Fifa 19 Player Valuation prediction Project - Capstone

Rishabh Babeley - Harvard Data Science Professional

May 24, 2019

Abstract

The Fifa dataset is a very comprehensive dataset which contains a lot of attibutes of footballers from a lot of clubs and leagues around the world.It provides us an opportunity to delve deep inside these attributes and try to understand what is it exactly that determines a players current valuation.

The purpose if the project is to predict the Valuation of a Player in the dataset using some of his improtant attributes with a decent level of accuracy.Preferably an RMSE of less than # Exploratory Data Analysis

The Dataset

```
## Loading the dataset
players_data <- read.csv("data.csv")
# Check dimensions
dim(players_data)
```

```
## [1] 18207    89
```

```
head(players_data)
```

```
##   ï..     ID              Name Age
## 1   0 158023          L. Messi  31
## 2   1  20801 Cristiano Ronaldo  33
## 3   2 190871         Neymar Jr  26
## 4   3 193080            De Gea  27
## 5   4 192985      K. De Bruyne  27
## 6   5 183277         E. Hazard  27
##                                             Photo Nationality
## 1 https://cdn.sofifa.org/players/4/19/158023.png   Argentina
## 2  https://cdn.sofifa.org/players/4/19/20801.png    Portugal
## 3 https://cdn.sofifa.org/players/4/19/190871.png      Brazil
## 4 https://cdn.sofifa.org/players/4/19/193080.png       Spain
## 5 https://cdn.sofifa.org/players/4/19/192985.png     Belgium
## 6 https://cdn.sofifa.org/players/4/19/183277.png     Belgium
##                                  Flag Overall Potential
## 1 https://cdn.sofifa.org/flags/52.png      94        94
## 2 https://cdn.sofifa.org/flags/38.png      94        94
## 3 https://cdn.sofifa.org/flags/54.png      92        93
## 4 https://cdn.sofifa.org/flags/45.png      91        93
## 5  https://cdn.sofifa.org/flags/7.png      91        92
## 6  https://cdn.sofifa.org/flags/7.png      91        91
##                  Club                                     Club.Logo Value
## 1        FC Barcelona https://cdn.sofifa.org/teams/2/light/241.png 110.5
```

```
## 2           Juventus  https://cdn.sofifa.org/teams/2/light/45.png  77.0
## 3 Paris Saint-Germain  https://cdn.sofifa.org/teams/2/light/73.png 118.5
## 4   Manchester United  https://cdn.sofifa.org/teams/2/light/11.png  72.0
## 5     Manchester City  https://cdn.sofifa.org/teams/2/light/10.png 102.0
## 6             Chelsea   https://cdn.sofifa.org/teams/2/light/5.png  93.0
##      Wage Special Preferred.Foot International.Reputation Weak.Foot
## 1 â‚¬565K    2202           Left                        5         4
## 2 â‚¬405K    2228          Right                        5         4
## 3 â‚¬290K    2143          Right                        5         5
## 4 â‚¬260K    1471          Right                        4         3
## 5 â‚¬355K    2281          Right                        4         5
## 6 â‚¬340K    2142          Right                        4         4
##   Skill.Moves      Work.Rate  Body.Type Real.Face Position Jersey.Number
## 1           4 Medium/ Medium      Messi       Yes       RF            10
## 2           5      High/ Low C. Ronaldo       Yes       ST             7
## 3           5   High/ Medium     Neymar       Yes       LW            10
## 4           1 Medium/ Medium       Lean       Yes       GK             1
## 5           4     High/ High     Normal       Yes      RCM             7
## 6           4   High/ Medium     Normal       Yes       LF            10
##      Joined Loaned.From Contract.Valid.Until Height Weight   LS   ST   RS
## 1  1-Jul-04                             2021    5'7 159lbs 88+2 88+2 88+2
## 2 10-Jul-18                             2022    6'2 183lbs 91+3 91+3 91+3
## 3  3-Aug-17                             2022    5'9 150lbs 84+3 84+3 84+3
## 4  1-Jul-11                             2020    6'4 168lbs
## 5 30-Aug-15                             2023   5'11 154lbs 82+3 82+3 82+3
## 6  1-Jul-12                             2020    5'8 163lbs 83+3 83+3 83+3
##     LW   LF   CF   RF   RW  LAM  CAM  RAM   LM  LCM   CM  RCM   RM  LWB
## 1 92+2 93+2 93+2 93+2 92+2 93+2 93+2 93+2 91+2 84+2 84+2 84+2 91+2 64+2
## 2 89+3 90+3 90+3 90+3 89+3 88+3 88+3 88+3 88+3 81+3 81+3 81+3 88+3 65+3
## 3 89+3 89+3 89+3 89+3 89+3 89+3 89+3 89+3 88+3 81+3 81+3 81+3 88+3 65+3
## 4
## 5 87+3 87+3 87+3 87+3 87+3 88+3 88+3 88+3 88+3 87+3 87+3 87+3 88+3 77+3
## 6 89+3 88+3 88+3 88+3 89+3 89+3 89+3 89+3 89+3 82+3 82+3 82+3 89+3 66+3
##    LDM  CDM  RDM  RWB   LB  LCB   CB  RCB   RB Crossing Finishing
## 1 61+2 61+2 61+2 64+2 59+2 47+2 47+2 47+2 59+2       84        95
## 2 61+3 61+3 61+3 65+3 61+3 53+3 53+3 53+3 61+3       84        94
## 3 60+3 60+3 60+3 65+3 60+3 47+3 47+3 47+3 60+3       79        87
## 4                                                    17        13
## 5 77+3 77+3 77+3 77+3 73+3 66+3 66+3 66+3 73+3       93        82
## 6 63+3 63+3 63+3 66+3 60+3 49+3 49+3 49+3 60+3       81        84
##   HeadingAccuracy ShortPassing Volleys Dribbling Curve FKAccuracy
## 1              70           90      86        97    93         94
## 2              89           81      87        88    81         76
## 3              62           84      84        96    88         87
## 4              21           50      13        18    21         19
## 5              55           92      82        86    85         83
## 6              61           89      80        95    83         79
##   LongPassing BallControl Acceleration SprintSpeed Agility Reactions
## 1          87          96           91          86      91        95
## 2          77          94           89          91      87        96
## 3          78          95           94          90      96        94
## 4          51          42           57          58      60        90
## 5          91          91           78          76      79        91
## 6          83          94           94          88      95        90
```

```
##   Balance ShotPower Jumping Stamina Strength LongShots Aggression
## 1      95        85      68      72       59        94         48
## 2      70        95      95      88       79        93         63
## 3      84        80      61      81       49        82         56
## 4      43        31      67      43       64        12         38
## 5      77        91      63      90       75        91         76
## 6      94        82      56      83       66        80         54
##   Interceptions Positioning Vision Penalties Composure Marking
## 1            22          94     94        75        96      33
## 2            29          95     82        85        95      28
## 3            36          89     87        81        94      27
## 4            30          12     68        40        68      15
## 5            61          87     94        79        88      68
## 6            41          87     89        86        91      34
##   StandingTackle SlidingTackle GKDiving GKHandling GKKicking GKPositioning
## 1             28            26        6         11        15            14
## 2             31            23        7         11        15            14
## 3             24            33        9          9        15            15
## 4             21            13       90         85        87            88
## 5             58            51       15         13         5            10
## 6             27            22       11         12         6             8
##   GKReflexes Release.Clause
## 1          8      â,¬226.5M
## 2         11      â,¬127.1M
## 3         11      â,¬228.1M
## 4         94      â,¬138.6M
## 5         13      â,¬196.4M
## 6          8      â,¬172.1M
```

