



Predicting NBA player Performance and Salary

By: Kai Itokazu, Noah Masur, Michael Truong, Matthew Mayemura

Motivation

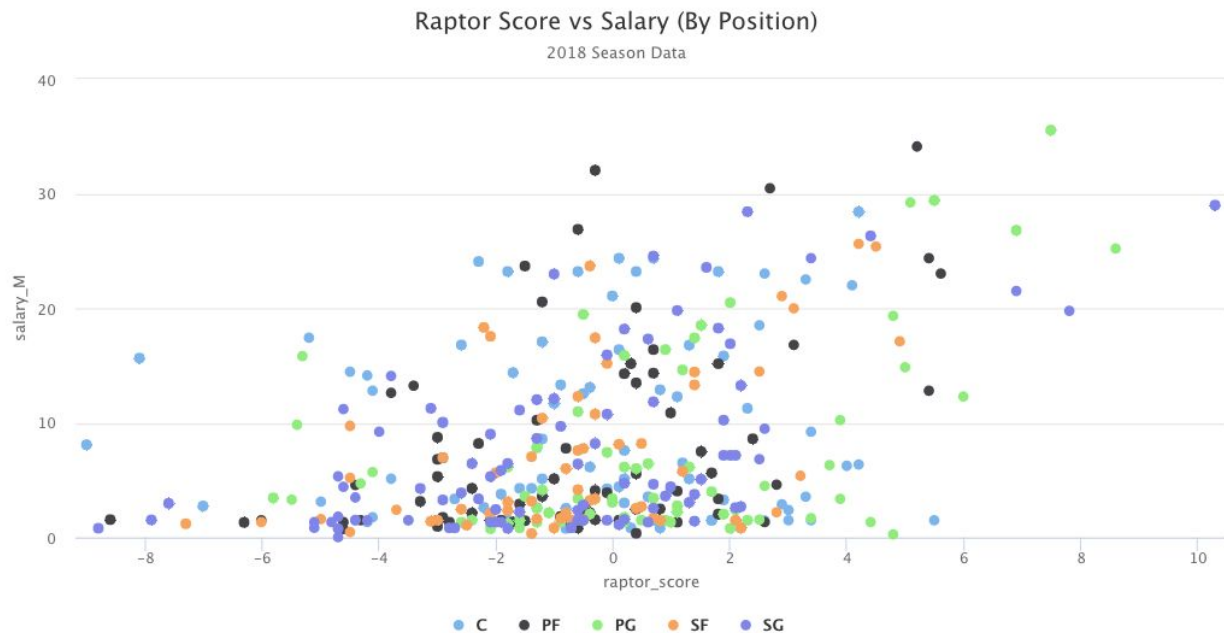
- Try to find the NBA players with the best value to performance
 - Like Moneyball
 - Which model best accomplishes this
- Business application
 - Helps teams make roster decisions
- To do this we forecast
 - Performance
 - Salary



RAPTOR

- Robust Algorithm (using) Player Tracking (and) On/Off Ratings
- Descriptive statistic based on Real Adjusted Plus Minus (RAPM)
 - Less noisy
 - Less computationally intensive
- Evaluates a players performance based on
 - Box scores
 - On off performance
- Limitations
 - Does not account for coaching, systems or synergies between teammates

RAPTOR Score vs Salary visual

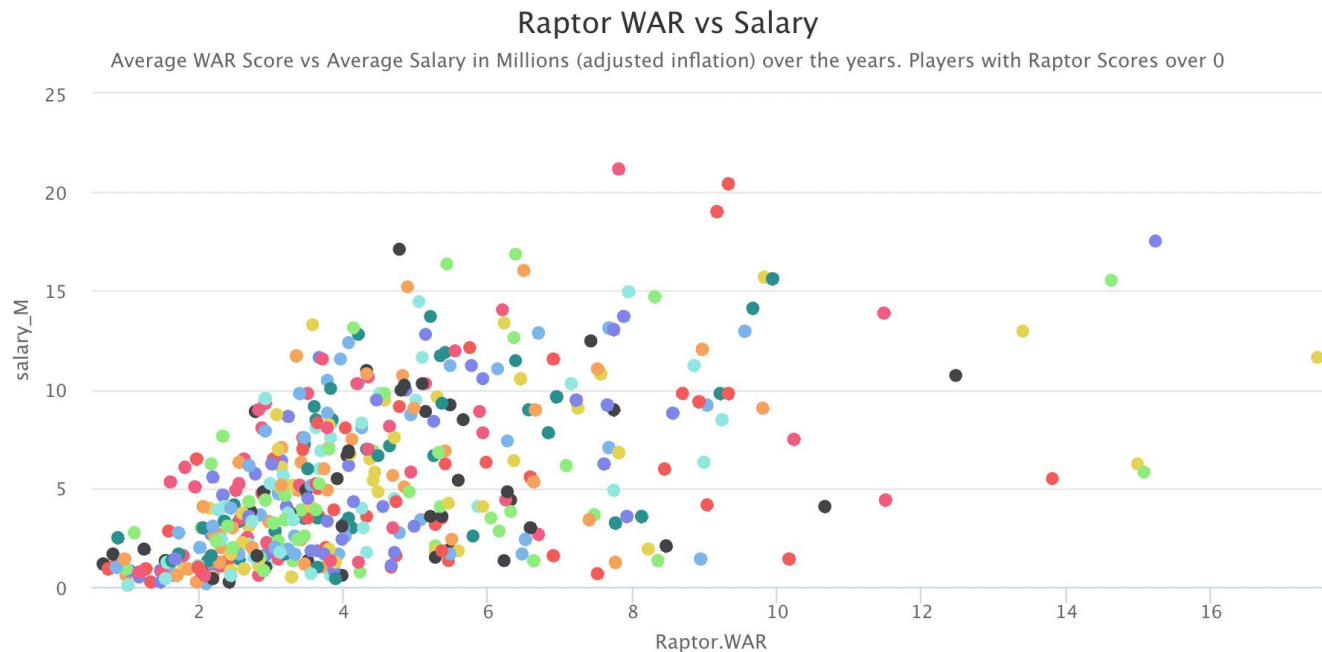


RAPTOR WAR

$WAR = (RAPTOR + Replacement\ level) \times Minutes\ Played \times ((League\ Pace + Individual\ Pace\ Impact) / League\ Pace) \times WARmultiplier$

- Raptor Wins Above Replacement
 - Wins created above average replacement player in the NBA
 - RAPTOR taking into account minutes played

RAPTOR WAR vs Salary visual



Overview of Data Sets

- Player statistics
 - Compiled by FiveThirtyEight
 - All NBA players and their performance from 1970 to 2020
- Salaries
 - Sourced from Basketball reference
 - Salaries of NBA players from 1985 - 2018
- Variables for analysis
 - **Demographic data:** Name, Age, Position
 - **Analysis data:** Raptor, Raptor WAR, Salary

Data Cleaning

1. Rename and filter out unnecessary columns
2. Join datasets

```
players <- players_historic %>%  
  inner_join(salaries, by = "player_id", "year_id") %>%  
  filter(year_id == season_end,) %>%  
  select(!season_end)
```

