Random Forest, Visuals

Andrew Henderson

2025-03-31

```
## Loading required package: randomForest
## randomForest 4.7-1.2
## Type rfNews() to see new features/changes/bug fixes.
## Loading required package: caret
## Loading required package: ggplot2
##
## Attaching package: 'ggplot2'
## The following object is masked from 'package:randomForest':
##
##     margin
## Loading required package: lattice
```

```
data <- read.csv("transfer_dataset.csv", stringsAsFactors = T)
```

```
str(data)
```

```
## 'data.frame':    4124 obs. of  32 variables:
##  $ transfer                   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ player                     : Factor w/ 2326 levels "Aapo Halme","Aaron Leya Iseka",..: 658 1874 1
##  $ age                        : int  24 26 26 30 21 25 21 22 22 31 ...
##  $ season                     : Factor w/ 4 levels "2017-2018","2018-2019",..: 1 1 1 1 1 1 1 1 1 1
##  $ matches_played             : int  34 26 14 29 34 16 24 22 38 33 ...
##  $ play_proportion            : num  1 0.867 0.5 1 1 0.889 0.8 0.688 1 0.917 ...
##  $ raw_goals                  : num  0.206 0.179 0 0.298 0.088 0 0.115 0.212 0.344 0.032 ...
##  $ raw_assists                : num  0.088 0.268 0.093 0.037 0 0.145 0.058 0.318 0 0.193 ...
##  $ raw_nonpenaltykick_goal    : num  0.088 0.179 0 0.149 0.088 0 0.115 0.212 0.344 0.032 ...
##  $ total_pass_attempts        : num  58.9 50.8 69.4 37.9 36.7 ...
##  $ Total_Cmp.                 : num  0.856 0.708 0.903 0.557 0.793 0.785 0.841 0.664 0.65 0.825 ...
##  $ Total_Passes_Leading_to_Shot: num  1.059 1.431 0.187 1.64 0.147 ...
##  $ Cmp_Passes_18_yard_box     : num  0.824 1.163 0 1.379 0.118 ...
##  $ prog_dist_per_pass         : num  37 31.8 44.4 39.1 82.4 ...
##  $ Shot_Creating_Actions      : num  2.706 4.205 1.119 3.913 0.676 ...
##  $ def_action_to_shot         : num  0.147 0 0 0.075 0 0 0 0.106 0.086 0 ...
##  $ goal_creating_action       : num  0.529 0.805 0.187 0.261 0 0.218 0.231 0.424 0.129 0.417 ...
##  $ takeon_to_goal             : num  0.118 0.089 0 0 0 0 0.058 0 0.043 0 ...
##  $ prog_dist_per_carry        : num  59 31.6 110 24.8 203.6 ...
##  $ Progressive_Passes_Received : num  1.676 11.362 0.56 11.776 0.029 ...
##  $ TakeOn_Attempts            : num  1.5 4.92 0.373 5.329 0.441 ...
##  $ TakeOn_Success_Percentage  : num  0.725 0.527 0.75 0.552 0.733 0.611 0.733 0.2 0.629 0.724 ...
##  $ tackle_ratio               : num  0.561 0.774 0.462 0.571 0.741 0.48 0.746 0.652 0.688 0.631 ...
```

