Football match winner prediction

Roger Comas Triadó

February 2020

Contents

1 Summary

This report is from a project for the **Capstone Project: IDV Learners** from the *HardvardX: PH125.9x Data Science: Capstone* course. This is the last course of the **Data Science Professional Certificate** of **HarvardX** in the edX online learning platform.

In this project, we are told to choose our own database to make the project. This project has to have the following parts:

- an introduction/overview/executive summary section that describes the dataset and variables, and summarizes the goal of the project and key steps that were performed;
- a methods/analysis section that explains the process and techniques used, including data cleaning, data exploration and visualization, insights gained, and your modeling approaches (you must use at least two different models or algorithms);
- a results section that presents the modeling results and discusses the model performance; and
- a conclusion section that gives a brief summary of the report, its potential impact, its limitations, and future work.

So this is what this project is going to be about.

1.1 Introduction

My name is Roger and I am 25 years old. I love watching football and playing it. I started playing this sport when I was 7 years old and I am still playing it in a semi-professional team in the 6th Divison of Spain. As football is my favourite hobbie, my project is going to be about football and its data analytics.

1.2 Database explanation

The database I selected for this project is one from the *Kaggle* website, called *European Soccer Database*. This database contains data of: - +25.000 matches - +10.000 players - 11 European Countries with their lead championship - Seasons 2008 to 2016 - Players and Teams' attributes sourced from EA Sports' FIFA video game series, including the weekly updates - Team line up with squad formation (X, Y coordinates) - Betting odds from up to 10 providers - Detailed match events (goal types, possession, corner, cross, fouls, cards etc...) for +10,000 matches

You can find the database in the following link: https://www.kaggle.com/hugomathien/soccer

The file you can download is a .sqlite file with all the following tables: - Country (11 x 2): contains the country name and its code - League (11 x 3): contains the league name, its code and the country where the league is from - Match (26.0k x 115): each row is a match with a lot of variables - Player (11.1k x 7): each row is a player with some variables like name, weight, height, birthday, etc. - Player_Attributes (184k x 42): contains all the player attributes of the FIFA game of each player - Team (299 x 5): contains information of all the teams of the 11 leagues - Team_Attributes (1458 x 25): contains all the information of the FIFA features for each team

1.3 Goal of the project

The goal I selected for this project is to find a model that predicts if the home team is going to win the match or if the away team is going to win the match. As I am from Spain and it is easier for me to understand the data, because I strongly know all the teams, I chose only to do the modelling for the Spanish league in this project.

2 Data exploration and Data analysis

2.1 Loading data

The first thing to do in this project is download the database and load it in Rstudio:

```
# Import libraries
library(tidyverse)
library(stringr)
library(caret)
library(RSQLite)
library(corrplot)
library(gridExtra)
# Load the database
database = dbConnect(RSQLite::SQLite(), dbname = "database.sqlite")
# Create the tables
matches <- as.data.frame(tbl(database, "Match"))
league <- as.data.frame(tbl(database,"League"))
country <- as.data.frame(tbl(database, "Country"))
teams <- as.data.frame(tbl(database, "Team"))
players <- as.data.frame(tbl(database, "Player"))
```

2.2 Data exploration

Now, there are five datasets loaded. The FIFA attributes are not loaded, which were in the *Player_Attributes* and the *Team_Attributes* of the .sqlite database, as this information is not going to be used for the prediction. We are only going to use real data, such as the matches, league, country, teams and players information.

As explained in the *Database explanation* in the league, country, teams and players tables only contain information of these different fields, but the really important table is the matches. This table contains a lot of variables:

```
ncol(matches)
```

```
## [1] 115
```

So there are 115 columns. Many of these variables could be very useful, but there is no information in most of the rows, so they are useless. To fix this, I am going to create a new table called "match_result" with the most important variables:

```
names(match_result)
```

```
##  [1] "id"               "country_id"       "league_id"        "season"
##  [5] "date"             "match_api_id"     "home_team_api_id" "away_team_api_id"
##  [9] "home_team_goal"   "away_team_goal"   "home_team_win"    "draw"
## [13] "away_team_win"    "result"
```

In the match_result, I have created new variables that are going to be useful:

- **home_team_win**: variable that is a 1 if the home team wins and a 0 in any other cases.
- **draw**: variable that is a 1 if there is a draw and a 0 in any other cases.

- **away_team_win**: variable that is a 1 if the away team wins and a 0 in any other cases.
- **result**: variable that is a "H" if the home team wins, "D" if there is a draw or "A" if it is the away team that wins.

Now that there is a table with all the relevant information, it is time to start looking to the goal of the project. In the football industry, it is known that if a team plays at his stadium there are more probabilities of winning than if they play away. Let's have a look at this information from our dataset:

```
match_result %>% summarize(perc_home_w = sum(home_team_win)/n(), perc_draw = sum(draw)/n() , perc_away_
```

```
##   perc_home_w perc_draw perc_away_w
## 1   0.4587167 0.2538974    0.287386
```

As seen from the last results, there is much more probabilities to win if a team plays home (46 %) than if it plays away (29%).

As it is known in the football industry, not all the teams win the same playing home or away: there are some teams that are very strong when playing home but they don't win any match when playing away, and there are the opposite ones that don't win a lot of games when playing home but then they win a lot when playing away. And there are also some teams that have a lot of draws during the season.

As well as the result can vary from playing home or away, the amount of goals scored or received can also vary a lot in a team when they play home or not.

It is also known that the same team does not act in the same way in two different seasons, as they can change the coach or some players.

The following steps are going to put this information into the datasets and I am going to create two databases: the home teams database and the away teams database. The two databases are going to have the information of the percentage of a win/draw/loss home and away, and the mean goals done and received.

