Configuration Manual

MSc Research Project
Data Analytics

Murtaza Saifi
Student ID: X18129463

School of Computing
National College of Ireland

Supervisor: Prof. Vladimir Milosavljevic

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	MURTAZA SAIFI
Student ID:	X1829463
Programme:	DATA ANALYTICS **Year:** 2019
Module:	MSC RESEARCH PROJECT
Lecturer:	Dr. Vladimir Milosavljevic
Submission Due Date:	12/12/2019
Project Title:	Implementation of Machine Learning Techniques to Predict Player Performance using Underlying Statistics
Word Count:	1288 **Page Count:** 13

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ………

Date: ………

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Murtaza Saifi
Student ID:x18129463
MSc Research Project in Data Analytics
11th December 2019

1 Introduction

The objective of this manual is to showcase the technical aspect of this project that involves system requirements and programming snippets that have not been covered in the main report. We will initiate with the basic system requirements utilized and discuss the implementation of the methodology.

1.1 System Requirement

- Hardware spec
    1. System Manufacturer: Dell Inc.
    2. Operating System: Windows 8.1 Pro 64-bit
    3. Processor: Intel(R) Core (TM) i5-4200U CPU @ 1.60GHZ (4 CPUs), ~2.3GHz
    4. Memory: 6 GB RAM

- Software spec
    1. R
    2. Tableau
    3. Microsoft Excel
    4. Kaggle (Kernel)

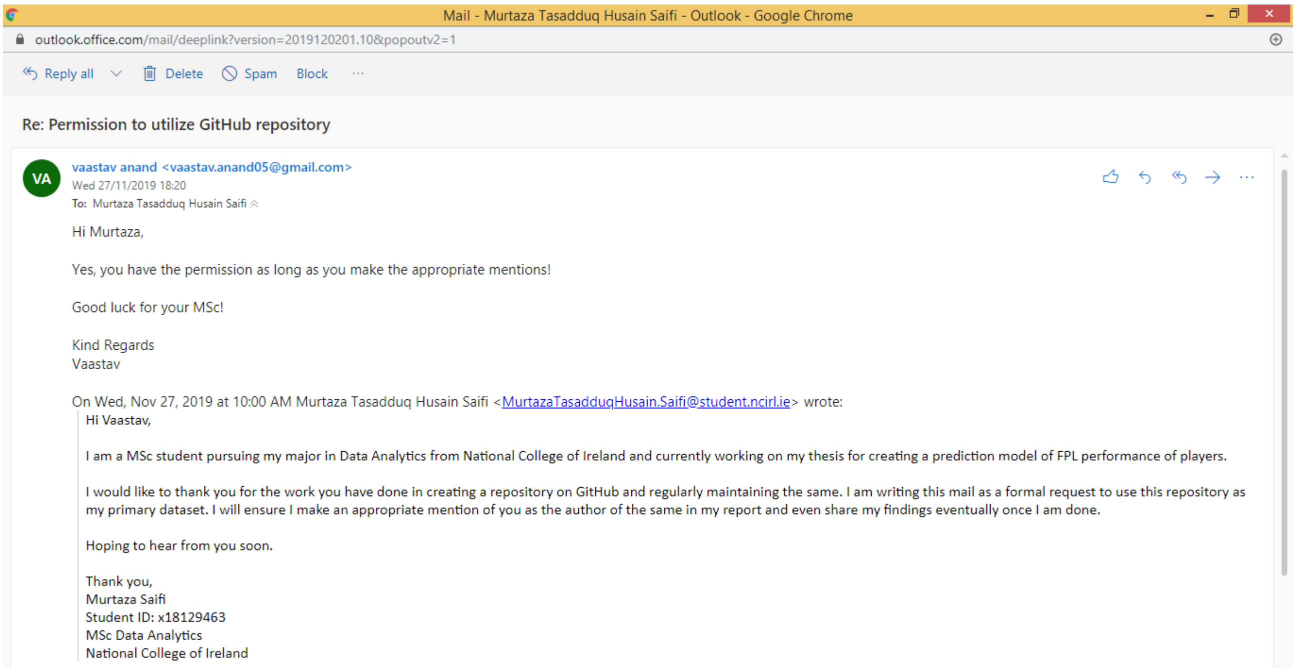
2 Project Development

Data preparation is done in multiple stages, between Excel & R studio. The code snapshots have been placed to avoid confusion of any kind.

2.1 Data Preparation

We primarily focus on 2 datasets in this study:

1. Fantasy Premier League Dataset: This is a Github repository (available here1) that has been active since 2016 and is sharing weekly updates sheets on match-weeks as they are conducted in the season. Permission to use the dataset has been taken via mail and can be observed below in Figure 1.



Mail - Murtaza Tasadduq Husain Saifi - Outlook - Google Chrome

outlook.office.com/mail/deeplink?version=2019120201.10&popoutv2=1

Reply all | Delete | Spam | Block

Re: Permission to utilize GitHub repository

vaastav anand <vaastav.anand05@gmail.com>
Wed 27/11/2019 18:20
To: Murtaza Tasadduq Husain Saifi

Hi Murtaza,

Yes, you have the permission as long as you make the appropriate mentions!

Good luck for your MSc!

Kind Regards
Vaastav

On Wed, Nov 27, 2019 at 10:00 AM Murtaza Tasadduq Husain Saifi <MurtazaTasadduqHusain.Saifi@student.ncirl.ie> wrote:

Hi Vaastav,

I am a MSc student pursuing my major in Data Analytics from National College of Ireland and currently working on my thesis for creating a prediction model of FPL performance of players.

I would like to thank you for the work you have done in creating a repository on GitHub and regularly maintaining the same. I am writing this mail as a formal request to use this repository as my primary dataset. I will ensure I make an appropriate mention of you as the author of the same in my report and even share my findings eventually once I am done.

Hoping to hear from you soon.

