NBA players' salary prediction model

Chun Yu Lee
cl5901@rit.edu

I Introduction

After taking the linear regression course, the simple linear regression model and multiple linear regression model concepts bring many thoughts to me about how to discover a set of data by building a model. Moreover, I am a big fan of sports, especially basketball. The biggest basketball league is National Basketball Association (NBA). Therefore, I wonder whether I could use the stats of players' performance to predict the reasonable salary of a player. That's why I choose this topic: NBA player's salary prediction.

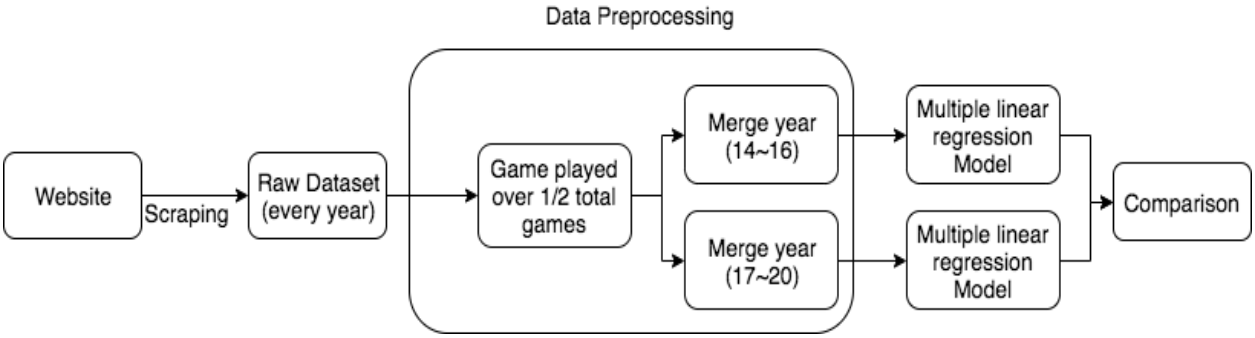
The expected result should be constructing two different models. One is from the year 2014 to 2016, and the other is from 2017 to 2020. Then, we could see the performance of these two models and the difference between each other.

II Data

Data is the foundation of a regression model. It is not possible having a good model without an appropriate dataset. The first step is to acquire the dataset. There are many ways to acquire a dataset. For example, surveys, web scraping, open data, enterprise databases are the main methods. In this project, web scraping with Python was implemented to obtain the dataset. The second step is to preprocess the dataset. How to define and use the data will determine the way to do the data preprocessing.

The year range of this dataset is from 2014 to 2020. There are over 25 variables in the original dataset. But I only used five to ten variables to build the model. Explanation of each term is shown at Table 1 in appendix.

Figure 1: Data Overview Diagram



II..1 Data acquisition

The data comes from two source. The data relevant to salary is from HOOPHYPE [1], and the data relevant to players' performance is from BASKETBALL REFERENCE [2]. The web scraping with Python is implemented here to acquire the data.

II..2 Data preprocessing

The regular games in NBA are 82 games. However, because of the pandemic, there were only 72 games in season 2019-2020. In this project, I tried to filter out the players playing less than half of the total games. The reason why choose half of the total games is because the players may encounter injury or retirement during the season. It did not reflect the relationship between expected players' performance and their salary. I sum up every player's record because the players traded, or released, or bought out by the original team is common.

The performance data every year has divided the games every individual player played. For example, a player got 72 points in a season, and he played 72 games. Then the point per game would be 1.

The next step is to merge the data from 2014 to 2016 and from 2017 to 2020 into an individual dataset for the model building. I can gain more information from the period of three years instead of only one year.

III Study Design

In this project, step-wise selection was implemented to find out the suitable variables. After I found the appropriate variables, I tried to build the model and test the accuracy. There are two ways to check the accuracy from residual. One is plotting residual with y predicted value, and the other is the normal probability plot.

