Data Science Midterm Project - FIFA 18_19

Team 4:
Dhairya Jaiswal,
Pragna Sonpal,
Shruti Walawalkar,
Xinyue Cai

Project Objective

- Using 2018 variables and 2019 values of FIFA data, to build a model so that in the future, the model can be used to predict a newly coming player’s value with his abilities, ratings, and etc.

Dataset Overview

- 2018 dataset: 17981 observations, 75 variables

  2019 dataset: 18201 observations, 89 variables

- Variables include players' basic profiles, like name, club, nationality, age, etc

  Ability, like speed, finishing, vision, etc

  Rating, like overall rating and ratings for positions

  Value, wages

- Missing values on columns like CAM, CD, CDM, CF, CM (for GK)
- Missing values for some players' ability features

Exploratory Data Analysis

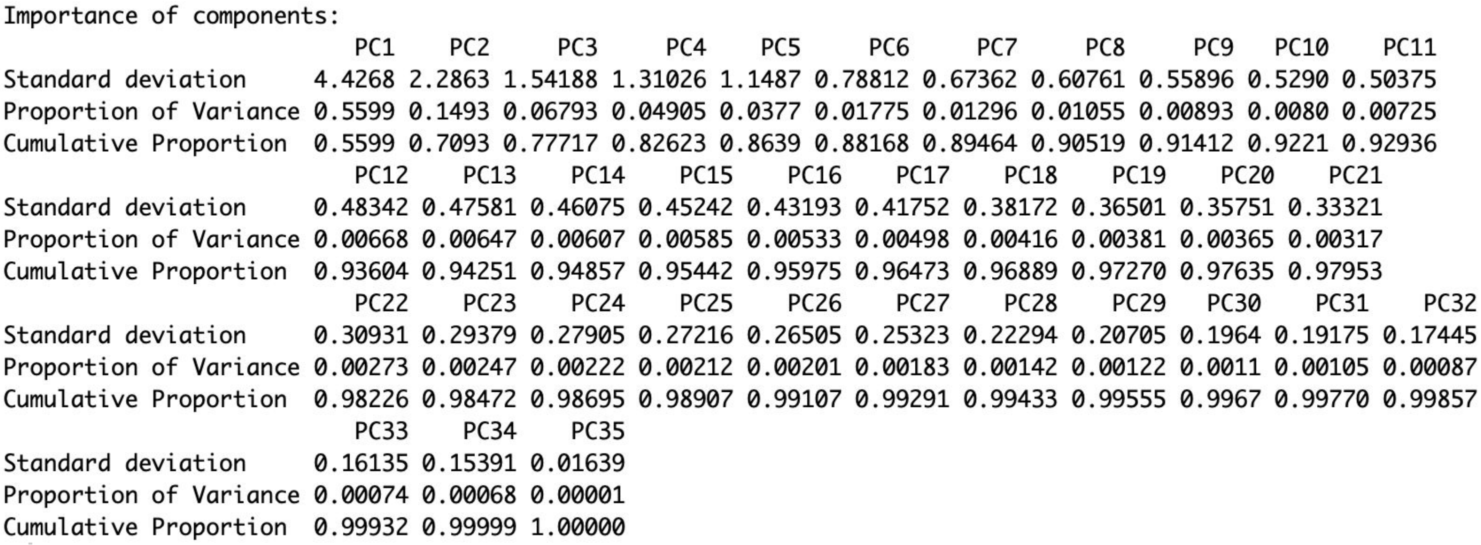
- Cross Referenced between 2018 dataset and 2019 dataset based on ID
  - Found 12487 players that participated in 2018 as well as in 2019
- Kept all the variables from 2018 and value from 2019
  - Independent variables: Variables from 2018
  - Target variable: Value from 2019
- Deleted variables: CAM, CD, CDM, CF, CM, ID, LAM, LB, LCM, LDM, LF, LM, LS, LW, LWB, RAM, RB, RCB, RCM, RDM, RF, RM, RW, RWB, ST, Overall
  - Missing values (for GK)
  - Derived variables from other variables like speed, finishing, vision, etc
    - Highly positively correlated and even multicollinearity
- Converted all Values into 000s and fill in with 0s for zero values
- Filled in 0s for missing values in ability features

Exploratory Data Analysis (Cont.)