Thank you,
Murtaza Saifi
Student ID: x18129463
MSc Data Analytics
National College of Ireland

Figure 1: Email discussion for permission of FPL Dataset use

2. Understat.com Dataset: This dataset is obtained from the understat.com team which includes underlying statistics such as Expected Goals (xG) and Expected Assists (xA). The dataset[2] was shared post requesting via mail as can be observed from Figure 2. As we can observe, the communication for both datasets have taken place via the student email address.

[1] https://github.com/vaastav/Fantasy-Premier-League

[2] https://docs.google.com/spreadsheets/d/1Jgapvetj5mGOOh1nzmsYiKrTWNHsRXB45o0uGnJVDcI/edit?usp=sharing

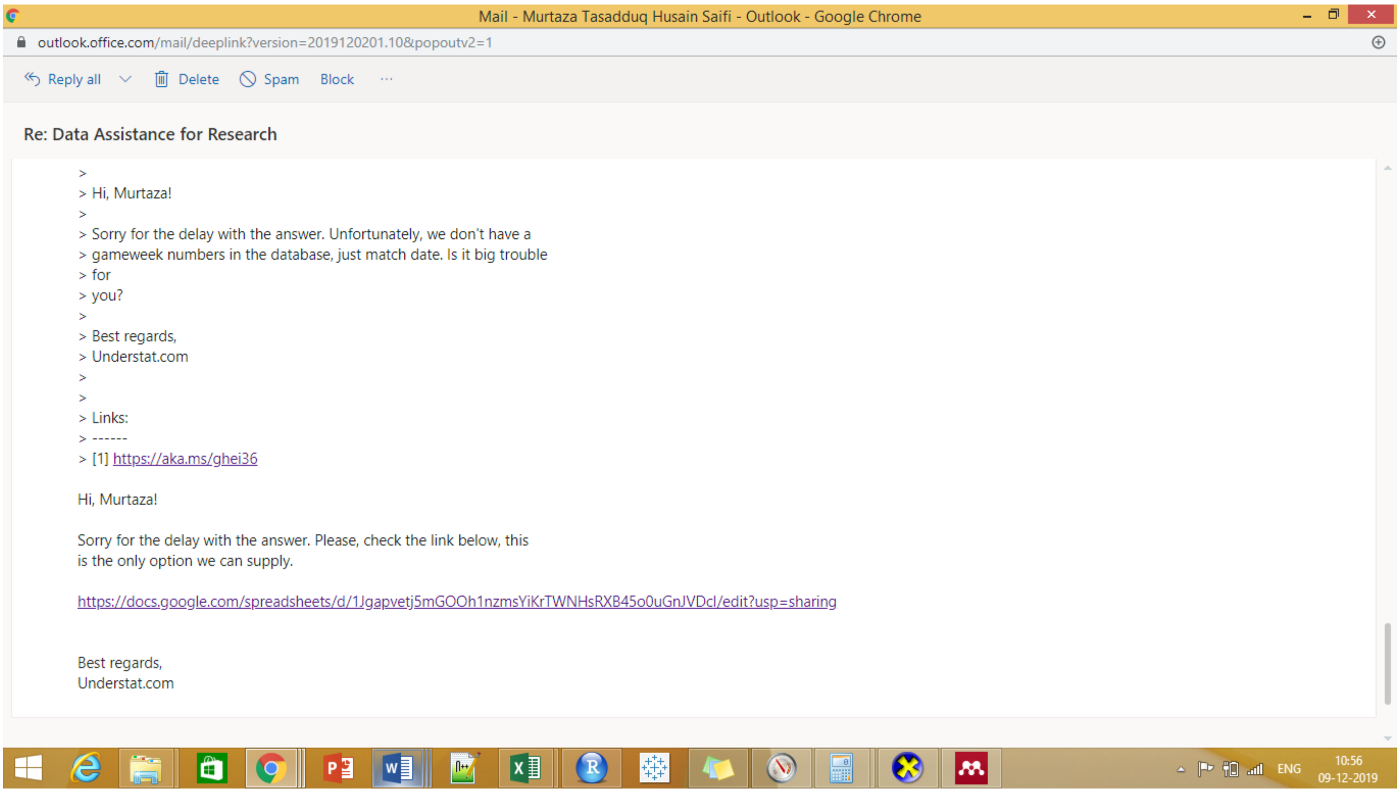


Figure 2: Mail response from understat.com sharing xG and xA data