Taking a look at the first 6 rows we can see that there are many columns which we might not require in our analysis.

```
# Removing unwanted columns
colnames(players_data)
```

```
##  [1] "ï.."                       "ID"
##  [3] "Name"                      "Age"
##  [5] "Photo"                     "Nationality"
##  [7] "Flag"                      "Overall"
##  [9] "Potential"                 "Club"
## [11] "Club.Logo"                 "Value"
## [13] "Wage"                      "Special"
## [15] "Preferred.Foot"            "International.Reputation"
## [17] "Weak.Foot"                 "Skill.Moves"
## [19] "Work.Rate"                 "Body.Type"
## [21] "Real.Face"                 "Position"
## [23] "Jersey.Number"             "Joined"
## [25] "Loaned.From"               "Contract.Valid.Until"
## [27] "Height"                    "Weight"
## [29] "LS"                        "ST"
## [31] "RS"                        "LW"
## [33] "LF"                        "CF"
## [35] "RF"                        "RW"
## [37] "LAM"                       "CAM"
## [39] "RAM"                       "LM"
```

```
## [41] "LCM"                      "CM"
## [43] "RCM"                      "RM"
## [45] "LWB"                      "LDM"
## [47] "CDM"                      "RDM"
## [49] "RWB"                      "LB"
## [51] "LCB"                      "CB"
## [53] "RCB"                      "RB"
## [55] "Crossing"                 "Finishing"
## [57] "HeadingAccuracy"          "ShortPassing"
## [59] "Volleys"                  "Dribbling"
## [61] "Curve"                    "FKAccuracy"
## [63] "LongPassing"              "BallControl"
## [65] "Acceleration"             "SprintSpeed"
## [67] "Agility"                  "Reactions"
## [69] "Balance"                  "ShotPower"
## [71] "Jumping"                  "Stamina"
## [73] "Strength"                 "LongShots"
## [75] "Aggression"               "Interceptions"
## [77] "Positioning"              "Vision"
## [79] "Penalties"                "Composure"
## [81] "Marking"                  "StandingTackle"
## [83] "SlidingTackle"            "GKDiving"
## [85] "GKHandling"               "GKKicking"
## [87] "GKPositioning"            "GKReflexes"
## [89] "Release.Clause"
```

```
players_data_filtered <- players_data[,-c(29:54)]
players_data_filtered <- players_data_filtered[,-c(1,5,7,63)]
colnames(players_data_filtered)
```

```
##  [1] "ID"                       "Name"
##  [3] "Age"                      "Nationality"
##  [5] "Overall"                  "Potential"
##  [7] "Club"                     "Club.Logo"
##  [9] "Value"                    "Wage"
## [11] "Special"                  "Preferred.Foot"
## [13] "International.Reputation" "Weak.Foot"
## [15] "Skill.Moves"              "Work.Rate"
## [17] "Body.Type"                "Real.Face"
## [19] "Position"                 "Jersey.Number"
## [21] "Joined"                   "Loaned.From"
## [23] "Contract.Valid.Until"     "Height"
## [25] "Weight"                   "Crossing"
## [27] "Finishing"                "HeadingAccuracy"
## [29] "ShortPassing"             "Volleys"
## [31] "Dribbling"                "Curve"
## [33] "FKAccuracy"               "LongPassing"
## [35] "BallControl"              "Acceleration"
## [37] "SprintSpeed"              "Agility"
## [39] "Reactions"                "Balance"
## [41] "ShotPower"                "Jumping"
## [43] "Stamina"                  "Strength"
## [45] "LongShots"                "Aggression"
## [47] "Interceptions"            "Positioning"
## [49] "Vision"                   "Penalties"
```

```
## [51] "Composure"               "Marking"
## [53] "StandingTackle"          "SlidingTackle"
## [55] "GKDiving"                "GKHandling"
## [57] "GKKicking"               "GKPositioning"
## [59] "GKReflexes"
```

```r
# Checking for NAs in the data
sapply(players_data_filtered, function(x) sum(is.na (x)))
```

```
##                       ID                     Name                      Age
##                        0                        0                        0
##              Nationality                  Overall                Potential
##                        0                        0                        0
##                     Club                Club.Logo                    Value
##                        0                        0                        0
##                     Wage                  Special           Preferred.Foot
##                        0                        0                        0
## International.Reputation                Weak.Foot              Skill.Moves
##                       48                       48                       48
##                Work.Rate                Body.Type                Real.Face
##                        0                        0                        0
##                 Position            Jersey.Number                   Joined
##                        0                       60                        0
##              Loaned.From      Contract.Valid.Until                   Height
##                        0                        0                        0
##                   Weight                 Crossing                Finishing
##                        0                       48                       48
##          HeadingAccuracy             ShortPassing                  Volleys
##                       48                       48                       48
##                Dribbling                    Curve               FKAccuracy
##                       48                       48                       48
##              LongPassing              BallControl             Acceleration
##                       48                       48                       48
##              SprintSpeed                  Agility                Reactions
##                       48                       48                       48
##                  Balance                ShotPower                  Jumping
##                       48                       48                       48
##                  Stamina                 Strength                LongShots
##                       48                       48                       48
##               Aggression            Interceptions              Positioning
##                       48                       48                       48
##                   Vision                Penalties                Composure
##                       48                       48                       48
##                  Marking           StandingTackle            SlidingTackle
##                       48                       48                       48
##                 GKDiving               GKHandling                GKKicking
##                       48                       48                       48
##            GKPositioning               GKReflexes
##                       48                       48
```