- Categorized Variables:
  - Club: UEFA vs noUEFA: (qualified for Union of European Football Associations champions league & in top 5 league)
    - UEFA: FC Barcelona, Real Madrid CF, Paris Saint-Germain, Manchester United, Manchester City, Atlético Madrid, FC Bayern Munich, Juventus, Tottenham Hotspur, Liverpool, Napoli, Borussia Dortmund, Olympique Lyonnais, Roma, Inter, FC Schalke 04, AS Monaco, Valencia CF
  - Preferred Position: GK vs DEF vs MID vs FWD
    - GK: GK
    - DEF: LWB, CB, RB, RWB
    - MID: LW, LM, CDM, CM, CAM, CM, CW
    - FWD: CF, ST
- 1-C & Max-Min Normalization on variables
- Final Dataset contains 12487 observations and 42 variables

PCA

- Performed PCA with the cleaned dataset on numeric variables (ability features)

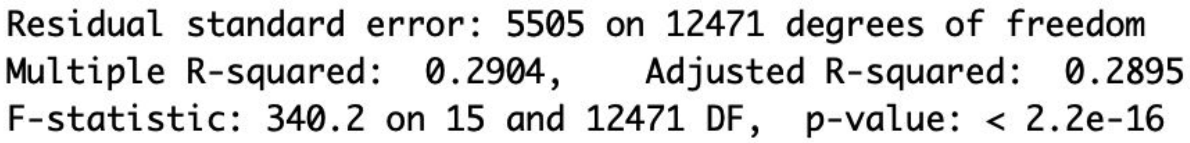


```
Importance of components:
                          PC1    PC2     PC3     PC4    PC5     PC6     PC7     PC8     PC9   PC10    PC11
Standard deviation     4.4268 2.2863 1.54188 1.31026 1.1487 0.78812 0.67362 0.60761 0.55896 0.5290 0.50375
Proportion of Variance 0.5599 0.1493 0.06793 0.04905 0.0377 0.01775 0.01296 0.01055 0.00893 0.0080 0.00725
Cumulative Proportion  0.5599 0.7093 0.77717 0.82623 0.8639 0.88168 0.89464 0.90519 0.91412 0.9221 0.92936
                          PC12    PC13    PC14    PC15    PC16    PC17    PC18    PC19    PC20    PC21
Standard deviation     0.48342 0.47581 0.46075 0.45242 0.43193 0.41752 0.38172 0.36501 0.35751 0.33321
Proportion of Variance 0.00668 0.00647 0.00607 0.00585 0.00533 0.00498 0.00416 0.00381 0.00365 0.00317
Cumulative Proportion  0.93604 0.94251 0.94857 0.95442 0.95975 0.96473 0.96889 0.97270 0.97635 0.97953
                          PC22    PC23    PC24    PC25    PC26    PC27    PC28    PC29   PC30    PC31    PC32
Standard deviation     0.30931 0.29379 0.27905 0.27216 0.26505 0.25323 0.22294 0.20705 0.1964 0.19175 0.17445
Proportion of Variance 0.00273 0.00247 0.00222 0.00212 0.00201 0.00183 0.00142 0.00122 0.0011 0.00105 0.00087
Cumulative Proportion  0.98226 0.98472 0.98695 0.98907 0.99107 0.99291 0.99433 0.99555 0.9967 0.99770 0.99857
                          PC33    PC34    PC35
Standard deviation     0.16135 0.15391 0.01639
Proportion of Variance 0.00074 0.00068 0.00001
Cumulative Proportion  0.99932 0.99999 1.00000
```

- Discovered that with PCA, 15 variables can explain more than 95% of the variations

PCA (Cont.)

- Performed linear regression on PCA-filtered 15 variables



```
Residual standard error: 5505 on 12471 degrees of freedom
Multiple R-squared:  0.2904,	Adjusted R-squared:  0.2895
F-statistic: 340.2 on 15 and 12471 DF,  p-value: < 2.2e-16
```

- Adjusted R-sq = 28.95%, Residual standard error = 5505
- Not as good as our linear regression model

Linear Regression

We began with splitting the cleaned data into training and test. The train data was 80% of the dataset while test was 20%

To determine statistical significance of each variable we took into consideration p-values for eliminating variables.