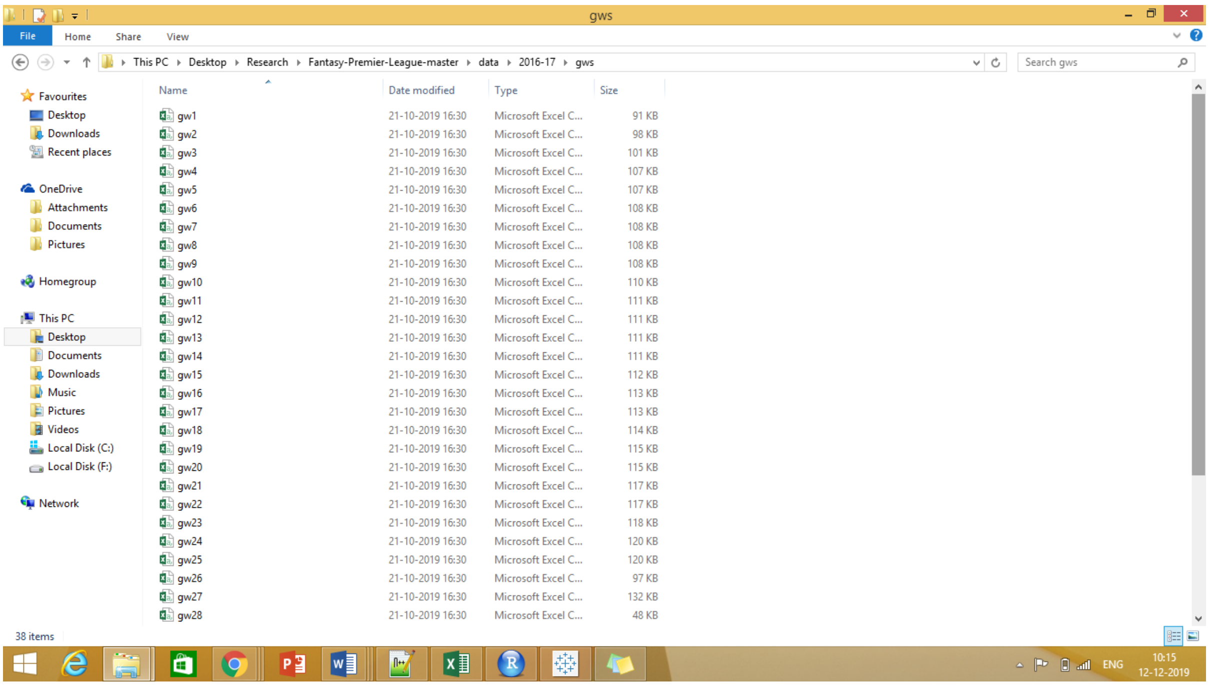


Figure 3: Gameweek wise datasets

Each week had its own dataset (which can be seen in Figure 3) which had to be merged to form a combined season dataset. Also, the numerical factor of the team name had to be converted back to its character as the same 20 teams to not play the following season. The code for this can be seen in Figure 4.

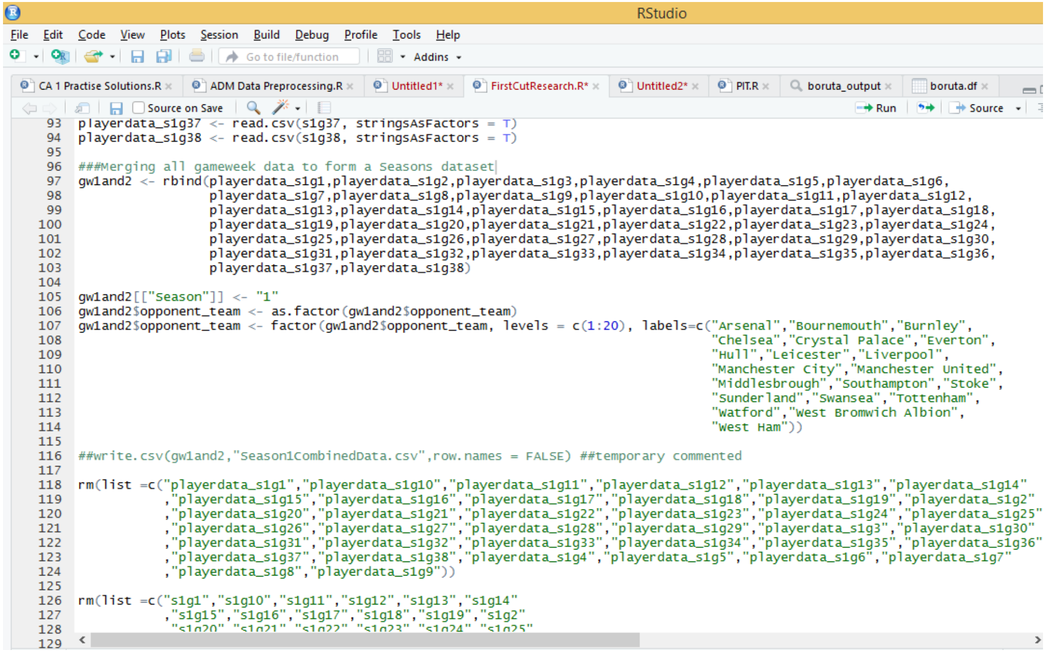


Figure 4: Merging Gameweek datasets

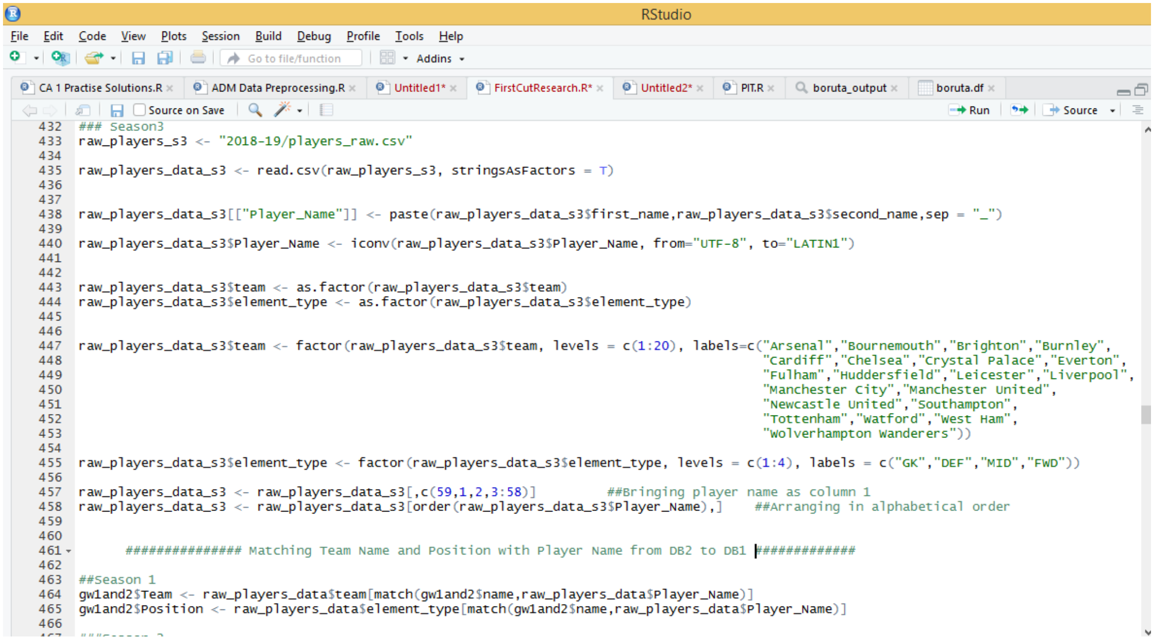


Figure 5: Matching Player Name from raw file to obtain Position

We then worked on the raw player dataset to obtain player position from the file. This would be done by matching player name and team. Hence, we had to make a naming conversion from ‘UTF-8’ to ‘LATIN1’ to handle the special characters.

