Soccer Predictions

Ricardo Zuñiga

16/9/2019

Introduccion

This project seeks to make the soccer teams winner prediction with better accuracy than currently has a betting site, however,*I strongly discourage the use of the method described in the project and I will not assume any responsibility of anyone who wants to try it.*

There is a Data set of 7299 matches from different countries, where they show us matching teams, goals scored by each one, as well as conditions in which they arrived,it means, won matches, table position and goals against rival team.

Project objective is to determine if betting sites will always be one step ahead and in long run people who bets will always lose money, or there is a valid way to be able to bet with greater winning money chances than losing risk, in case there is a robust enough method so that when changing the odds "x" percentage loses effectiveness again.

Within the project you wil see how the data set is manipulated a bit, attempts are made to make predictions without machine learning, machine learning predictions,improvement of algorithm by ensembles, gains/losses calculations and profit percentages with each method.

Preprocessing

To get project started, you must first have downloaded, installed and enabled the following packages that we are going to use:

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(dplyr)) install.packages("dplyr", repos = "http://cran.us.r-project.org")
if(!require(ggplot2)) install.packages("ggplot2", repos = "http://cran.us.r-project.org")
if(!require(stringr)) install.packages("stringr", repos = "http://cran.us.r-project.org")
if(!require(readr)) install.packages("readr", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")

library(dplyr)
library(ggplot2)
library(stringr)
library(readr)
library(caret)
```

In this project several algorithms will be made to try to see which or whiches can give us a positive profit at bet ending, for this several functions will help us speed up process, so I propose 3 functions that I will constantly use. First one "profit percentage" is the percentage of money earned compared to the money invested.

Second function helps us calculate algorithm efficiency, this function is really a compressed use of Confusion-Matrix function and instead of having it with decimal point we multiply it by 100 to pass it to percentage.

Last one computes profit based on whether our prediction was correct, that is, we bet 10 dollars in each match, multiply it by the corresponding odd in case of being correct, if it was incorrect, 10 dollars are subtracted from the total amount.

```r
#function computes the percentage profit
profit_percentage<-function(x,a){
  10*x/(nrow(a))
}

#computes the efficiency of that model
efficiency<-function(a,b){
  100*confusionMatrix(a,as.factor(b$HomeResult))$overall[["Accuracy"]]
}

#computes the total profit
profit<-function(a){
  test_set3<-test_set2%>%mutate(my_bet=a)
sum(test_set3%>%mutate(gain=ifelse(HomeResult==my_bet,ifelse(my_bet=="WIN",(ODD.HOME-1)*10,ifelse(my_be
}
```

Once functions are generated next step is read the database, in the Github repository is already attached the file "MASTER.cvs", which contains all the matches data to be used, variables are: country, teams, home score, away score, odd for home, odd for draw, odd for away, home team table position, away team table position, home team win rate, away team win rate and the mean goals from matches head to head

```r
master<-read.csv("MASTER.csv")
```

To facilitate names location in the future it is easier to separate competing teams variable into two, first one for local team and another for away team, then I verify that if this separation is correct in the first observation

```r
#separate teams into home and away

master<-master%>%mutate(HomeTeam=TEAMS.H.A.)
master$HomeTeam<-gsub("\\ /.*","",master$HomeTeam)

master<-master%>%mutate(AwayTeam=TEAMS.H.A.)
master$AwayTeam<-gsub(".*/ ","",master$AwayTeam)