```
##  $ Shot_Blocks                : num  0.382 0 0.093 0.075 0.735 0.291 0.115 0 0.043 0.385 ...
##  $ Pass_Blocks                : num  1.029 1.521 0.746 1.379 0.382 ...
##  $ Clearances                 : num  1.824 0.179 0.933 1.118 3.765 ...
##  $ Aerial_Win_Percentage      : num  0.676 0.29 0.5 0.206 0.613 0.385 0.667 0.357 0.389 0.51 ...
##  $ aerials_total              : num  4.09 2.77 0.56 2.35 2.73 ...
##  $ FW                         : int  0 1 0 1 0 0 0 1 1 0 ...
##  $ MF                         : int  1 1 1 1 0 0 1 0 1 1 ...
##  $ DF                         : int  0 0 0 0 1 1 1 0 0 0 ...
##  $ X90minutes                 : num  34 11.2 10.7 26.8 34 ...
```

```
summary(data)
```

```
##     transfer                    player          age              season
##  Min.   :0.00000   Abdoulaye Bamba :   4   Min.   :16.00   2017-2018: 379
##  1st Qu.:0.00000   Abdoulaye Touré :   4   1st Qu.:22.00   2018-2019:1208
##  Median :0.00000   Adrien Hunou    :   4   Median :25.00   2019-2020:1244
##  Mean   :0.06159   Adrien Thomasson:   4   Mean   :25.54   2020-2021:1293
##  3rd Qu.:0.00000   Alexander Djiku :   4   3rd Qu.:28.00
##  Max.   :1.00000   Andrei Girotto  :   4   Max.   :42.00
##                    (Other)         :4100
##  matches_played  play_proportion    raw_goals      raw_assists
##  Min.   : 5.00   Min.   :0.1670   Min.   :0.000   Min.   :0.00000
##  1st Qu.:18.00   1st Qu.:0.7270   1st Qu.:0.000   1st Qu.:0.00000
##  Median :25.00   Median :0.8750   Median :0.069   Median :0.06100
##  Mean   :25.15   Mean   :0.8259   Mean   :0.126   Mean   :0.08561
##  3rd Qu.:32.00   3rd Qu.:0.9690   3rd Qu.:0.190   3rd Qu.:0.13300
##  Max.   :46.00   Max.   :1.0000   Max.   :1.268   Max.   :0.75800
##
##  raw_nonpenaltykick_goal total_pass_attempts   Total_Cmp.