Once the data from Dataset 1 had been processed, we initiated our work on dataset 2. Similar to working on the raw player file, we were going to match multiple parameters to pull the underlying stats of Expected Goals (xG) and Expected Assists (xA). These parameters were: Player Name, Player Team, Player Opponent and Match Date. After a first round of attempting a match we found a lot of missing values in the combined dataset. On further inspection, it was observed that certain games had a difference of 1 day in their date and also the dataset provided by understat held player names as the popularly known names or nicknames while FPL had their official names. Hence, we had to pull the names from the two datasets and find a workaround on Excel (shown is Figure 6).

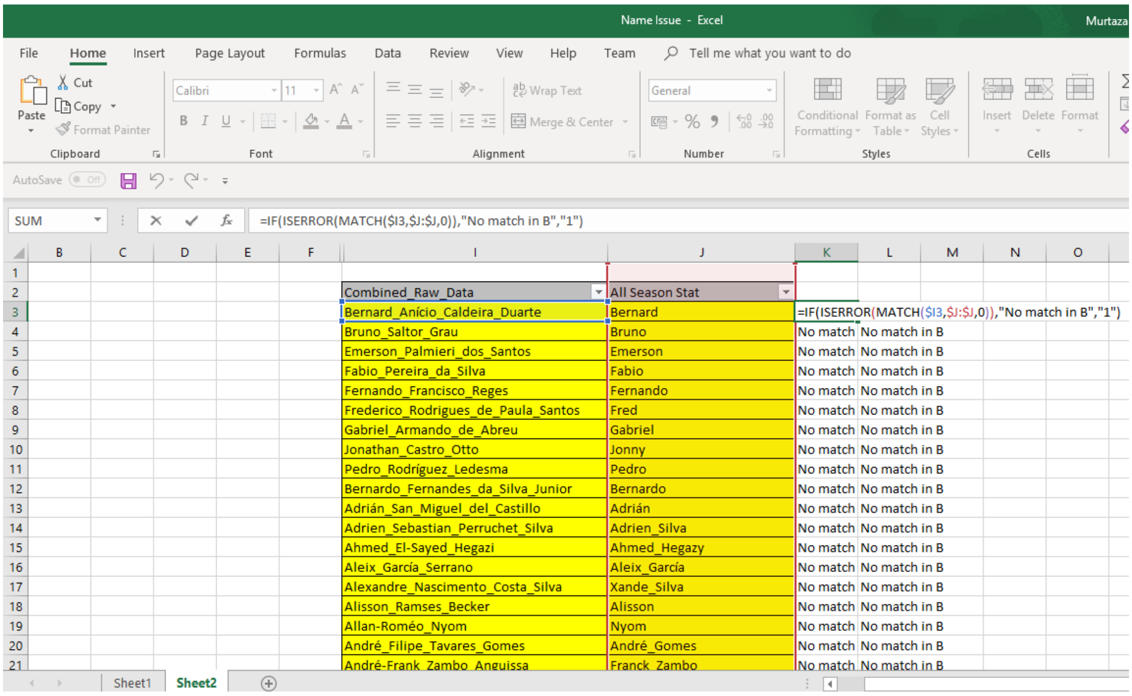


Figure 6: Player Name mismatch analysis in Excel

There were around 85 players with name mismatches which had to be then manually substituted by using the sub() function in RStudio.