We can see the head of the home team database:

```
head(team_database_home)
```

```
##   home_team_api_id team_long_name team_short_name    season perc_home_win
## 1             9987       KRC Genk             GEN 2008/2009     0.4117647
## 2             9987       KRC Genk             GEN 2009/2010     0.2857143
## 3             9987       KRC Genk             GEN 2010/2011     0.7333333
## 4             9987       KRC Genk             GEN 2011/2012     0.6666667
## 5             9987       KRC Genk             GEN 2012/2013     0.6666667
## 6             9987       KRC Genk             GEN 2014/2015     0.5333333
##   perc_home_draw mean_goals_h
## 1     0.29411765     1.352941
## 2     0.50000000     1.142857
## 3     0.06666667     2.466667
## 4     0.20000000     2.400000
## 5     0.20000000     2.400000
## 6     0.33333333     1.666667
```

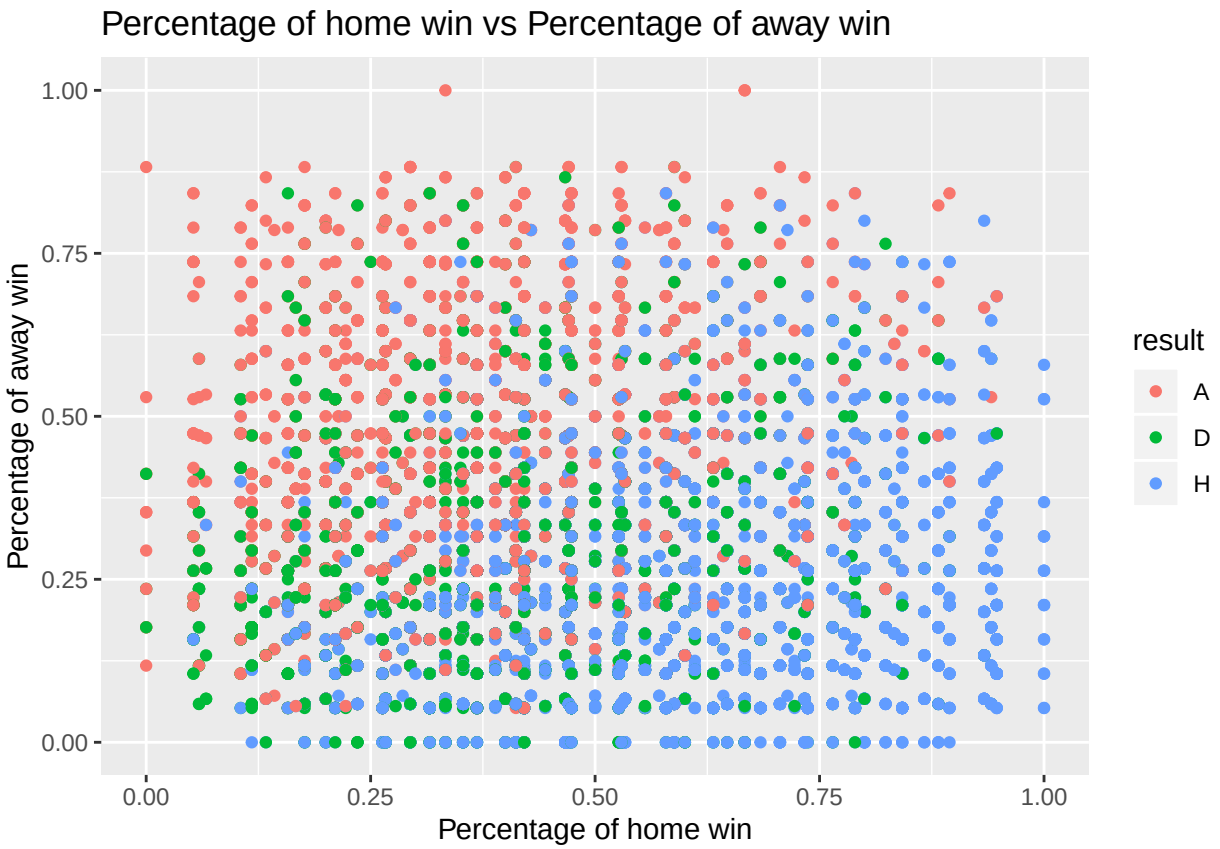
Finally all the information of these databases is going to be added in a final table with the informations of each match and all the information of the databases.

```r
match_final <- match_result %>% select(country_id,
                                       league_id,
                                       season,
                                       match_api_id,
                                       home_team_api_id,
                                       away_team_api_id,
                                       home_team_win,
                                       result) %>%
  left_join(team_database_home, by = c("season", "home_team_api_id")) %>%
  left_join(team_database_away, by = c("season", "away_team_api_id")) %>%
  select(country_id, league_id, season, match_api_id, home_team_api_id, perc_home_win, mean_goals_h, pe
head(match_final)
```

```
##   country_id league_id    season match_api_id home_team_api_id perc_home_win
## 1          1         1 2008/2009       492473             9987     0.4117647
## 2          1         1 2008/2009       492474            10000     0.5294118
## 3          1         1 2008/2009       492475             9984     0.5294118
## 4          1         1 2008/2009       492476             9991     0.5294118
## 5          1         1 2008/2009       492477             7947     0.2941176
## 6          1         1 2008/2009       492478             8203     0.3529412
##   mean_goals_h perc_home_draw away_team_api_id perc_away_win mean_goals_a
## 1     1.352941      0.2941176             9993     0.1176471    0.7058824
## 2     1.823529      0.3529412             9994     0.3529412    1.0588235
## 3     1.705882      0.1176471             8635     0.5882353    1.5882353
## 4     1.823529      0.1764706             9998     0.0000000    0.7058824
## 5     1.235294      0.2352941             9985     0.5294118    1.4117647
## 6     1.411765      0.4117647             8342     0.4117647    1.2941176
##   perc_away_draw result
## 1      0.3529412      D
## 2      0.2941176      D
## 3      0.2352941      A
## 4      0.1764706      H
## 5      0.2941176      A
## 6      0.1764706      D
```

We can use this table to plot some of the variables and see if they have some correlation: First, I will plot the percentage of home win against the percentage of away win, and color it depending on the result:

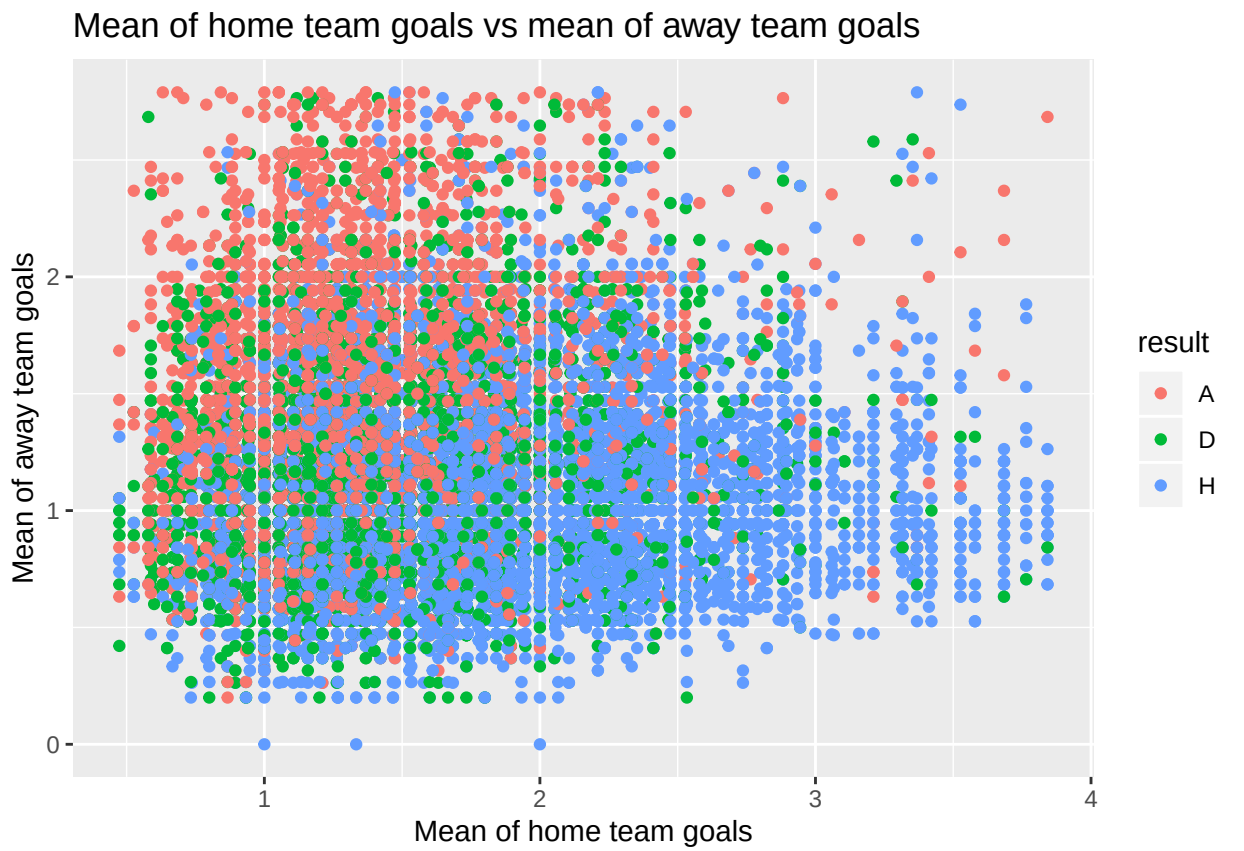
```r
match_final %>% ggplot(aes(perc_home_win, perc_away_win, color = result)) + geom_point() + ggtitle("Per
```



From this plot we can see two diferent areas: the "A" area (up left) and the "H" area (down right), and the "T" are all around the graphic. We can think that there is some correlation in this variables and the final result.

Now, let's plot the mean of home team goals against the mean of away team goals, and color it depending on the result:

```
match_final %>% ggplot(aes(mean_goals_h, mean_goals_a, color = result)) + geom_point() + ggtitle("Mean
```



Despite it is not as clear as the last plot, in this plot there are also two different color zones.

To analyze the data with more detail, as said in the *Summary* point, I am going to take only the Spanish league data.

2.3 Data Analysis

The dataset with the Spanish league information is the following:

```
head(laliga_match)
```

```
##      id country_id league_id    season                date match_api_id
## 1 21518      21518     21518 2008/2009 2008-08-30 00:00:00       530023
## 2 21519      21518     21518 2008/2009 2008-08-31 00:00:00       530084
## 3 21520      21518     21518 2008/2009 2008-08-31 00:00:00       530085
## 4 21521      21518     21518 2008/2009 2008-08-31 00:00:00       530086
## 5 21522      21518     21518 2008/2009 2008-08-31 00:00:00       530087
## 6 21523      21518     21518 2008/2009 2008-08-31 00:00:00       530088
##   home_team_api_id away_team_api_id home_team_goal away_team_goal home_team_win
## 1            10267             8661              3              0             1
## 2             8371            10205              1              1             0
## 3             9783             8633              2              1             1
## 4             8388             8634              1              0             1
## 5             8696             8302              1              1             0
## 6             9869             8305              1              2             0
##   draw away_team_win result
```

```
## 1    0              0      H
## 2    1              0      D
## 3    0              0      H
## 4    0              0      H
## 5    1              0      D
## 6    0              1      A
```

The percentage of home win, tie and away win is the following:

```
laliga_match %>% summarize(perc_home = sum(home_team_win/n()),
                                          perc_away = sum(away_team_win/n()),
                                          perc_draw = sum(draw/n()))
```

```
##   perc_home perc_away perc_draw
## 1 0.4884868 0.2799342 0.2315789
```

