 4.43**
- Marginally better than Candidate #1
- FINAL MODEL: Candidate #2



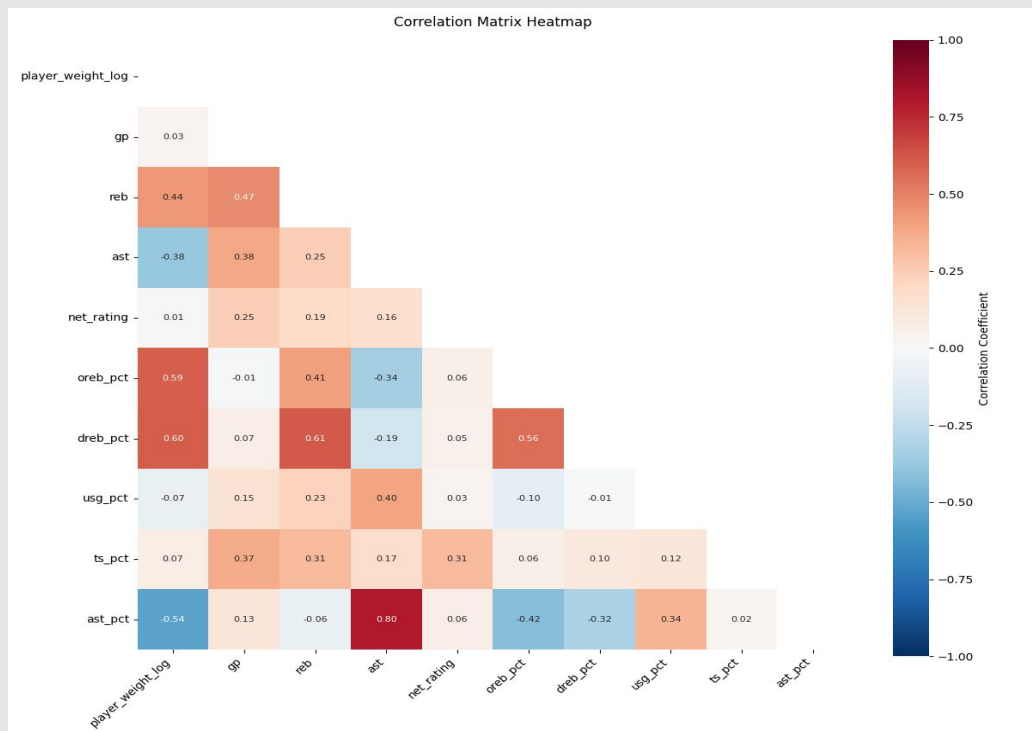
Is Multicollinearity a Problem?

Feature correlations are high... ex: the correlation between assists and assist % is 0.8



Addressing Multicollinearity

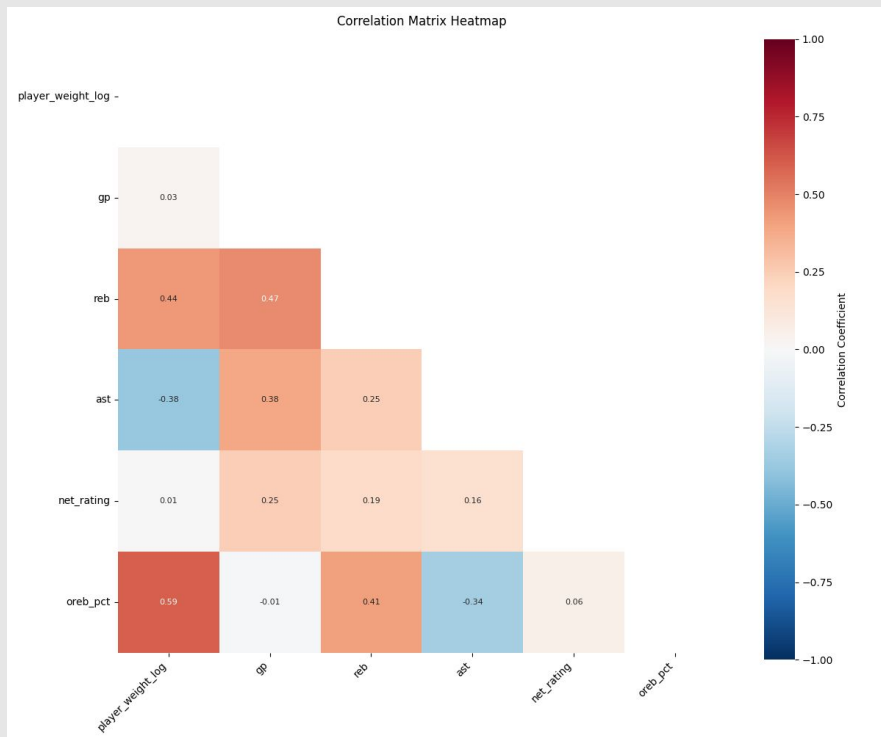
- We can compute the VIF (variance inflation factor) to determine each feature's net linear relation to other features → greater than 10 is extreme