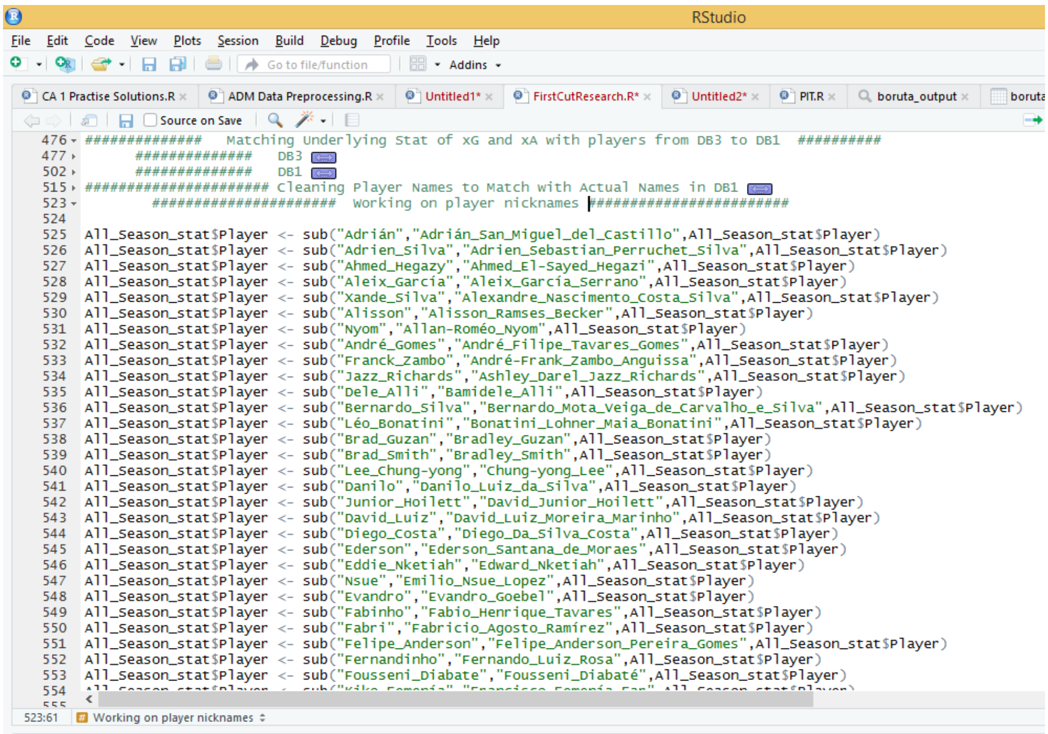
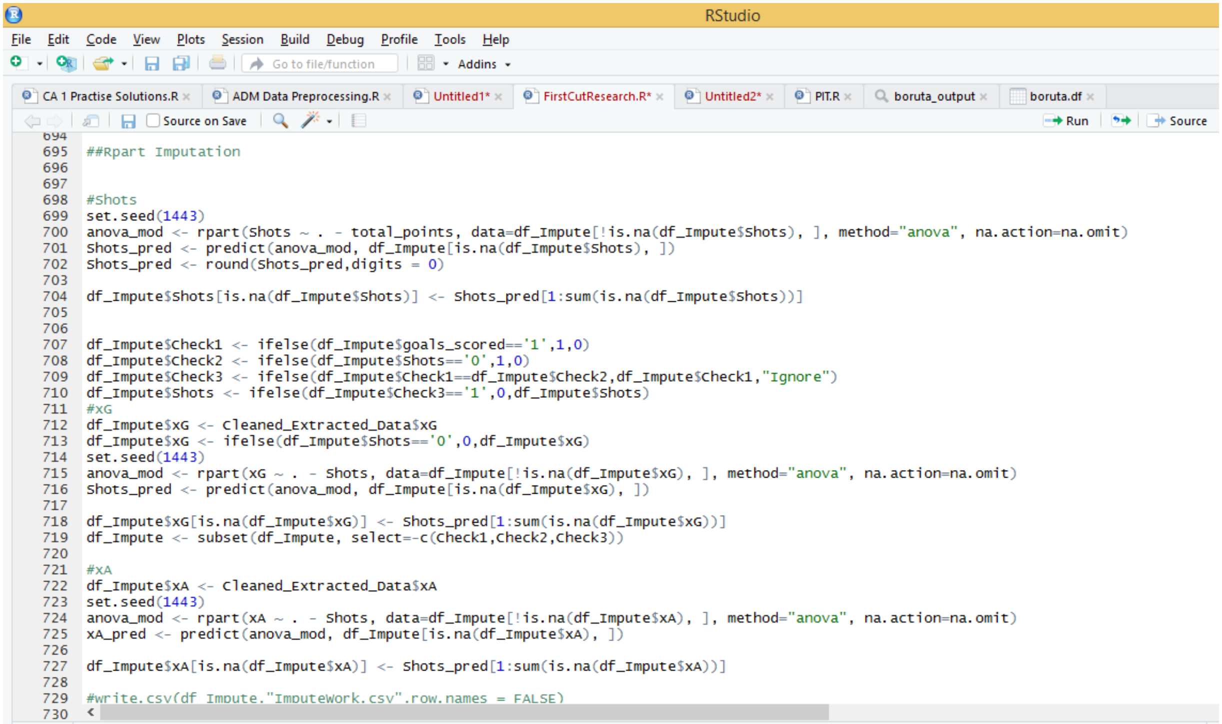


Figure 7: Player Name mismatch handling in RStudio

This was followed by data cleaning where duplicate rows and unwanted columns were removed. Player value did not have a decimal value and was hence divided by 10. Also, upon merging 2 columns highlighting key passes shared the same data but did not match in values. This was handled by assigning a new column which held the maximum of the two on comparison. A new parameter called player form was introduced as it was not present in the dataset. As per FPL, the player form is the average of the points earned in the last 30 days (roughly 4 gameweeks). Missing values in xG, xA and shots were observed after merging datasets. This was handled by assigning 0 value to players who did not play a game on that day. As xG and xA are sum of the probabilities of shots being converted to goals and key passes being possible assists respectively. A value of 1 was placed to any remaining player with missing data in shots if he had scored a goal. The remaining values were then imputed by prediction using the ANOVA method and the rpart function which can be observed in Figure 8.

2.2 Data Transformation and Feature Selection




```r
##Rpart Imputation


#Shots
set.seed(1443)
anova_mod <- rpart(Shots ~ . - total_points, data=df_Impute[!is.na(df_Impute$Shots), ], method="anova", na.action=na.omit)
Shots_pred <- predict(anova_mod, df_Impute[is.na(df_Impute$Shots), ])
Shots_pred <- round(Shots_pred,digits = 0)

df_Impute$Shots[is.na(df_Impute$Shots)] <- Shots_pred[1:sum(is.na(df_Impute$Shots))]


df_Impute$Check1 <- ifelse(df_Impute$goals_scored=='1',1,0)
df_Impute$Check2 <- ifelse(df_Impute$Shots=='0',1,0)
df_Impute$Check3 <- ifelse(df_Impute$Check1==df_Impute$Check2,df_Impute$Check1,"Ignore")
df_Impute$Shots <- ifelse(df_Impute$Check3=='1',0,df_Impute$Shots)
#xG
df_Impute$xG <- Cleaned_Extracted_Data$xG
df_Impute$xG <- ifelse(df_Impute$Shots=='0',0,df_Impute$xG)
set.seed(1443)
anova_mod <- rpart(xG ~ . - Shots, data=df_Impute[!is.na(df_Impute$xG), ], method="anova", na.action=na.omit)
Shots_pred <- predict(anova_mod, df_Impute[is.na(df_Impute$xG), ])

df_Impute$xG[is.na(df_Impute$xG)] <- Shots_pred[1:sum(is.na(df_Impute$xG))]
df_Impute <- subset(df_Impute, select=-c(Check1,Check2,Check3))

#xA
df_Impute$xA <- Cleaned_Extracted_Data$xA
set.seed(1443)
anova_mod <- rpart(xA ~ . - Shots, data=df_Impute[!is.na(df_Impute$xA), ], method="anova", na.action=na.omit)
xA_pred <- predict(anova_mod, df_Impute[is.na(df_Impute$xA), ])

df_Impute$xA[is.na(df_Impute$xA)] <- Shots_pred[1:sum(is.na(df_Impute$xA))]

#write.csv(df_Impute,"ImputeWork.csv",row.names = FALSE)
```

Figure 8: Handling missing values

Once the missing data had been taken care of, we worked on utilizing the in-game statistics in our dataset. As we cannot use these values to predict that particular entry, we make a summation of all its previous values and shift it to the next week entry where it acts as historical data. This transformation can be observed in Figure 9.