With the completed model accuracy checking, I utilize the three different transformations as equations 1, 2, 3 to correct the model inadequacies.

$$y' = log(y) \tag{1}$$

$$y' = sqrt(y) \tag{2}$$

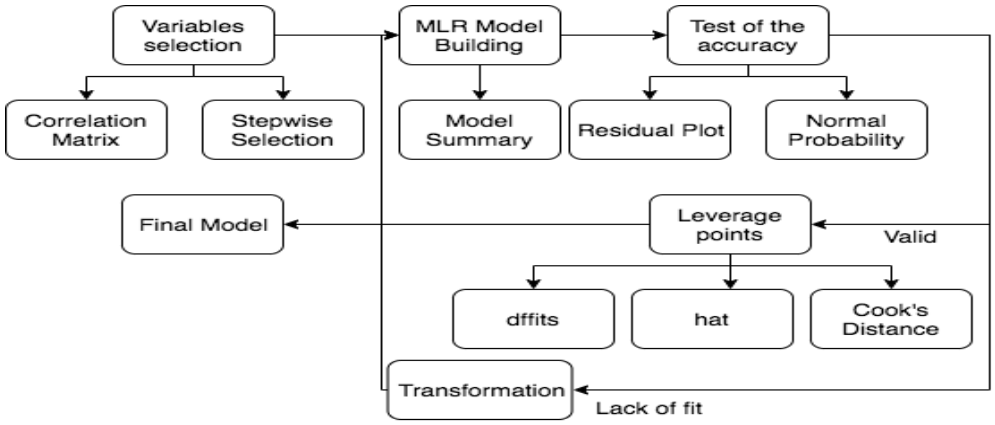
$$y' = square(y) \tag{3}$$

Once I confirmed my model is a valid model, then my next step is trying to enhance the performance of the model. For example, I examined the leverage points by using dffits, hat, cook's Distance.

[1] https://hoopshype.com/salaries/players/2016-2017/

[2] https://www.basketball-reference.com/leagues/NBA_2017_totals.html

Figure 2: Study Design



IV Model

There are two dataset. One is season 2014 to season 2016, and the other is season 2017 to season 2020. I would like to see whether there is any difference in the model between two different period.

IV.I season 2014 to season 2016

This model is building with the data range from season 2014 to season 2016. In the first beginning, I check the correlation of every variable. It seems that there is no severe multicollinearity issue here from the perspective of the correlation, as Figure 3.

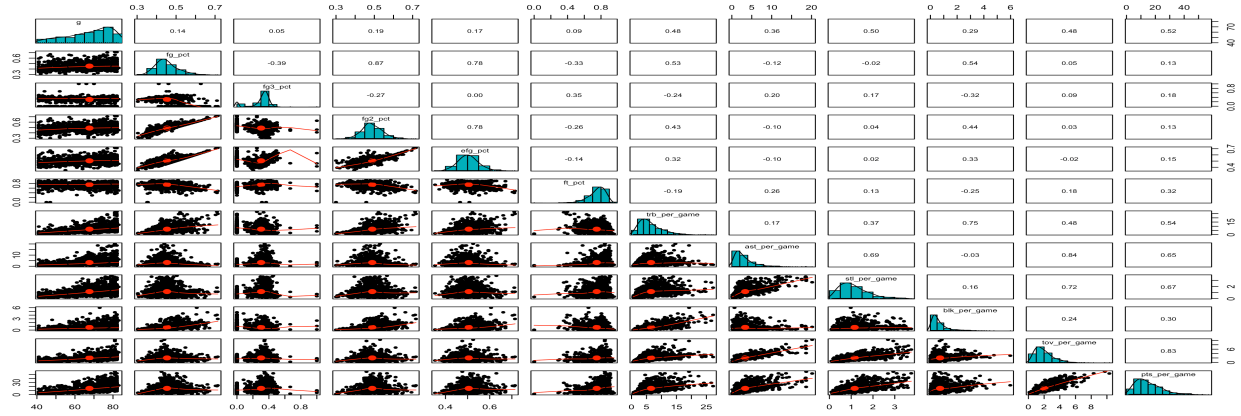
The model was built through the step-wise selection method. The R Square is 42.3%, and the p-value is smaller than 2.2e-16. It means that this is a valid model. You could see the details in the Figure in the appendix. The fitted model here is:

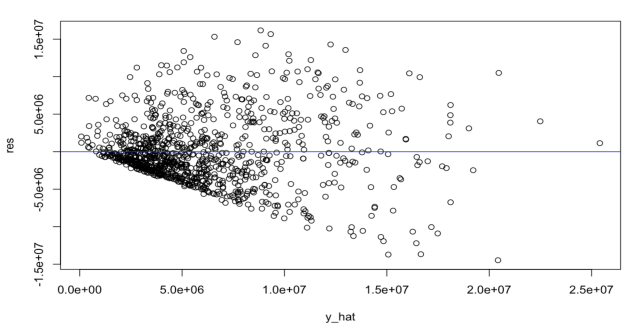
salary = 6661597 + 350814*x1 - 114743*x2 + 545867*x3 + 699907*x4 - 1230766*x5 - 1073984*x6 - 602858*x7