Since we want a comprehensive data for our analysis we can either drop the columns with the NA's or use only the rows for which data is present in all the columns.

```r
# Removing columns with NA values

players_data_filtered<-players_data_filtered %>%
```

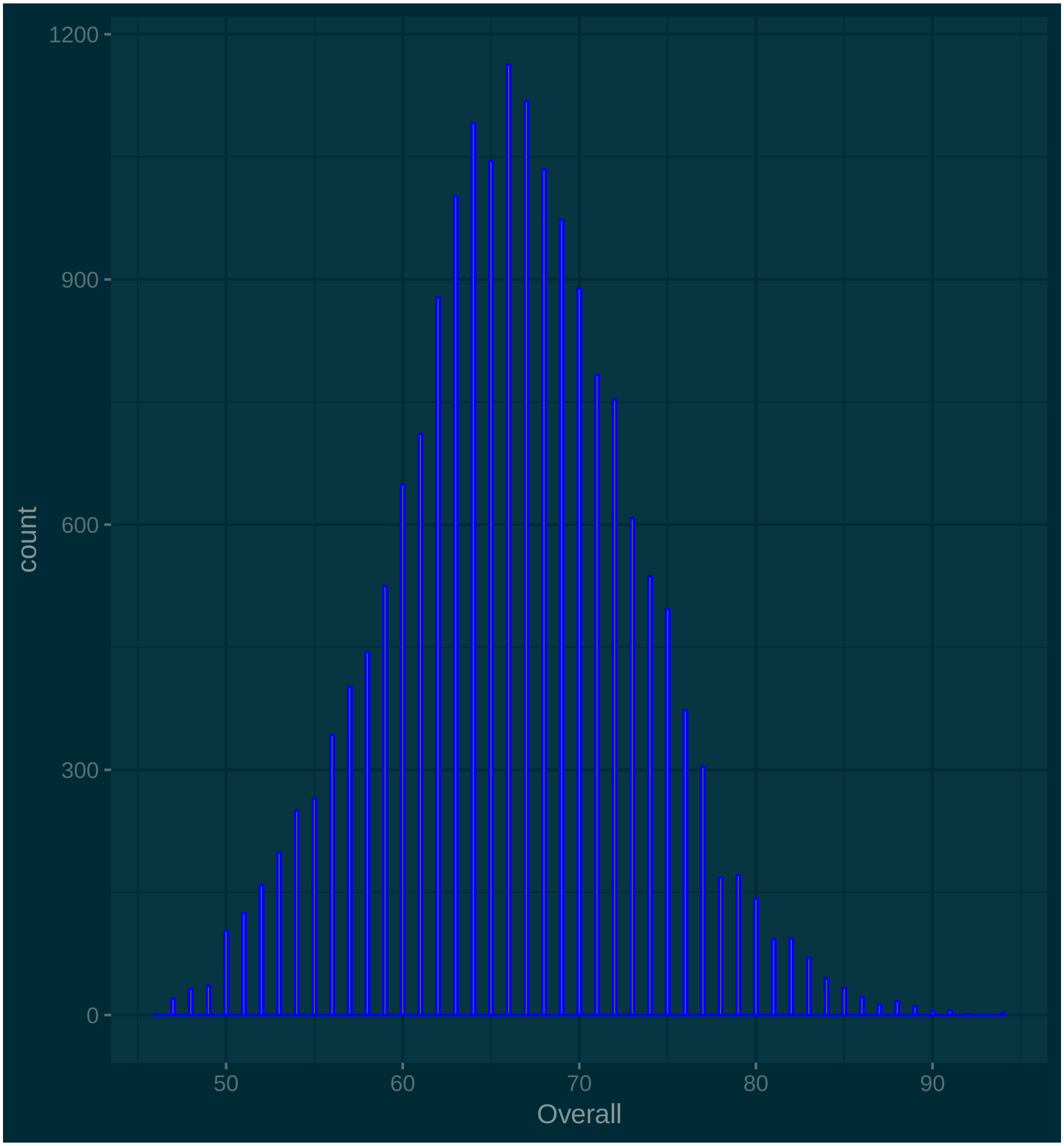
```
                 select_if(~ !any(is.na(.)))
```

There are still a lot of columns that we might not end up using but let's keep them as of now for exploratory purposes. Let's visulaize the current data and gather some insights first.