3. Filter out players who have multiple sets of statistics per season

```
players %<>% group_by(player_id, year_id) %>%  
  filter(n() == 1) %>%  
  arrange(year_id) %>% arrange(player_id)
```


Feature Engineering

1. Adjust salary based on inflation

```
players$salary_adjusted <- c(adjust_for_inflation(players$salary, players$year_id, "US", to_date = max(players$year_id)))
players %<>% mutate(salary_M = round(salary_adjusted / 1000000, 3),
  Raptor_per_million = Raptor_T/salary_M,
  WAR_per_million = Raptor_WAR/salary_M)
```

2. Create references to next year's statistics

```
players_clean <- players %>% group_by(player_id) %>%
  mutate(Raptor_F = lead(Raptor_T, order_by = year_id),
    Raptor_WAR_F = lead(Raptor_WAR, order_by = year_id),
    salary_M_F = lead(salary_M, order_by = year_id),
    WAR_per_million_F = lead(WAR_per_million, order_by = year_id),
    Raptor_per_million_F = lead(Raptor_per_million, order_by = year_id))
  filter(!is.na(Raptor_F))
```



Predicting Raptor Score through Linear Regression

Linear Regression Model Stats

Summary Statistics of LM

```
mod1 <- lm(Raptor_F ~ age + age_squared + pos + MPG,  
           data = players_clean_train)
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-2.021786	1.336345	-1.513	0.1303
age	-0.226560	0.098676	-2.296	0.0217 *
age_squared	0.003074	0.001781	1.726	0.0845 .
posPF	0.423776	0.104972	4.037	0.0000545277166942 ***
posPG	1.532475	0.108839	14.080	< 0.0000000000000002 ***
posSF	0.834868	0.109502	7.624	0.0000000000000267 ***
posSG	1.062533	0.108513	9.792	< 0.0000000000000002 ***
MPG	0.173320	0.003610	48.011	< 0.0000000000000002 ***

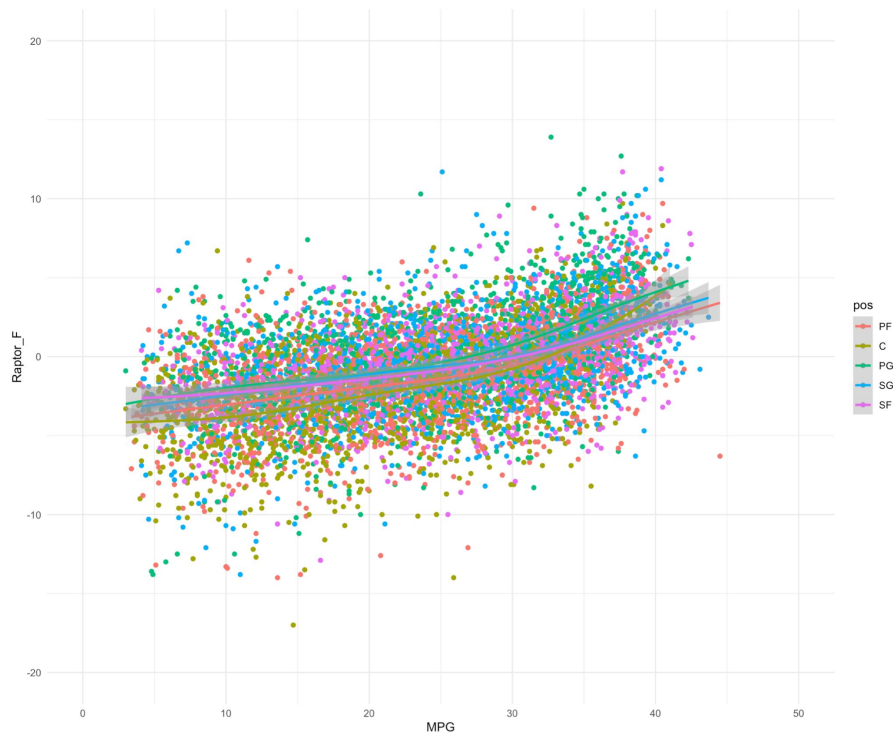
Residual standard error: 2.78 on 7287 degrees of freedom

Multiple R-squared: 0.2753, Adjusted R-squared: 0.2746

F-statistic: 395.5 on 7 and 7287 DF, p-value: < 0.0000000000000002

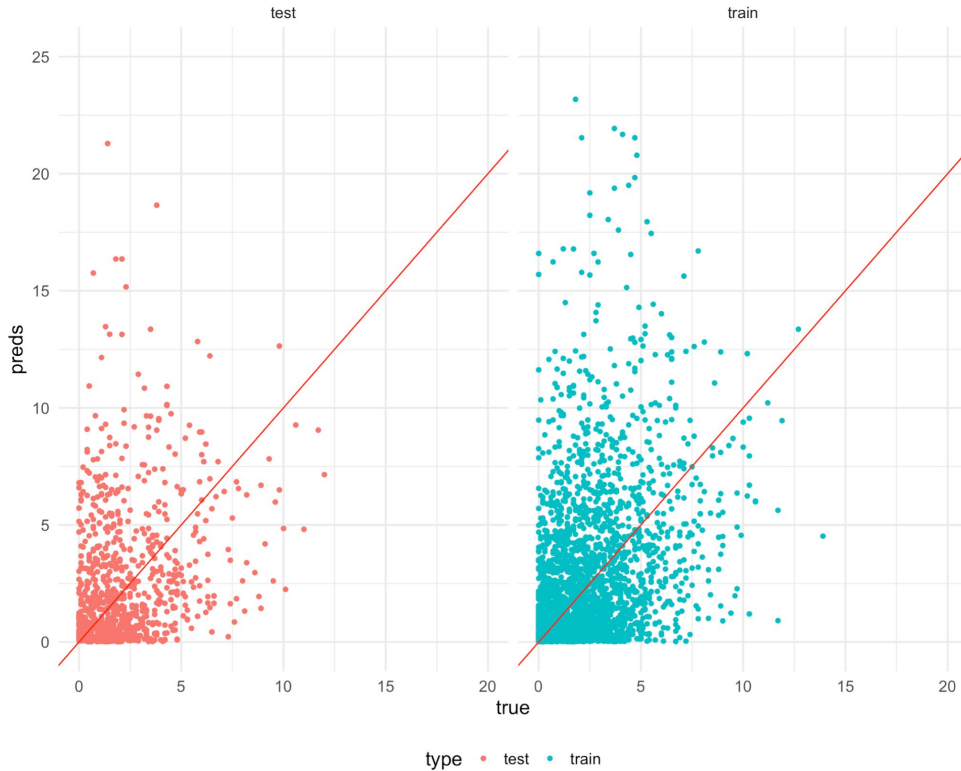
- Baseline/Foundation of Predictions
- Raptor_F (raptor_score) increases by 0.1733 as MPG increases by 1 unit
- Age_squared: supposed to represent that as age increases, raptor_score would increase
 - To a certain extent
- R-squared: 0.2753
- Point Guards tend to have the highest Raptor_Score

Linear Regression Model of Data



- As MPG increases, Raptor Score increases
- Point Guards have the best raptor score
- Power Forwards/Centers have the worst

Predicted True Plots



- Model is not very accurate
 - Data isn't aligned along the diagonal line
- Model predicts significantly higher raptors
- Model fails to take into account
 - Max raptor score
 - Ratio of average players and all star level players is unproportionate
 - More players above 5 raptor scores

Linear Regression Model Stats

- Testing error < Training error
 - Model is Underfit