(x1 = pts_per_game, x2 = g, x3 = trb_per_game, x4 = ast_per_game, x5 = stl_per_game, x6 = tov_per_game, x7 = blk_per_game)

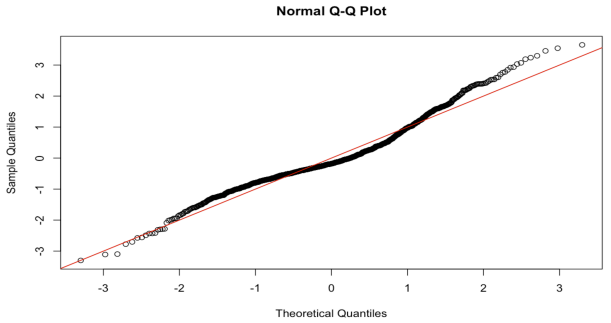
Because of the testing of the accuracy, I plotted the residual with the prediction value and the normal probability. Figure 4,5 indicates this model is not good. The Residual plot suggests no constant variance, and the normal probability plot shows that the model may be sensitive to the small subset of the data.

Figure 3: Correlation Matrix



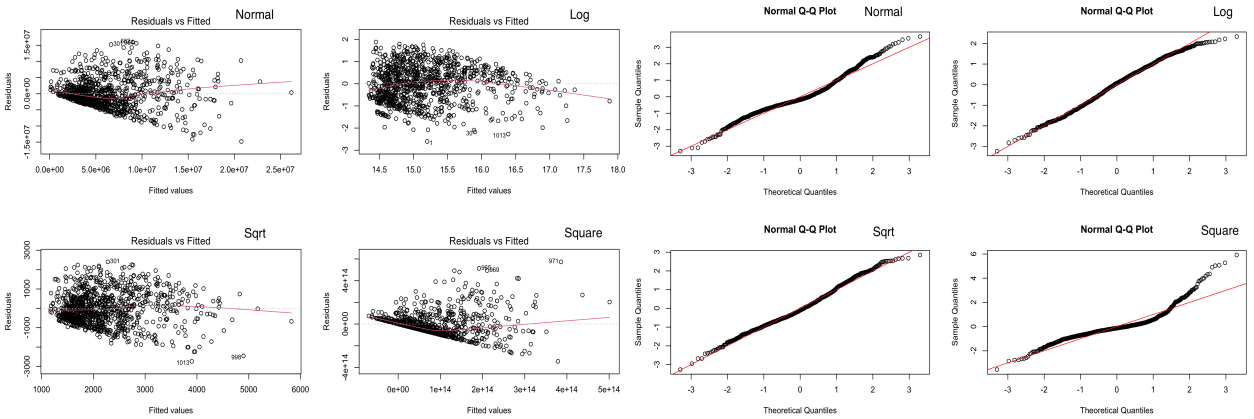


(a) Residual plot



(b) Normal Probability

Due to the lack of fit Regression Model, I am trying to implement the transformation to see whether I could find a more proper model to predict the salary. I try to transform the salary by logarithm, square root, square to find the best one. The residual plot and normal probability, the original and square root, have funnel shape in the residual plot. It shows the variance is getting bigger when the predicted value is becoming bigger. Thus, the transformation by logarithm has the best result for now. There are more details of the variables selection process and the summary of each model in the appendix.



(c) Residual plot

(d) Normal Probability

The model after transformation: log(salary) = 14.72 + 0.055845*x1 - 0.009379*x2 + 0.0823*x3 + 0.095673*x4 - 0.141021*x5 - 0.187254*x6 - 0.1288*x7
(x1 = pts_per_game, x2 = g, x3 = trb_per_game, x4 = ast_per_game, x5 = stl_per_game, x6 = tov_per_game, x7 = blk_per_game)

After finding a valid model, I am going to improve the R Square value. Removing the leverage points could improve the R Square value. The first method to find out the leverage points is diffits.

$$|dffits| > 2 * \sqrt{p/n} = 2 * \sqrt{8/1027} = 0.1765 \tag{4}$$

The second method to find out the leverage points is hat.

$$hat > 2 * \frac{p}{n} = 2 * \frac{8}{1027} = 0.01558 \tag{5}$$

The last method to find out the leverage points here is cook's Distance.

$$D_i > 1 \tag{6}$$

From the result, I found 104 possible leverage points by hat value and 54 possible leverage points by dffits value and 0 leverage points by cook's value. There are more details in Table 2, 3 in the appendix.