#Verify if Teams were correctly separaded and remove the spaces
identical("Nurnberg II",master$HomeTeam[1])
## [1] TRUE
identical("Pipinsried",master$AwayTeam[1])
## [1] TRUE
```

Due to data set does not have a variable that tells us who really won,is necessary add a column for the result of the local team and for the away team this is calculated by knowing who scored more goals than rival.

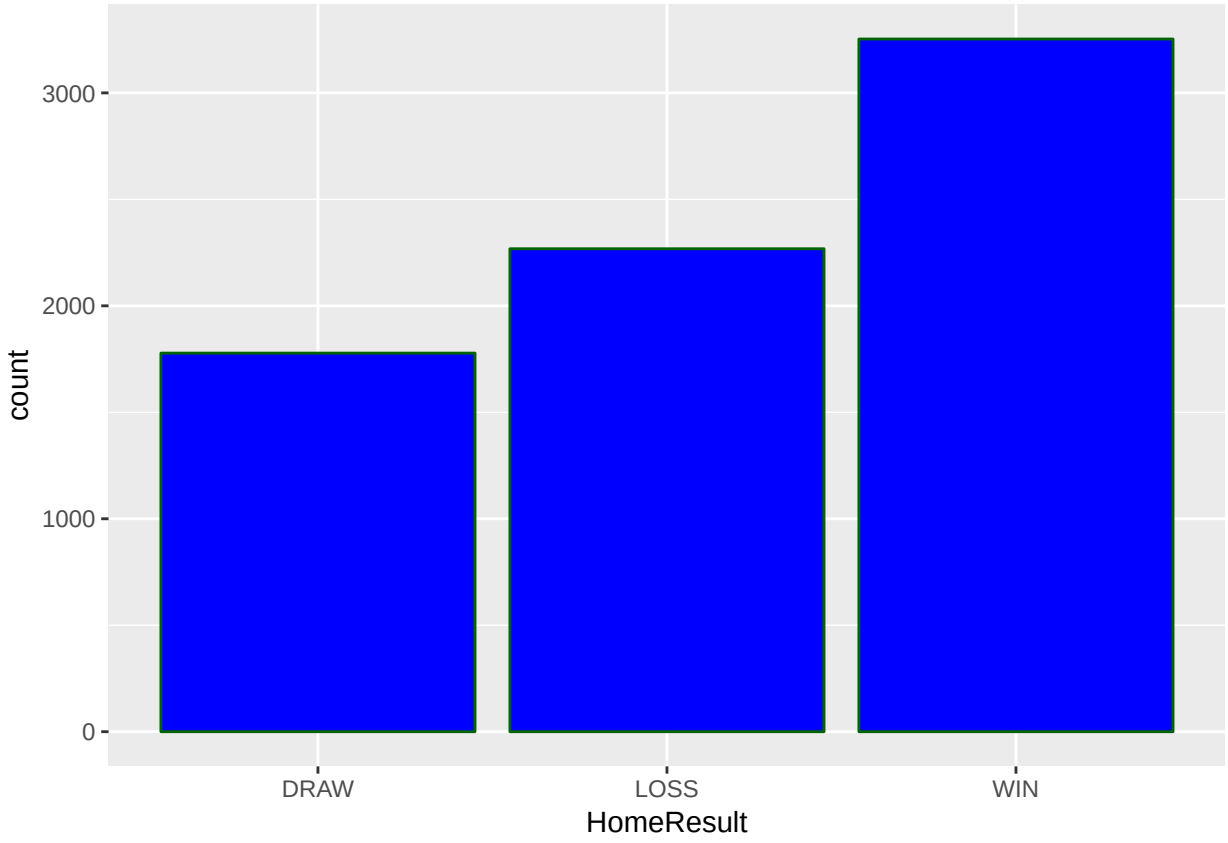
```r
#Add column to know which Team Won
master<-master%>%mutate(HomeResult=ifelse(HOME.SCORE.F.==AWAY.SCORE..F.,"DRAW",ifelse(HOME.SCORE.F.>AWA
master<-master%>%mutate(AwayResult=ifelse(HomeResult=="DRAW","DRAW",ifelse(HomeResult=="WIN","LOSS","WI
```

Analysis (tendencies)

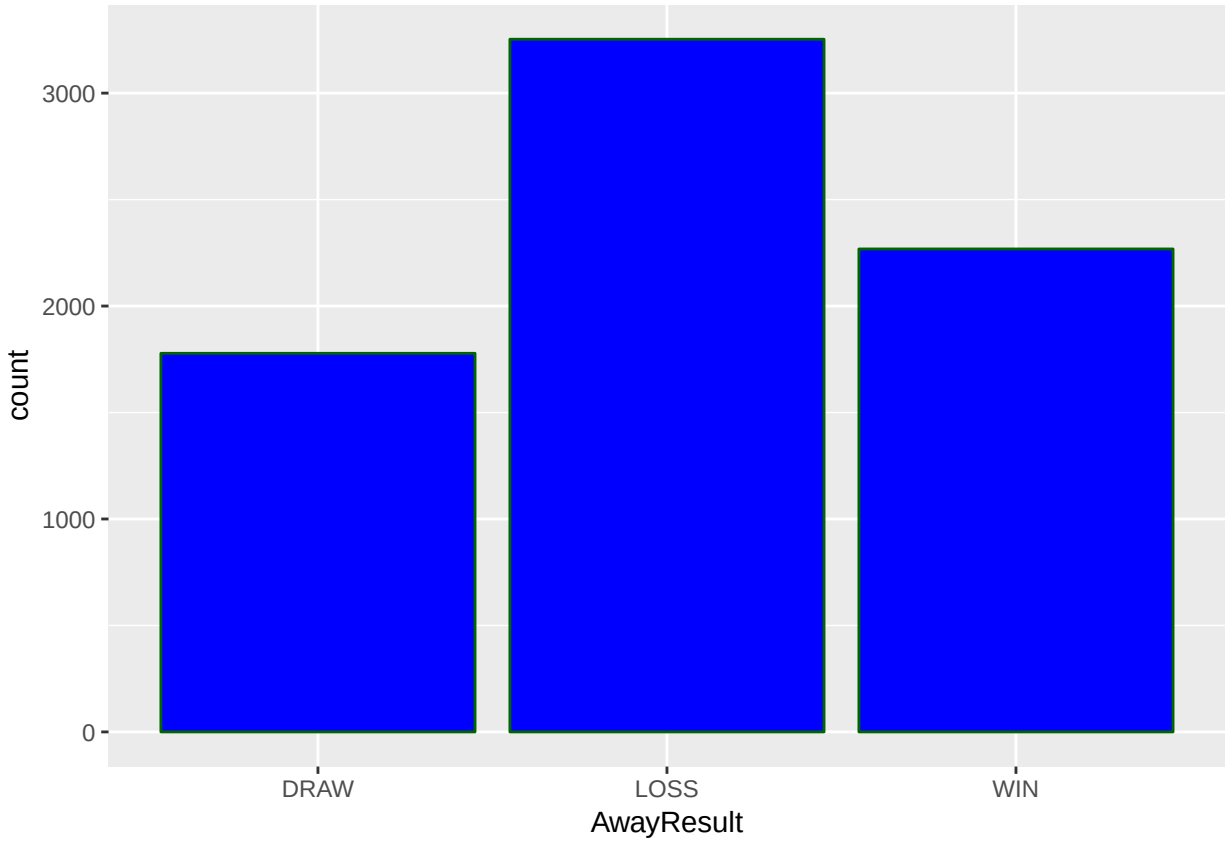
Now to start analysis, here it should be noted that in each bet made it will be $10 to have an equal metric in each bet.

It is known that Home team tends to win more times than an away team, so I make a histogram of how a Home team performs and another of how does an Away team.

```
master%>%group_by(HomeResult)%>%ggplot(aes(HomeResult))+geom_histogram(stat="count",fill="blue",color="
```




```
master%>%group_by(AwayResult)%>%ggplot(aes(AwayResult))+geom_histogram(stat="count",fill="blue",color="
```

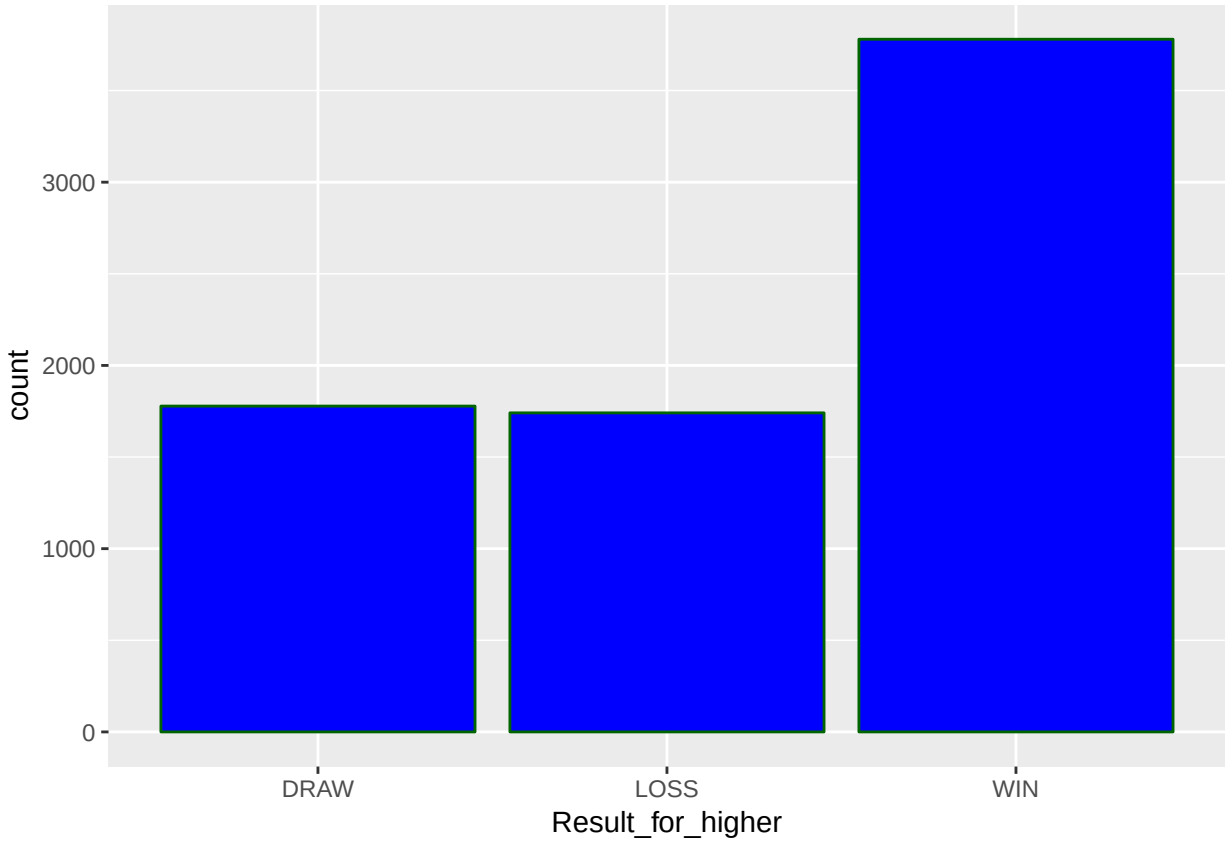