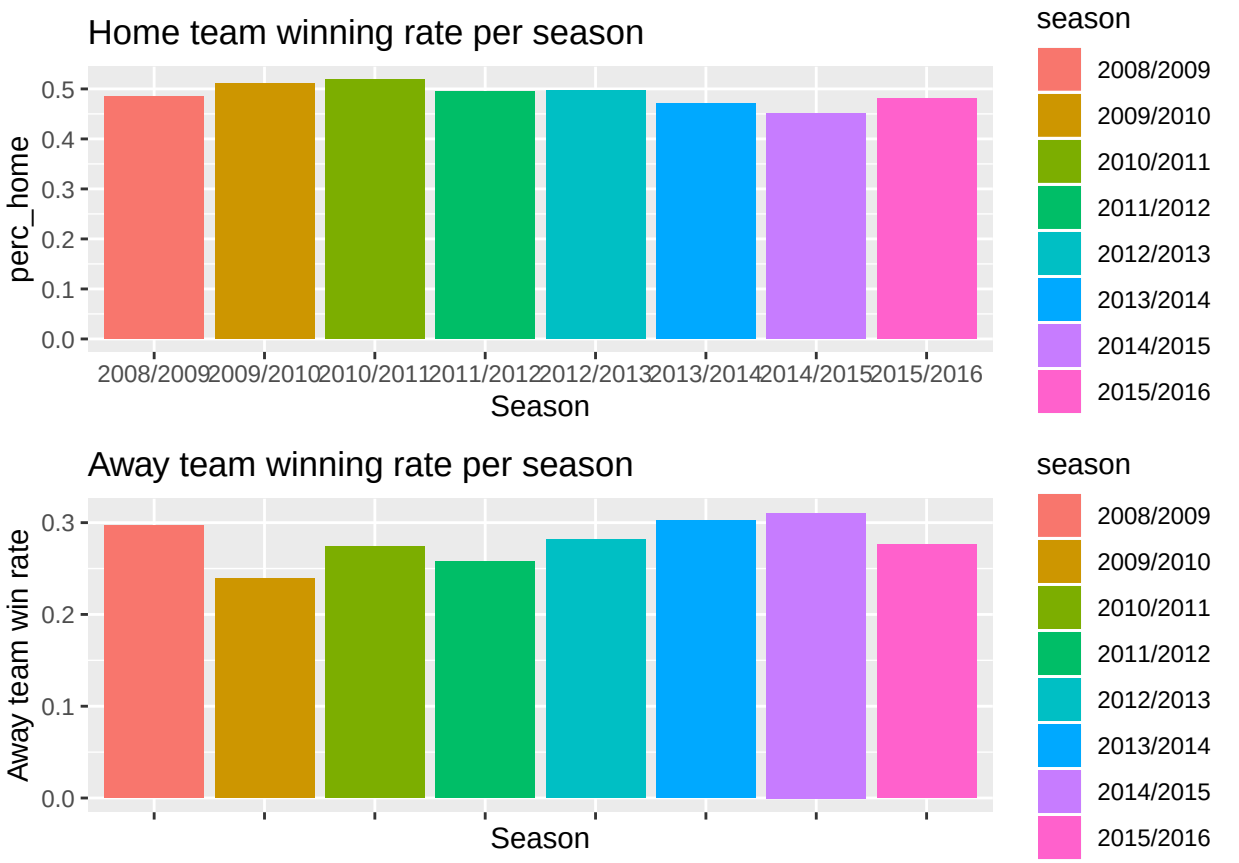
Now, I am going to plot the percentage of home team winning across the different seasons:

```
a <- laliga_match %>%
  group_by(season) %>%
  summarize(perc_home = sum(home_team_win/n()),
            perc_away = sum(away_team_win/n()),
            perc_draw = sum(draw/n())) %>%
  ggplot(aes(season,perc_home, fill = season)) +
  geom_bar(stat="identity") +
  ggtitle("Home team winning rate per season") +
  xlab("Season")
  ylab("Home team win rate") +
  theme(axis.text.x = element_blank())
```

```
## NULL
```

```
b <- laliga_match %>%
  group_by(season) %>%
  summarize(perc_home = sum(home_team_win/n()),
            perc_away = sum(away_team_win/n()),
            perc_draw = sum(draw/n())) %>%
  ggplot(aes(season,perc_away, fill = season)) +
  geom_bar(stat="identity") +
  ggtitle("Away team winning rate per season") +
  xlab("Season") +
  ylab("Away team win rate") +
  theme(axis.text.x = element_blank())

grid.arrange(a, b)
```



We see that the home team win rate changes depending on the season, but it is always between 45% and 50% more or less. It is more or less the same with the away team win rate.Despite it changes depending on the season it always stays between 25% and 30% approximately.

Now, I am going to create two databases: one for home teams and another one for away teams with some important new variables: percentage of win/tie/loss, mean of goals done and mean of goals received, and as an output I will only have the winning variable (1 if there is a win and 0 if there is not).

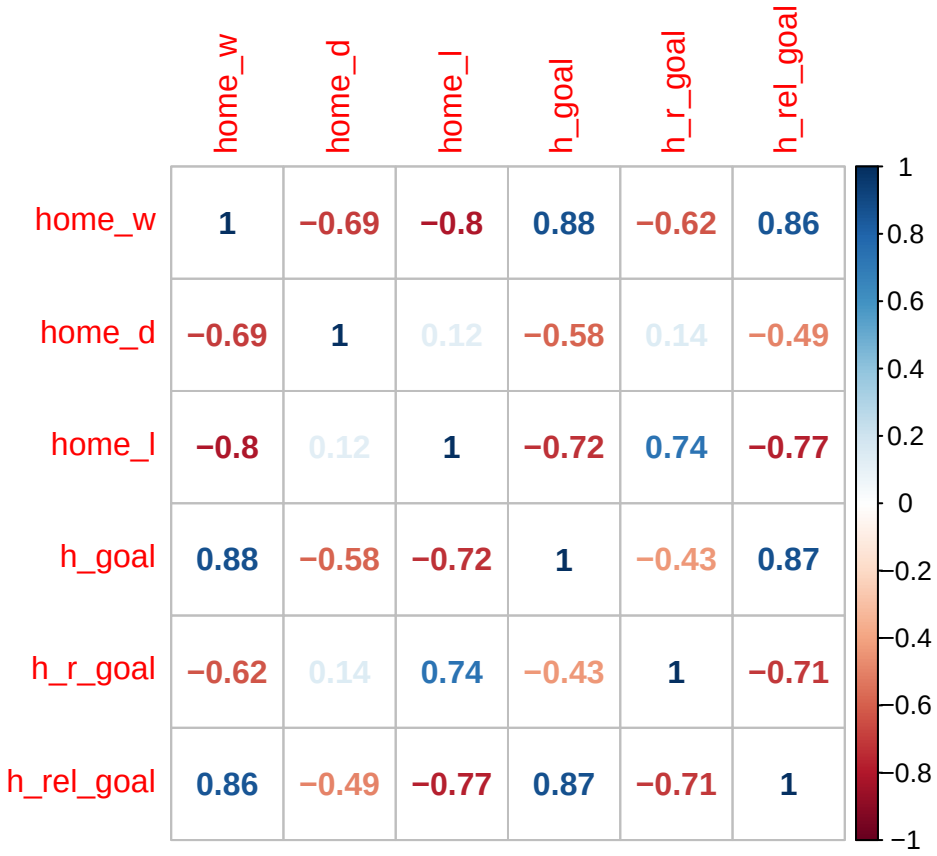
2.3.1 Home/away team database

In this section, a home team and an away team databases are created with the win/draw/loss ratio and the mean of goals scored and received for each team and season. I also create a new variable which is the ratio between the mean of goals scored and the mean of goals received for each team and season.

2.3.2 Home team database analysis

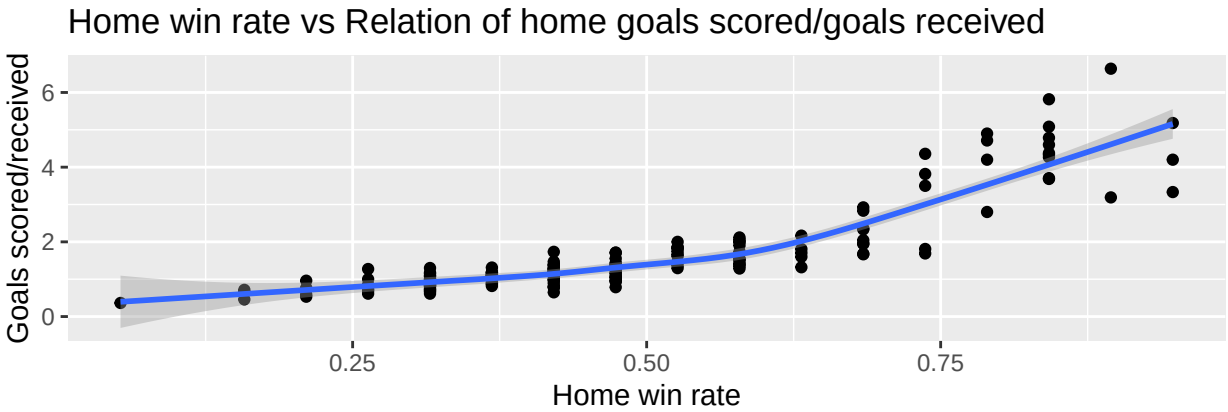
Let's see the correlation between the home teams variables:

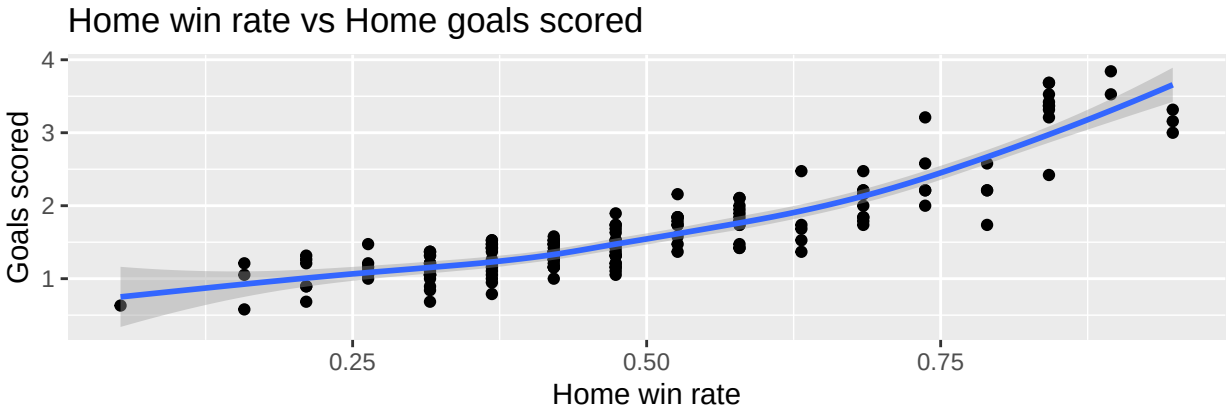
```
laliga_home_correlation <- cor(laliga_home_database[,3:8])
corrplot(laliga_home_correlation, method = "number")
```



As we see from the correlation matrix, the percentage of a home team win is very correlated to the home mean goal and the factor between the mean of the goals scored divided by the mean of the goals received. This is demonstrated in the following plots:

```
h_rel_goal <- laliga_home_database %>% ggplot(aes(home_w, h_rel_goal)) + geom_point() + geom_smooth() +
h_goal <-laliga_home_database %>% ggplot(aes(home_w, h_goal)) + geom_point() + geom_smooth() + ggtitle(
grid.arrange(h_rel_goal, h_goal)
```





As the winning ratios are very correlated to the goals scored and the relation between goals scored and goals received, let's create a new dataset with these variables.

After that, let's join all this information in one dataset with the information of the matches and the ratios of winning and goals explained in the last paragraph.

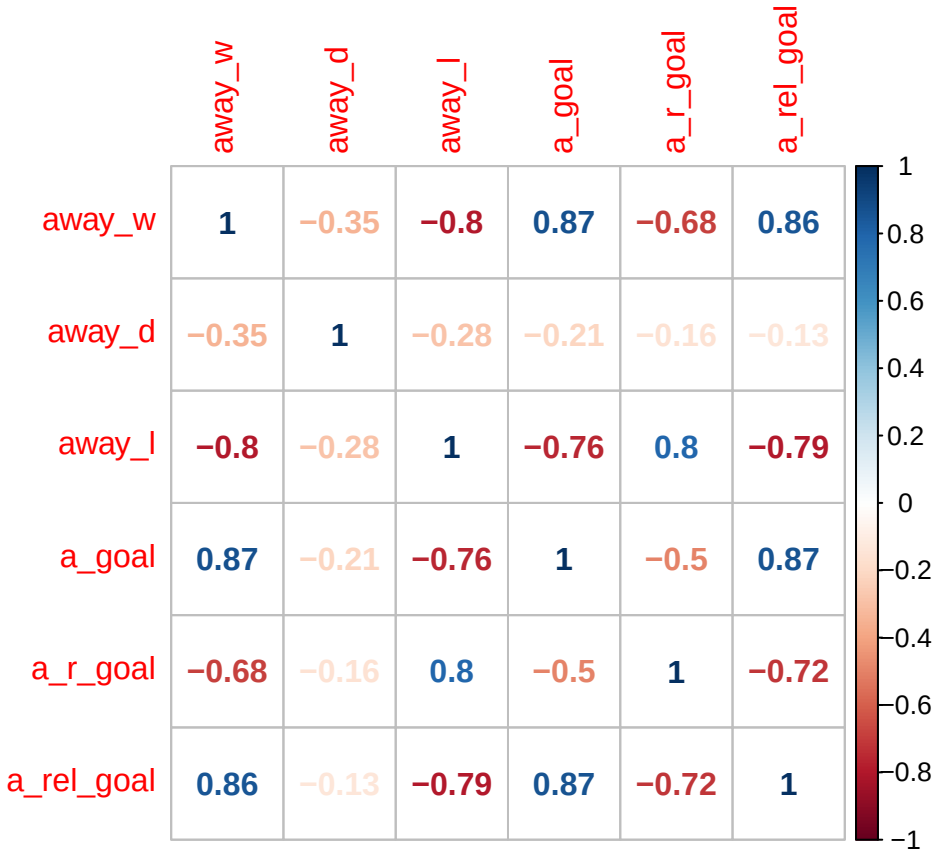
```
head(h_laliga_win)
```

```
##      season home_team_api_id away_team_api_id home_team_win    home_w   h_goal
## 1 2008/2009            10267             8661             1 0.6315789 2.473684
## 2 2008/2009             8371            10205             0 0.4210526 1.421053
## 3 2008/2009             9783             8633             1 0.5263158 1.578947
## 4 2008/2009             8388             8634             1 0.4736842 1.210526
## 5 2008/2009             8696             8302             0 0.2631579 1.473684
## 6 2008/2009             9869             8305             0 0.4210526 1.263158
##   h_rel_goal  h_r_goal    away_w   a_goal a_rel_goal a_r_goal
## 1  1.8076923 1.3684211 0.2631579 1.052632  0.5555556 1.894737
## 2  1.2272727 1.1578947 0.3157895 1.473684  0.9655172 1.526316
## 3  1.6666667 0.9473684 0.5789474 1.789474  1.4782609 1.210526
## 4  1.0454545 1.1578947 0.6842105 2.315789  2.0952381 1.105263
## 5  1.2727273 1.1578947 0.5263158 1.368421  1.3684211 1.000000
## 6  0.6486486 1.9473684 0.1578947 1.210526  0.6969697 1.736842
```