Data Visualization

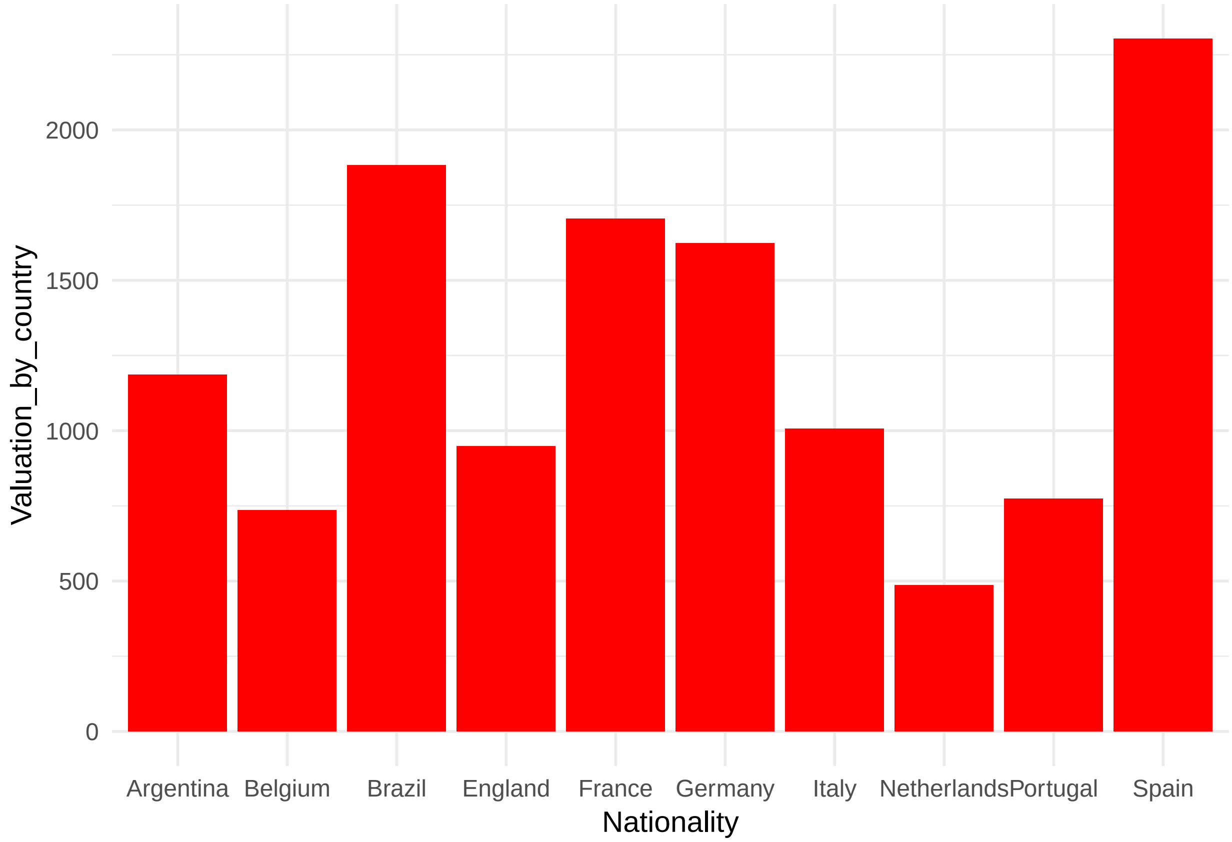
Let's see how the Overall ratings are distributed with a Histogram .

```
# Distribution of Overall ratings

  ggplot(players_data_filtered,aes(Overall))+theme_solarized_2(light=FALSE)+
    geom_histogram(color="blue",binwidth = 0.2)
```



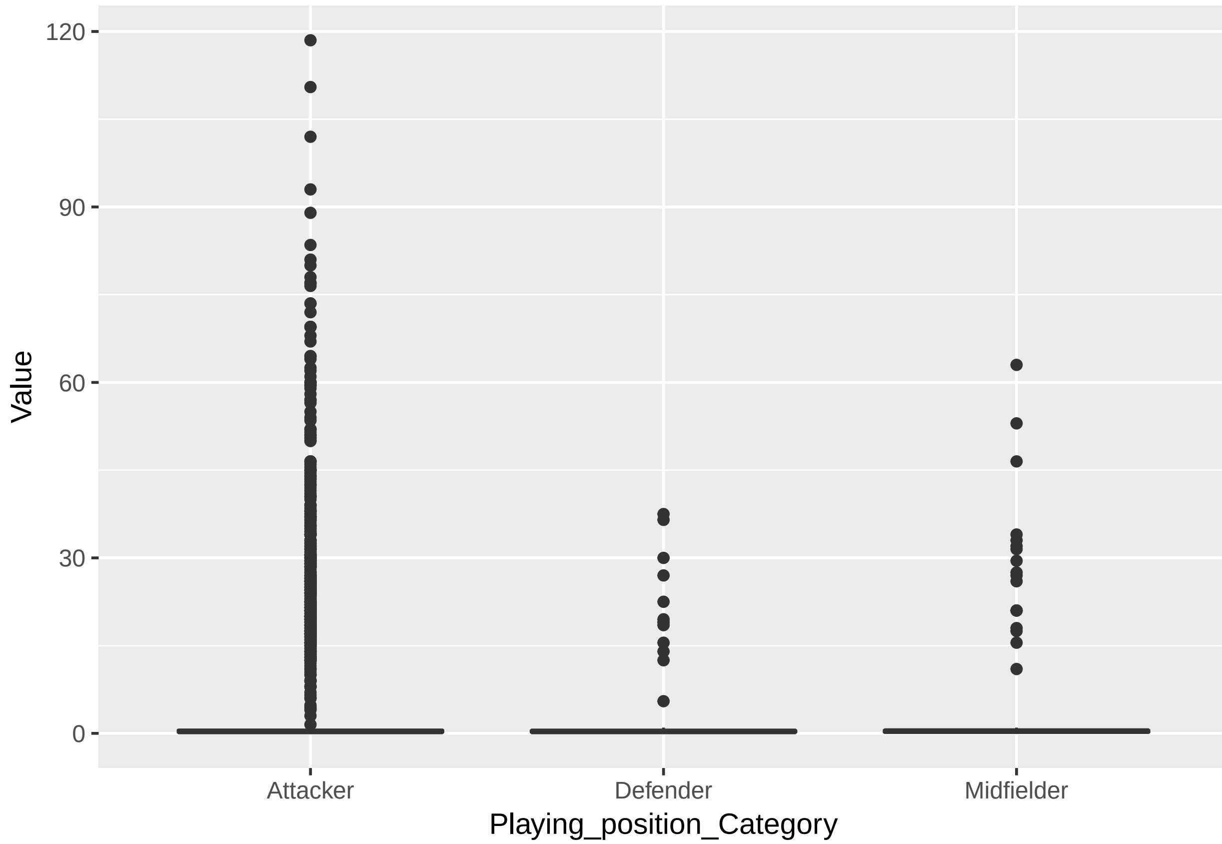
Although Football is a global sport with presense in almost all countries, there are some countries whose players have valuation predominantly higher that others. Let us explore this.

```r
players_data_filtered$Value <- as.numeric(players_data_filtered$Value)

Valuation_by_country<-players_data_filtered %>% group_by(Nationality) %>% summarise(Valuation_by_countr

# Top 10 countries with largest valuations
  Valuation_by_country[order(-Valuation_by_country$Valuation_by_country),] %>% head(10) %>%
    ggplot(aes(Nationality,Valuation_by_country)) + geom_bar(stat="identity", fill="red")+
  theme_minimal()
```