```
> get_rmse(players_clean_train$Raptor_F, preds_train)
[1] 3.471226
> get_rmse(players_clean_test$Raptor_F, preds_test)
[1] 3.398541
```

- Lower error in Train vs Test
 - Add more variables
 - To increase model complexity

```
> get_mae(results_train$true, results_train$preds)
[1] 3.000744
> get_mae(results_test$true, results_test$preds)
[1] 3.040833
```

- RMSE: measure of how far the points are from the line of best fit
 - How concentrated the data is
 - Around 3 residuals away from the line of best fit
- Underfit: model is unable to capture the relationship between my variables
- MAE: Expect a 3.0 error from the forecast on average
- RMSE is always larger than MAE
 - Difference is the variance in the individual errors in the sample



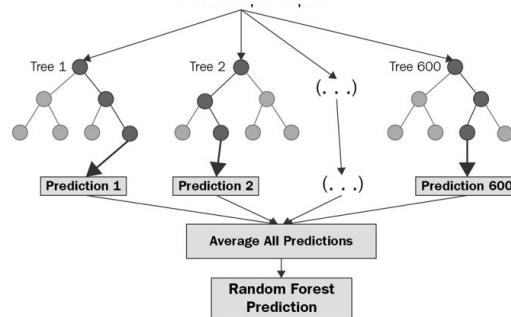
Predicting Salary through a Random Forest

Random Forest Model

(Predict a players salary for the following season)

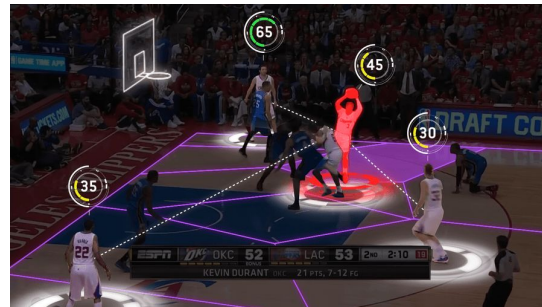
Why Random Forest?

- At each split, model considers only a small subset of features rather than all of the features of the model.
- More variables does not lead to overfitting
- It is flexible to both classification and regression problems.



Predictors

- current salary, minutes per game (MPG), raptor score, raptor WAR, age, position



Random Forest Model

- Chose $mtry = 3$

```
Call:
randomForest(formula = salary_M_F ~ MPG + Raptor_T + Raptor_WAR,
              ntree = 500, mtry = 3, importance = TRUE)
Type of random forest: regression
Number of trees: 500
No. of variables tried at each split: 3

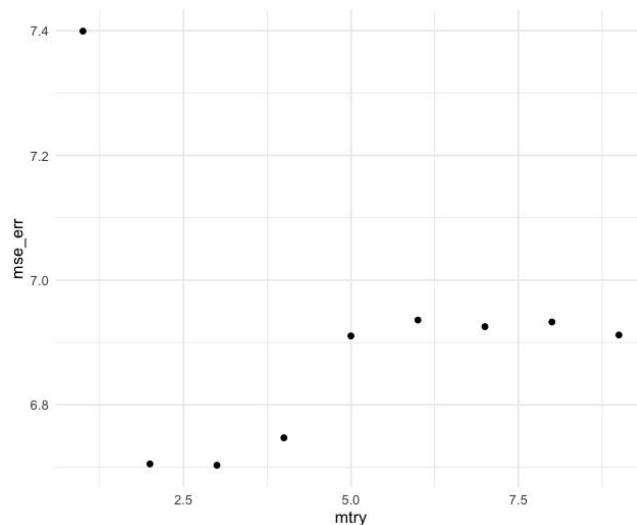
Mean of squared residuals: 6.733305
% Var explained: 75.31
> |
```

Tuning Random Forests To Determine Optimal Parameters (mtry)

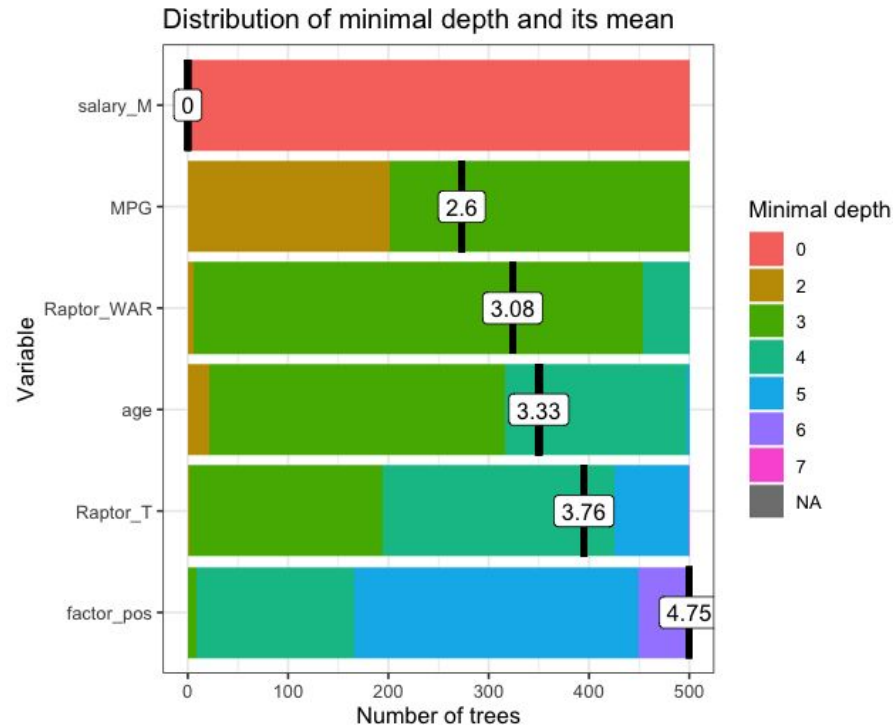
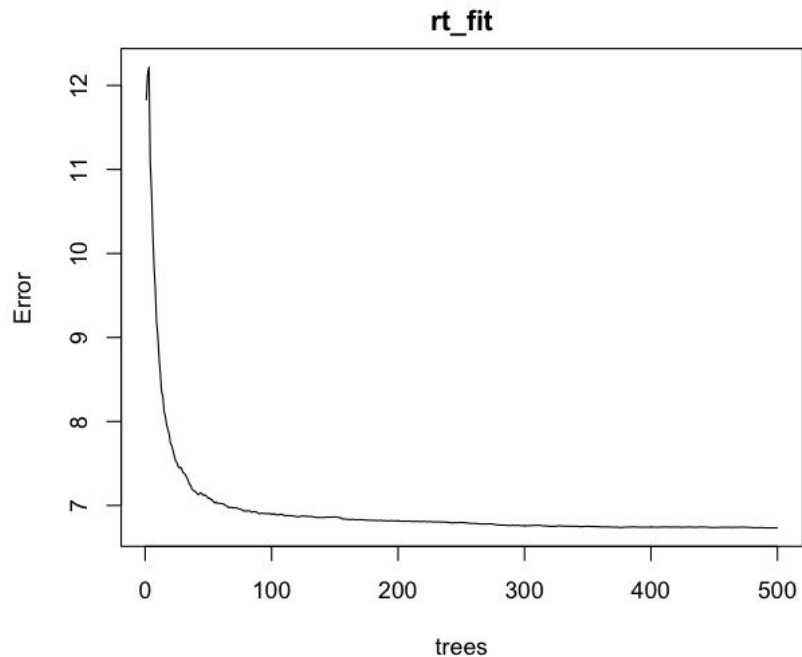
```
#THIS TAKES FOREVER
for(mtry in 1:9){
  rt_fit <- randomForest(salary_M_F ~ MPG + Raptor_T + Raptor_WAR + age,
                        data = players_clean,
                        type = regression,
                        ntree = 500,
                        mtry = mtry,
                        importance = TRUE)