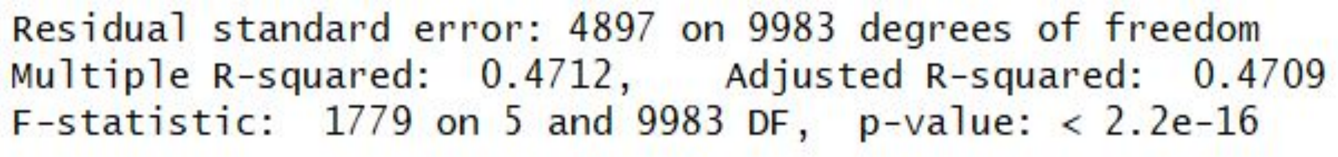
Our final model had only 5 variables that were the most significant wrt our output variable Value

Our equation thus becomes

$$\text{Value} = -10240.1 - 10426.7 * \text{Age} + 11401.3 * \text{Composure} + 18337.4 * \text{Reactions} + 13611.7 * \text{Club} + 2940.6 * \text{GK}$$

Linear Regression

The final chosen model proves to be the best because it's residual spread obtained was smaller

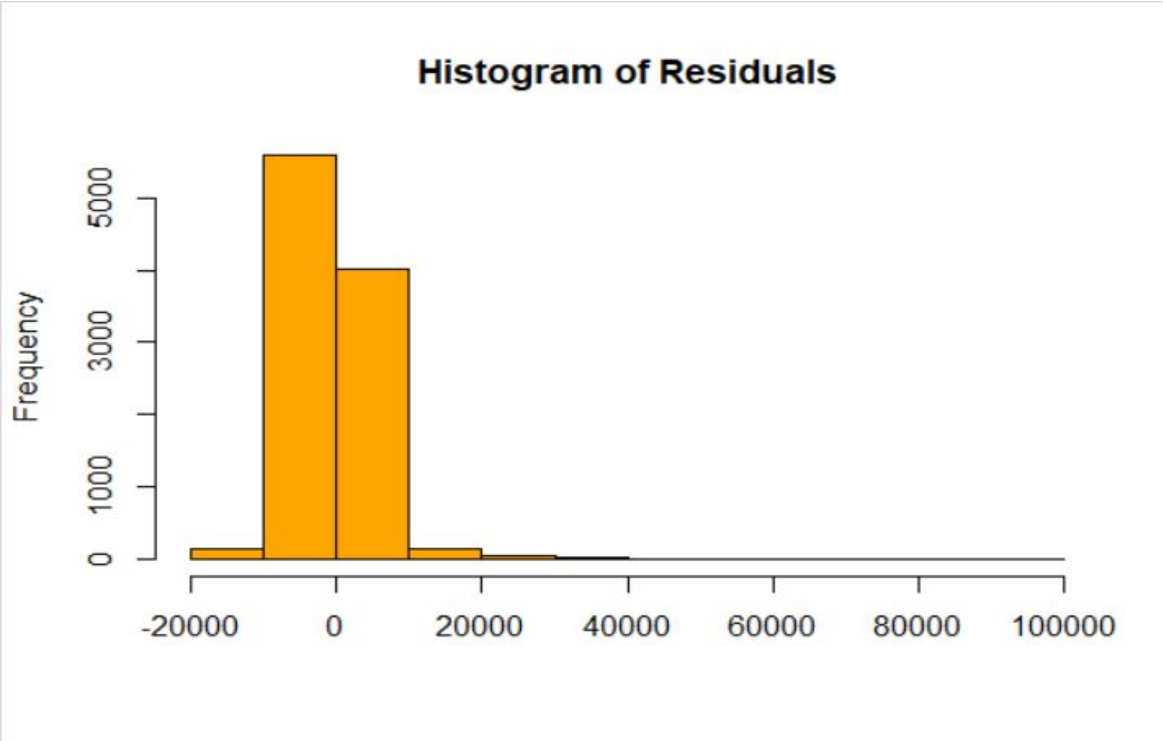


```
Residual standard error: 4897 on 9983 degrees of freedom
Multiple R-squared:  0.4712,    Adjusted R-squared:  0.4709
F-statistic:  1779 on 5 and 9983 DF,  p-value: < 2.2e-16
```