```
results<-data.frame() #create a data frame where results would be stored
results[1,1]<-"Bet Home"
results[1,2]<-100*mean(master%>%mutate(gain=ifelse(HomeResult=="WIN",1,0))%>%pull(gain))
results[1,2]
## [1] 44.56775
results[1,3]<-sum(master%>%mutate(gain=ifelse(HomeResult=="WIN",(ODD.HOME-1)*10,-10))%>%pull(gain)) #ca
results[1,3]
## [1] -5744.2
results[1,4]<-profit_percentage(results[1,3],master)
```

According to histogram home team tends to win suspicion and away team to lose is confirmed, however, if was realized a bet following this methodology we will end up losing money.

Also by logic is possible to deduce that if a Team is higher in general table will win against teams with a lower position, histogram is done to probe that.

```
master_selected<-master%>%mutate(Result_for_higher=ifelse(HOME.POSITION<AWAY.POSITION,HomeResult,AwayRes
master_selected%>%group_by(Result_for_higher)%>%ggplot(aes(Result_for_higher))+geom_histogram(stat="cou
```



Next try is to bet only in matches where Home team has higher table position even though code show that also would result in losing money

```
#filter  for match with Local Team has higher Table Position
master_selected<-master%>%filter(HOME.POSITION<AWAY.POSITION)

results[2,1]<-"Home and Higher"
results[2,2]<-100*mean(master_selected%>%mutate(gain=ifelse(HomeResult=="WIN",1,0))%>%pull(gain))
results[2,2]
## [1] 52.20588
results[2,3]<-sum(master_selected%>%mutate(gain=ifelse(HomeResult=="WIN",(ODD.HOME-1)*10,-10))%>%pull(ga
results[2,3]
## [1] -2819.9
results[2,4]<-profit_percentage(results[2,3],master_selected)
```

Before continue eliminate certain observations such as World Cup or World Cup Classification is needed due to does not exist neither Home Team nor Away Team because is neutral stadium, also Table position is not stable.

```
x<-unique(master$COUNTRY)
x<-x[1:99]
master<-master%>%filter(COUNTRY %in% x)
```

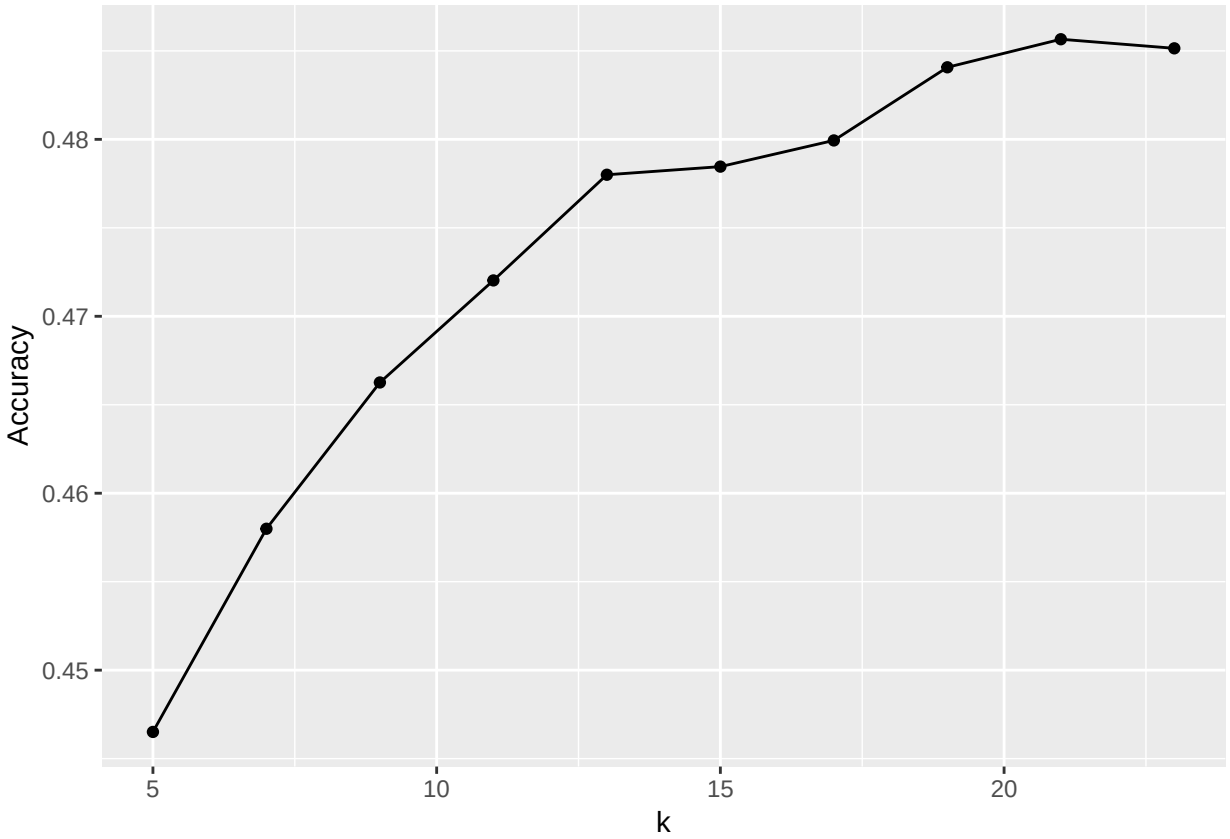
Analysis (Machine learning)

First thing I did to start machine learning is set a set.seed () to be able to replicate results, then next step is to split master into two data sets one for training and another for tests.