2.3.3 Away team database analysis

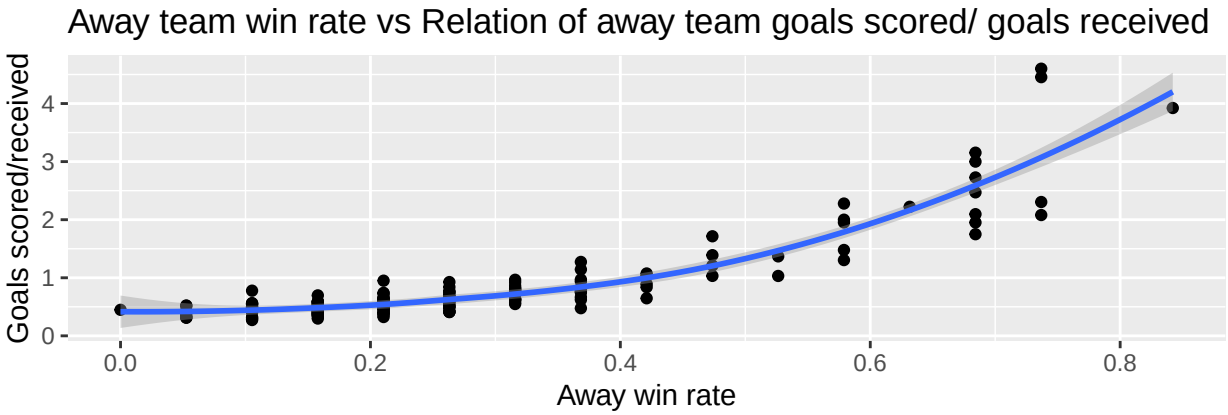
Now, let's see the correlation between the away teams variables:

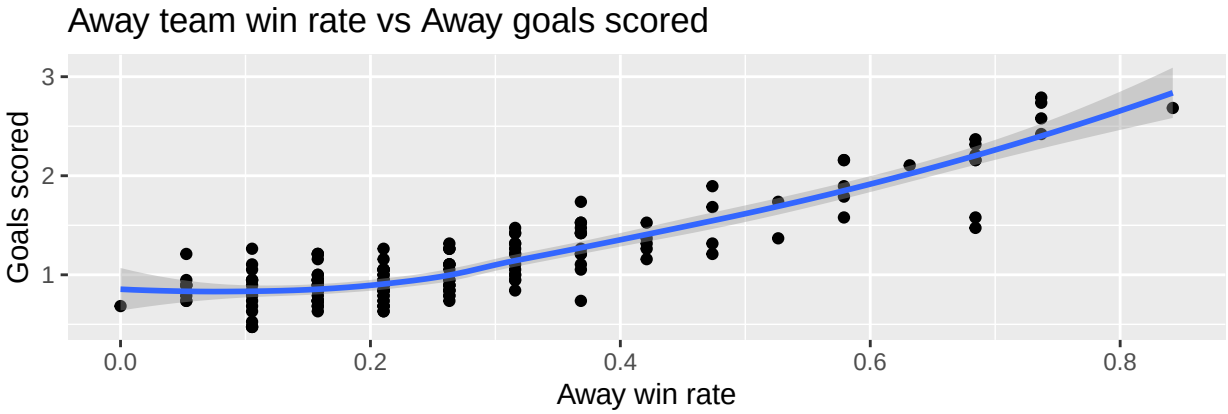
```
laliga_away_correlation <- cor(laliga_away_database[, 3:8])
corrplot(laliga_away_correlation, method = "number")
```



As we can see from this last correlation matrix, the percentage of an away team win is also very correlated to their mean goal scored and the factor of the mean of goals scored divided by the mean of the goals received. This is demonstrated in the following plots:

```
rel_goal <- laliga_away_database %>% ggplot(aes(away_w, a_rel_goal)) + geom_point() + geom_smooth() + g
goal <- laliga_away_database %>% ggplot(aes(away_w, a_goal)) + geom_point() + geom_smooth()  + ggtitle(
grid.arrange(rel_goal, goal)
```





As the winning ratios are very correlated to the goals scored and the relation between goals scored and goals received, let's create a new dataset with these variables.

After that, let's join all this information in one dataset with the information of the matches and the ratios of winning and goals explained in the last paragraph.

```
head(a_laliga_win)
```

```
##      season home_team_api_id away_team_api_id away_team_win    home_w   h_goal
## 1 2008/2009            10267             8661             0 0.6315789 2.473684
## 2 2008/2009             8371            10205             0 0.4210526 1.421053
## 3 2008/2009             9783             8633             0 0.5263158 1.578947
## 4 2008/2009             8388             8634             0 0.4736842 1.210526
## 5 2008/2009             8696             8302             0 0.2631579 1.473684
## 6 2008/2009             9869             8305             1 0.4210526 1.263158
##   h_rel_goal  h_r_goal    away_w   a_goal a_rel_goal a_r_goal
## 1  1.8076923 1.3684211 0.2631579 1.052632  0.5555556 1.894737
## 2  1.2272727 1.1578947 0.3157895 1.473684  0.9655172 1.526316
## 3  1.6666667 0.9473684 0.5789474 1.789474  1.4782609 1.210526
## 4  1.0454545 1.1578947 0.6842105 2.315789  2.0952381 1.105263
## 5  1.2727273 1.1578947 0.5263158 1.368421  1.3684211 1.000000
## 6  0.6486486 1.9473684 0.1578947 1.210526  0.6969697 1.736842
```

With these two datasets created, it is time to create the prediction models.

3 Model creation

In this section I am going to create two models: one for the home team winning prediction and another one for the away team winning prediction.

For each model I will use as a variables the win/draw/loss ratios for each team and the mean goals scored, received and the ratio between goals scored/goals received, to predict the output variable:

- Home model: the output variable will be a **1** if the home team wins the match and the output will be a **0** if there is a draw or the away team wins.
- Away model: he output variable will be a **1** if the away team wins the match and the output will be a **0** if there is a draw or the home team wins.