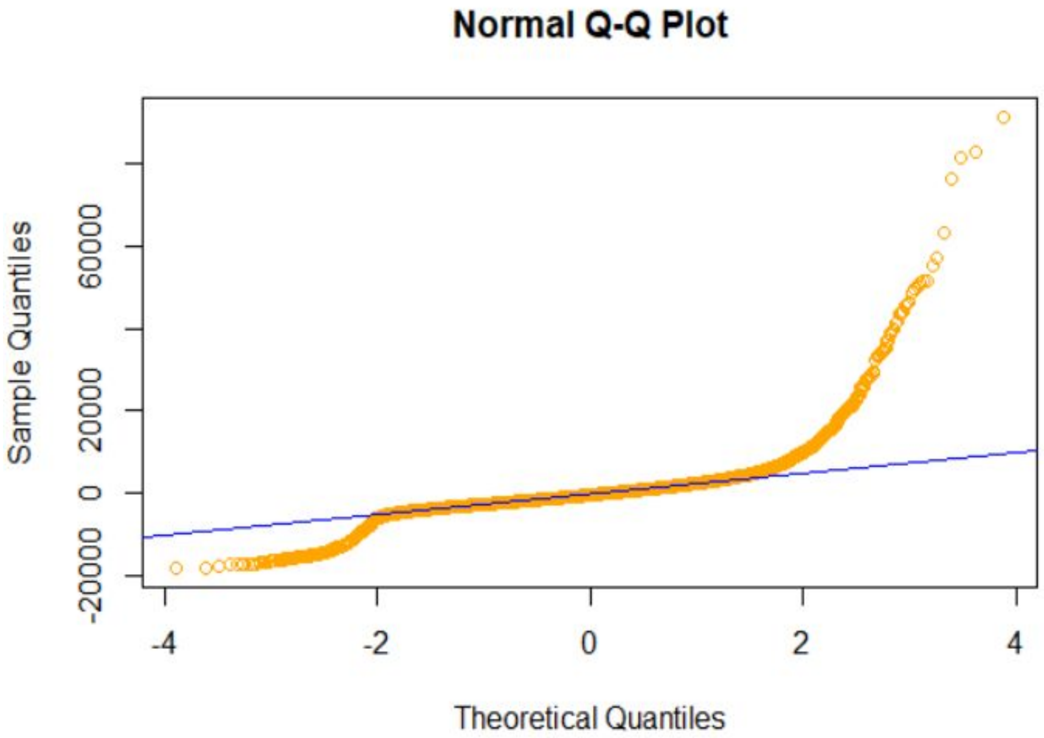
The predictors jointly explain 47.12 of observed variance on "Value"
(Adjusted- R^2 = 0.4709)
F (9983, 5) = 1779 (a large F indicates strong relationship)
P < 0.05 we reject Null Hypothesis

RMSE(trainData) < RMSE(testData)

Linear Regression



Linear Regression



Decision Tree

- We have done splitting of our preprocessed data in train and test with the ratio of 80% and 20% respectively.
- Using rpart, we are fitting decision tree which takes top down approach for recursive partition.
- Based on minimum sum of square error dividing data in subgroups.
- Rpart is automatically applying different range for cost of complexity to achieve best result.
- Using complexity variable cp=0.001 we are getting least rmse for prediction.

```
fit <- rpart(y~., data = train, method="anova",control = rpart.control(minsplit = 4,cp=0.001))
```

Decision Tree

Plotting cost of complexity

with respect to relative error.

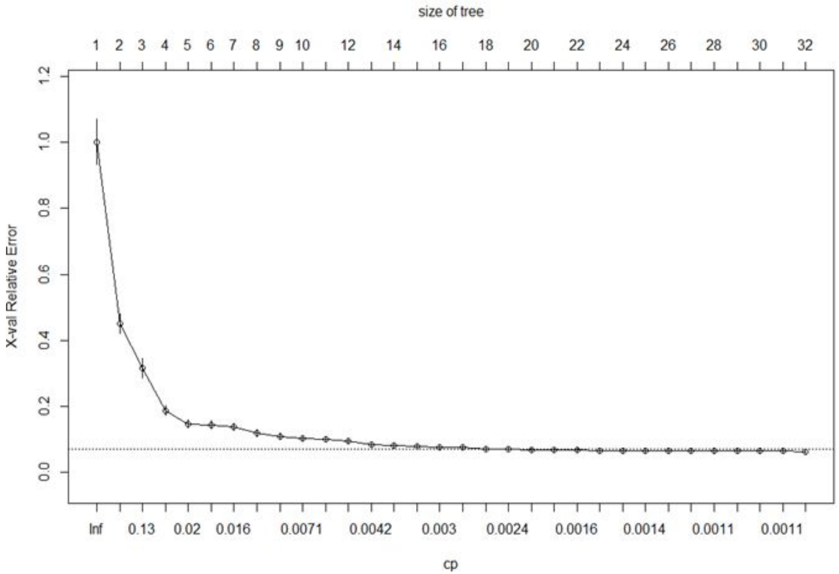
From observation we can see

cp= 0.001 gives best result

And smallest margin of error.

