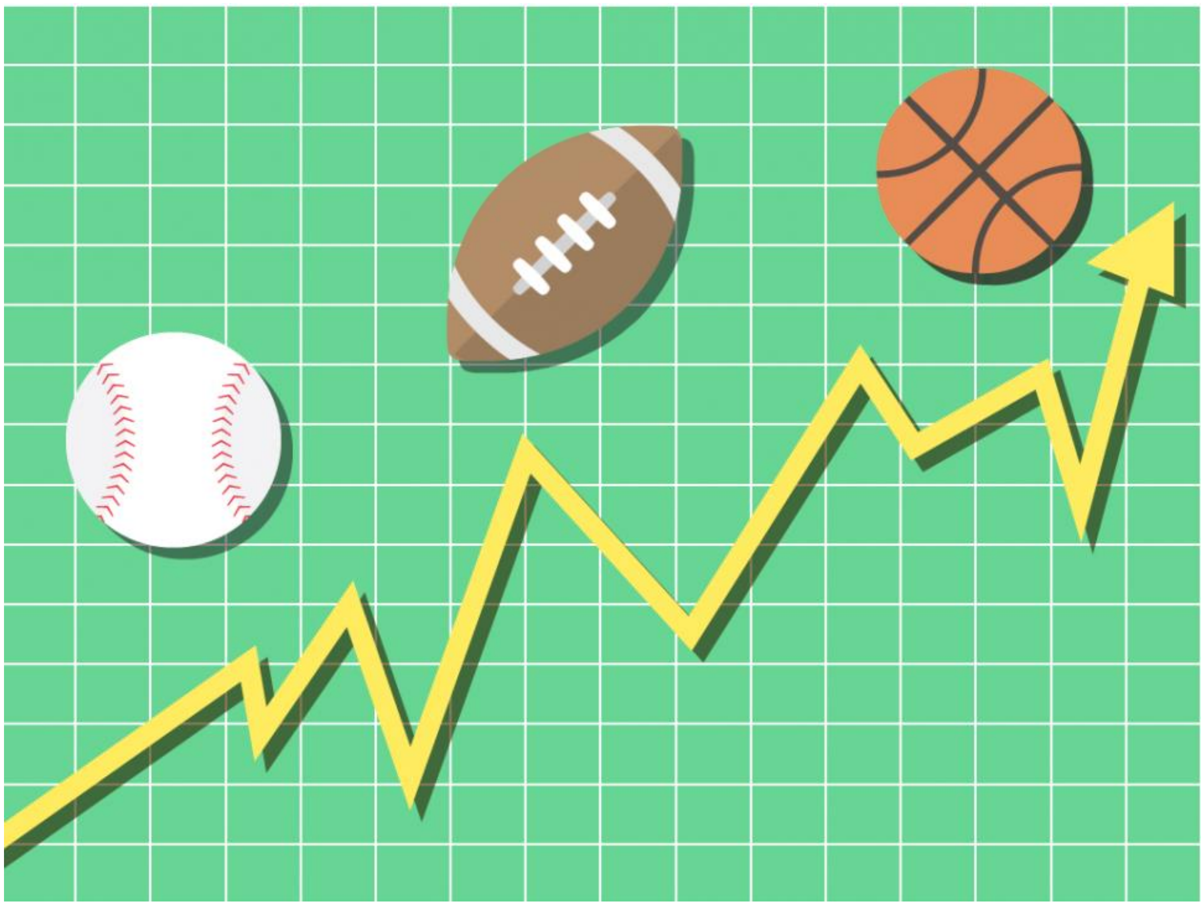
EXPLOITING BOOKMAKERS BIASES TO CREATE HIGHLY PROFITABLE AI-BASED BETTING STRATEGIES



To do :

- *Grammar and typos*
- *State-of-the-art*
- *ML models explanation*
- *Include training times*
- *Football simulations*

Side Personal Project

Sébastien CARARO

2020

TABLE OF CONTENTS

I - ABSTRACT

I created a pipeline for forecasting NBA games outcome and I developed a couple tools to exploit this pipeline in order to place bets according to the market odds. The most precise model is capable of predicting the winner 69,34 % of the time over the last season (2018/2019 – 1230 games). I also developed similar algorithms towards various sports such as football, tennis, baseball and eSports : the software is able to make predictions on a daily basis and yielt a highly competitive return on investment (ROI) of about 7,30 % on the first 700 bets - over a mean duration of 2h (average duration of a sport match) – between January 15th and March 13rd. Given the fact that the most recent models - presented in this report - are outperforming the previous ones, we can even expect this method to give higher profit in the next months.