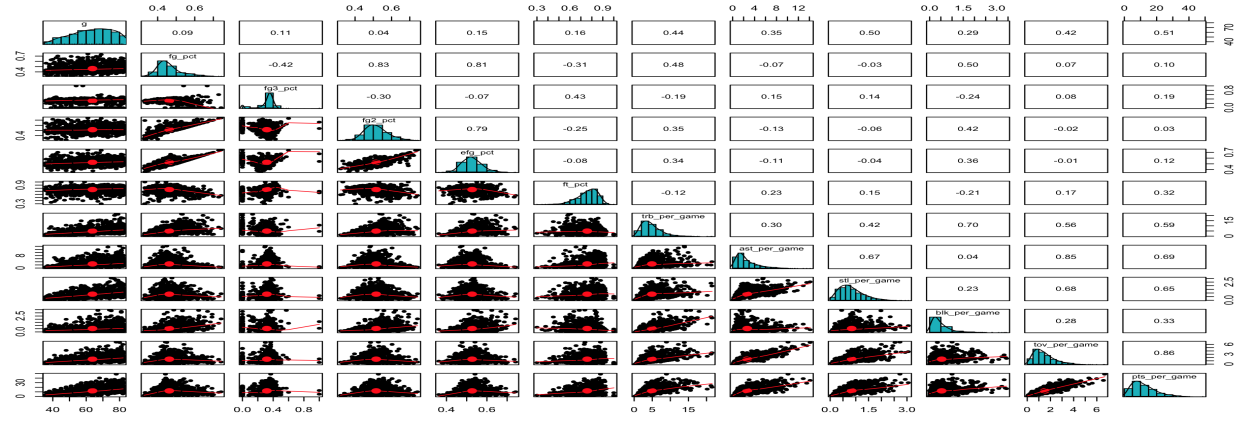
IV.II season 2017 to season 2020

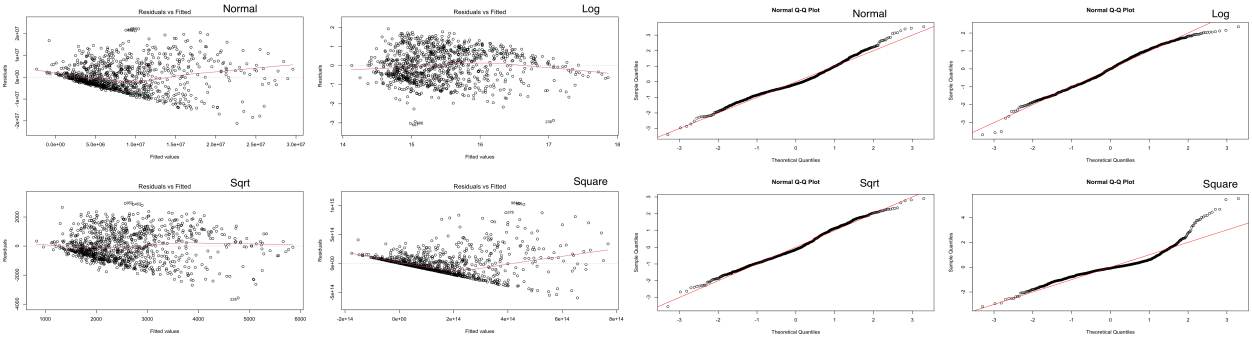
You can see a similar result in season 2017 to season 2020. The most suitable model is still the Logarithm model. There are more details of the model selection and summary in the appendix. All the potential leverage points are shown in the appendix as well.
The model without transformation:
salary = 9271901 + 647399*x1 - 193254*x2 + 862598*x3 + 1448346*x4 - 3031190*x6
(x1 = pts_per_game, x2 = g, x3 = trb_per_game, x4 = ast_per_game, x6 = tov_per_game)

The model after transformation:

log(salary) = 15.255 + 0.063363*x1 - 0.017746*x2 + 0.1031*x3 + 0.1186*x4 - 0.119573*x5 - 0.229*x6 - 2.118*x8 + 3.9138*x9 - 0.6455*x10 - 1.7272*x11
(x1 = pts_per_game, x2 = g, x3 = trb_per_game, x4 = ast_per_game, x5 = stl_per_game, x6 = tov_per_game, x8 = fg2_pct, x9 = efg_pct, x10 = fg3_pct, x11 = fg_pct)

Figure 4: Correlation Matrix





(a) Residual plot (b) Normal Probability

V Results & Conclusion

The p-value of the two log models is smaller than 0.05. It means that the regression is statistically significant. So the salary transformed by logarithm has a linear relationship with these variables. Furthermore, the residual plot and normal probability show that they are valid models. However, they have lower R Square now. For example, the 14-17 log model has 35 percent, and the 17-20 log model has 37 percent in R square. Moreover, I found out the potential leverage points in the two models. Verifying the leverage points is a good direction I could work on in the future.