##  Min.   :0.0000          Min.   : 11.08      Min.   :0.3580
##  1st Qu.:0.0000          1st Qu.: 32.24      1st Qu.:0.6890
##  Median :0.0650          Median : 43.06      Median :0.7485
##  Mean   :0.1149          Mean   : 42.83      Mean   :0.7426
##  3rd Qu.:0.1740          3rd Qu.: 52.91      3rd Qu.:0.8040
##  Max.   :1.2290          Max.   :108.65      Max.   :0.9630
##
##  Total_Passes_Leading_to_Shot Cmp_Passes_18_yard_box prog_dist_per_pass
##  Min.   :0.0000               Min.   :0.0000         Min.   :  0.00
##  1st Qu.:0.4400               1st Qu.:0.2900         1st Qu.: 29.24
##  Median :0.8320               Median :0.6215         Median : 41.73
##  Mean   :0.9219               Mean   :0.7034         Mean   : 55.17
##  3rd Qu.:1.2660               3rd Qu.:1.0040         3rd Qu.: 70.46
##  Max.   :4.6400               Max.   :4.8180         Max.   :380.50
##
##  Shot_Creating_Actions def_action_to_shot goal_creating_action
##  Min.   :0.000         Min.   :0.00000    Min.   :0.0000
##  1st Qu.:1.268         1st Qu.:0.00000    1st Qu.:0.0750
##  Median :2.067         Median :0.00000    Median :0.1780
##  Mean   :2.188         Mean   :0.03669    Mean   :0.2149
##  3rd Qu.:2.942         3rd Qu.:0.06100    3rd Qu.:0.3130
##  Max.   :8.869         Max.   :0.57100    Max.   :1.7130
##
##  takeon_to_goal    prog_dist_per_carry Progressive_Passes_Received
##  Min.   :0.00000   Min.   :   0.00     Min.   : 0.000
##  1st Qu.:0.00000   1st Qu.:  30.50     1st Qu.: 1.087
```

```
##  Median :0.00000   Median :  41.74     Median : 3.774
##  Mean   :0.01427   Mean   :  80.31     Mean   : 3.958
##  3rd Qu.:0.00000   3rd Qu.:  78.80     3rd Qu.: 6.201
##  Max.   :0.41200   Max.   :2533.00     Max.   :16.000
##
##  TakeOn_Attempts   TakeOn_Success_Percentage  tackle_ratio     Shot_Blocks
##  Min.   : 0.0260   Min.   :0.0000            Min.   :0.0000   Min.   :0.0000
##  1st Qu.: 0.5677   1st Qu.:0.4948            1st Qu.:0.5500   1st Qu.:0.0770
##  Median : 1.2190   Median :0.5880            Median :0.6250   Median :0.1990
##  Mean   : 1.5453   Mean   :0.5989            Mean   :0.6221   Mean   :0.2837
##  3rd Qu.: 2.1482   3rd Qu.:0.7040            3rd Qu.:0.7000   3rd Qu.:0.4153
##  Max.   :10.6830   Max.   :1.0000            Max.   :1.0000   Max.   :1.9820
##
##   Pass_Blocks       Clearances    Aerial_Win_Percentage aerials_total
##  Min.   :0.0000   Min.   :0.000   Min.   :0.0000        Min.   : 0.049
##  1st Qu.:0.4280   1st Qu.:0.698   1st Qu.:0.3520        1st Qu.: 1.991
##  Median :0.6640   Median :1.393   Median :0.4675        Median : 3.030
##  Mean   :0.7112   Mean   :1.925   Mean   :0.4565        Mean   : 3.804
##  3rd Qu.:0.9273   3rd Qu.:2.801   3rd Qu.:0.5683        3rd Qu.: 4.626
##  Max.   :2.7550   Max.   :9.432   Max.   :1.0000        Max.   :31.329
##
##        FW               MF               DF           X90minutes
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   : 4.444
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:11.111
##  Median :0.0000   Median :0.0000   Median :0.0000   Median :18.216
##  Mean   :0.3654   Mean   :0.4968   Mean   :0.4178   Mean   :19.453
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:26.547
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :46.000
##
```

```
data$transfer <- as.factor(data$transfer)
data$MF <- as.factor(data$MF)
data$DF <- as.factor(data$DF)
data$FW <- as.factor(data$FW)

data <- na.omit(data)
```

```
set.seed(82)
mf_down <- downSample(x = data[, setdiff(names(data), "transfer")],
                      y = data$transfer,