The two models have to use a classification algorithm: with the input variables, the models should classify the output in a 1 or 0, with its meaning depending on the model. In this project the classification models that will be used are:

- K-Nearest Neighbors (knn)
- Random forest (rf)
- Naive Bayes (nb)
- Support Vector Machine (svm)
- Kernel Support Vector Machine (ksvm)

For each model, all of these models will be used with different samples to find the different accuracies of each model to find the best algorithm for each model.

3.1 Home team model

In this section I am going to create a function in order to select the best algorithm for the home team winning prediciton model. As explained in the previous point, with this function I am going to evaluate the model with 5 different algorithms (knn, rf, nb, svm, ksvm) and measure the accuracy of each of the algorithms. The accuracy of each algorithm depends on the training and testing set used, so the function will run each algorithm as many times as we want (using different "set.seed()" functions) and save each of these accuracies for each model and each time. The function used for the home team model is the following:

```
# initialize the dataframes used in the function
h_model1 = data.frame()
h_model2 = data.frame()
h_model3 = data.frame()
h_model4 = data.frame()
h_model5 = data.frame()
h_total_avg = data.frame()

#Home win predictor function

model = function(x){
  set.seed(x)
  #Create the validation dataset
  h_validation_index <- createDataPartition(h_laliga_win$home_team_win, times = 1, p= 0.2, list= FALSE)
  h_validation_set <- h_laliga_win[h_validation_index,]
  h_working_set <- h_laliga_win[-h_validation_index, ]
```

```
    #Create the training and testing datasets
  h_testing_index <- createDataPartition(h_working_set$home_team_win, times = 1, p = 0.2, list = FALSE)
  h_testing_set <- h_working_set[h_testing_index,]
  h_training_set <- h_working_set[-h_testing_index,]


  #First Model
  train_x <- h_training_set[,5:12]
  train_y <- as.factor(h_training_set[,4])
  test_x <- h_testing_set[,5:12]
  test_y <- as.factor(h_testing_set[,4])

  fit_knn <- train(train_x, train_y, method = "knn")
  y_hat_knn <- predict(fit_knn, test_x)
  h_testing_set1 <- h_testing_set %>% bind_cols(y_hat_knn = y_hat_knn)

  fit_rf <- train(train_x, train_y , method ="rf")
  y_hat_rf <- predict(fit_rf, test_x)
  h_testing_set1 <- h_testing_set1 %>% bind_cols(y_hat_rf = y_hat_rf)

  fit_nb <- train(train_x, train_y, method ="nb")
  y_hat_nb <- predict(fit_nb, test_x)
  h_testing_set1 <- h_testing_set1 %>% bind_cols(y_hat_nb = y_hat_nb)
  library(e1071)
  fit_svm <- svm(home_team_win ~ home_w + away_w + h_goal + h_rel_goal + h_r_goal + a_goal + a_rel_goal
  y_hat_svm <- predict(fit_svm, test_x)
  h_testing_set1 <- h_testing_set1 %>% bind_cols(y_hat_svm = y_hat_svm)

  fit_ksvm <- svm(home_team_win ~ home_w + away_w + h_goal + h_rel_goal + h_r_goal + a_goal + a_rel_goal
  y_hat_ksvm <- predict(fit_ksvm, test_x)
  h_testing_set1 <- h_testing_set1 %>% bind_cols(y_hat_ksvm = y_hat_ksvm)

  h_acc_model1 <- h_testing_set1  %>% summarize(knn_acc = mean(y_hat_knn == home_team_win),
                                          rf_acc = mean(y_hat_rf == home_team_win),
                                          nb_acc = mean(y_hat_nb == home_team_win),
                                          svm_acc = mean(y_hat_svm == home_team_win),
                                          ksvm_acc = mean(y_hat_ksvm == home_team_win))

  h_model1<- bind_rows(h_model1,h_acc_model1)


  h_accuracy = data.frame(h_model1)


  h_knn = mean(h_accuracy$knn_acc)
  h_rf = mean(h_accuracy$rf_acc)
  h_nb = mean(h_accuracy$nb_acc)
  h_svm = mean(h_accuracy$svm_acc)
  h_ksvm=mean(h_accuracy$ksvm_acc)

  h_avg = data.frame(knn = h_knn, rf = h_rf, nb = h_nb, svm = h_svm, ksvm = h_ksvm)
  return(h_avg)
}
```

Now it is time to apply this function 10 times and get the output:

```
B = c(1:10)
h_final_model = sapply(B, model)
# Conversion of the output to a data frame
df_h_model = as.data.frame((h_final_model))
df_h_model = data.frame(t(h_final_model))

# Calculation of the accuracies of the algorithms for this model
home_accuracies = data.frame(h_knn = mean(t(as.data.frame(df_h_model$knn))),
                             h_rf = mean(t(as.data.frame(df_h_model$rf))),
                             h_nb = mean(t(as.data.frame(df_h_model$nb))),
                             h_svm = mean(t(as.data.frame(df_h_model$svm))),
                             h_ksvm = mean(t(as.data.frame(df_h_model$ksvm))))
home_accuracies
```

```
##       h_knn      h_rf      h_nb     h_svm    h_ksvm
## 1 0.6589322 0.6425051 0.6774127 0.6915811 0.6632444
```

The best home win prediction model is:

```
h_best_model <- which.max(home_accuracies)
h_best_model
```

```
## h_svm
##     4
```