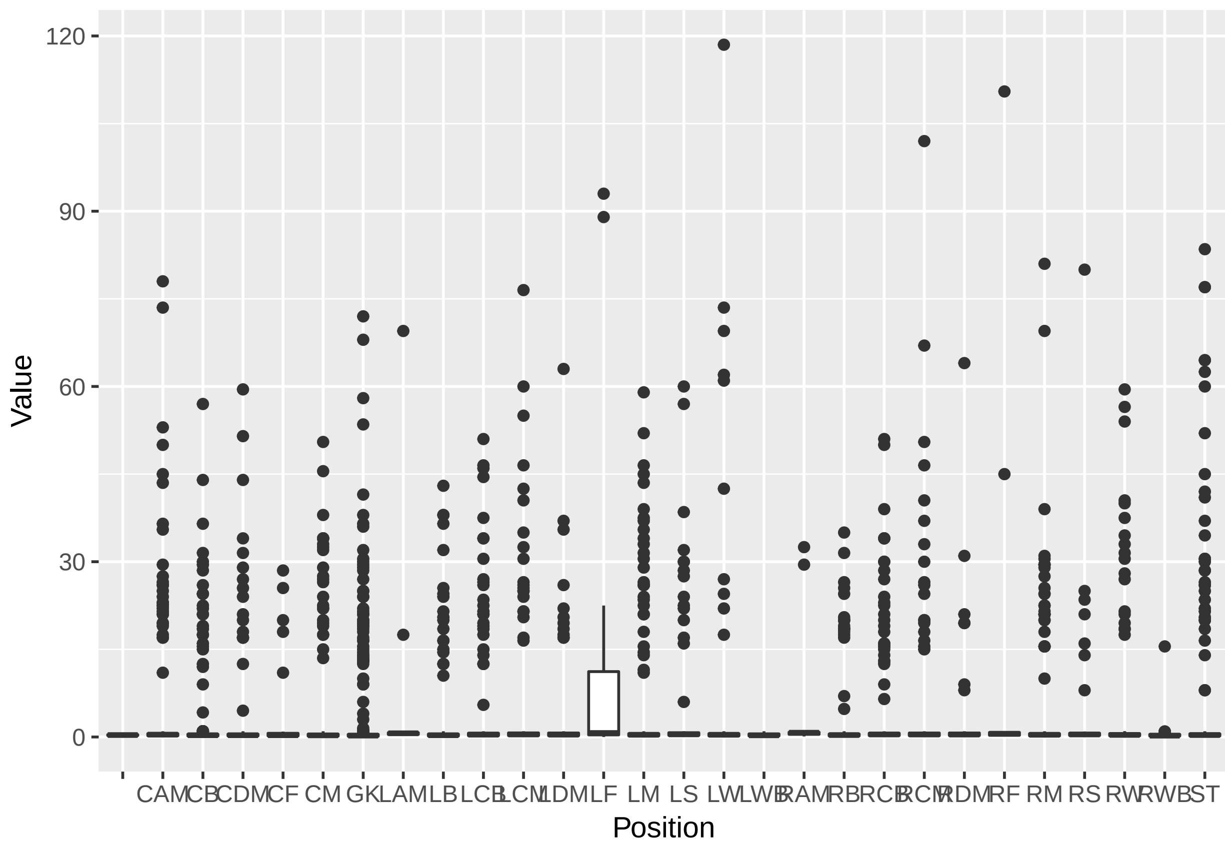
Now let us look at the dependence of playing poition on valuation.

```r
Position_category<-players_data_filtered %>%  mutate(Playing_position_Category="Attacker",Playing_posit

ggplot(Position_category ,aes(Playing_position_Category,Value))+geom_boxplot()+
  scale_color_brewer(palette="Dark2")
```



This representation tells us that the Valuation of Attackers overall is higher than both the other categories. We can still dig deeper into sub-categories of these.

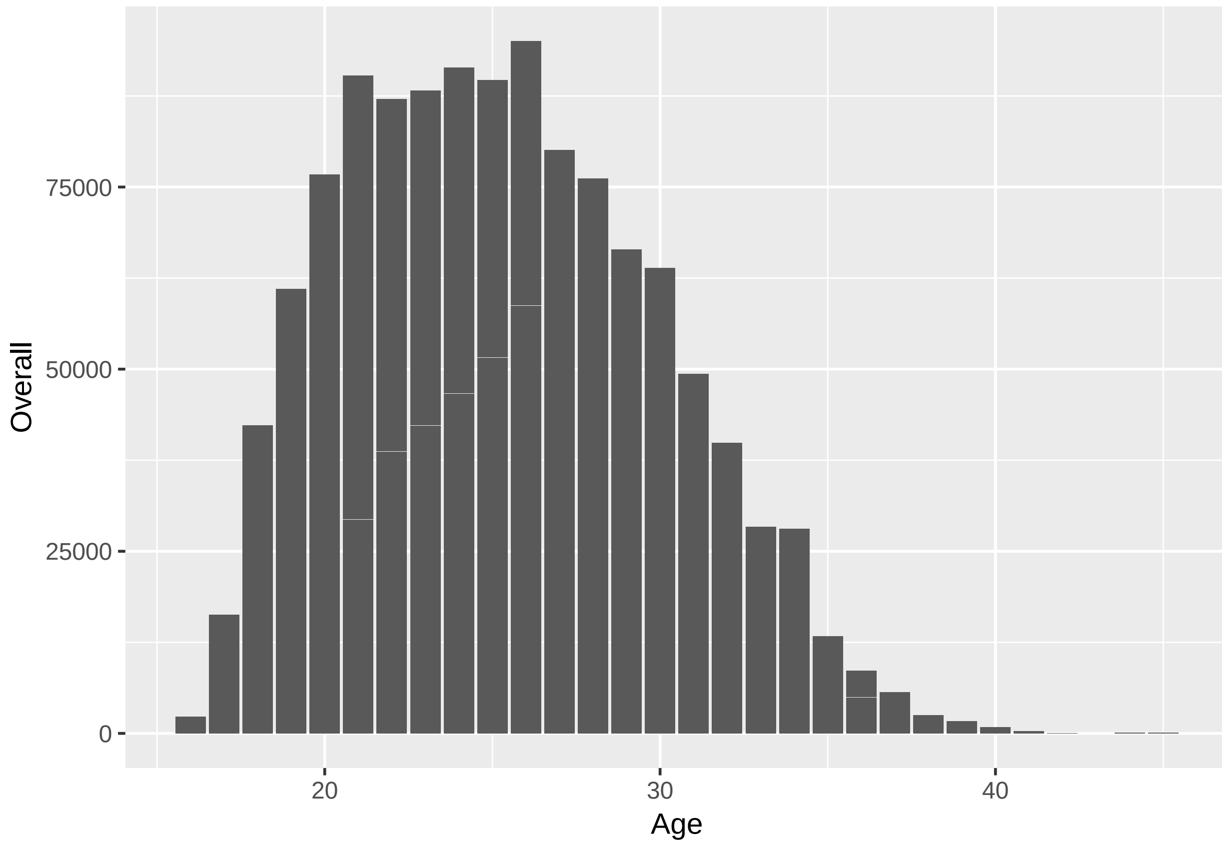
```
ggplot(Position_category ,aes(Position,Value))+geom_boxplot()+
  scale_color_brewer(palette="Dark2")
```