We could see there is a bit of difference between these two models. The significant coefficients are different. The question here is whether the league focuses on other performance in these two

periods. That will be an interesting question to be investigated in the future.

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  14.720197   0.157935  93.204  < 2e-16 ***
pts_per_game  0.055845   0.005140  10.865  < 2e-16 ***
trb_per_game  0.082302   0.011110   7.408 2.69e-13 ***
g            -0.009379   0.002736  -3.428 0.000633 ***
ast_per_game  0.095673   0.018789   5.092 4.22e-07 ***
tov_per_game -0.187254   0.054896  -3.411 0.000673 ***
stl_per_game -0.141021   0.056190  -2.510 0.012237 *
blk_per_game -0.128800   0.052611  -2.448 0.014527 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8053 on 1019 degrees of freedom
Multiple R-squared:  0.3536,    Adjusted R-squared:  0.3491
F-statistic: 79.63 on 7 and 1019 DF,  p-value: < 2.2e-16
```

(c) 14-16 Log- Model Summary

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  15.254563   0.325078  46.926  < 2e-16 ***
pts_per_game  0.063363   0.007876   8.045 2.40e-15 ***
trb_per_game  0.103097   0.013746   7.500 1.40e-13 ***
g            -0.017746   0.002715  -6.537 9.93e-11 ***
ast_per_game  0.118606   0.025569   4.639 3.97e-06 ***
tov_per_game -0.228989   0.079989  -2.863 0.004287 **
fg2_pct      -2.118152   0.765607  -2.767 0.005767 **
efg_pct       3.913812   1.126810   3.473 0.000536 ***
stl_per_game  0.119573   0.079956   1.495 0.135104
fg3_pct      -0.645530   0.308491  -2.093 0.036639 *
fg_pct       -1.727196   0.984111  -1.755 0.079549 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8273 on 1009 degrees of freedom
Multiple R-squared:  0.3701,    Adjusted R-squared:  0.3639
F-statistic: 59.29 on 10 and 1009 DF,  p-value: < 2.2e-16
```

(d) 17-20 Log- Model Summary

VI Future Work

First, the leverage points could affect the accuracy of the model. Therefore, I would try to verify the potential leverage points I found. Secondly, I have enough data points now, so I don't need to acquire more data points. Instead, I would try to gain more new features. This method could help me build up a model with higher accuracy. Last but not least, there is a difference between these two models. I will try to discover more about the difference. Hopefully, I could make this model better.

VII Appendix

VII.I Terms

Table 1: Terms

Terms	Explanation
Age	Age of the player.
Pos	Position- C, PF, SF, SG, PG
G	Games Played
GS	Number of games being Starting lineup
MP	Minute played
FG	Field Goal Made
FGA	Field Goal Attempts
fg_pct	Field Goal Percentage
3P	Three points Made
3PA	Three points Attempts
fg3_pct	Three points percentage
2P	Two points Made
2PA	Two points Attempts
fg2_pct	Two points percentage
efg_pct	Effective Field Goal percentage – adjust the stat that the 3-point field goal being one more score than 2-point field goal.
FT	Free Throws Made
FTA	Free Throws Attempts
FT%	Free Throws Percentage
ORB	Offensive Rebounds
DRB	Defensive Rebounds
TRB	Total Rebounds
AST	Assists
STL	Steals
BLK	Blocks
TOV	Turnover
PF	Personal Foul
PTS	Points
Salary	Salary in a specific year
pts_per_game	points per game
g	games played
trb_per_game	total rebounds per game
ast_per_game	assistance per game
stl_per_game	steal per game
tov_per_game	turnover per game
blk_per_game	block per game

VII.II Model Building

VII.II.1 season 2014 to season 2016

```
Call:
lm(formula = salary ~ pts_per_game + g + trb_per_game +
 ast_per_game +
    stl_per_game + tov_per_game + blk_per_game, data = n
ba_data)

Coefficients:
 (Intercept)  pts_per_game             g
     6661597        350813       -114743
trb_per_game  ast_per_game  stl_per_game
      535867        699907      -1230766
tov_per_game  blk_per_game
    -1073984       -602858
```