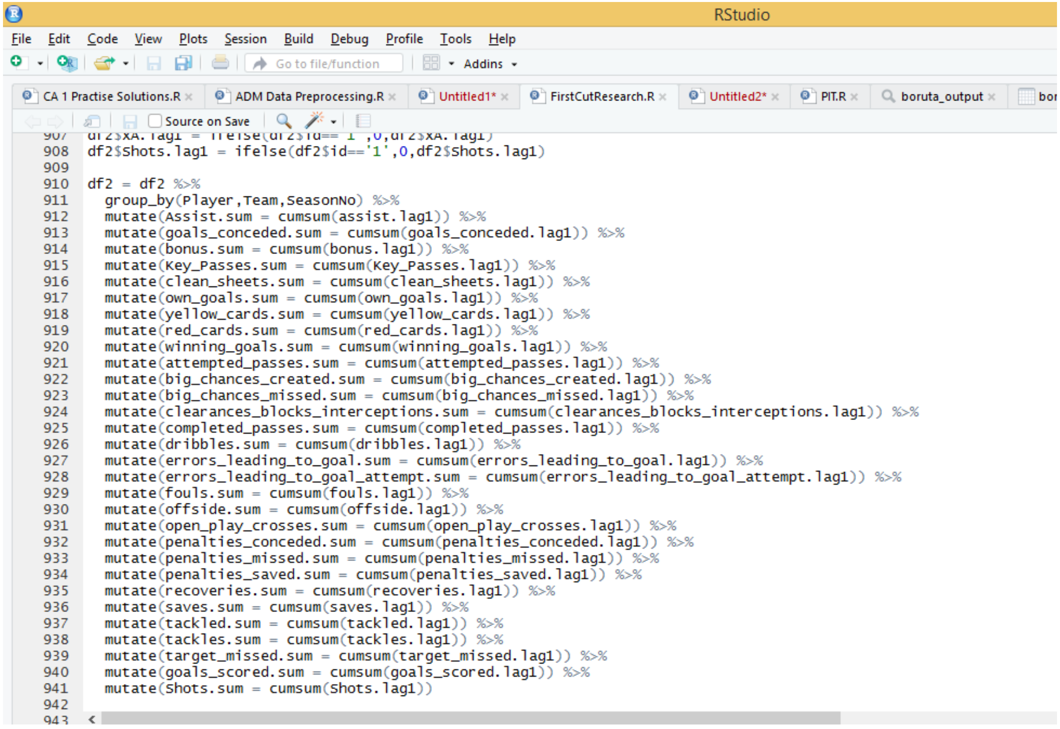


Figure 9: Summation of in-game statistics used as historic data

The dataset had been completely processed and ready to use for modelling, but we had to ensure we did appropriate feature selection and hence applied Boruta algorithm on around 50 attributes which rejected only “penalties missed” and accepted all the other parameters. Below is the plot of the analysis. As we can see in Figure 11, the underlying statistics of FPL i.e, Influence, Threat, Creativity and ICT index are all important parameters. While xG and xA have a relatively low importance, we hold on to them and remove the 9 least important variables including the rejected variable which can be seen in figure 10.