  mse_err[mtry] <- rt_fit$mse[500]

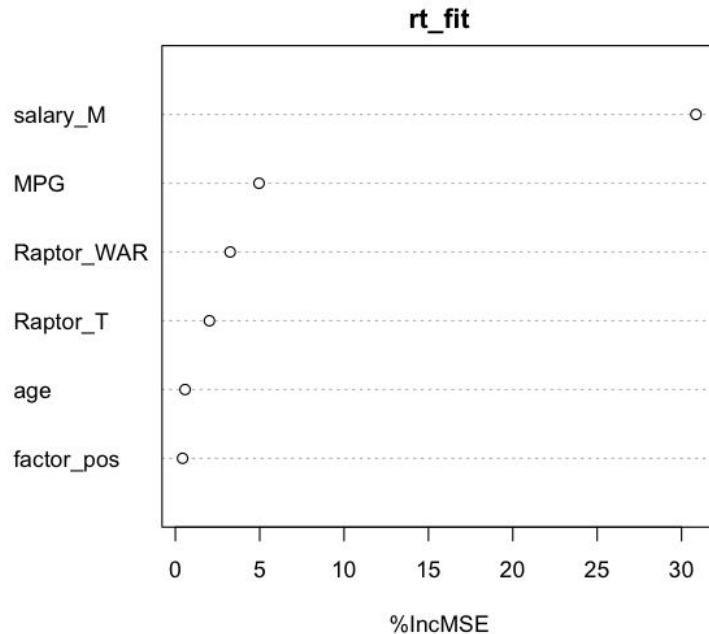
  cat(mtry, " ")
}
```



Random Forest Model



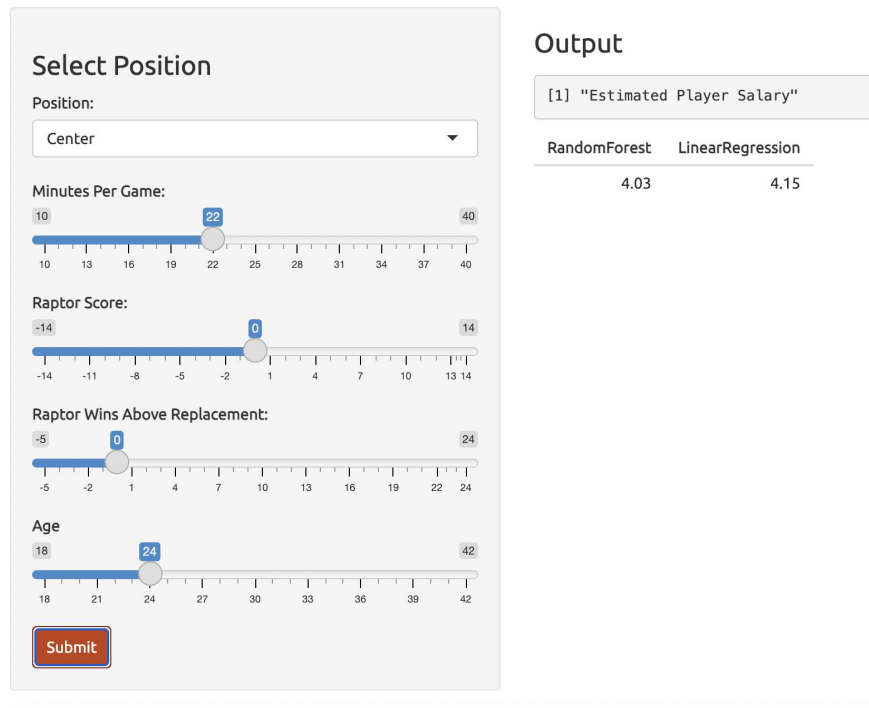
Random Forest Model



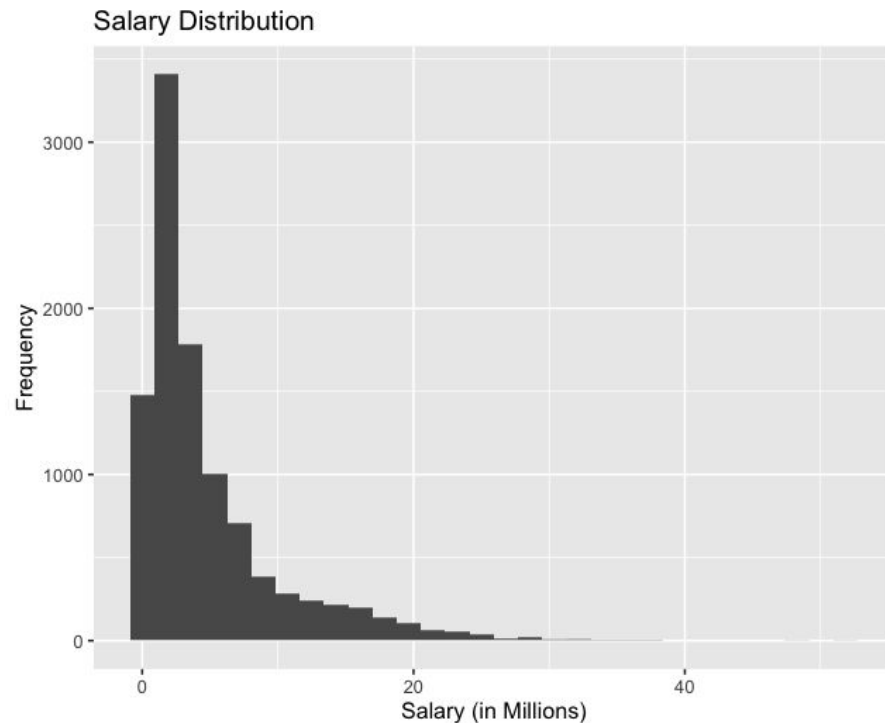
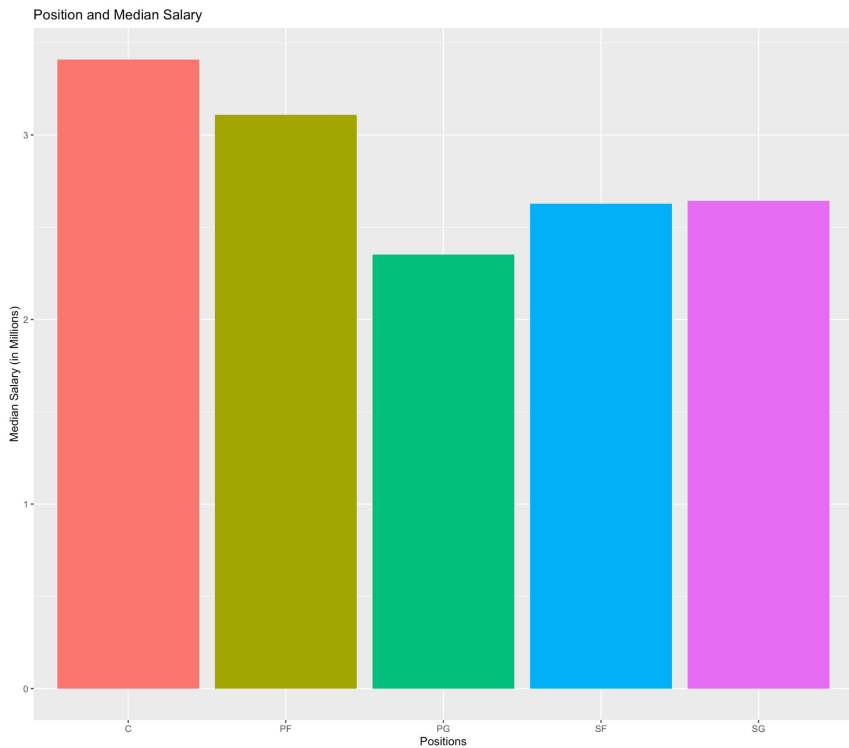
```
> importance(rt_fit, type=1, scale = FALSE)
      %IncMSE
MPG      4.9591749
Raptor_T  2.0183297
Raptor_WAR 3.2491891
age       0.5767285