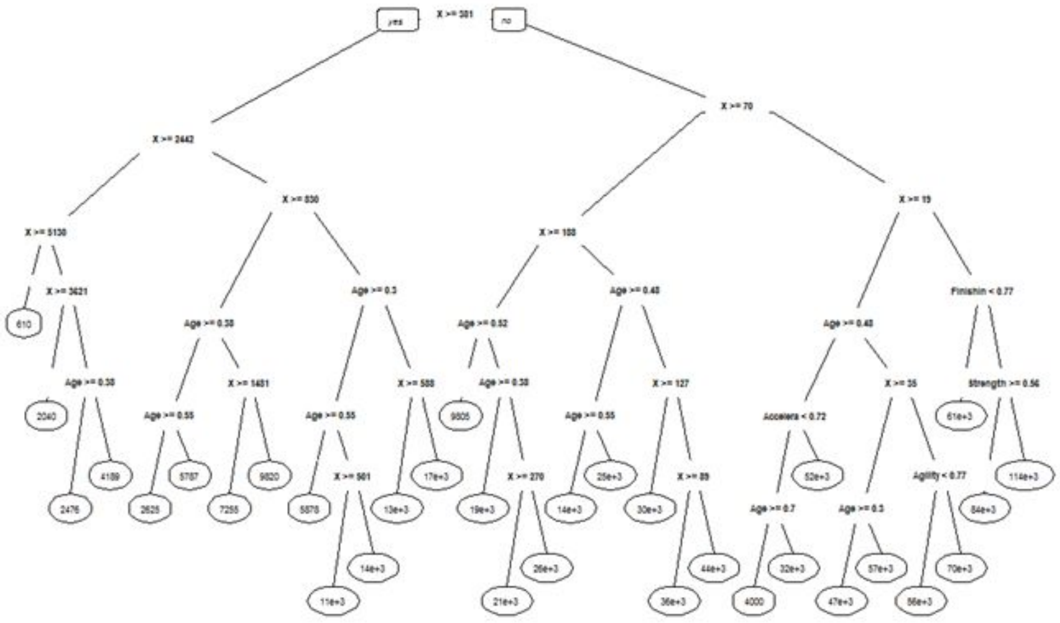
rmse=1160.38

Rmse is 3.6%



Decision Tree

Decision tree with regression



Random Forest

A Random Forest consists of a collection or ensemble of simple tree predictors, each capable of producing a response when presented with a set of predictor values.

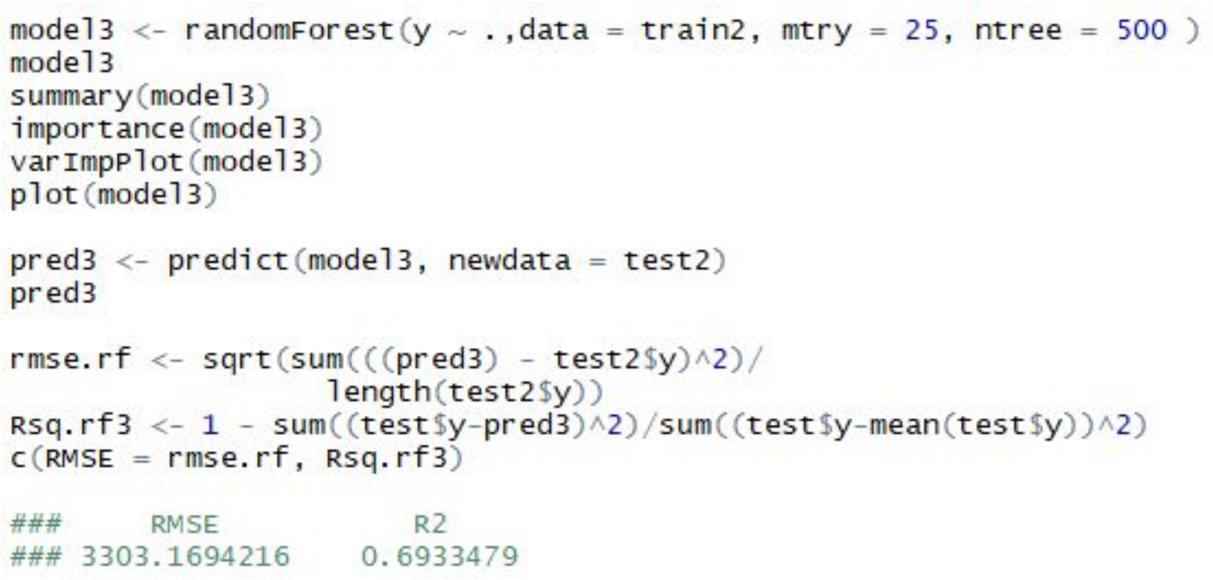
Implementation Steps :