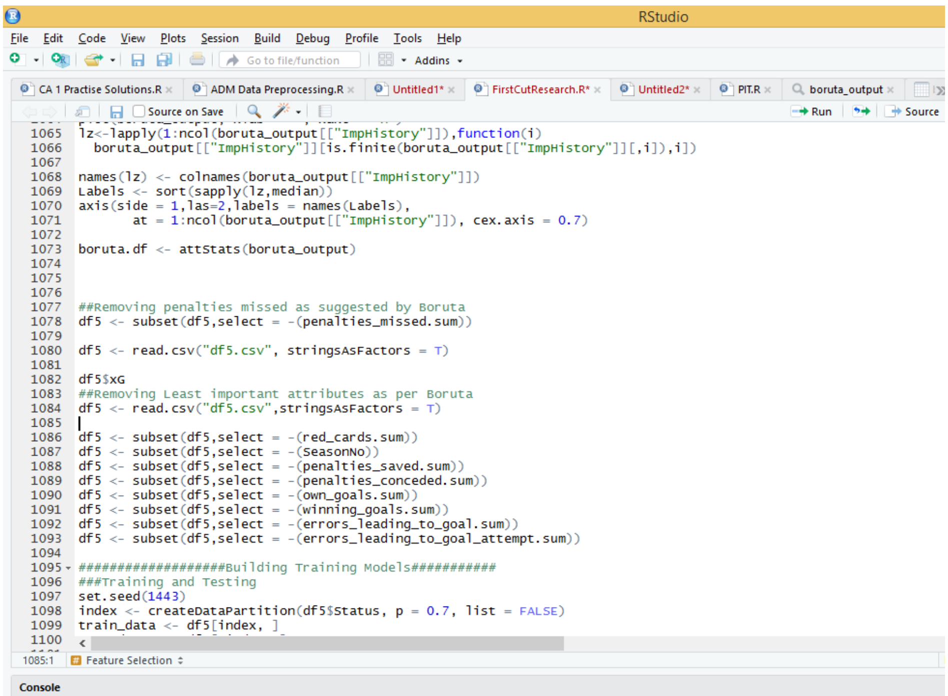


Figure 10: Removal of attributes based on Boruta Algorithm

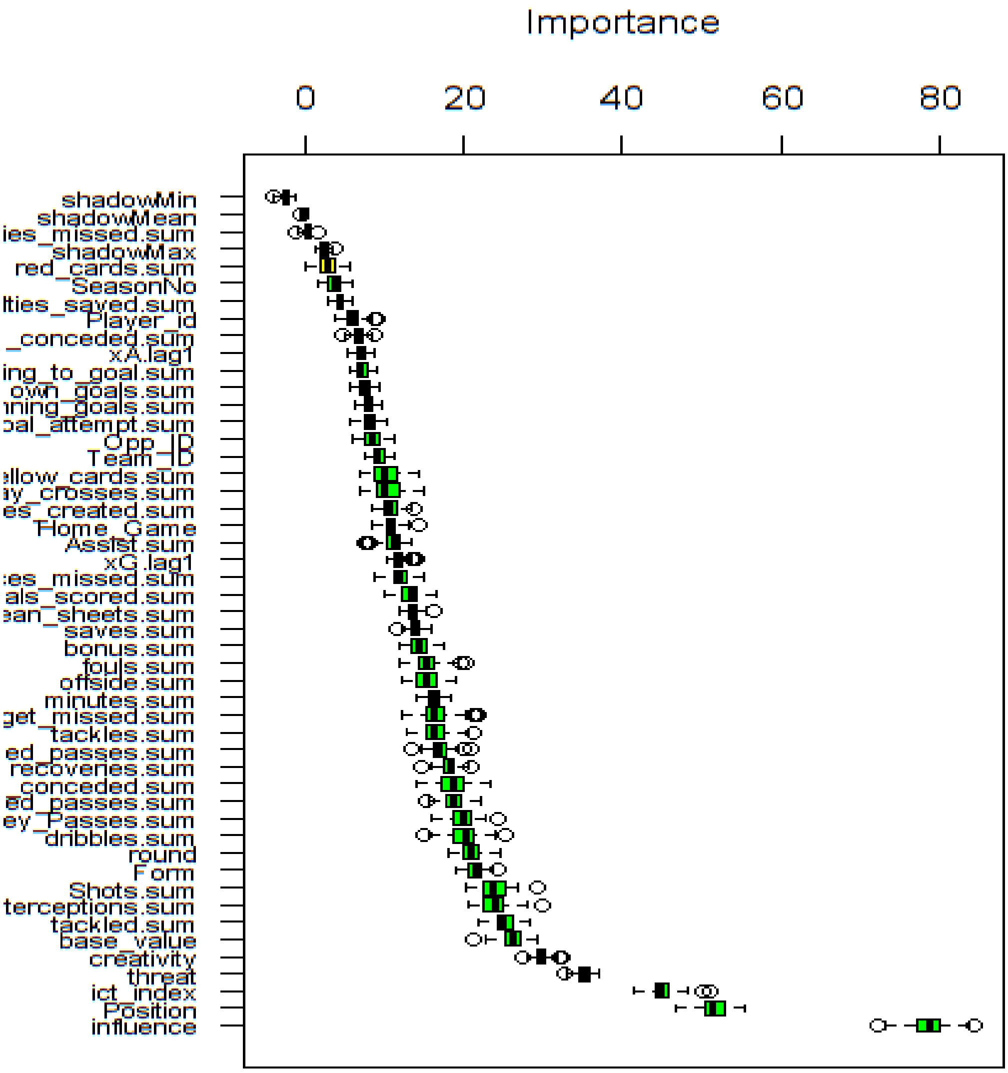


Figure 11: Boruta Algorithm plot

3 Modelling

We have applied 2 cases of Random Forest and XGBoost with and without underlying stats. Also 4 different types of sampling have been done to handle class imbalance. Validation of training data sets have been done by k-fold validation.

Sampling

```
###################Building Training Models###########
###Training and Testing
set.seed(1443)
index <- createDataPartition(df5$Status, p = 0.7, list = FALSE)
train_data <- df5[index, ]
test_data  <- df5[-index, ]
df6 <- df5

df6 <- subset(df6,select= -(xG))
df6 <- subset(df6,select= -(xA))

set.seed(1443)
index <- createDataPartition(df5$Status, p = 0.7, list = FALSE)
train_data <- df6[index, ]
test_data  <- df6[-index, ]
###Handling Class Imbalance

#over sampling
data_balanced_over <- ovun.sample(Status ~ ., data = train_data, method = "over",N = 60940)$data  ##N = no. of rows
table(data_balanced_over$Status)

#Under Sampling
data_balanced_under <- ovun.sample(Status ~ ., data = train_data, method = "under", N = 9922, seed = 1443)$data
table(data_balanced_under$Status)

#Combination of undersampling and oversampling
data_balanced_both <- ovun.sample(Status ~ ., data = train_data, method = "both", p=0.5,
table(data_balanced_both$Status)

##Using Rose function for handling class imbalance
data_balanced_rose <- ROSE(Status ~ ., data = train_data, seed = 1)$data
table(data.rose$Status)
```