```
set.seed(13)
index<- createDataPartition(y = master$HomeResult,p = 0.9,list = FALSE)
train_set<- master[index,]
test_set<- master[-index,]
```

I will create the first machine learning training with "knn" method, after training is ready will plot number of near neighbors against algorithm precision, it can be seen that algorithm accuracy is very low, at its best point does not exceed 50%, so Not very reliable.

```
control <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
mod_fit <- train(HomeResult~ODD.HOME+ODD.DRAW+ODD.AWAY+HOME.POSITION+AWAY.POSITION+WIN.RATE.HOME+WIN.RA
                 method = "knn",
                 trControl=control,
                 preProcess = c("center", "scale"),
                 tuneLength = 10)
mod_fit
## k-Nearest Neighbors
##
## 6530 samples
##    9 predictor
##    3 classes: 'DRAW', 'LOSS', 'WIN'
##
## Pre-processing: centered (9), scaled (9)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 5876, 5878, 5877, 5877, 5877, 5877, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    5  0.4465088  0.1269882
##    7  0.4579922  0.1389610
##    9  0.4662598  0.1471996
##   11  0.4720275  0.1533431
##   13  0.4779993  0.1584006
##   15  0.4784592  0.1558082
##   17  0.4799400  0.1553154
##   19  0.4840733  0.1599316
##   21  0.4856591  0.1603980
##   23  0.4851455  0.1575353
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 21.
mod_fit$results%>%ggplot(aes(k,Accuracy))+geom_point()+geom_line()
```



Due to Knn method do not give us the precision expected, several machine learning methods would be trained in order to compare accuracy among them and select the best.

```
models<- c("glm", "lda", "naive_bayes", "svmLinear", "knn", "gamLoess", "multinom", "qda", "rf", "adaboo
fits<- lapply(models, function(model){
       print(model)
       train(HomeResult~ODD.HOME+ODD.DRAW+ODD.AWAY+HOME.POSITION+AWAY.POSITION+WIN.RATE.HOME+WIN.RATE.AW
   })
## [1] "glm"
## [1] "lda"
## [1] "naive_bayes"
## [1] "svmLinear"
## [1] "knn"
## [1] "gamLoess"
## [1] "multinom"
## [1] "qda"
## [1] "rf"
## [1] "adaboost"

## Precision for each attempt
fit_accuracy<-data.frame()
for (i in 1:10) {
  fit_accuracy[i,1]<-models[i]
}
for (i in 1:10) {
  fit_accuracy[i,2]<-fits[[i]]$results$Accuracy[1]
```

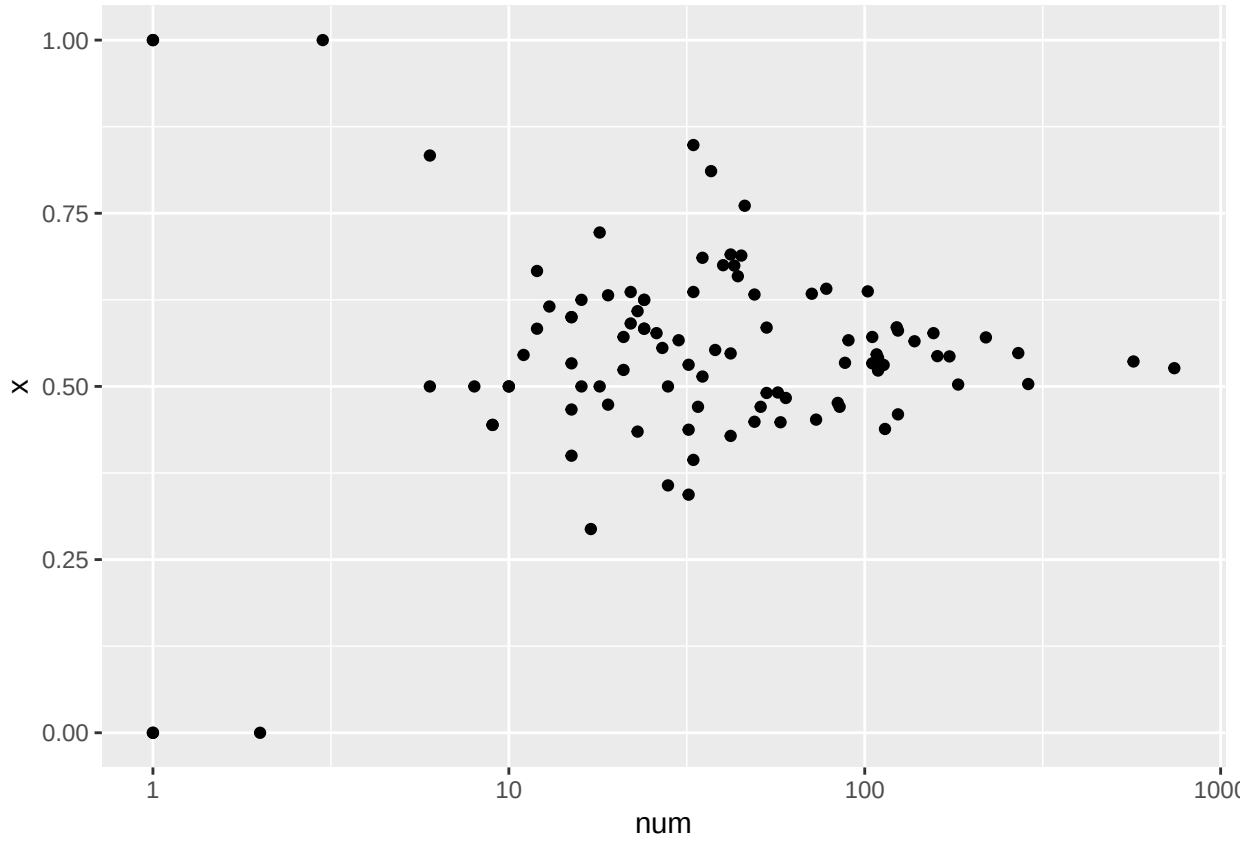
```
}