1. First, we divided our cleaned data into training and testing sets with training set accounting for 80% of the data and test set 20% of the data.

2. Then tried fitting our model with all the variables taking the default Random Forest Parameters and found a limitation in our model for the categorical columns – “Nationality” and “Club”, the limitation is that the random forest model cannot process categorical variables with more than 53 levels and our categorical columns have over 100 levels, hence we remove these columns for our model to be fit.

So, without tuning any parameters and taking the default mtry(Total variables/3) which is 13 in our case and ntree values we got an accuracy of 71.4% and rmse of 1519 which was the lowest for our model, but we have a high OOB error rate for this model and we should not use it.

Random Forest



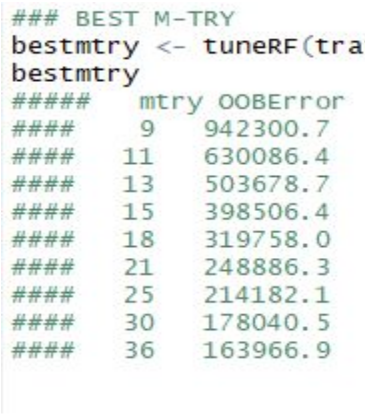
```
model3 <- randomForest(y ~ .,data = train2, mtry = 25, ntree = 500 )
model3
summary(model3)
importance(model3)
varImpPlot(model3)
plot(model3)

pred3 <- predict(model3, newdata = test2)
pred3

rmse.rf <- sqrt(sum(((pred3) - test2$y)^2)/
                  length(test2$y))
Rsq.rf3 <- 1 - sum((test$y-pred3)^2)/sum((test$y-mean(test$y))^2)
c(RMSE = rmse.rf, Rsq.rf3)

###      RMSE           R2
### 3303.1694216    0.6933479
```