                      yname = "transfer")

trainIndex <- createDataPartition(mf_down$transfer, p = 0.7, list = FALSE)
trainData <- mf_down[trainIndex, ]
testData <- mf_down[-trainIndex, ]
```

```
rf_model <- randomForest(transfer ~ . - season - player,
                         data = trainData,
                         ntree = 100,
                         mtry = floor(sqrt(ncol(trainData) - 1)),
                         importance = TRUE)
```

```
print(rf_model)
```

```
##
```

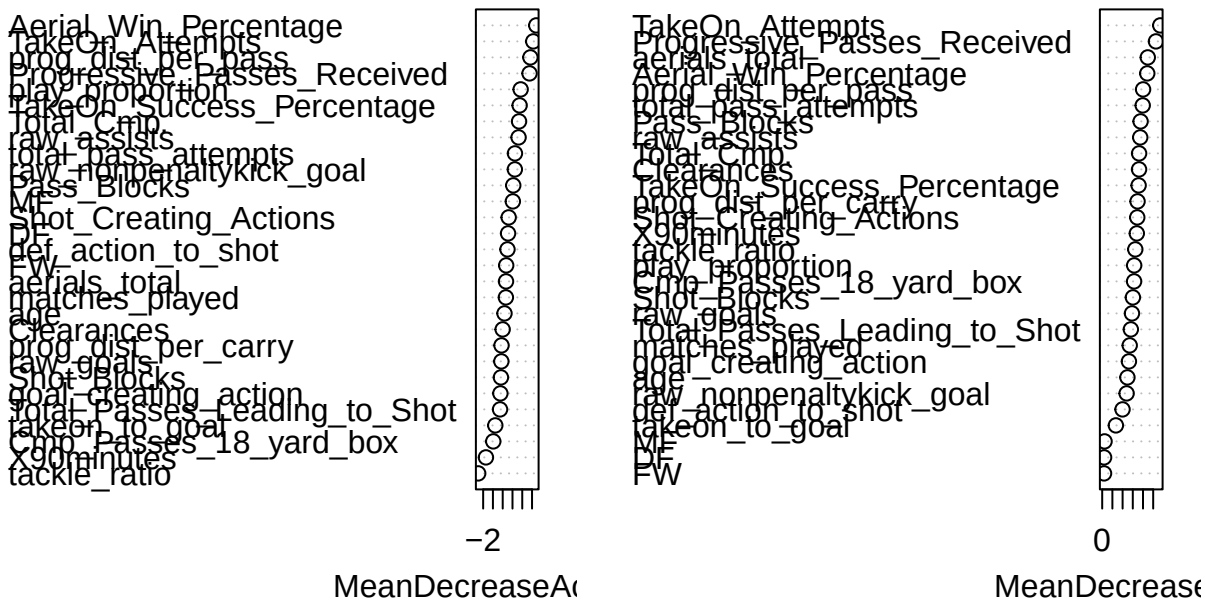
```
## Call:
##  randomForest(formula = transfer ~ . - season - player, data = trainData,      ntree = 100, mtry = fl
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 5
##
##         OOB estimate of  error rate: 44.66%
## Confusion matrix:
##     0  1 class.error
## 0 109 69   0.3876404
## 1  90 88   0.5056180
```

```
predictions <- predict(rf_model, testData)
```

```
confMat <- confusionMatrix(predictions, testData$transfer)
print(confMat)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 37 26
##          1 39 50
##
##                Accuracy : 0.5724
##                  95% CI : (0.4897, 0.6522)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : 0.04409
##
##                   Kappa : 0.1447
##
##  Mcnemar's Test P-Value : 0.13664
##
##             Sensitivity : 0.4868
##             Specificity : 0.6579
##          Pos Pred Value : 0.5873
##          Neg Pred Value : 0.5618
##              Prevalence : 0.5000
##          Detection Rate : 0.2434
##    Detection Prevalence : 0.4145
##       Balanced Accuracy : 0.5724
##
##        'Positive' Class : 0
##
```

```
varImpPlot(rf_model)
```



Random Forest Models / Position

```
positions <- c("MF", "FW", "DF")

models <- list()
conf_matrices <- list()

for (pos in positions) {