Feature	VIF
player_weight_log	53.59
ts_pct	33.28
usg_pct	16.16
dreb_pct	13.83
ast_pct	10.77
reb	9.43
ast	9.25
gp	8.44
oreb_pct	4.48
net_rating	1.19

Addressing Multicollinearity

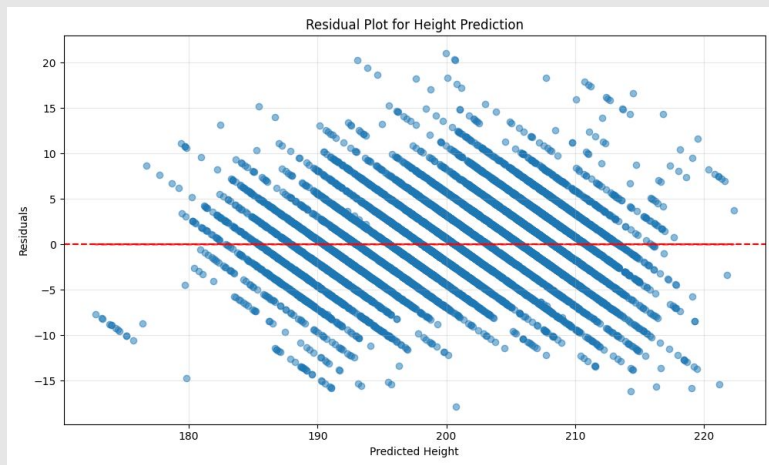
- Remove features with VIF >10 (in the following motivated order):
 - ts_pct, ast_pct, dreb_pct, usg_pct



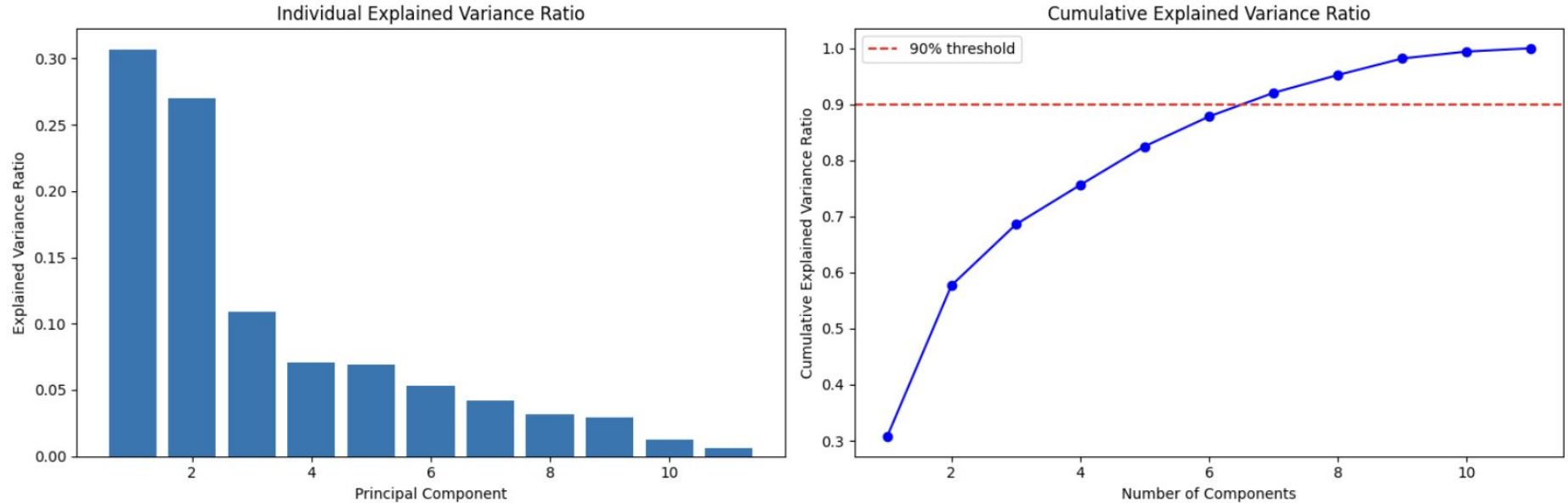
Feature	VIF
player_weight_log	8.45
gp	7.64
reb	5.56
oreb_pct	4.23
ast	2.99
net_rating	1.12