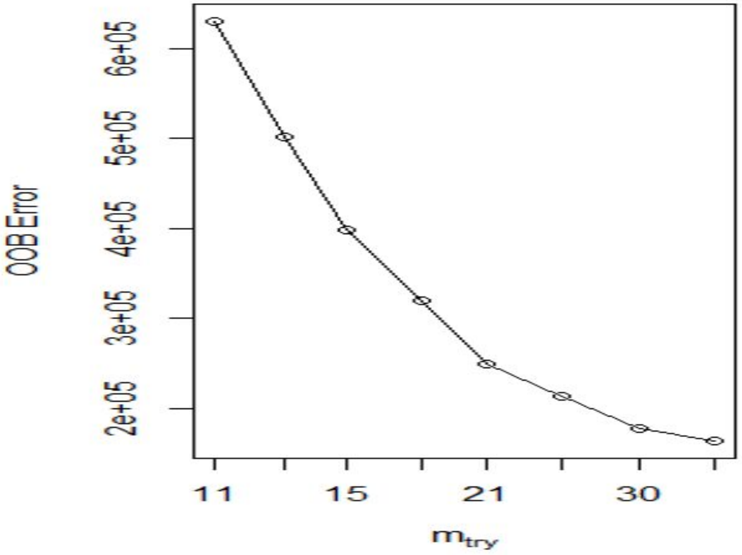
Random Forest



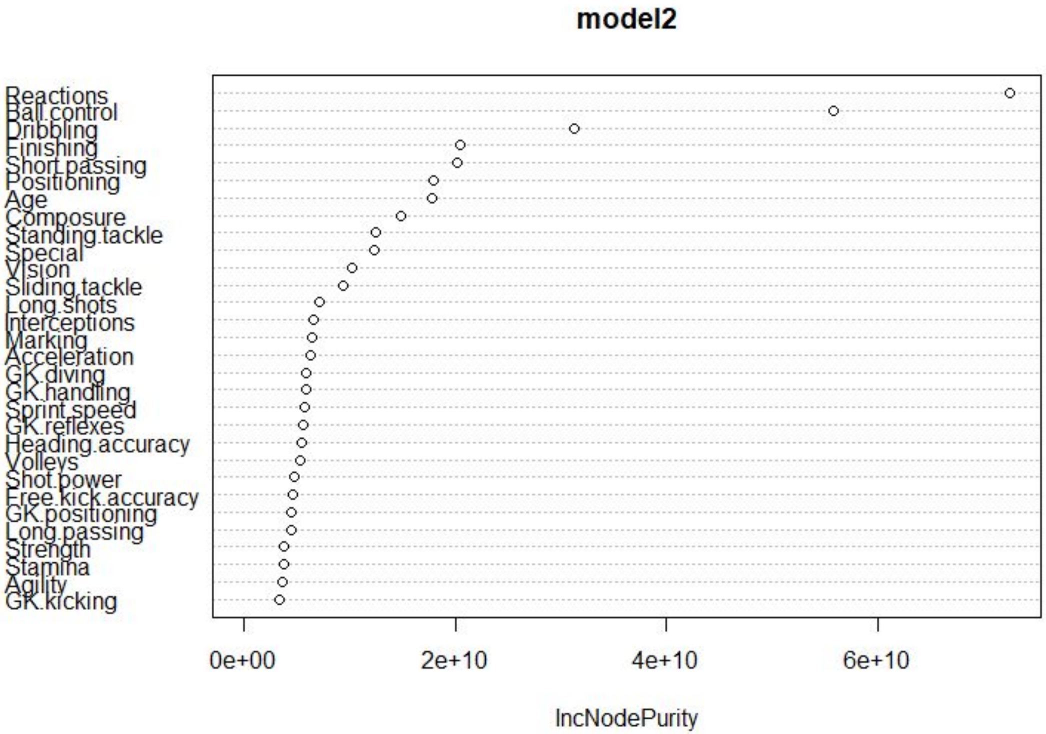
```
### BEST M-TRY
bestmtry <- tuneRF(train2, train2$y, ntreeTry = 500, stepFactor = 1.5, improve = 0.1, trace = T, plot = T )
bestmtry
#####   mtry OOBError
####    9   942300.7
####   11   630086.4
####   13   503678.7
####   15   398506.4
####   18   319758.0
####   21   248886.3
####   25   214182.1
####   30   178040.5
####   36   163966.9
```