factor_pos 0.4288049
salary_M  30.8555439
> |
```

Random Forest Model and Linear Regression Model - R Shiny

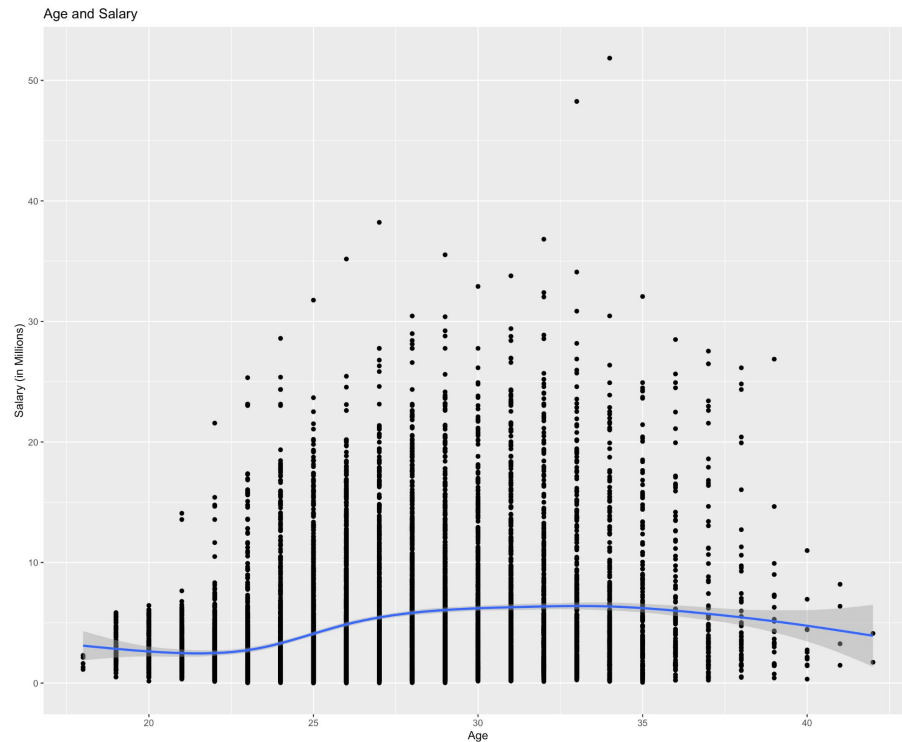
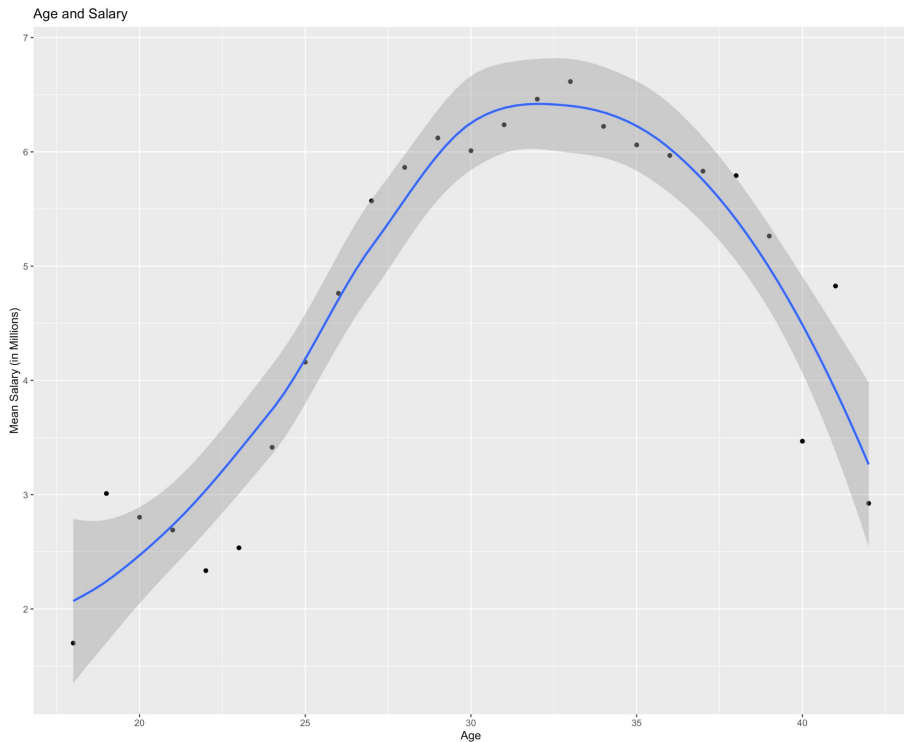
Estimating Salary (in Millions)



Random Forest Model and Linear Regression Model - R Shiny



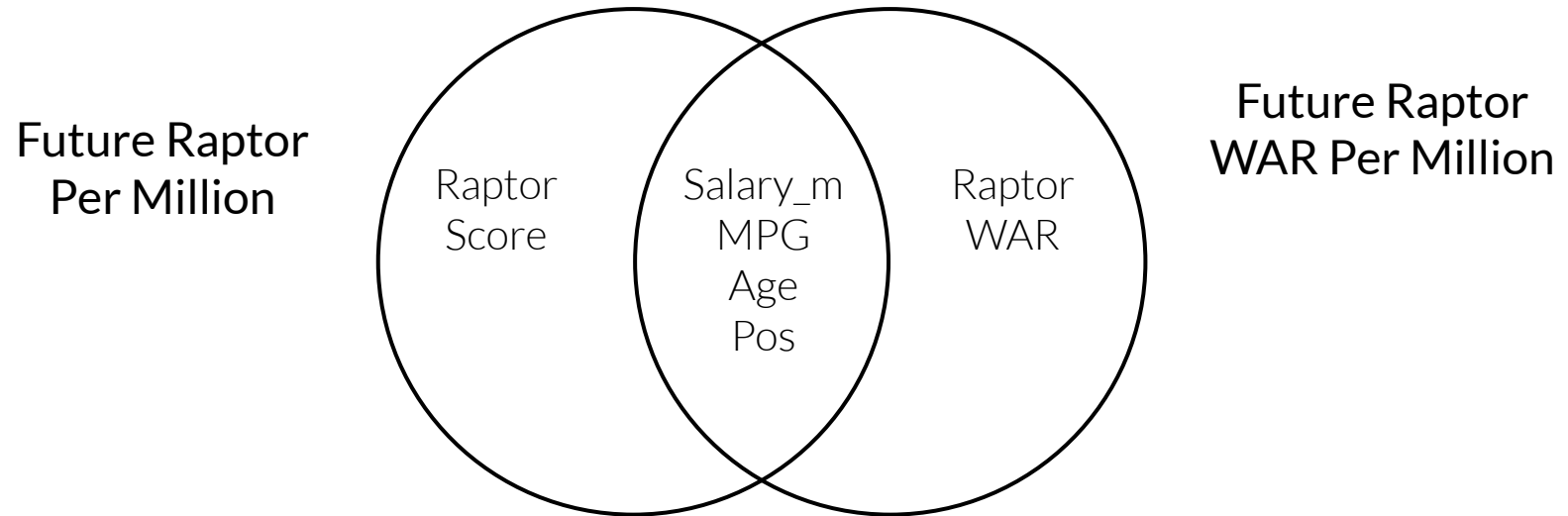
Random Forest Model and Linear Regression Model - R Shiny





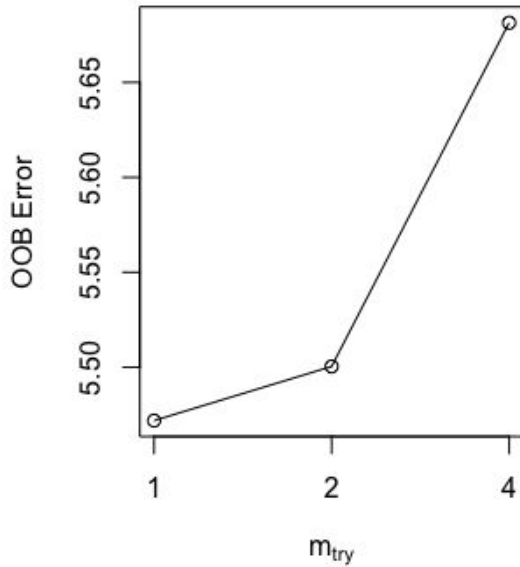