A Parsimonious Model for Height

- We can fit all subsets of these six features using the AIC criterion (again)
 - 'net_rating' and 'gp' gets dropped as a result
- This leads to our final linear regression model for height:
 - $\text{Height Estimate} = -12 + 46 \cdot \log(\text{weight}) + 0.66 \cdot \text{reb} - 1.16 \cdot \text{ast} + 10.96 \cdot \text{oreb_pct}$
- Resulting $R^2 = 0.74$ and **Test RMSE = 4.60** → and only four features!



Principal Component Analysis (since features are correlated)



- First 7 component uses: ast, ast_pct, pts, usg_pct, player_weight_log
- PCA Model Performance: $R^2 = 0.72$ and **Test RMSE = 4.85** → slightly worse

Regression for Points (ppg)

Note that avg. points per game is a variable with high variance because it is influenced by numerous dynamic and context-dependent factors. Unlike height, which is biometric, how much points a player scores depends on hundreds of factors including team role, team strategy, overall form, matchups, injuries, and game-to-game factors.

To start off the full dataset had 20 possible predictors including 1 target variable in this case “points” .

- 15 variables are numerical
- 5 variables are categorical [Player name, Team, College, Country, Season]

To predict points we chose a more expansive approach with many more variables included in the model because factors from the team you play for to what season it is could influence a players points.

We decided to keep all categorical variables, not including player name, to prepare for encoding.

Encoding & Feature Selection

For two variables, team and season, we directly encoded all entries into our dataset increasing its number of features. For the other two variables, country and college, we encoded only the 11 and 23 of the most frequently appearing countries and colleges respectively. [For scale there were 64 different countries and 246 different colleges]

We used Forward Selection on the Numerical data, and 11 of the 15 variables were kept.

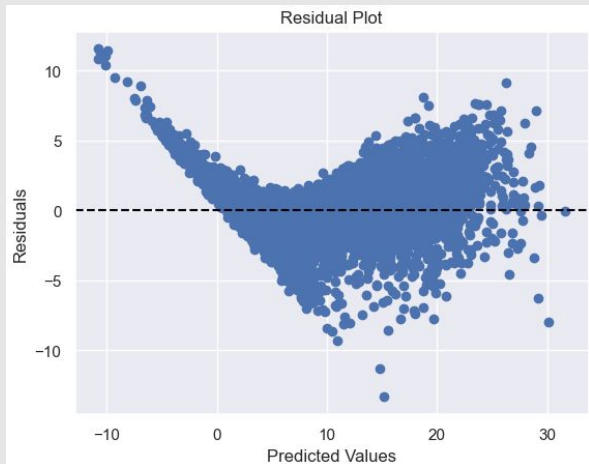
- The dropped variables were player height and weight, draft year, and net rating

Next we combined our numerical and newly encoded data for a grand total of 108 features!