  pos_data <- subset(data, data[[pos]] == 1)
  pos_data <- pos_data[, !(names(pos_data) %in% c("season", "player", "MD", "FW", "DF"))]
  pos_data$transfer <- as.factor(pos_data$transfer)

  if(nrow(pos_data) < 10) {
    cat("Not enough data for position", pos, "\n")
    next
  }
   set.seed(42)

  trainIndex <- createDataPartition(pos_data$transfer, p = 0.7, list = FALSE)
  trainData <- pos_data[trainIndex, ]
  testData <- pos_data[-trainIndex, ]

  # Random Forest model
  rf_model <- randomForest(transfer ~ .,
                           data = trainData,
                           ntree = 500,
```

```
                          mtry = floor(sqrt(ncol(trainData) - 1)),
                          importance = TRUE)

  models[[pos]] <- rf_model

  # Predictions on the test set
  predictions <- predict(rf_model, testData)

  # Evaluate the model performance
  conf_mat <- confusionMatrix(predictions, testData$transfer)
  conf_matrices[[pos]] <- conf_mat

  cat("Confusion Matrix for", pos, "players:\n")
  print(conf_mat)
  cat("\n--------------------------------\n")
}
```

```
## Confusion Matrix for MF players:
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 579  33
##          1   0   2
##
##                Accuracy : 0.9463
##                  95% CI : (0.9253, 0.9627)
##     No Information Rate : 0.943
##     P-Value [Acc > NIR] : 0.4064
##
##                   Kappa : 0.1026
##
##  Mcnemar's Test P-Value : 2.54e-08
##
##             Sensitivity : 1.00000
##             Specificity : 0.05714
##          Pos Pred Value : 0.94608
##          Neg Pred Value : 1.00000
##              Prevalence : 0.94300
##          Detection Rate : 0.94300
##    Detection Prevalence : 0.99674
##       Balanced Accuracy : 0.52857
##
##        'Positive' Class : 0
##
##
## --------------------------------
## Confusion Matrix for FW players:
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 422  28