fit_accuracy
##              V1        V2
## 1           glm 0.4953969
## 2           lda 0.4935115
## 3  naive_bayes 0.4947464
## 4    svmLinear 0.4941031
## 5           knn 0.4944182
## 6     gamLoess 0.4952316
## 7     multinom 0.4946371
## 8           qda 0.4971141
## 9            rf 0.4941023
## 10    adaboost 0.4963833
```

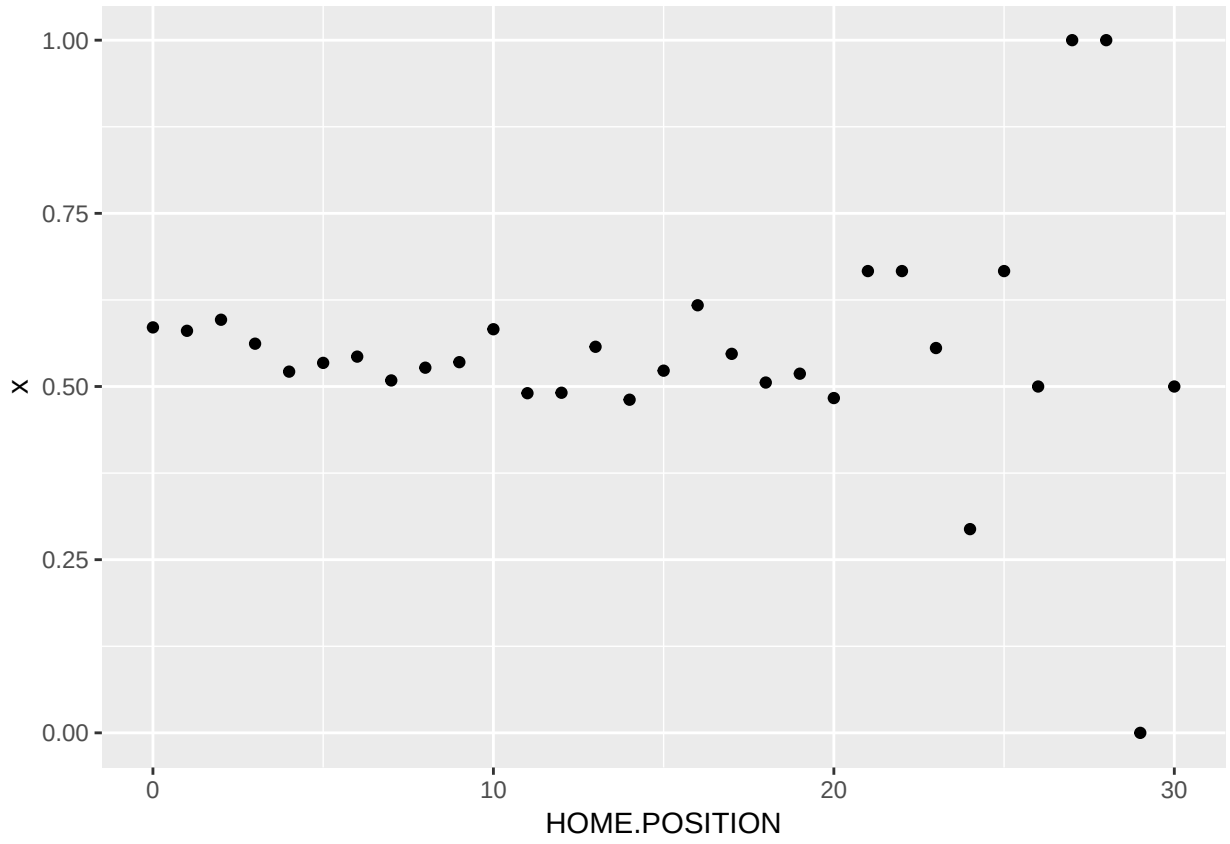
Table shows that there is not significant improvement between methods, even though the more confiable at this point are "naive_bayes" and "svmLinear".

To see how is possible to create an improvement 3 different plots were made, First one vary the accuracy among countries, next two is to observe if the tends do not overestimate or underestimate some Home or Away Position.

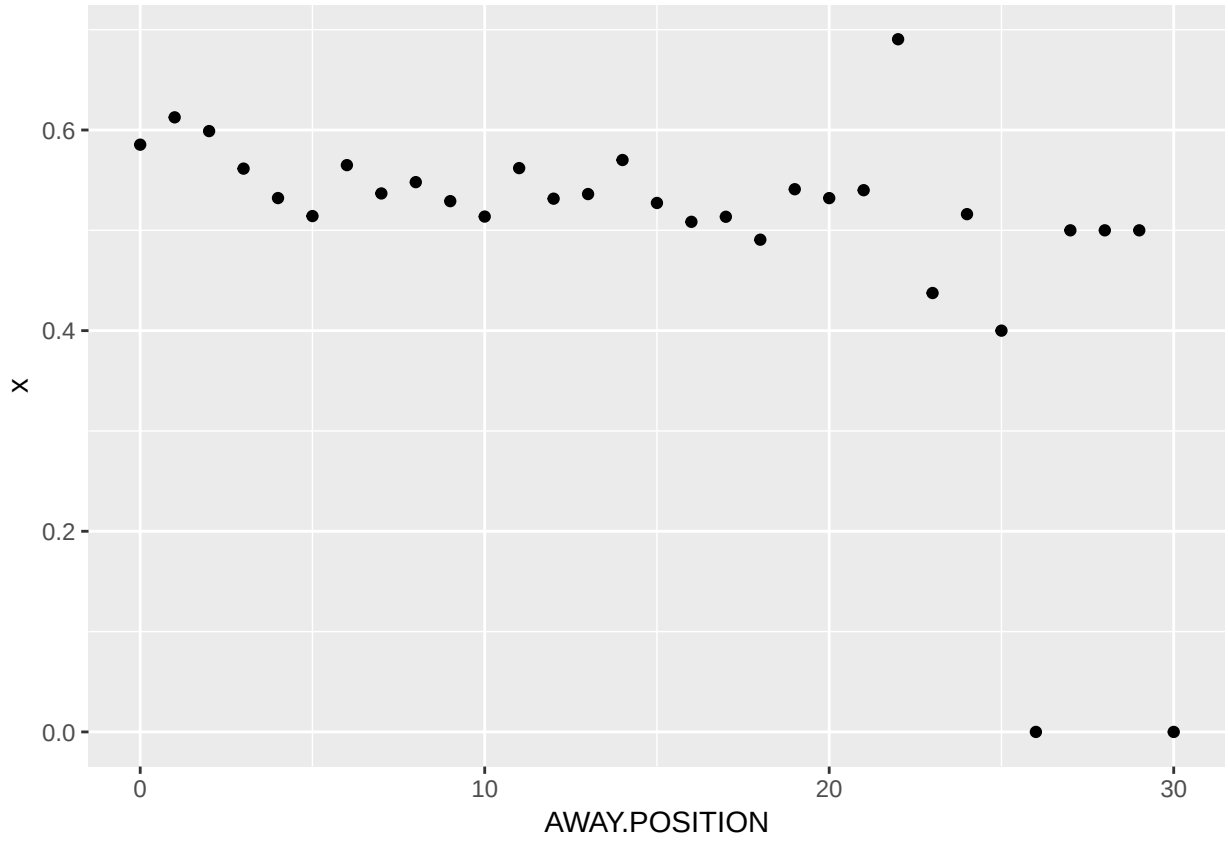
```
train_set2<-train_set%>%mutate(pred=predict(mod_fit))
correction<-train_set2%>%mutate(correct=ifelse(pred==HomeResult,1,0))%>%group_by(COUNTRY)%>%summarise(x=
correction%>%ggplot(aes(num,x))+geom_point()+scale_x_log10()
```