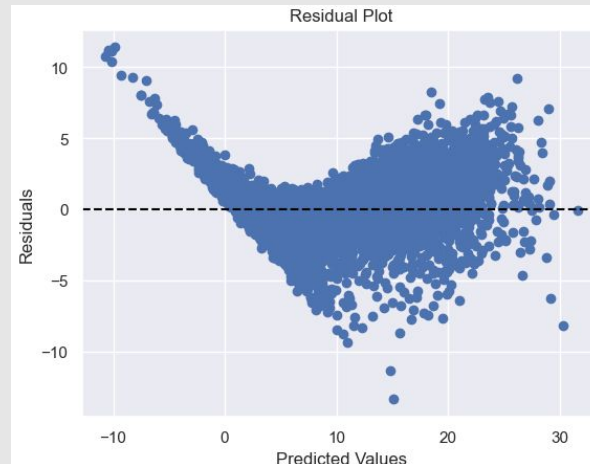
Numerical	Team	Season	Country	College
11	36	27	11	23

More Feature Selection

We then dropped all features with $p\text{-value} > 0.05$ to reduce complexity and keep only significant features. This reduced our number of features from 108 \rightarrow 68. Linear Regression Plots:



Features: 108
R2: 0.921
Adj. R2: 0.920



Features: 68
R2: 0.921
Adj. R2: 0.920

This is good news we dropped 40 features and had little to no effect on our scores!

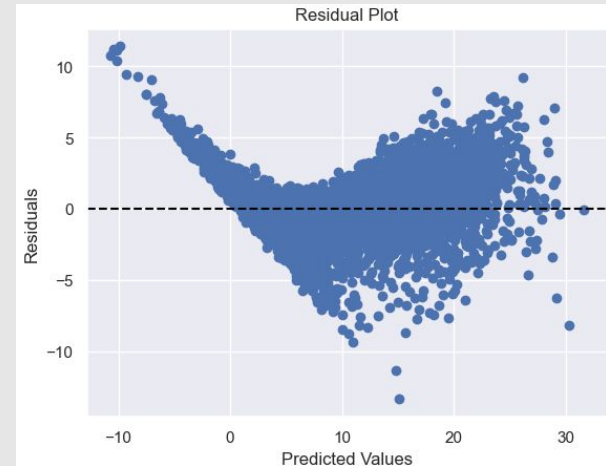
Splitting and Model Selection

In order to avoid multicollinearity we split our data in a way to make sure players that appear multiple times are grouped together during the split. [We used a conventional 80% train 20% test split]

Now because our linear regression Residual vs. Predicted Points plot is obviously non-linear, this indicates that the relationship between Points and the features cannot adequately be captured by a linear model. (Without heavy feature engineering)

Because of this we will fit some other models that can handle complex and non-linear relationships without requiring explicit feature transformations.

We hope these models will improve scores like R^2 and MSE.