##          1   0   2
```

```
##
##                Accuracy : 0.9381
##                  95% CI : (0.9117, 0.9584)
##     No Information Rate : 0.9336
##     P-Value [Acc > NIR] : 0.3982
##
##                   Kappa : 0.1177
##
##  Mcnemar's Test P-Value : 3.352e-07
##
##             Sensitivity : 1.00000
##             Specificity : 0.06667
##          Pos Pred Value : 0.93778
##          Neg Pred Value : 1.00000
##              Prevalence : 0.93363
##          Detection Rate : 0.93363
##    Detection Prevalence : 0.99558
##       Balanced Accuracy : 0.53333
##
##        'Positive' Class : 0
##
##
## ---------------------------------
## Confusion Matrix for DF players:
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 485  30
##          1   0   1
##
##                Accuracy : 0.9419
##                  95% CI : (0.918, 0.9604)
##     No Information Rate : 0.9399
##     P-Value [Acc > NIR] : 0.4739
##
##                   Kappa : 0.059
##
##  Mcnemar's Test P-Value : 1.192e-07
##
##             Sensitivity : 1.00000
##             Specificity : 0.03226
##          Pos Pred Value : 0.94175
##          Neg Pred Value : 1.00000
##              Prevalence : 0.93992
##          Detection Rate : 0.93992
##    Detection Prevalence : 0.99806
##       Balanced Accuracy : 0.51613
##
##        'Positive' Class : 0
##
##
## ---------------------------------
```

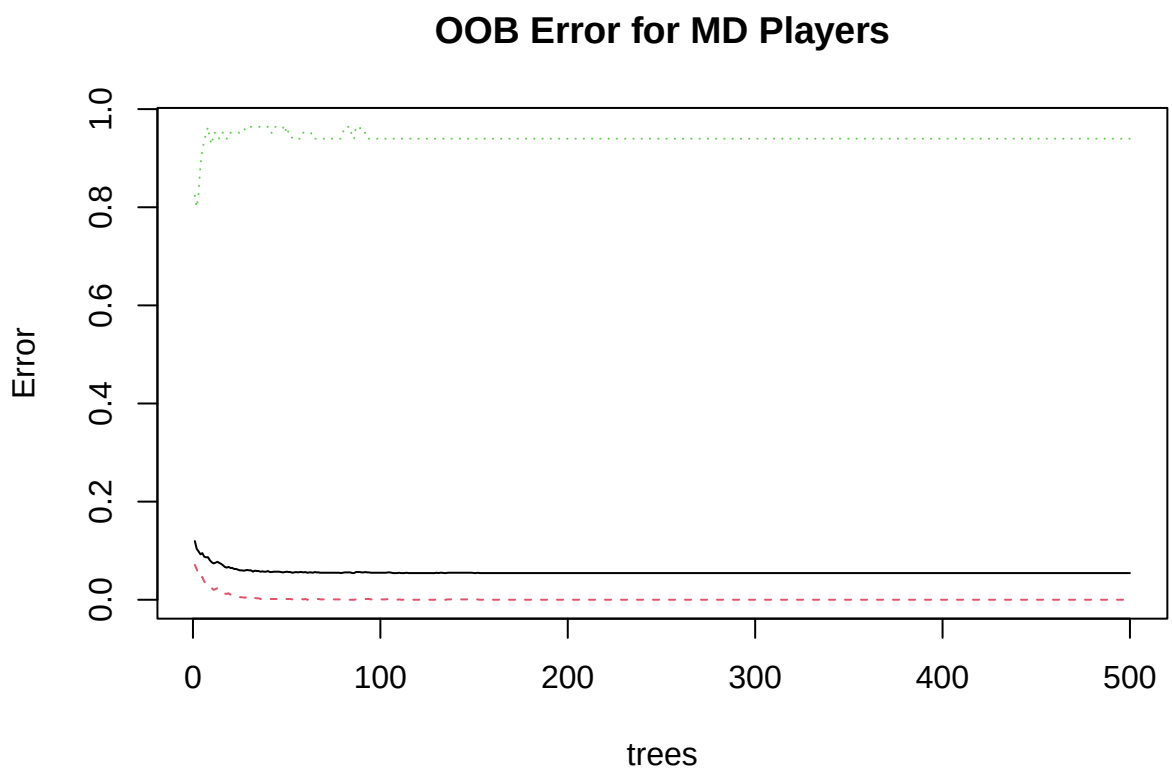
```r
#MF players:
cat("Model summary for MF players:\n")
```

```
## Model summary for MF players:
```

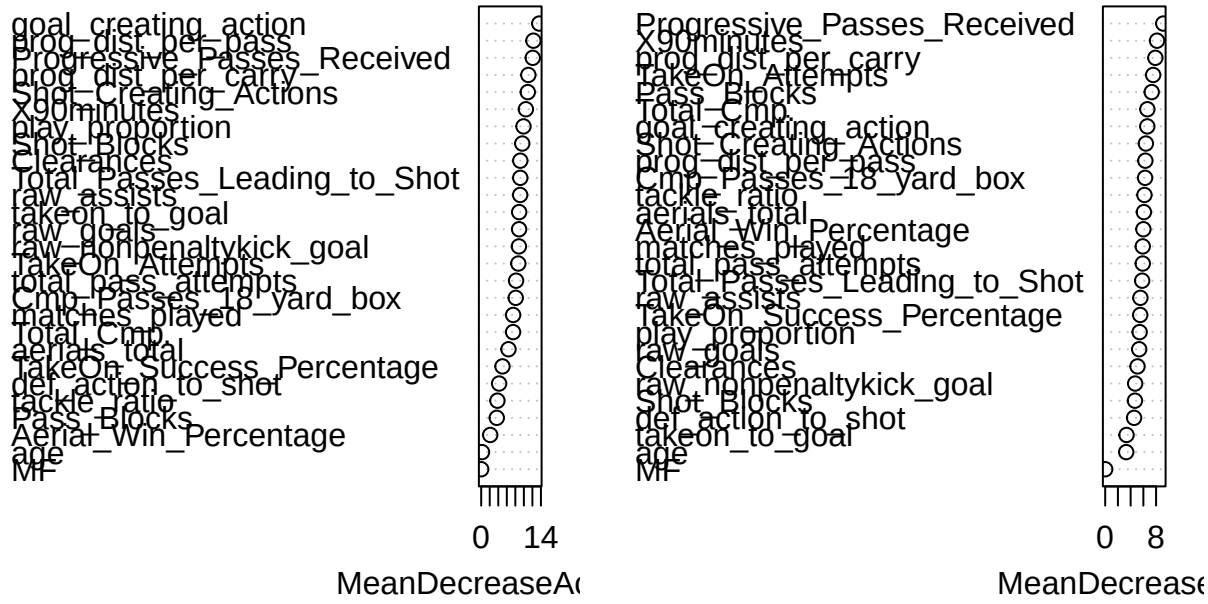
```r
print(models[["MF"]])
```

```