```
Home_stats<-train_set2%>%mutate(correct=ifelse(pred==HomeResult,1,0))%>%group_by(HOME.POSITION)%>%summa
Home_stats%>%ggplot(aes(HOME.POSITION,x))+geom_point()
```




```
Away_stats<-train_set2%>%mutate(correct=ifelse(pred==HomeResult,1,0))%>%group_by(AWAY.POSITION)%>%summa
Away_stats%>%ggplot(aes(AWAY.POSITION,x))+geom_point()
```



Is clearly in the plots that exist considerable bias between countries, but not for Teams Positions, there is somewhat normal, so there be created 2 new datasets the train and test set filter just for countries with accuracy of 60% or more

```
countries<-correction%>%filter(x>=.6)
countries<-countries%>%select(COUNTRY)

#eliminate the unnecesary countries for train set and test set
test_set2<-test_set%>%filter(COUNTRY %in% (countries$COUNTRY))
train_set_new<-train_set%>%filter(COUNTRY %in% (countries$COUNTRY))
```

Results

With previous analysis is possible to test different methods, first one a naive_bayes method predictions with test set that has all countries, then the same method but only with countries that are consider safe.

```
naive_bayes_new<-train(HomeResult~ODD.HOME+ODD.DRAW+ODD.AWAY+HOME.POSITION+AWAY.POSITION+WIN.RATE.HOME+W

#Make predictions
a<-predict(fits[[3]],test_set, type = "raw")
b<-predict(fits[[3]],test_set2, type = "raw")

results[3,1]<-"All Matches"

results[3,2]<-efficiency(a,test_set)
```

```
test_set3<-test_set%>%mutate(my_bet=a)
results[3,3]<-sum(test_set3%>%mutate(gain=ifelse(HomeResult==my_bet,ifelse(my_bet=="WIN",(ODD.HOME-1)*1
results[3,4]<-profit_percentage(results[3,3],test_set)
results[3,3]
## [1] -657

results[4,1]<-"60% Countries Matches FD"
results[4,2]<-efficiency(b,test_set2)

results[4,3]<-profit(b)
results[4,3]
## [1] -99.2
results[4,4]<-profit_percentage(results[4,3],test_set2)
```

Exists a considerable improvement in the profit if a bet is realized just for safe countries against with all, so for this results next step is make predictions by a machine learning methods ensemble.

```
#make an ensemble
naive_bayes_p<-predict(fits[[3]],test_set2, type = "prob")
qda_p<-predict(fits[[8]],test_set2, type = "prob")
svmLinear_p<-predict(fits[[4]],test_set2, type = "prob")

pred_avg<-(naive_bayes_p+qda_p+svmLinear_p)/3

pred_final <- factor(apply(pred_avg, 1, which.max)-1)
levels(pred_final) <- c("DRAW", "LOSS","WIN")

results[5,1]<-"Ensemble FD"
results[5,2]<-efficiency(pred_final,test_set2)
results[5,3]<-profit(pred_final)
results[5,4]<-profit_percentage(results[5,3],test_set2)
```

Last step is to train again, the difference is that the new train set just have safe countries, and would be compared against an ensemble of ”knn”, “svmLinear”,”naive_bayes” methods to improve accuracy.

```
#predict for the new data set
c<-predict(naive_bayes_new,test_set2, type = "raw")
results[6,1]<-"60% Countries Matches PD"
results[6,2]<-efficiency(c,test_set2)
results[6,3]<-profit(c)
results[6,4]<-profit_percentage(results[6,3],test_set2)

#make new ensemble
svmLinear_new<-train(HomeResult~ODD.HOME+ODD.DRAW+ODD.AWAY+HOME.POSITION+AWAY.POSITION+WIN.RATE.HOME+WI
knn_new<-train(HomeResult~ODD.HOME+ODD.DRAW+ODD.AWAY+HOME.POSITION+AWAY.POSITION+WIN.RATE.HOME+WIN.RATE
naive_bayes_newp<-predict(naive_bayes_new,test_set2, type = "prob")
knn_newp<-predict(knn_new,test_set2, type = "prob")
svmLinear_newp<-predict(svmLinear_new,test_set2, type = "prob")

pred_avg<-(naive_bayes_newp+knn_newp+svmLinear_newp)/3

pred_final_new<- factor(apply(pred_avg, 1, which.max)-1)
levels(pred_final_new) <- c("DRAW", "LOSS","WIN")
```