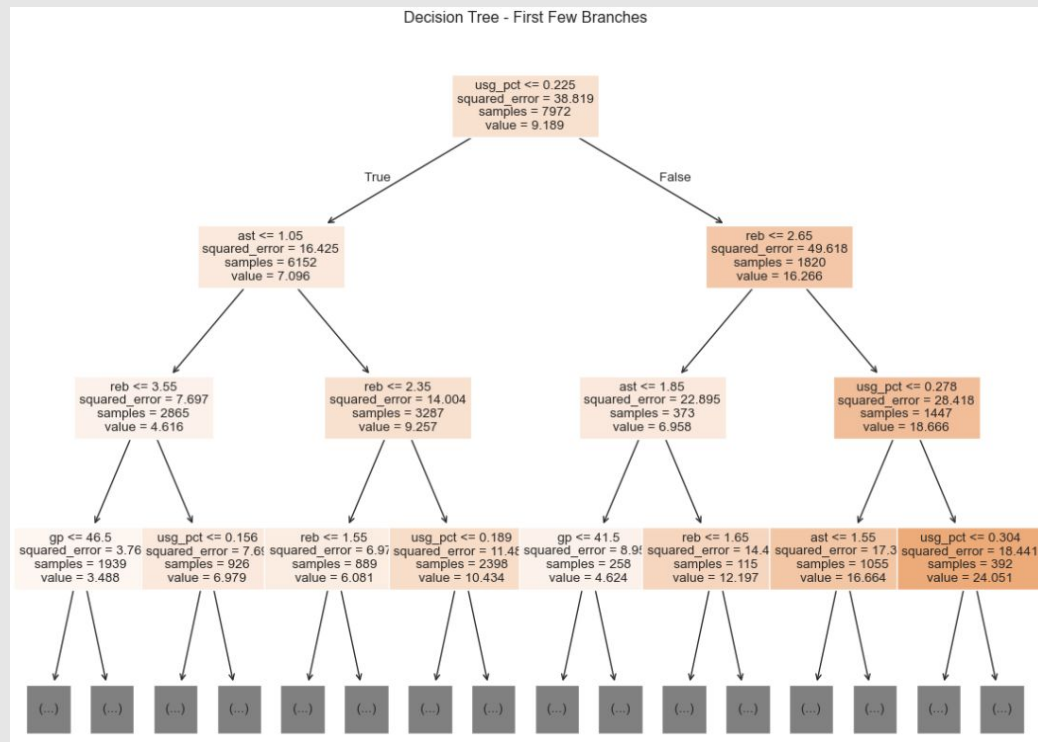


Model 1: Decision Tree Regressor

Decision tree regression works by using a tree-like structure to predict a continuous target variable by splitting the data into smaller regions based on feature thresholds. Each split is based on a feature and threshold that minimizes error. (MSE in this case)

To reduce overfitting and improve generalization we also pruned the tree which works by removing the 'branches' that have little predictive power.

To fit all of our different models we used cross validation to ensure our models generalize well to unseen data and reduces bias and variance in performance evaluation.



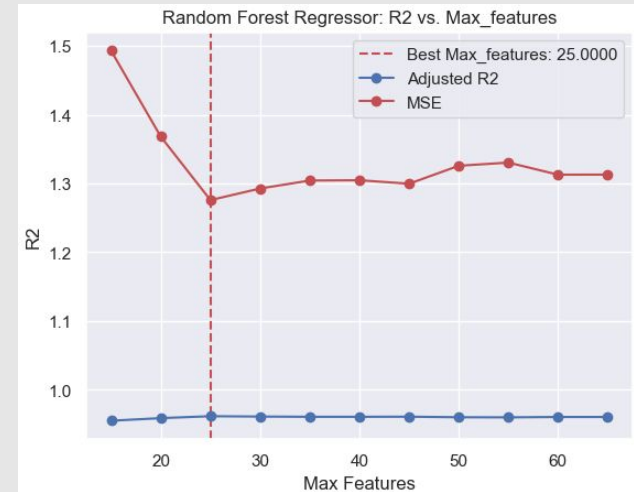
Model 2 : Random Forest Regressor

Random forest regression is an ensemble learning method that combines multiple decision trees to predict continuous values. It creates multiple subsets of the training data and fits a regression tree to each subset, splitting data at nodes based on features that minimize error.

We decided to optimize RFR by first evaluating its main parameter 'Max_features' and finding which value maximizes its R2 value.

We then used cross validation along with a grid search to optimize four other parameters that could potentially improve its performance.