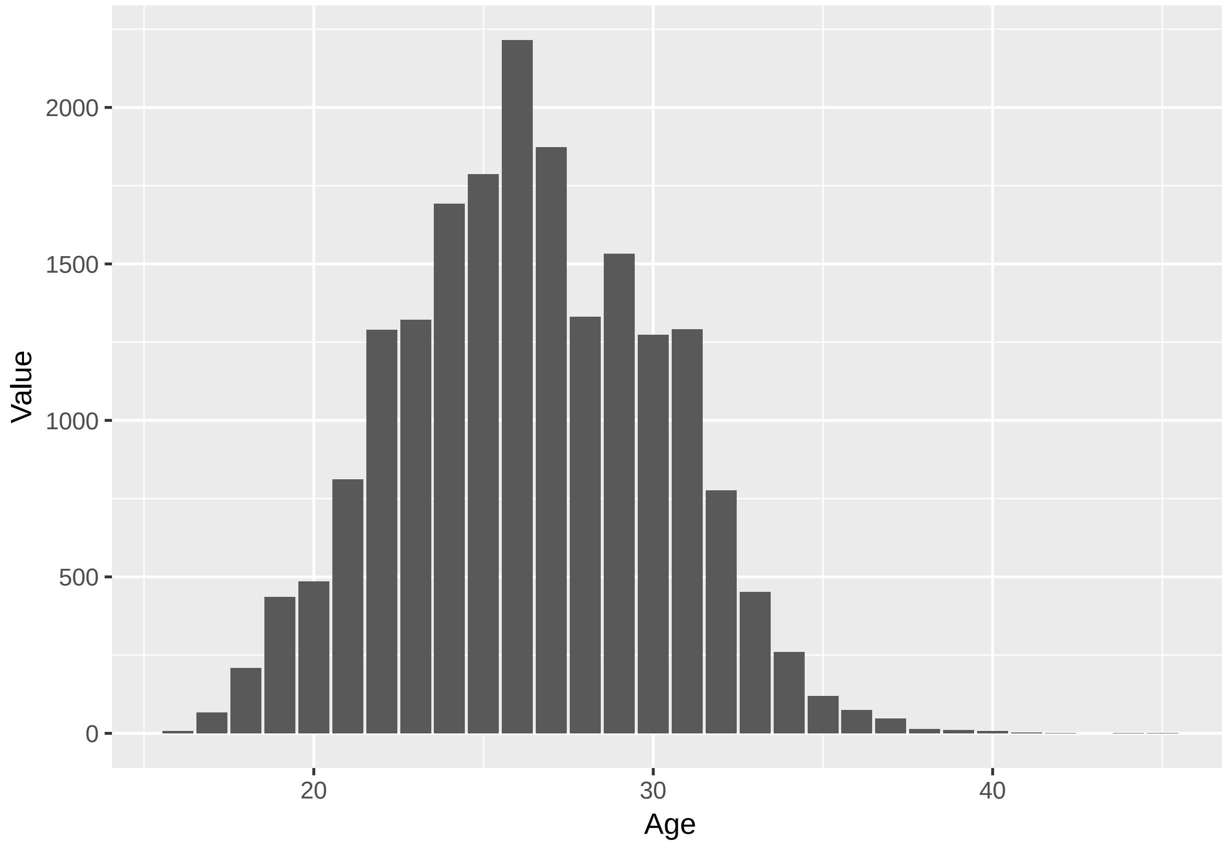
This boxplot tells us that some playing positions have much higher valuations that others.

Now lets look at the dependence of Player's age on his overall rating and valuation

```
ggplot(players_data_filtered,aes(Age,Overall))+geom_bar(stat="identity")
```




```
ggplot(players_data_filtered,aes(Age,Value))+geom_bar(stat="identity")
```



We can conclude from these graph that on an broader scale ,most players reach their peak overall performance at the age of 26-27 and decreases after that.

Data Pre-Processing

Now let us proceed towards building our models.We will first make testing and training datasets.