(e) Variables selection

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   6661597     868834   7.667 4.09e-14 ***
pts_per_game   350814      28276  12.407  < 2e-16 ***
g             -114743      15051  -7.623 5.64e-14 ***
trb_per_game   535867      61121   8.767  < 2e-16 ***
ast_per_game   699907     103360   6.772 2.15e-11 ***
stl_per_game -1230766     309112  -3.982 7.33e-05 ***
tov_per_game -1073984     301996  -3.556 0.000393 ***
blk_per_game  -602858     289426  -2.083 0.037506 *
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4430000 on 1019 degrees of freedom
Multiple R-squared:  0.423,     Adjusted R-squared:  0.4191
F-statistic: 106.7 on 7 and 1019 DF,  p-value: < 2.2e-16
```

(f) Model Summary

```
Call:
lm(formula = y_log ~ pts_per_game + trb_per_game + g + ast_
per_game +
    tov_per_game + stl_per_game + blk_per_game, data = x)

Coefficients:
 (Intercept)  pts_per_game  trb_per_game
   14.720197      0.055845      0.082302
           g  ast_per_game  tov_per_game
   -0.009379      0.095673     -0.187254
stl_per_game  blk_per_game
   -0.141021     -0.128800
```

(g) Log model Variables selection

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  14.720197   0.157935  93.204  < 2e-16 ***
pts_per_game  0.055845   0.005140  10.865  < 2e-16 ***
trb_per_game  0.082302   0.011110   7.408 2.69e-13 ***
g            -0.009379   0.002736  -3.428 0.000633 ***
ast_per_game  0.095673   0.018789   5.092 4.22e-07 ***
tov_per_game -0.187254   0.054896  -3.411 0.000673 ***
stl_per_game -0.141021   0.056190  -2.510 0.012237 *
blk_per_game -0.128800   0.052611  -2.448 0.014527 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8053 on 1019 degrees of freedom
Multiple R-squared:  0.3536,    Adjusted R-squared:  0.3491
F-statistic: 79.63 on 7 and 1019 DF,  p-value: < 2.2e-16
```

(h) Log- Model Summary

```
Call:
lm(formula = y_sqrt ~ pts_per_game + trb_per_game + g + ast
_per_game +
    tov_per_game + stl_per_game + blk_per_game, data = x)

Coefficients:
 (Intercept)  pts_per_game  trb_per_game
     2059.41         64.77         97.88
           g  ast_per_game  tov_per_game
     -16.92        122.62       -212.84
stl_per_game  blk_per_game
     -189.55       -126.54
```

(i) Sqrt model Variables selection

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  2059.407    165.315  12.457  < 2e-16 ***
pts_per_game   64.768      5.380  12.038  < 2e-16 ***
trb_per_game   97.877     11.630   8.416  < 2e-16 ***
g             -16.925      2.864  -5.910 4.67e-09 ***
ast_per_game  122.618     19.667   6.235 6.61e-10 ***
tov_per_game -212.838     57.461  -3.704 0.000224 ***
stl_per_game -189.550     58.816  -3.223 0.001310 **
blk_per_game -126.541     55.070  -2.298 0.021773 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 842.9 on 1019 degrees of freedom
Multiple R-squared:  0.4046,    Adjusted R-squared:  0.4005
F-statistic: 98.91 on 7 and 1019 DF,  p-value: < 2.2e-16
```

(j) Sqrt- Model Summary

```
Call:
lm(formula = y_square ~ pts_per_game + g + trb_per_game + ast_per_game
 +
    stl_per_game + tov_per_game + blk_per_game, data = x)

Coefficients:
 (Intercept)  pts_per_game             g  trb_per_game
   1.143e+14     7.284e+12    -2.834e+12     1.098e+13
ast_per_game  stl_per_game  tov_per_game  blk_per_game
   1.484e+13    -3.460e+13    -1.805e+13    -1.174e+13
```

(k) Square model Variables selection

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.143e+14  1.920e+13   5.953 3.62e-09 ***
pts_per_game  7.284e+12  6.248e+11  11.658  < 2e-16 ***
trb_per_game  1.098e+13  1.351e+12   8.126 1.28e-15 ***
g            -2.834e+12  3.326e+11  -8.522  < 2e-16 ***
ast_per_game  1.484e+13  2.284e+12   6.496 1.28e-10 ***
tov_per_game -1.805e+13  6.673e+12  -2.704  0.00696 **
stl_per_game -3.460e+13  6.831e+12  -5.066 4.82e-07 ***
blk_per_game -1.174e+13  6.396e+12  -1.836  0.06671 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 9.79e+13 on 1019 degrees of freedom
Multiple R-squared:  0.3944,    Adjusted R-squared:  0.3902
F-statistic:  94.8 on 7 and 1019 DF,  p-value: < 2.2e-16
```