Random Forest

```
##############RANDOM FOREST ################


###Creating Training and Testing Data

set.seed(1443)
index <- createDataPartition(df5$Status, p = 0.7, list = FALSE)
train_data <- df5[index, ]
test_data  <- df5[-index, ]

table(train_data$Status) ##Clear case of Class Imbalance

###Handling Class Imbalance

#over sampling
data_balanced_over <- ovun.sample(Status ~ ., data = train_data, method = "over",N = 60886)$data  ##N = no. of rows i
table(data_balanced_over$Status)

#Under Sampling
data_balanced_under <- ovun.sample(Status ~ ., data = train_data, method = "under", N = 9976, seed = 1443)$data
table(data_balanced_under$Status)

#Combination of undersampling and oversampling
data_balanced_both <- ovun.sample(Status ~ ., data = train_data, method = "both", p=0.5,
table(data_balanced_both$Status)

##Using Rose function for handling class imbalance
data.rose <- ROSE(Status ~ ., data = train_data, seed = 1)$data
table(data.rose$Status)


####Cross Validation
```

```
####Cross Validation


set.seed(1443)
inTrain_over = createDataPartition(data_balanced_over$Status, p = 0.05, list = F)
inTrain_under = createDataPartition(data_balanced_under$Status, p = 0.05, list = F)
inTrain_both = createDataPartition(data_balanced_both$Status, p = 0.05, list = F)
inTrain_rose = createDataPartition(data.rose$Status, p = 0.05, list = F)

crossv = train_data[-inTrain_over, ]
crossv = train_data[-inTrain_under, ]
crossv = train_data[-inTrain_both, ]
crossv = train_data[-inTrain_rose, ]

training2over = data_balanced_over[inTrain_over, ]
training2under = data_balanced_under[inTrain_under, ]
training2both = data_balanced_both[inTrain_both, ]
training2rose = data.rose[inTrain_rose, ]

####Training validated Models
modover = suppressMessages(
  train(Status ~ ., method = "rf", data = training2over,
        trControl = trainControl(method = "cv"), number = 25)
)

modover$finalModel

modunder = suppressMessages(
  train(Status ~ ., method = "rf", data = training2under,
        trControl = trainControl(method = "cv"), number = 25)
)

modunder$finalModel

modboth = suppressMessages(
```

```
####Training validated Models
modover = suppressMessages(
  train(Status ~ ., method = "rf", data = training2over,
        trControl = trainControl(method = "cv"), number = 25)
)

modover$finalModel

modunder = suppressMessages(
  train(Status ~ ., method = "rf", data = training2under,
        trControl = trainControl(method = "cv"), number = 25)
)

modunder$finalModel

modboth = suppressMessages(
  train(Status ~ ., method = "rf", data = training2both,
        trControl = trainControl(method = "cv"), number = 25)
)

modboth$finalModel

modrose = suppressMessages(
  train(Status ~ ., method = "rf", data = training2rose,
        trControl = trainControl(method = "cv"), number = 25)
)

modrose$finalModel
```

XGBoost

```
#####XGBOOST#######

new_train_over <- model.matrix(~ . + 0, data = data_balanced_over[, 1:37])   ###Conversion to Matrix required for XGB
new_train_under <- model.matrix(~ . + 0, data = data_balanced_under[, 1:37])   ###Conversion to Matrix required for X
new_train_both <- model.matrix(~ . + 0, data = data_balanced_both[, 1:37])   ###Conversion to Matrix required for XGB
new_train_rose <- model.matrix(~ . + 0, data = data.rose[, 1:37])   ###Conversion to Matrix required for XGBoost


new_test <- model.matrix(~ . + 0, data = test_data[, 1:37])

xgb_train_over <- xgb.DMatrix(data = new_train_over, label = data_balanced_over$Status) ###Preparing Matrices
xgb_train_under <- xgb.DMatrix(data = new_train_under, label = data_balanced_under$Status) ###Preparing Matrices
xgb_train_both <- xgb.DMatrix(data = new_train_both, label = data_balanced_both$Status) ###Preparing Matrices
xgb_train_rose <- xgb.DMatrix(data = new_train_rose, label = data.rose$Status) ###Preparing Matrices
xgb_test <- xgb.DMatrix(data = new_test, label = test_data$Status)

# Set parameters(default)
params <- list(booster = "gbtree", objective = "multi:softprob", num_class = 6, eval_metric = "mlogloss")

# Applying 10 folds for cross-validation
xgbcv_over <- xgb.cv(params = params, data = xgb_train_over, nrounds = 100, nfold = 10, showsd = TRUE,
                   stratified = TRUE, print_every_n = 10, early_stop_round = 20, maximize = FALSE, prediction = TRUE)
xgbcv_under <- xgb.cv(params = params, data = xgb_train_under, nrounds = 100, nfold = 10, showsd = TRUE,
                    stratified = TRUE, print_every_n = 10, early_stop_round = 20, maximize = FALSE, prediction = TRU
xgbcv_both <- xgb.cv(params = params, data = xgb_train_both, nrounds = 100, nfold = 10, showsd = TRUE,
                   stratified = TRUE, print_every_n = 10, early_stop_round = 20, maximize = FALSE, prediction = TRUE)
xgbcv_rose <- xgb.cv(params = params, data = xgb_train_rose, nrounds = 100, nfold = 10, showsd = TRUE,
                    stratified = TRUE, print_every_n = 10, early_stop_round = 20, maximize = FALSE, prediction = TRUE

##Prediction and Confusion Matrix of Oversampling Training
OOF_prediction_over <- data.frame(xgbcv_over$pred) %>%
  mutate(max_prob = max.col(., ties.method = "last"),
         label = data_balanced_over$Status + 1)
```

```
##Prediction and Confusion Matrix of Oversampling Training
OOF_prediction_over <- data.frame(xgbcv_over$pred) %>%
  mutate(max_prob = max.col(., ties.method = "last"),
         label = data_balanced_over$Status + 1)

confusionMatrix(factor(OOF_prediction_over$max_prob),
                factor(OOF_prediction_over$label),
                mode = "everything")

##Prediction and Confusion Matrix of Undersampling Training
OOF_prediction_under <- data.frame(xgbcv_under$pred) %>%
  mutate(max_prob = max.col(., ties.method = "last"),
         label = data_balanced_under$Status + 1)

confusionMatrix(factor(OOF_prediction_under$max_prob),
                factor(OOF_prediction_under$label),
                mode = "everything")

##Prediction and Confusion Matrix of Both Training
OOF_prediction_both <- data.frame(xgbcv_both$pred) %>%
  mutate(max_prob = max.col(., ties.method = "last"),
         label = data_balanced_both$Status + 1)

confusionMatrix(factor(OOF_prediction_both$max_prob),
                factor(OOF_prediction_both$label),
                mode = "everything")

##Prediction and Confusion Matrix of Rose Training
OOF_prediction_rose <- data.frame(xgbcv_rose$pred) %>%
  mutate(max_prob = max.col(., ties.method = "last"),
         label = data.rose$Status + 1)

confusionMatrix(factor(OOF_prediction_rose$max_prob),
                factor(OOF_prediction_rose$label),
```