Value to Performance through Random Forest

Independent and Dependent Variables

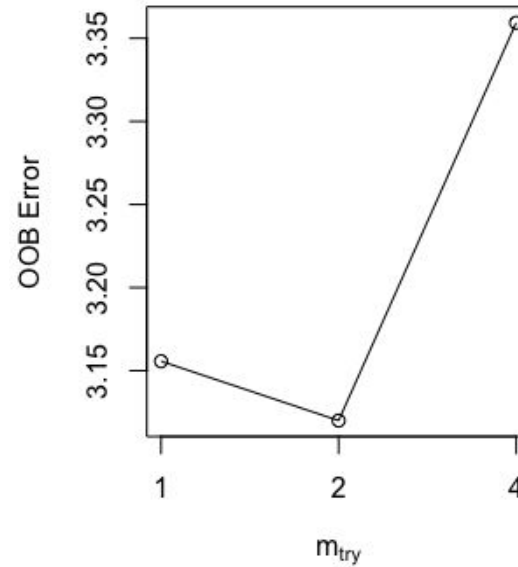


Tuning Mtry

Raptor



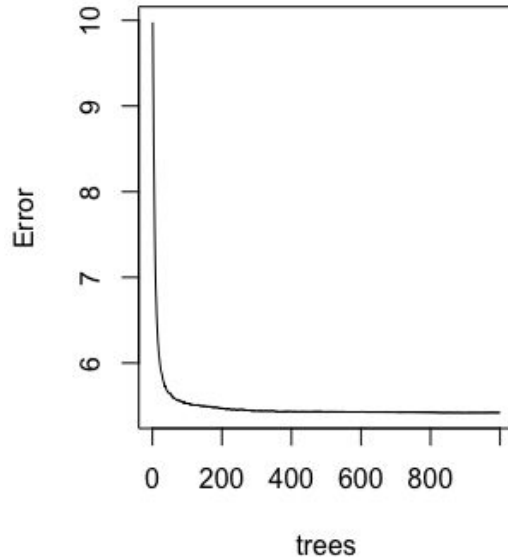
Raptor WAR



Tuning Number of Trees

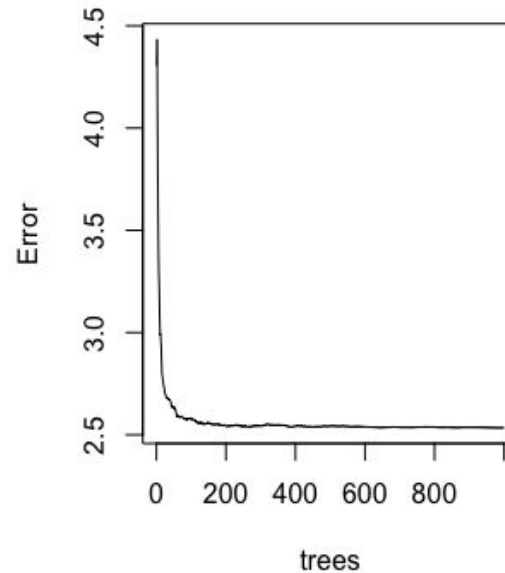
Raptor

rf_fit_Raptor



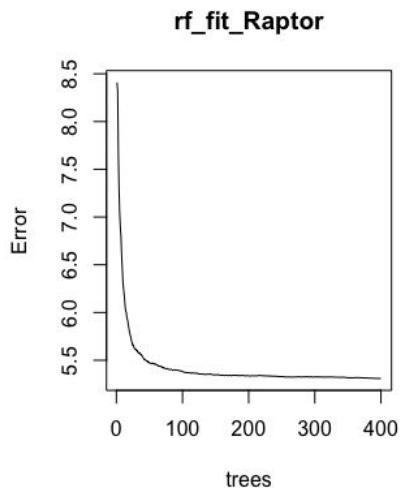
Raptor WAR

rf_fit_WAR



Model Results

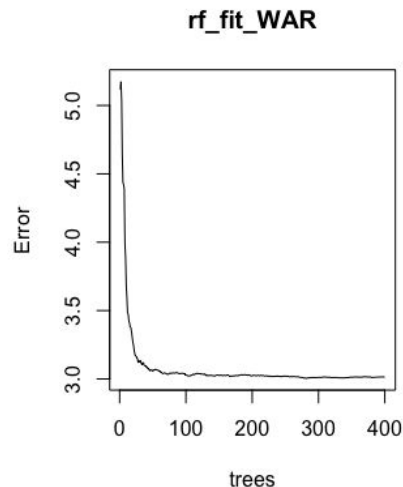
Raptor