Feature	Importance
usg_pct	0.378
ast	0.213
reb	0.190

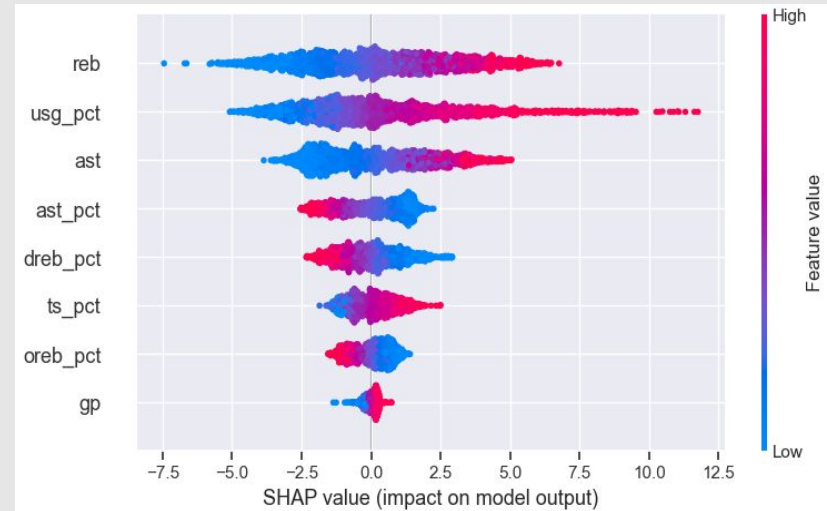
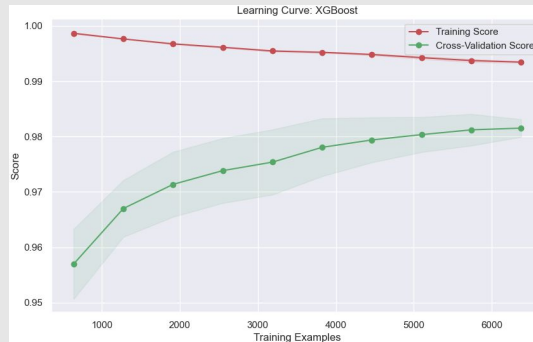


Model 3: XGBoost Regressor

XGBoost regression is machine learning algorithm that is built to handle complex non-linear relationships between features and a target variable. It builds decisions trees where each new tree corrects the errors of the previous ones and uses complex optimization techniques.

For this model we used cross validation along with a comprehensive grid search to optimize ten parameters with the goal of increasing R2.

Now it's time to see how our models performed on the test data!

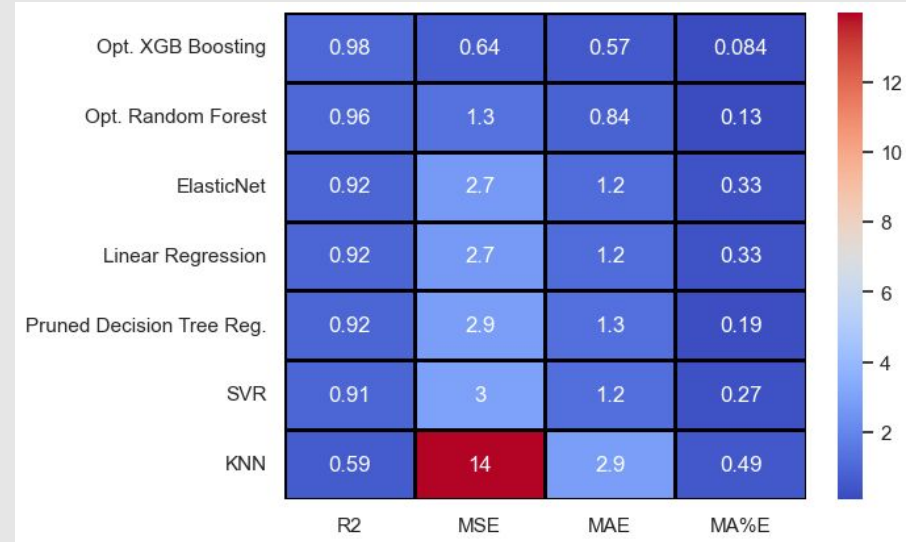


Model Results

All of the models we used were trained on the training data and evaluated on the test data and the results are:

It's no surprise that models like our optimized random forest or XGBoost were able to capture the non-linear relationships in the data most effectively.

However, our linear regression model given its low complexity compared to other models performed decently well; able to explain 92% of the variance in the points variable.