```
results[7,1]<-"Ensemble PD"
results[7,2]<-efficiency(pred_final_new,test_set2)
results[7,3]<-profit(pred_final_new)
results[7,4]<-profit_percentage(results[7,3],test_set2)
```

Once different algorithms were wrote in a table, column names be changed into "Situation, Precision %, Profit, Profit %" (Situation = algorithm used, Precision% = accuracy percentage, Profit= money earned after betting, Profit % = percentage of money earned after betting) and show all results

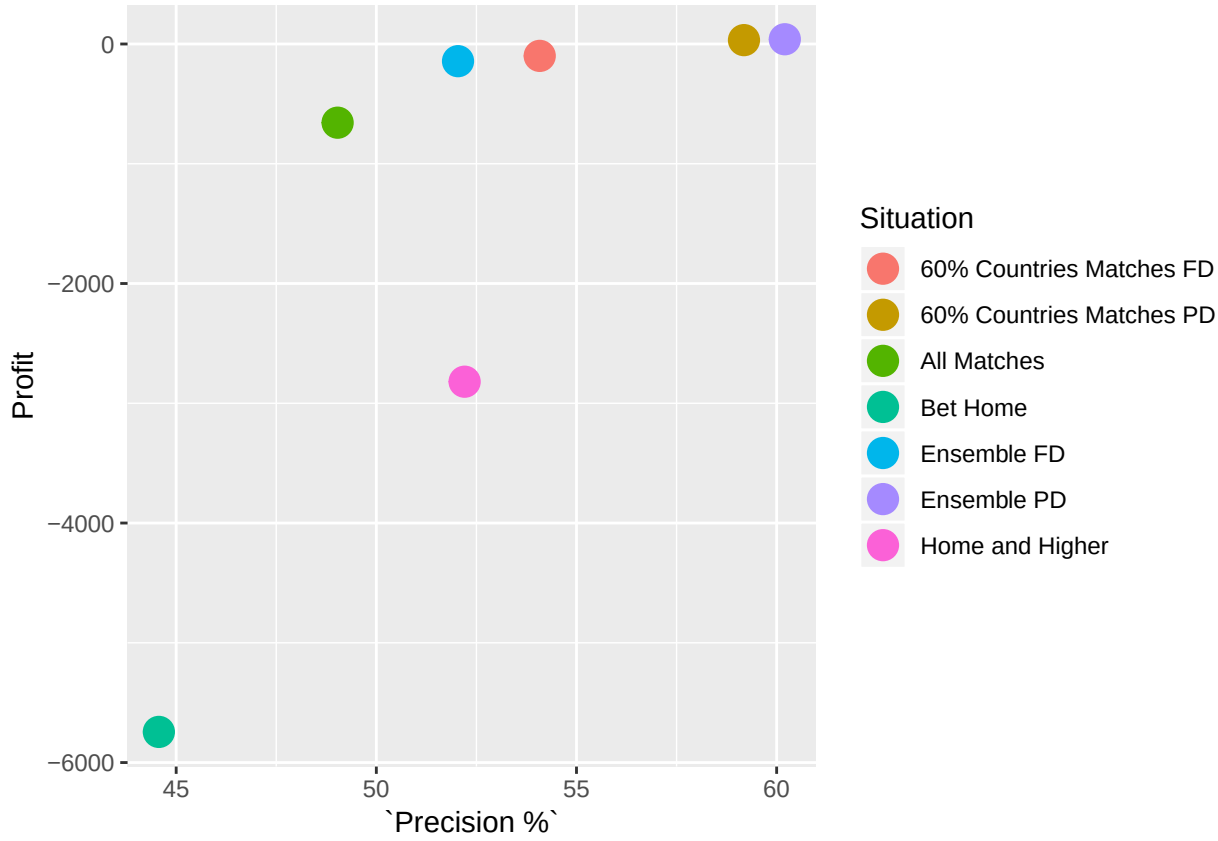
```
colnames(results)<-c("Situation","Precision %","Profit","Profit %")
results
##                  Situation Precision %  Profit   Profit %
## 1                  Bet Home    44.56775 -5744.2  -7.869845
## 2           Home and Higher    52.20588 -2819.9  -5.456463
## 3               All Matches    49.03315  -657.0  -9.074586
## 4 60% Countries Matches FD    54.08163   -99.2 -10.122449
## 5               Ensemble FD    52.04082  -143.3 -14.622449
## 6 60% Countries Matches PD    59.18367    32.4   3.306122
## 7               Ensemble PD    60.20408    39.8   4.061224
```

A guarantee of this results are that if odds are changed in 5% the last results should keep positive, so if odds are changed the net profit is:

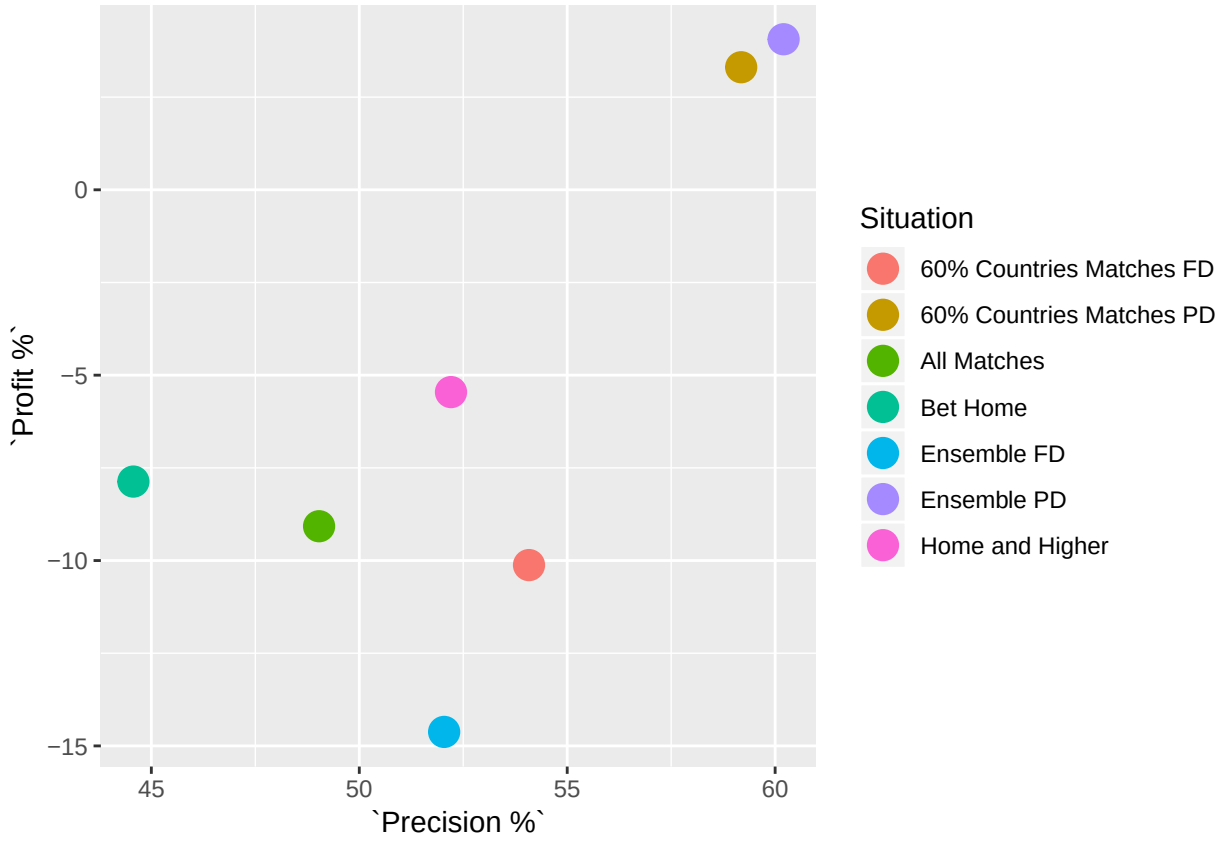
```
test_set3<-test_set2%>%mutate(my_bet=pred_final_new)
sum(test_set3%>%mutate(gain=ifelse(HomeResult==my_bet,ifelse(my_bet=="WIN",((ODD.HOME*.95)-1)*10,ifelse
## [1] -11.19
```

Finally, I add a new variable to each observation that tell us if some algorithm had a positive performance, it would be classified as success otherwise Fail, I plot accuracy vs profit, another precision plot vs % earned and finally a geom_col that tells us how much win or lose.

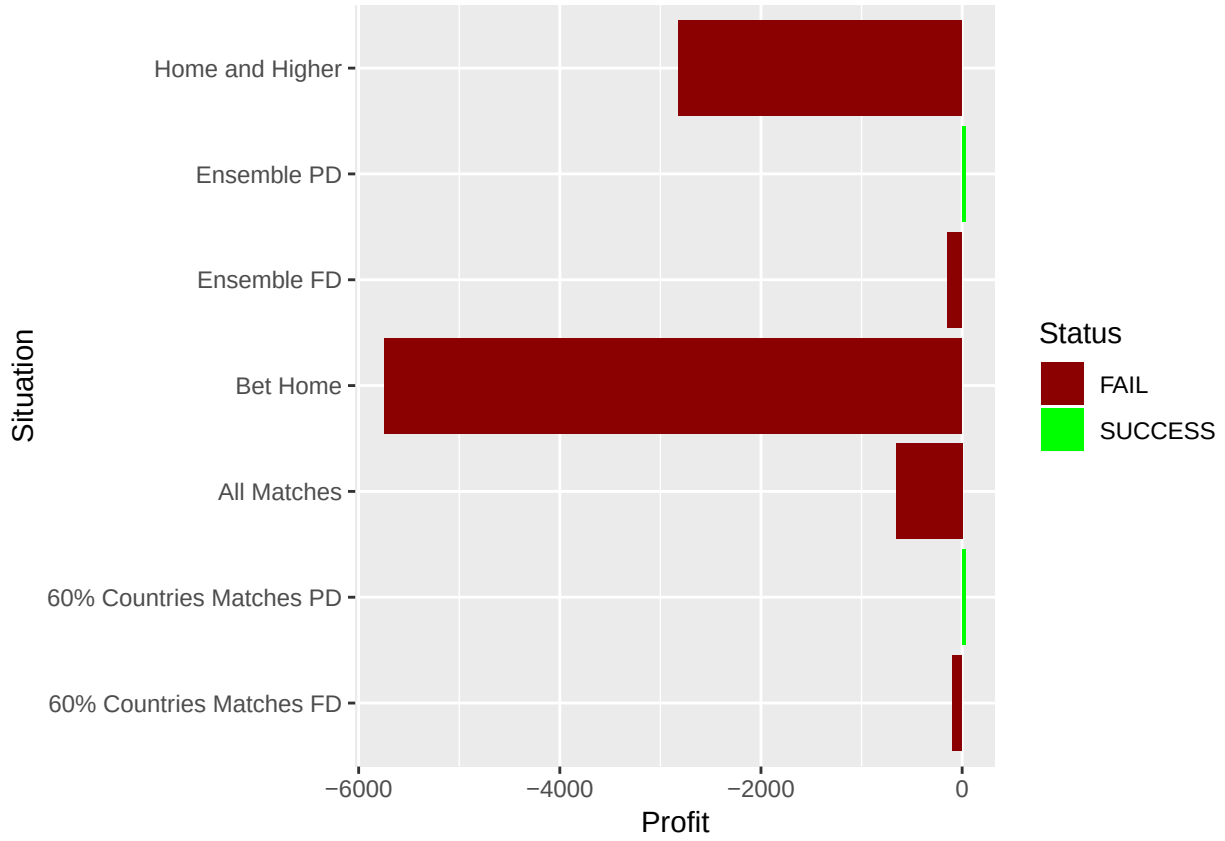
```
results<-results%>%mutate(Status=ifelse(Profit>0,"SUCCESS","FAIL"))
results%>%ggplot(aes(`Precision %`,Profit,colour=Situation))+geom_point(size=5)
```