```
Type of random forest: regression
Number of trees: 400
No. of variables tried at each split: 2

Mean of squared residuals: 5.310003
% Var explained: 49.54
```

Raptor WAR

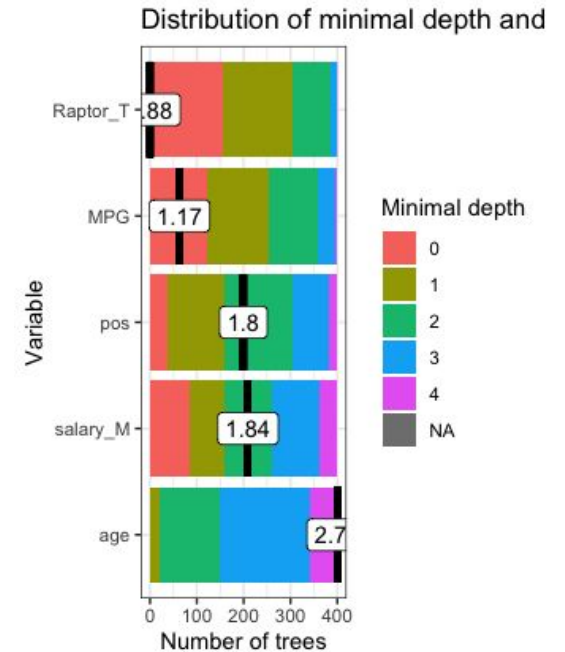
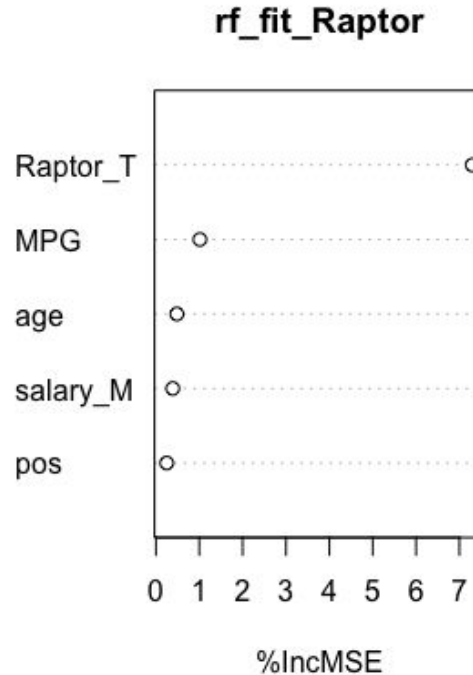


```
Type of random forest: regression
Number of trees: 400
No. of variables tried at each split: 2

Mean of squared residuals: 3.012913
% Var explained: 29.95
```

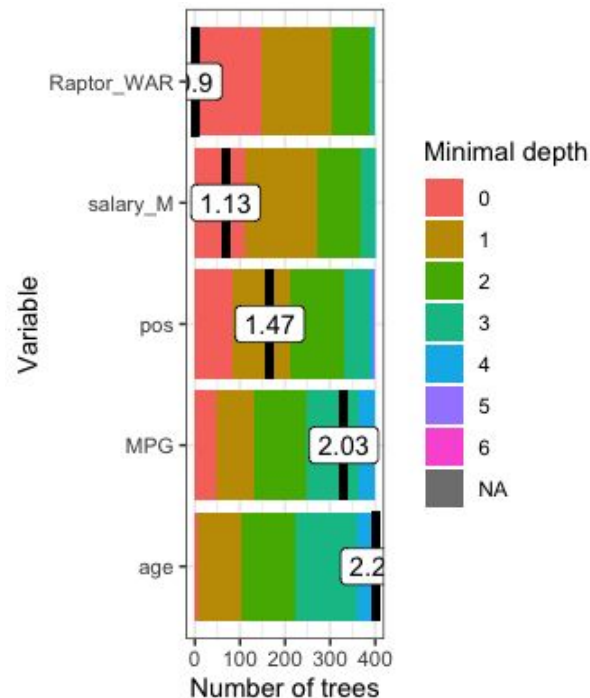
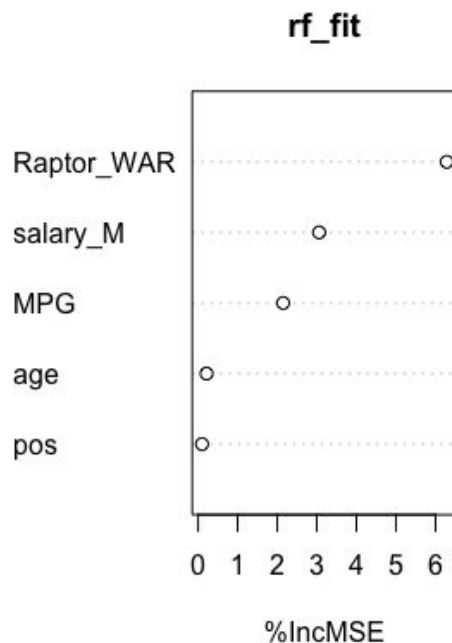
Variable Importance/ Depth: Raptor


	%IncMSE
age	0.4833279
pos	0.2490412
MPG	1.0133035
salary_M	0.3873888
Raptor_T	7.2963135



Variable Importance/ Depth: Raptor WAR

	%IncMSE
age	0.21442600
pos	0.09885498
MPG	2.14917736
salary_M	3.05856970
Raptor_WAR	6.28462878



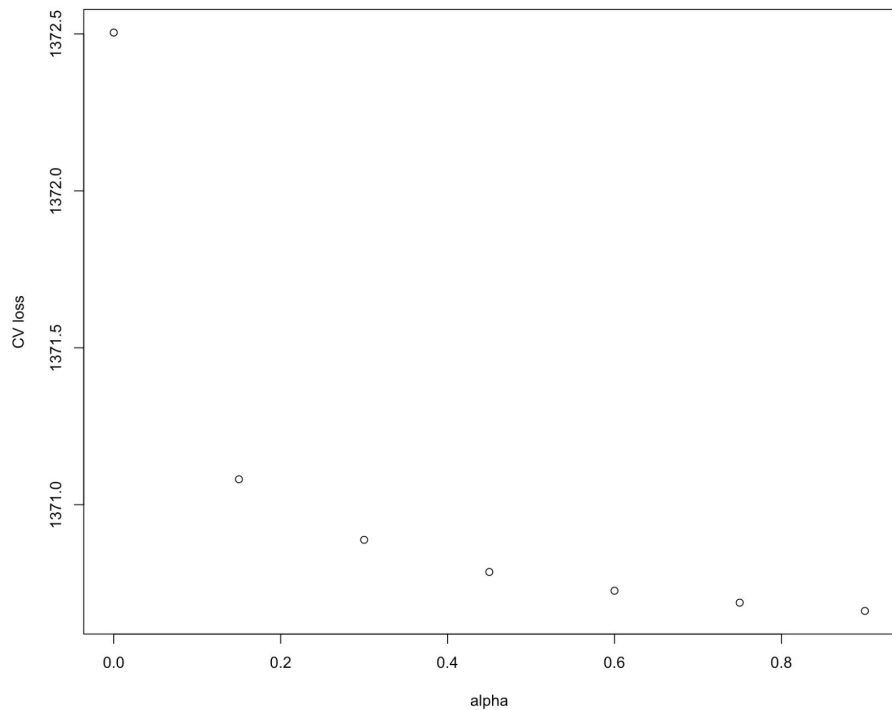


Value to Performance through Elastic Net Regression

Elastic Net Regression Model

- Why?
 - More optimal model than Lasso or Ridge by themselves
 - Allow me to have the Lasso ability to reduce the features and reduce the overfitting of our model and the Ridge ability to eliminate non important variables or variables that are highly correlated.
 - Wanted to see how good of a fit our models were at given levels of alpha
 - Wanted to see how the cross validated Mean Squared Error varies as we change alpha

Min Loss Plot (Raptor_per_mil_F)



- What does this show?

- This min loss plot of the testing set shows us that the optimal alpha value for this particular dataset is at 0.9
 - Shows us that a lasso model is a better fit
 - Alpha closer to 1
- Mean squared error is kind of high on the graph
 - This can be attributed to breakout seasons players have had
 - This is followed by huge drop offs after the fact
 - Furthermore, some players had seasons cut short which affect their score in the future
- **How did we solve this problem**

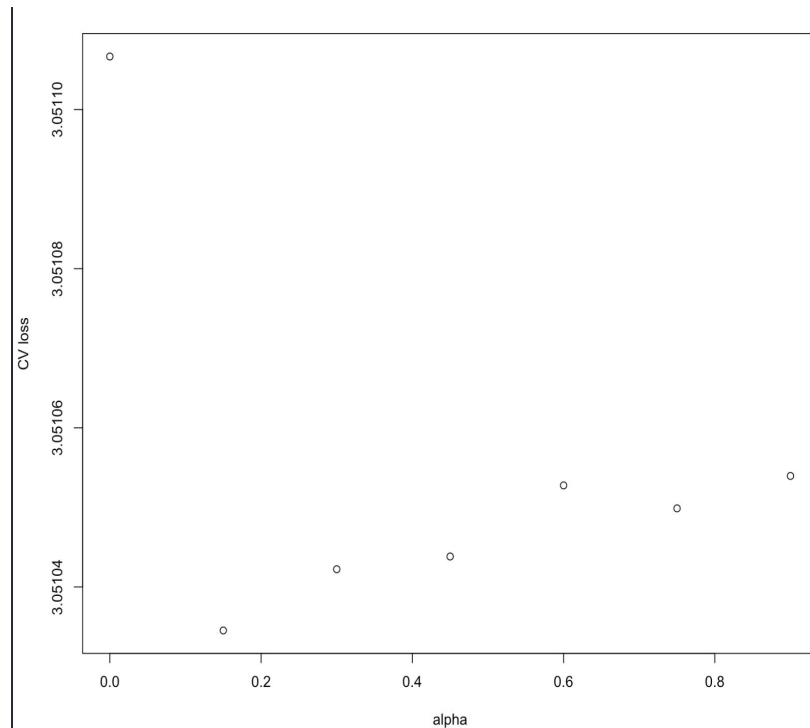
Min Loss Plot (WAR_per_million_F)

- How has it changed?
 - This min loss plot of the testing set shows us that the optimal alpha value for this particular dataset is at 0.15
 - Shows us that a ridge model is a better fit
 - Alpha closer to 0
- We adjusted this model by using a statistic called WAR (wins above replacement) which takes into account the number of games a player has played