##
## Call:
##  randomForest(formula = transfer ~ ., data = trainData, ntree = 500,      mtry = floor(sqrt(ncol(tra
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##         OOB estimate of  error rate: 5.44%
## Confusion matrix:
##      0 1 class.error
## 0 1352 0    0.000000
## 1   78 5    0.939759
```

```r
plot(models[["MF"]], main = "OOB Error for MD Players")
```




```r
varImpPlot(models[["MF"]], main = "Variable Importance for MD Players")
```



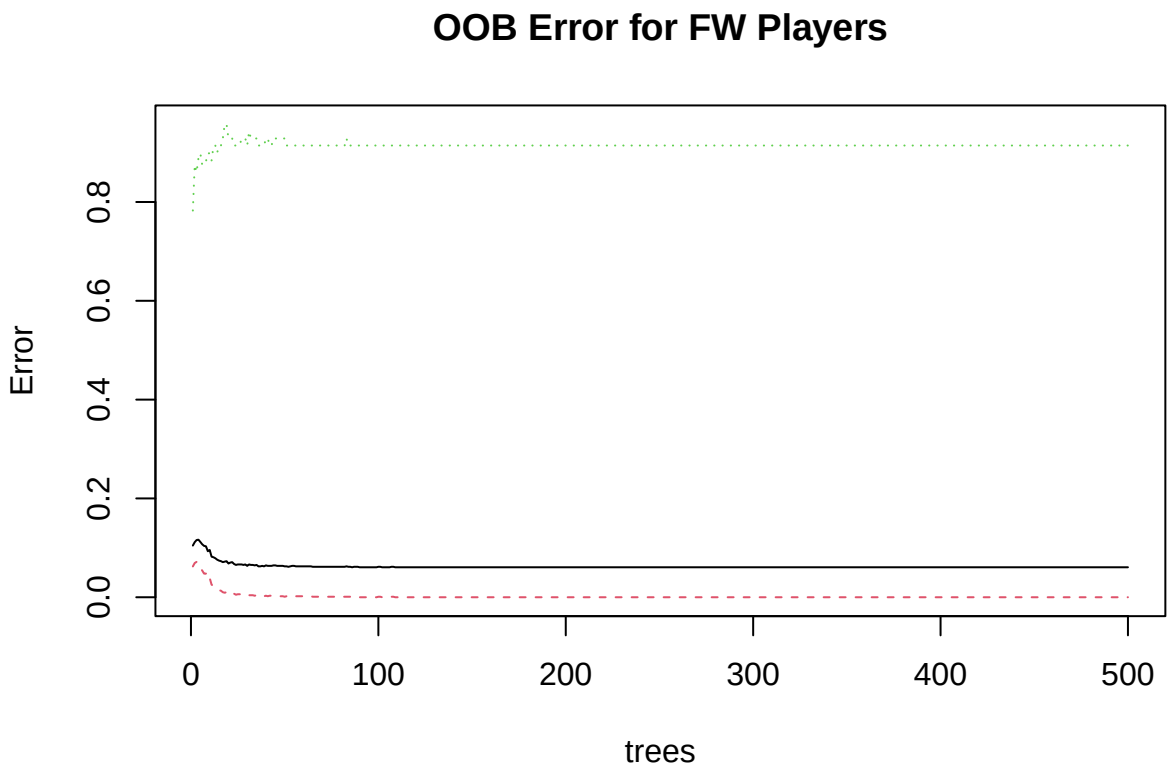

```
#FW players:
cat("Model summary for FW players:\n")
```

```
## Model summary for FW players:
```

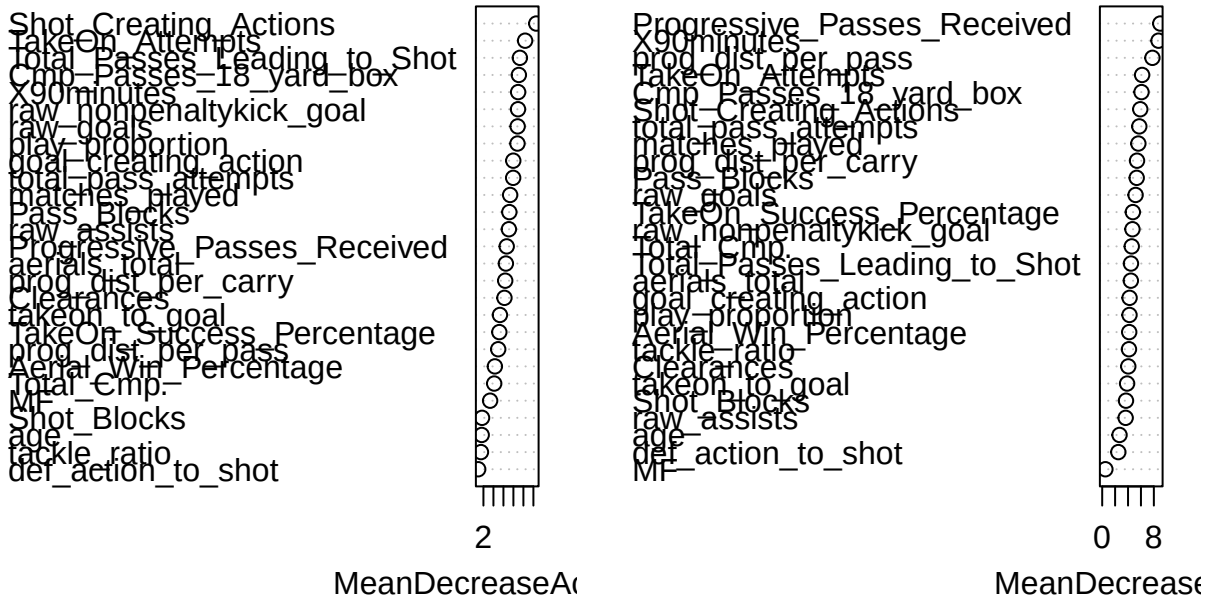
```
print(models[["FW"]])
```

```
##
## Call:
##  randomForest(formula = transfer ~ ., data = trainData, ntree = 500,      mtry = floor(sqrt(ncol(tra
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##         OOB estimate of  error rate: 6.07%
## Confusion matrix:
##     0 1 class.error
## 0 985 0   0.0000000
## 1  64 6   0.9142857
```

```
plot(models[["FW"]], main = "OOB Error for FW Players")
```




```
varImpPlot(models[["FW"]], main = "Variable Importance for FW Players")
```



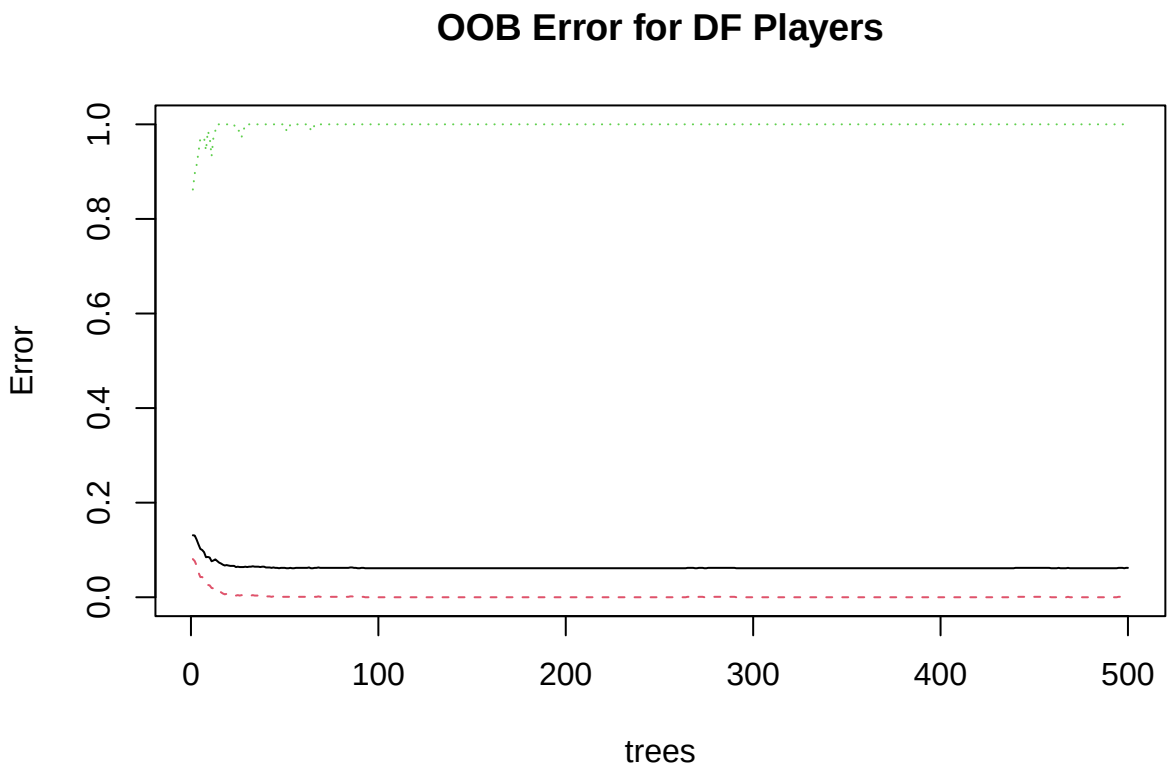

```r
#DF players:
cat("Model summary for DF players:\n")
```

```
## Model summary for DF players:
```

```r
print(models[["DF"]])
```

```
##
## Call:
##  randomForest(formula = transfer ~ ., data = trainData, ntree = 500,      mtry = floor(sqrt(ncol(tra
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##         OOB estimate of  error rate: 6.21%
## Confusion matrix:
##      0 1  class.error
## 0 1132 1 0.0008826125
## 1   74 0 1.0000000000
```

```r
plot(models[["DF"]], main = "OOB Error for DF Players")
```




```
varImpPlot(models[["DF"]], main = "Variable Importance for DF Players")
```

Variable Importance for DF Players