3.2 Away team model

In this section, it is time to create the away winning prediction model. I am also going to create a function that will evaluate the model with the 5 different algorithms and get the accuracy of these algorithms as many times as we want, using different "set.seed()" functions:

```
#initialize dataframes
a_model1 = data.frame()
a_model2 = data.frame()
a_model3 = data.frame()
a_model4 = data.frame()
a_model5 = data.frame()
a_total_avg = data.frame()

# Away team winning function
a_model <- function(x){
  set.seed(x)
  #Create the validation dataset
  a_validation_index <- createDataPartition(a_laliga_win$away_team_win, times = 1, p= 0.2, list= FALSE)
  a_validation_set <- a_laliga_win[a_validation_index,]
  a_working_set <- a_laliga_win[-a_validation_index, ]

  #Create the training and testing datasets
  a_testing_index <- createDataPartition(a_working_set$away_team_win, times = 1, p = 0.2, list = FALSE)
  a_testing_set <- a_working_set[a_testing_index,]
```

```
  a_training_set <- a_working_set[-a_testing_index,]


  #First Model

  train_x <- a_training_set[,5:12]
  train_y <- as.factor(a_training_set[,4])
  test_x <- a_testing_set[,5:12]
  test_y <- as.factor(a_testing_set[,4])

  fit_knn <- train(train_x, train_y, method = "knn")
  y_hat_knn <- predict(fit_knn, test_x)
  a_testing_set1 <- a_testing_set %>% bind_cols(y_hat_knn = y_hat_knn)

  fit_rf <- train(train_x, train_y , method ="rf")
  y_hat_rf <- predict(fit_rf, test_x)
  a_testing_set1 <- a_testing_set1 %>% bind_cols(y_hat_rf = y_hat_rf)

  fit_nb <- train(train_x, train_y, method ="nb")
  y_hat_nb <- predict(fit_nb, test_x)
  a_testing_set1 <- a_testing_set1 %>% bind_cols(y_hat_nb = y_hat_nb)

  library(e1071)
  fit_svm <- svm(away_team_win ~ home_w + away_w + h_goal + h_rel_goal + h_r_goal + a_goal + a_rel_goal
  y_hat_svm <- predict(fit_svm, test_x)
  a_testing_set1 <- a_testing_set1 %>% bind_cols(y_hat_svm = y_hat_svm)

  fit_ksvm <- svm(away_team_win ~ home_w + away_w + h_goal + h_rel_goal + h_r_goal + a_goal + a_rel_goal
  y_hat_ksvm <- predict(fit_ksvm, test_x)
  a_testing_set1 <- a_testing_set1 %>% bind_cols(y_hat_ksvm = y_hat_ksvm)

  a_acc_model1 <- a_testing_set1  %>% summarize(knn_acc = mean(y_hat_knn == away_team_win),
                                                rf_acc = mean(y_hat_rf == away_team_win),
                                                nb_acc = mean(y_hat_nb == away_team_win),
                                                svm_acc = mean(y_hat_svm == away_team_win),
                                                ksvm_acc = mean(y_hat_ksvm == away_team_win))

  a_model1<- bind_rows(a_model1,a_acc_model1)

  a_accuracy = data.frame(a_model1)


  a_knn = mean(a_accuracy$knn_acc)
  a_rf = mean(a_accuracy$rf_acc)
  a_nb = mean(a_accuracy$nb_acc)
  a_svm = mean(a_accuracy$svm_acc)
  a_ksvm= mean(a_accuracy$ksvm_acc)

  a_avg = data.frame(knn = a_knn, rf = a_rf, nb = a_nb, svm = a_svm, ksvm = a_ksvm)

  return(a_avg)
}
```

Now it is time to apply this function 10 times and get the output:

```
a_final_model <- sapply(B,a_model)
#Conversion of the output to a data frame
df_a_model <- as.data.frame(a_final_model)
df_a_model <- data.frame(t(a_final_model))

# Calculation of the accuracies of the algorithms of the model
away_accuracies = data.frame(a_knn = mean(t(as.data.frame(df_a_model$knn))),
                             a_rf = mean(t(as.data.frame(df_a_model$rf))),
                             a_nb = mean(t(as.data.frame(df_a_model$nb))),
                             a_svm = mean(t(as.data.frame(df_a_model$svm))),
                             a_ksvm = mean(t(as.data.frame(df_a_model$ksvm))))
away_accuracies
```

```
##      a_knn      a_rf      a_nb     a_svm    a_ksvm
## 1 0.737577 0.7252567 0.7622177 0.7603696 0.7585216
```

The best away win prediction model is:

```
a_best_model <- which.max(away_accuracies)
a_best_model
```

```
## a_nb
##    3
```

4 Results

Now that the models are created and the best accuracy algorithm is selected, based on the training and testing sets, it is time to use the validation set to evaluate each model and see the final accuracy they have.

4.1 Home win prediction model

A validation set and a working set have to be created from the original dataset:

```
# Create the validation dataset
h_validation_index <- createDataPartition(h_laliga_win$home_team_win, times = 1, p= 0.2, list= FALSE)
h_validation_set <- h_laliga_win[h_validation_index,]
h_working_set <- h_laliga_win[-h_validation_index, ]


h_working_x <- h_working_set[,5:12]
h_working_y <- as.factor(h_working_set[,4])
h_valid_x <- h_validation_set[,5:12]
h_valid_y <- as.factor(h_validation_set[,4])
```

Now it is time to fit the working dataset using the svm function and evaluate it with the validation dataset. The accuracy of the model is the following:

```
h_fit_svm <- svm(home_team_win ~ home_w + away_w + h_goal + h_rel_goal + h_r_goal + a_goal + a_rel_goal
h_y_hat_svm <- predict(h_fit_svm, h_valid_x)
mean(h_y_hat_svm == h_valid_y)
```

```
## [1] 0.7236842
```

4.2 Away win prediction model

A validation set and a working set have to be created from the original dataset:

```
# Create the validation dataset
a_validation_index <- createDataPartition(a_laliga_win$away_team_win, times = 1, p= 0.2, list= FALSE)
a_validation_set <- a_laliga_win[a_validation_index,]
a_working_set <- a_laliga_win[-a_validation_index, ]

a_working_x <- a_working_set[,5:12]
a_working_y <- as.factor(a_working_set[,4])
a_valid_x <- a_validation_set[,5:12]
a_valid_y <- as.factor(a_validation_set[,4])
```

It is time to fit the working dataset using the nb function and evaluate it with the validation dataset. The accuracy of the model is the following:

```
a_fit_nb <- train(a_working_x, a_working_y, method = "nb")
a_y_hat_nb <- predict(a_fit_nb, a_valid_x)
mean(a_y_hat_nb == a_valid_y)
```

```
## [1] 0.7384868
```

5 Conclusion

This project is the conclusion of the **Data Science Professional Certificate** from HarvardX. This is the final project of the *Data Scicence: Capstone* course, the last of the 9 courses of the Professional Certificate. Through all these courses I have learned to obtain data, read data, transform data, analyse it and make prediction models using the data. This project has al little bit of all these concepts with a goal I have achieved: predict if a team will win a football match.

After analysing the data of the dataset, I have finally created two models: one for home team win prediction and another one for away team win prediction. The two models have between 70-75% of accuracy in their predictions, what means that they are quite good models.

If I think about real applications of the models, the first two things that come to my mind are:

- Use the models for betting. With an accuracy of 70-75%, depending on the betting odds it is possible to have a positive money balance.
- A football coach could use this information to know which is their probability to win the next match and decide if they want to train more the deffense that week or some other specific tactical aspects.

Finally, obviously the models can be improved. Here are some proposals to do with the model in order to make it better and more useful: - Add more recent seasons information to update the models, and specially add the information of the current season in order to make predictions nowadays. - Add some other features of the matches. For example: the possession, shots, shots, faults, yellow cards, red cards, etc. I'm sure that this extra features could use to improve the models and make better predictions.