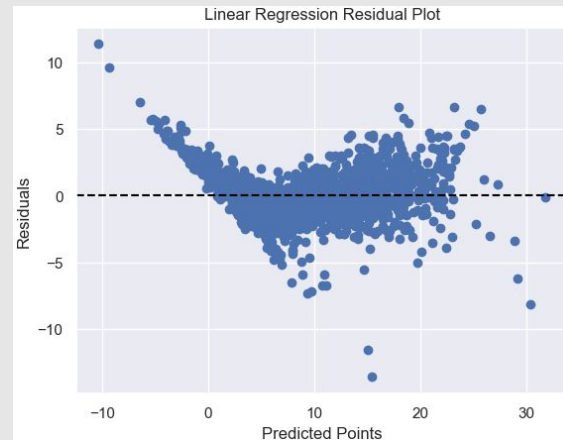
A heatmap table showing performance metrics for seven models. The columns are R2, MSE, MAE, and MA%E. The rows are Opt. XGB Boosting, Opt. Random Forest, ElasticNet, Linear Regression, Pruned Decision Tree Reg., SVR, and KNN. A color scale on the right ranges from 2 (dark blue) to 12 (dark red).

Opt. XGB Boosting	0.98	0.64	0.57	0.084
Opt. Random Forest	0.96	1.3	0.84	0.13
ElasticNet	0.92	2.7	1.2	0.33
Linear Regression	0.92	2.7	1.2	0.33
Pruned Decision Tree Reg.	0.92	2.9	1.3	0.19
SVR	0.91	3	1.2	0.27
KNN	0.59	14	2.9	0.49
	R2	MSE	MAE	MA%E

Model Comparison: XGBoost vs. Linear Reg.



0.98	R2	0.92
0.64	MSE	2.7
0.57	MAE	1.2
97.6%	$ e < 2$	84.27%



The residuals in the second plot are curved suggesting that the linear model does not fully capture the relationship. While in the first plot, the residuals are more randomly distributed indicating better performance.

Search for a Simple Linear Model

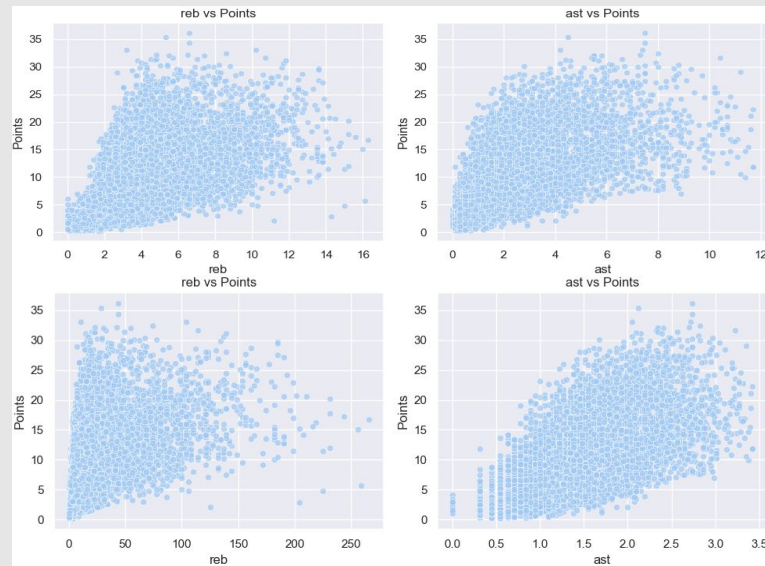
Lastly we would like to explore a simple and easily interpretable linear model for predicting 'points'.

We will start by only using the 11 numerical features since they seem to have the most importance when it comes to predicting 'points'. Then we dropped the features that crossed a certain variance inflation factor threshold bringing our features from 11 \rightarrow 6.

- Features: [draft_number, gp, reb, ast, oreb_pct, usg_pct]

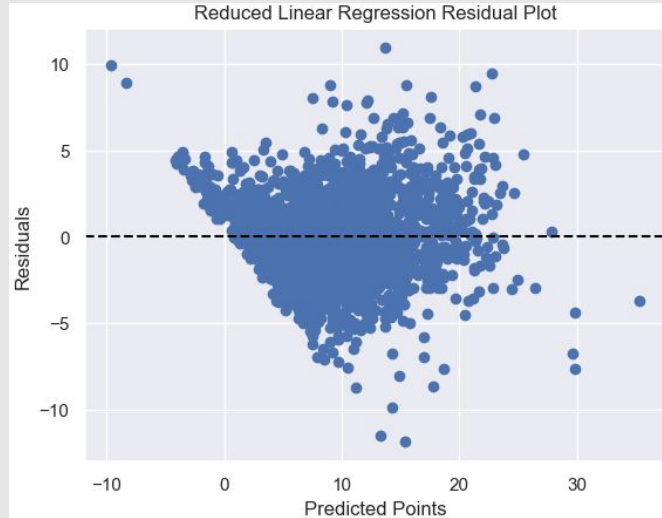
Next we transform the 'reb' and 'ast' features because they seem to have a non-linear relationship with 'points'

reb \rightarrow reb²
ast \rightarrow sqrt(ast)