```
results%>%ggplot(aes(`Precision %`,`Profit %`,colour=Situation))+geom_point(size=5)
```




```
ggplot(results,aes(Situation,Profit))+
geom_col(aes(fill = Status))+
  coord_flip()+
  scale_fill_manual(values = c("darkred","green"))
```



The final percentage of money earned is:

```
results[7,4]
## [1] 4.061224
```

Conclusions

I conclude bet for certain method without a really machine learning philosophy in the long run will finish in losing money, even though having more complexes algorithms and betting just in secure countries is possible to create better predictions and in the long win, in other words, is possible to create a Machine Learning algorithm whose predict with advantage a match results, in the other hand if casino decrease by 5% odds the profit of gains would reduce to almost 0 so the algorithm needs to improve

Future work for project, I need to increase observations of the data set, it means add more matches, with less risk countries, this due to have more information would results in higher accuracy, second would be update data base for odds update for new matches, also add more variables like tournament,injured players, expelled players, if match represent a special prize for a team such as classify to another tournament, keep in the league, historical mark, etc.

Another future work is to increase algorithm difficulty by using Deep Learning, even though first I need to learn it and start to develop it,*I remain you, I strongly discourage the use of the method described in the project and I will not assume any responsibility of anyone who wants to try it.*

Acknowledgments

I would like thank Professor Rafael A. Irizarry for guiding us and introducing us along path of Data science, in the same way special thanks to the entire Edx and HarvardX team for always being on the lookout and answering our questions, finally to all the classmates of the course we were together throughout this trip.

References

Rafael A. Irizarry. (2019) Introduction to Data Science. Data Analysis and Prediction Algorithms with R, HARVARDX, web site: https://rafalab.github.io/dsbook/index.html