(l) Square- Model Summary

VII.II.2 season 2017 to season 2020

```
Call:
lm(formula = salary ~ pts_per_game + g + trb_per_game + ast_per_game +
    tov_per_game, data = nba_data)

Coefficients:
 (Intercept)  pts_per_game             g  trb_per_game
     9271901       647399      -193254       862599
ast_per_game  tov_per_game
     1448346     -3031190
```

(m) Variables selection

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  15.254563   0.325078  46.926  < 2e-16 ***
pts_per_game  0.063363   0.007876   8.045 2.40e-15 ***
trb_per_game  0.103097   0.013746   7.500 1.40e-13 ***
g            -0.017746   0.002715  -6.537 9.93e-11 ***
ast_per_game  0.118606   0.025569   4.639 3.97e-06 ***
tov_per_game -0.228989   0.079989  -2.863 0.004287 **
fg2_pct      -2.118152   0.765607  -2.767 0.005767 **
efg_pct       3.913812   1.126810   3.473 0.000536 ***
stl_per_game  0.119573   0.079956   1.495 0.135104
fg3_pct      -0.645530   0.308491  -2.093 0.036639 *
fg_pct       -1.727196   0.984111  -1.755 0.079549 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8273 on 1009 degrees of freedom
Multiple R-squared:  0.3701,    Adjusted R-squared:  0.3639
F-statistic: 59.29 on 10 and 1009 DF,  p-value: < 2.2e-16
```

(n) Model Summary

```
Call:
lm(formula = y_log ~ pts_per_game + trb_per_game + g + ast_per_game +
    tov_per_game + fg2_pct + efg_pct + stl_per_game + fg3_pct +
    fg_pct, data = x)

Coefficients:
 (Intercept)  pts_per_game  trb_per_game             g
    15.25456       0.06336       0.10310      -0.01775
ast_per_game  tov_per_game       fg2_pct       efg_pct
     0.11861      -0.22899      -2.11815       3.91381
stl_per_game       fg3_pct        fg_pct
     0.11957      -0.64553      -1.72720
```

(o) Log model Variables selection

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  15.254563   0.325078  46.926  < 2e-16 ***
pts_per_game  0.063363   0.007876   8.045 2.40e-15 ***
trb_per_game  0.103097   0.013746   7.500 1.40e-13 ***
g            -0.017746   0.002715  -6.537 9.93e-11 ***
ast_per_game  0.118606   0.025569   4.639 3.97e-06 ***
tov_per_game -0.228989   0.079989  -2.863 0.004287 **
fg2_pct      -2.118152   0.765607  -2.767 0.005767 **
efg_pct       3.913812   1.126810   3.473 0.000536 ***
stl_per_game  0.119573   0.079956   1.495 0.135104
fg3_pct      -0.645530   0.308491  -2.093 0.036639 *
fg_pct       -1.727196   0.984111  -1.755 0.079549 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8273 on 1009 degrees of freedom
Multiple R-squared:  0.3701,    Adjusted R-squared:  0.3639
F-statistic: 59.29 on 10 and 1009 DF,  p-value: < 2.2e-16
```

(p) Log- Model Summary

VII.III Leverage Points

Table 2: 14-16: Log model - dffits(index/value)

1	2	12	30	33	46
-0.2663497	-0.217752	-0.2524373	-0.2931715	-0.2236937	-0.1966318
57	100	117	151	206	220
-0.275748	-0.2228255	-0.2989792	-0.1961171	-0.2726245	-0.192257
249	261	267	280	291	297
0.1880334	0.1825913	0.1766852	0.1947357	0.1815379	0.2429564
322	336	352	353	366	371
-0.1788429	-0.2876984	-0.2060172	-0.1942716	-0.5619844	-0.1964885
385	421	473	477	501	577
-0.1953322	-0.1894477	-0.2211226	-0.3894009	-0.246501	-0.1820373
582	607	631	649	682	687
0.1864535	0.2568365	0.1794969	-0.2375646	-0.2244141	-0.1871359
712	753	755	762	770	785
-0.209625	-0.3468202	-0.2518123	-0.1815305	-0.1767029	-0.3248466
792	852	914	921	924	931
-0.2056146	-0.2601558	0.1774383	0.210681	0.1831047	0.1977725
933	949	970	998	1013	1021
0.1919225	0.2226857	-0.3227395	-0.3837757	-0.3688399	-0.2798411

Table 3: 14-16: Log model - hat(index/value)