We define the RMSE function as following:

RMSE <- function(true_ratings = NULL, predicted_ratings = NULL) { sqrt(mean((true_ratings - predicted_ratings)^2)) }

Analysis - Models Building and Comparison

Random Forest

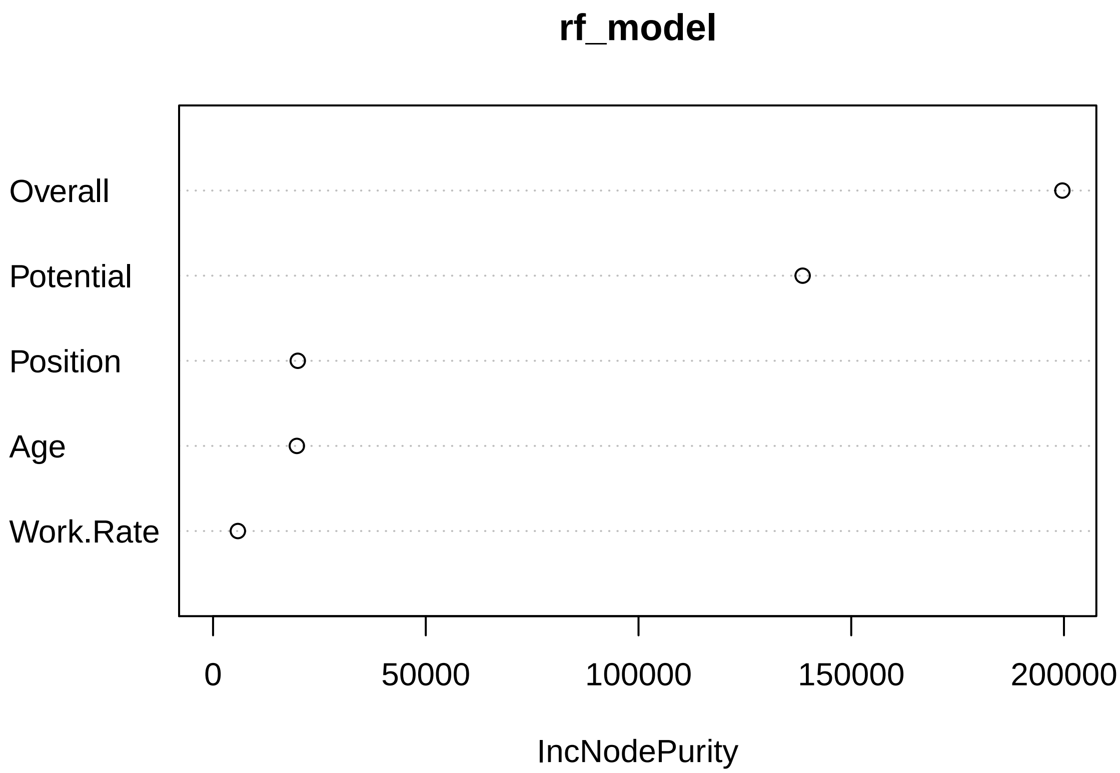
Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

```
# Set seed 1234 for reproducibility

set.seed(1234)
```

```r
# Build a Random Forest Model with Value as Target and all other
# variables as predictors. The number of trees is set to 500


rf_model <- randomForest(Value ~ ., data = train, ntree = 500,proximity=TRUE)

  # Get the feature importance

feature_imp_rf <- data.frame(importance(rf_model))

# Make predictions based on this model

predictions <- predict(rf_model, newdata=test)

errors = abs(predictions - test$Value)

#Calculating the Root MEan Squared Error

rmse<-RMSE(test$Value,predictions)

# Adding the respective metrics to the results dataset


results<-data.frame(Model=as.character(),rmse=as.double())
results <- results %>% add_row(
  Model = "Random Forest",
  rmse = rmse )

# Show results on a table

results %>%
  kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed","responsive"),
      position = "center",
      font_size = 10,
      full_width = FALSE)
```

Model	rmse
Random Forest	0.8350321

```r
# Show feature importance on a table
varImpPlot(rf_model)
```



Thus our Random forest algorithm gave an RMSE of **0.8350321** predicting the Value of a player using Overall rating,Potential,Work Rate, age and Position. This can be considered a is a good prediction.Although it is good enough we can further refine and stablize our model if we want using cross validation.

Taking a look at the feature importance table the Overall rating and the Potential are way more important variables to predict the Value of a player.

We now come on to a different machine learning algorithm which is Support Vector Machines.

SVM - Support Vector Machines

"Support Vector Machine" (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges.In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we find the hyper-plane that differentiate the two classes very well.

Here we are building an SVM model cost function 1000, gamma 0.01. Here we are performing Cross Validation of 2 fold is being performed in order to avoid overfitting and increase stability.

```
# Set seed 1234 for reproducibility
set.seed(1234)

# Build a SVM Model with Value as Target and all other
# variables as predictors. The kernel is set to default which is linear
#Cross Validation of 2 fold is being performed in order to avoid overfitting and increase stability.
svm_model <- svm(Value ~ ., data = train,cost=1000,gamma=0.01,cross=2)
```

```
## Warning in cret$cresults * scale.factor: Recycling array of length 1 in vector-array arithmetic is d
##   Use c() or as.vector() instead.
```

```
# Make predictions based on this model

predictions <- predict(svm_model, newdata=test)

rmse=RMSE(test$Value,predictions)

# Adding the respective metrics to the results dataset

results <- results %>% add_row(
  Model = "SVM Result",
  rmse = rmse )

# Show results on a table

results %>%
  kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed",           "responsive"),
      position = "center",
      font_size = 10,
      full_width = FALSE)
```

Model	rmse
Random Forest	0.8350321
SVM Result	1.4996891

The SVM Model with a default linear Kernel is a big a step back as it has a Root mean squared error of **1.4996891** which is larger as compared to Random Forest.This is the case even after applying 2 fold cross validation.Thus SVM , although is a fast method does not produce satisfactory results for us.

We thus move on to our next method which is XGBoost.

XGBoost

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. In prediction problems involving unstructured data (images, text, etc.) artificial neural networks tend to outperform all other algorithms or frameworks. However, when it comes to small-to-medium structured/tabular data, decision tree based algorithms are considered best-in-class right now.

XGBoost are a top class model. It always stays on TOP5 (or wins them) in every competitions on Kaggle and in this case, its' very fast to train and its performance are awesome.