Search for a Simple Linear Model

We then trained our linear regression model on the reduced training data and evaluated its performance on the reduced testing data.



R2	0.82
MSE	6.2
MAE	1.9
MA%E	0.41
e <2	61.60%

This leads us to our estimate for 'points':

$$\text{Points Estimate} = -8.8 + 57(\text{usg_pct}) - 12(\text{oreb_pct}) + 3.2\text{sqrt}(\text{ast}) + 0.06(\text{reb}^2) + 0.05(\text{gp}) - 0.02(\text{draft \#})$$

Inference Example #1

$$\text{Points Estimate} = -8.8 + 57(\text{usg_pct}) - 12(\text{oreb_pct}) + 3.2\text{sqrt}(\text{ast}) + 0.06(\text{reb}^2) + 0.05(\text{gp}) - 0.02(\text{draft \#})$$

Example: Victor Wembemyama 1st Year Stats in 2023 (Out of Sample):

31.2 Usage Rate, 7.4 OREB%, 3.9 APG, 10.6 RPG, 71 games played, #1 overall pick

Using our formula:

$$\text{Points Estimate} = -8.8 + 57(.312) - 12(.074) + 3.2\text{sqrt}(3.9) + 0.06(10.6^2) + 0.05(71) - 0.02(1) = 24.99$$

Prediction Interval:

$$\hat{y}_0 \pm t^* \cdot \sqrt{\text{MSE} \cdot (1 + \mathbf{x}_0^\top (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{x}_0)}$$

We get the following 95% prediction interval for Victor's ppg: [19.8, 30.1], which contains his actual ppg, 21.4
Could adjust predictors to gain insight about future seasons (i.e increase usage)

Inference Example #2

$$\text{Height Estimate} = -12 + 46 \cdot \log(\text{weight}) + 0.66 \cdot \text{reb} - 1.16 \cdot \text{ast} + 10.96 \cdot \text{oreb_pct}$$

Example: Victor Wembemyama 1st Year Stats in 2023 (Out of Sample):

107 kg, 7.4 OREB%, 10.6 RPG, 3.9 APG

Using our formula:

$$\text{Height Estimate} = -12 + 46 \cdot \log(107) + 0.66 \cdot 10.6 - 1.16 \cdot 3.9 + 10.96 \cdot 0.074 = 206 \text{ cm}$$

Prediction Interval:

$$\hat{y}_0 \pm t^* \cdot \sqrt{\text{MSE} \cdot (1 + \mathbf{x}_0^\top (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{x}_0)}$$

We get the following 95% prediction interval for Victor's height: [197.2, 215.4] → but he is 221cm!
He is an outlier in height, and low weight and agile, so high assists → he is a counterexample to the model



Conclusion

Summary of Analysis and Interpretation of Results

The Big Picture

- **Linear Regression vs Other Models**

- Linear Regression did not perform quite as well as Random Forest and XGBoost models in predicting points, but still did well overall
- Linear regression allowed us to uncover and interpret critical relationships between covariates
- Linear Regression outperformed PCA in predicting player height

- **NBA Insights**

- We can pretty accurately predict height using only four features, which is cool!
- In a practical sense, we can more accurately predict points given all basic statistics, which is likely used for scouting and coaching analytics in various basketball organizations

Next Steps / Potential for Improvement

- We can apply all the ML models we applied to predicting points to predict height
- We can predict other variables, like rebounds or games played
- We can use college performance statistics to predict NBA statistics - this would be even more useful for scouting and drafting purposes
- We can use data that spans more seasons
- We can seek out even more features per player per season, such as minutes per game or W/L result