57	91	93	109	117	134
0.02703963	0.02232608	0.01565064	0.02047724	0.02255659	0.02316997
148	191	206	210	220	237
0.02994584	0.02095736	0.05262069	0.01864772	0.02404011	0.01981573
239	252	256	257	261	264
0.0188821	0.01743428	0.03399967	0.03478404	0.02176356	0.01755272
265	271	272	274	280	283
0.02146067	0.03042049	0.02824285	0.0216366	0.016307	0.01790815
285	288	294	332	336	338
0.01917286	0.02247721	0.06085026	0.0205766	0.01648202	0.01614315
343	352	366	464	473	477
0.01682198	0.01734049	0.06227907	0.01654551	0.02502618	0.03623621
483	501	521	531	545	547
0.0224305	0.01828748	0.01976676	0.01785547	0.01752249	0.02039352
556	566	571	577	580	586
0.02281934	0.04665986	0.01636979	0.02396016	0.01826144	0.01824499
589	590	598	599	601	602
0.01670447	0.03290048	0.01681201	0.02691068	0.0242125	0.02874195
605	608	610	613	616	617
0.01734469	0.03126404	0.01653117	0.01924832	0.03142398	0.02433058
619	621	627	628	712	753
0.0204446	0.02593528	0.03701419	0.01798605	0.01881918	0.03136976
755	762	785	839	843	852
0.02488473	0.02645589	0.02210959	0.02220603	0.01996742	0.02729051
860	865	895	897	906	912
0.02442835	0.01845385	0.01707631	0.01846645	0.01593039	0.0284549
921	925	926	930	938	939
0.03335605	0.01815355	0.02930276	0.01668912	0.0208885	0.01928055
942	949	950	951	954	955
0.01685723	0.02661261	0.02592081	0.01782527	0.03662835	0.02869271
958	959	961	962	966	967
0.02226757	0.02513145	0.02067021	0.02044318	0.0179675	0.05795164
968	970	971	983	998	1002
0.01778836	0.05183218	0.0230776	0.01616917	0.02593312	0.01562399
1016	1021				
0.01759189	0.02290841				

Table 4: 17-20: Log model - dffits(index/value)

21	22	41	77	111	113
-0.317776	0.2635158	-0.2909371	-0.2170989	0.6537055	0.3425215
157	169	170	229	238	284
-0.2124535	-0.2178315	-0.3338842	0.2128811	-0.4759664	-0.3749864
344	348	358	359	412	418
0.2260351	0.2273148	-0.2881299	0.2164223	-0.2437352	-0.2162884
419	445	518	578	580	687
-0.2240938	-0.3213505	-0.2598498	-0.2236498	-0.2519927	-0.3427232
688	690	692	694	697	701
-0.2924947	-0.2215381	-0.2731471	-0.2893892	-0.2618326	-0.2755144
710	721	772	806	808	850
-0.2916623	-0.2964097	-0.221203	-0.2101513	-0.2417232	-0.2154602
878	903	942	980	1000	
0.2198853	0.2345768	0.221513	-0.3310169	-0.2489072	

Table 5: 17-20: Log model - hat(index/value)

9	13	21	59	62	80
0.04945895	0.03087764	0.03684033	0.0410354	0.03119088	0.02456943
82	92	111	123	128	149
0.03194506	0.0245707	0.09981361	0.03057255	0.02467959	0.02531661
169	170	191	197	253	256
0.02221851	0.02644141	0.02682723	0.02912444	0.02977277	0.02329413
270	284	320	328	347	351
0.03782414	0.02594905	0.0344896	0.03258635	0.04864787	0.02232951
358	378	386	401	412	418
0.03131826	0.02343746	0.02248822	0.02774594	0.02367187	0.0237697
422	423	429	445	475	495
0.03240126	0.02508329	0.02735524	0.06859865	0.04903982	0.02420476
498	518	541	561	589	591
0.02215935	0.0231574	0.02231163	0.02202195	0.02368686	0.02618503
597	603	619	620	638	660
0.02992482	0.03210811	0.02940799	0.03942893	0.0237733	0.03397954
680	694	697	701	710	721
0.02861645	0.03748572	0.02986569	0.02386166	0.02758248	0.02876226
745	749	755	763	779	780
0.02219441	0.10475005	0.02321314	0.02274405	0.04702798	0.03475421
816	850	869	890	934	940
0.02292127	0.02704868	0.02249442	0.02438113	0.02539885	0.03499956
942	944	951	962	963	1000
0.03983222	0.02440828	0.02675904	0.02708249	0.02935275	0.02325407
1006					
0.02199342					