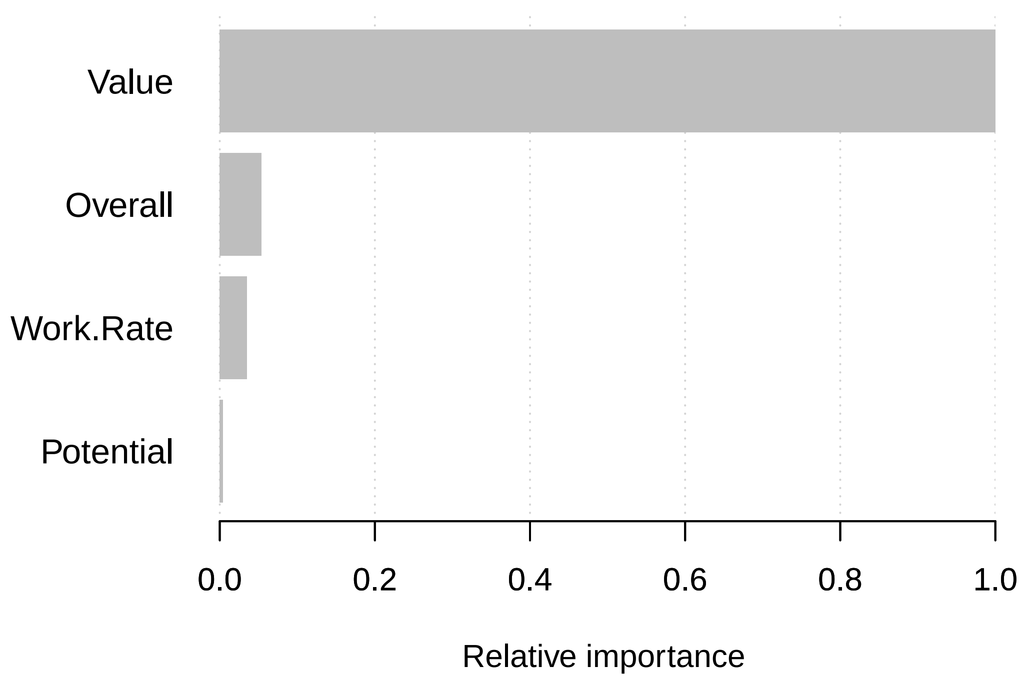
Here we perform XGBoost training with cross validation and try to see if this method gives us a better result.

```r
# Set seet 1234 for reproducibility

set.seed(1234)

# Prepare the training dataset

xgb_train <- xgb.DMatrix(
  as.matrix(train[, colnames(train) != c("Value","Position")]),
  label = train$Value)


# Prepare the test dataset

xgb_test <- xgb.DMatrix(
  as.matrix(test[, colnames(test) != c("Value","Position")]),
  label = test$Value)


test_cv<-test[1:1000,]

# Prepare the cv dataset

xgb_cv <- xgb.DMatrix(
  as.matrix(test_cv[, colnames(test_cv) != c("Value","Position")]),
  label = test_cv$Value)


# Prepare the parameters list.

xgb_params <- list(
  eta = 0.01,
  max.depth = 5,
  nthread = 6
)

# Train the XGBoost Model

xgb_model <- xgb.train(
  data = xgb_train,
  params = xgb_params,
  watchlist = list(test = xgb_test, cv = xgb_cv),
  nrounds = 1000,
  early_stopping_rounds = 20,
  print_every_n = 50
```

```
)
```

```
## [1]  test-rmse:4.099204  cv-rmse:7.807679
## Multiple eval metrics are present. Will use cv_rmse for early stopping.
## Will train until cv_rmse hasn't improved in 20 rounds.
##
## [51] test-rmse:2.516181  cv-rmse:4.789492
## [101]    test-rmse:1.625311  cv-rmse:3.089906
## [151]    test-rmse:1.127940  cv-rmse:2.139398
## [201]    test-rmse:0.879168  cv-rmse:1.663109
## [251]    test-rmse:0.763001  cv-rmse:1.440672
## [301]    test-rmse:0.719781  cv-rmse:1.358225
## [351]    test-rmse:0.705893  cv-rmse:1.332039
## Stopping. Best iteration:
## [353]    test-rmse:0.705681  cv-rmse:1.331650
```

```
# Get feature importance

feature_imp_xgb <- xgb.importance(colnames(train), model = xgb_model)

xgb.plot.importance(feature_imp_xgb, rel_to_first = TRUE, xlab = "Relative importance")
```




```
# Make predictions based on this model

predictions_xgboost = predict(
  xgb_model,
  newdata = as.matrix(test[, colnames(test) != c("Value","Position")]),
```

```r
  ntreelimit = xgb_model$bestInd
)

errors_xgboost = abs(predictions_xgboost - test$Value)

#Calculating the Mean Absolute percentage Error

rmse=RMSE(test$Value,predictions_xgboost)

# Adding the respective metrics to the results dataset

results <- results %>% add_row(
  Model = "XG_BOOST",
  rmse = rmse )


# Show feature importance on a table

feature_imp_xgb %>%
  kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed","responsive"),
      position = "center",
      font_size = 10,
      full_width = FALSE)
```

Feature	Gain	Cover	Frequency	Importance
Value	0.9152253	0.8921940	0.3955831	0.9152253
Overall	0.0490937	0.0943324	0.1591696	0.0490937
Work.Rate	0.0319768	0.0109772	0.3038876	0.0319768
Potential	0.0037043	0.0024964	0.1413597	0.0037043

The above analysis with XG-Boost with Cross validation suggests that is method is much superior to both our above methods.It gives us an RMSE of **0.7056808**.This shows its superiority over other tree based methods.

Results

This is the summary results for all the models builted, trained and validated.

Conclusion

We started out with exploring out dataset which contained many attributes of a player and his valuation at that point of time. We built some graphs, plots and tables to gather some insights.

We saw that the distribution of the Overall rating was normal.We then saw that some countries have players who have much higher combined valuation that other countries.

We observerved that the Attackers we on an average valued more that other categories this was also somewhat expected.Although there were some intresting patterns on how various sub - positions were valued compared to others.

We then saw that at about the age of 26-27 a player generally reaches his peakperformance , which then devaluates.

We then moved on to our models.We started out by running Random Forest Regression and and got a decent RMSE value of **0.8350321** which can be considered acceptable and concluded that Random Forest can be a good way of valuating players.

We also ploted the feature importances here to see which variables are most important.

We then tried out the Support Vector Machines algorithm with cross validation.This gave us an RMSE of **1.4996891**.This was not a satisfactory performance.

Finally we delved into the XGBoost algorithm with cross validation.XGBoost outperformed both our previous algorithms and gave us an RMSE of **0.7056808**.This demostrated its superiority and usability to predict the valuation if players.

Appendix

Acknowledgements

Sources

1.Wikipedia

2.www.towardsdatascience.com

3.www.analyticsvidhya.com