Random Forest

OOB ERROR RATE CHART FOR Mtry



Variable Importance



RESULTS

Best FIT Parameters for Random Forest

Mtry = 25 Least OOB = 214182.1

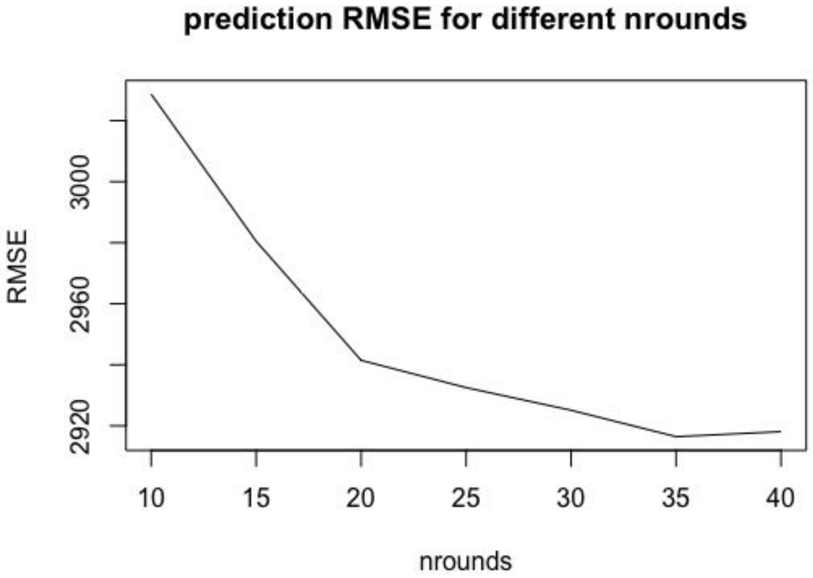
Acuuracy(R2) = 69% RMSE = 3303

XGBoost

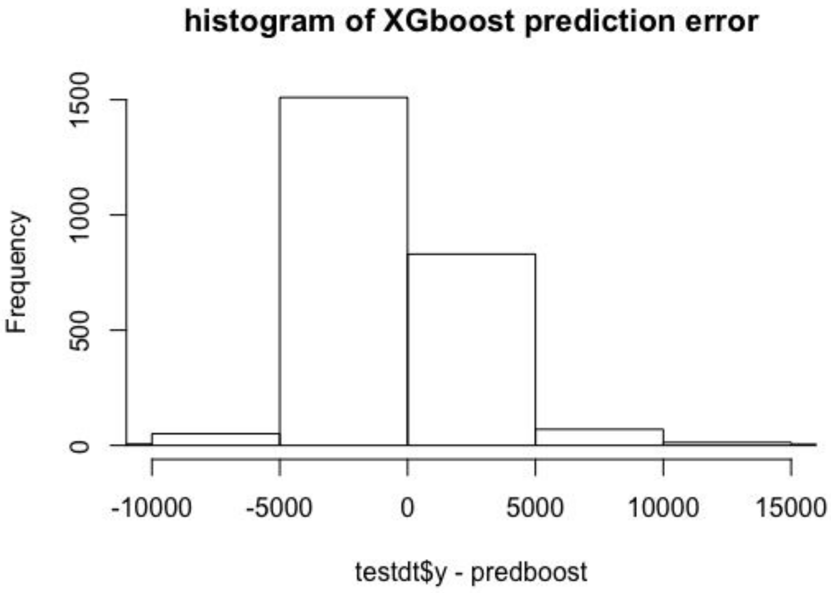
- We started with splitting the cleaned dataset into 80% for training and 20% for testing.
- Then, we fitted the training data in XGboost and set the parameter nrounds to be 10
- After that, we used the model to predict the value in testing data and compared the results with the true value to calculate RMSE predicted by the model
- modelboost<-xgboost(as.matrix(traindt[,1:41]),traindt$y,nrounds=35)

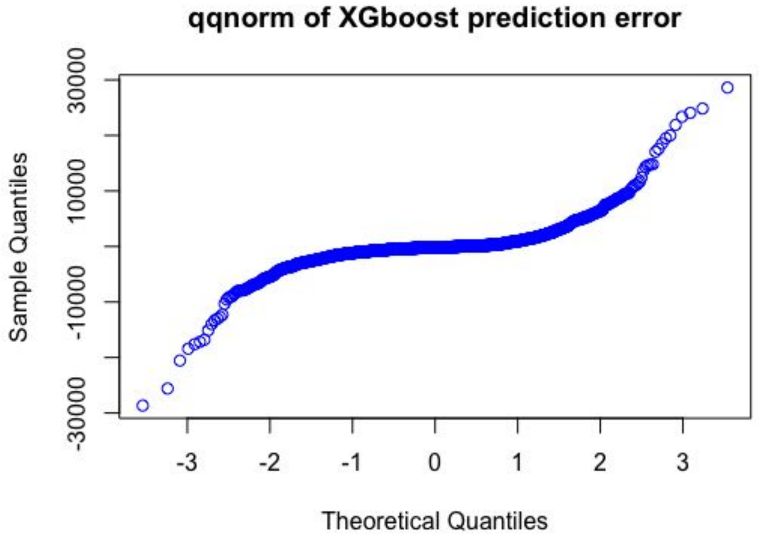
Picking Optimal Parameter

- We change the parameter nrounds until the prediction RMSE is minimized
- We find the optimal parameter of nrounds to be 35. The result of the model is plotted in the next page. The RMSE of prediction using test data is 2916.449



Plot for prediction error (XGBoost)





Residuals are not normally distributed, it is thin-tail. But it is OK to have residuals with thin-tail.

Conclusion

- Linear Regression: RMSE = 7920.3350, R-sq = 46%
- Decision Tree: RMSE=1160.637 , R-sq= 96.8%
- Random Forest: RMSE = 3303 , R-sq = 69%
- XGBoost: RMSE = 2916.449; R-sq = 77.8%

Here Decision tree is getting R squared value as 96% which means it remembers most of the data and so model is not generic or we might say it is overfitting the data.

Random Forest and XGBoost gives us the best results.