```
[r]
best_alpha <- get_alpha(enet_mod)
print(best_alpha)
get_model_params(enet_mod)

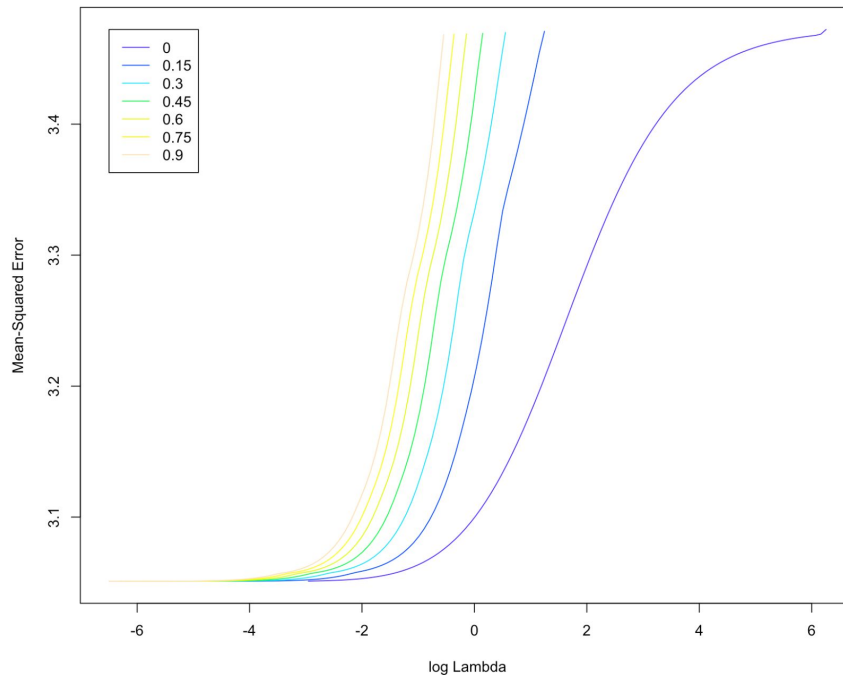
best_alpha <- get_alpha(enet_mod1)
print(best_alpha)
get_model_params(enet_mod1)
***
```

alpha	lambdaMin	lambdaSE	error
0.15	0.009899412	1.370956	3.051035



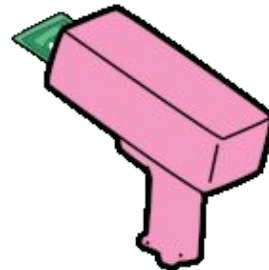
Elastic Net Regression Model

- What does this show?
 - This graph shows us how changes in lambda affect the mean squared error
 - More specifically shows how the number of non-zero variables present in the model affect the mean squared error for the models that are looking for WAR per million squared
- This model is also adjusted for the players that have not played a large majority of games over the course of a season



Conclusion

- Player performance is difficult to predict
 - Using models increases accuracy
 - Need to explore more models and variables
- Salary is a more predictable variable
 - Helps teams understand what they need to pay to get a player



Further Research

- Improve this model
 - Take into account more demographic data
 - Normalize for NBA capital
 - Test model to build specific team
- New Models
 - What is the makeup of successful NBA teams (high raptor players vs low raptor players)?