II - INTRODUCTION

This report contains information and techniques concerning the creation of a full pipeline from scratch to help predicting the outcomes of multiple sports and, upon everything, create a highly profitable betting strategy. The following report is furnished along with multiple result metrics, all contained in a directory called *Results.* For any information or question concerning this work, one can contact me at sebcararo@hotmail.fr.

The project contains multiple steps which are developed in this report, I tried as much as possible to make it intelligible towards a non-scientific audience, but also to display a few very technical points which were crucial in the development of this project. I created the NBA program first, therefore I will focus on the NBA algorithm for the main explanations, then I will eventually describe how I adapted these techniques to create the other sports' algorithms afterwards.

This project was built under *Creative Commons* license CC BY-NC-SA, which means any reader has the right to remix, adapt and build upon this work for non-commercial purposes, as long as he credits myself and this license along with their publication. [xx]

III - WHY THIS PROJECT ?

As a 22-years old sports enthusiast, I had always been fascinated by sports analytics and have been focusing on studying the many various statistics in the NBA for years. As I started my gap year before my final Master's year of education, I wanted to involve myself towards a personal project such like this one because then I wouldn't be able to spend so much time when working full-time. I was largely inspired by some of the publications that I present in the next section, and I was willing to mix my passion for sport with my growing interest for Data Science. Furthermore, such a project was a huge challenge for me, as I had to plan everything from scratch, from the conception of the pipeline to the code implementation, along with the many others related tasks such as data scraping, interface construction, or even computation-time optimization. The difficulty of the project was reinforced by the fact that is is quite a *niche* problem, because even if there are a lot of related works published online, only few of them publish their code, plus many of these works don't yield profitable betting strategies. Taking all these elements into considerations, I was fully enthusiastic to work on this project for a few months.

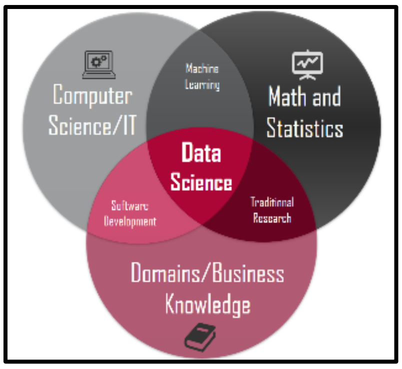


Figure xx : Data Science's Venn Diagram

IV - STATE-OF-THE-ART CONCERNING PREDICTIVE ML ALGORITHMS IN SPORTS

I started this project by spending a few weeks looking for previous works related to Machine Learning applications for sport events predictions. As a modern field with a lot of interest, this subject is already well documented and there are plenty of papers written each year. The main papers which I was influenced by were :

- Beating bookmakers with their own odds – and how the online sports betting market is rigged (Kaunitz et al., 2017, [xx])
- Non-Linear classification as a tool for predicting tennis matches (Jakub Hostacny, 2018, [xx])
- A statistical approach to sports betting (Anton Altmann, 2004, [xx])
- Predicting football results using Machine Learning techniques (Corentin Herbinet, 2018, [xx])
- Prediction Markets : Theory and Applications (Michael Edward Ruberry, 2013, [xx])
- Favorite-longshot bias in European Football betting market : Differences between popular and non-popular football competitions (Daniël van Raaij, 2019, [xx])

I would say that the first two papers in this list inspired me the most,

I also documented myself towards GitHub repositories made available by their creators. These projects were really helpful to me, because even though none of them was coded in R, the structure of their pipeline and the explanations represented a huge time gain to me :

- *Bet on Sibyl* application code ([xx])
- Beating the bookmakers on tennis matches ([xx])
- Soccer betting ([xx])

V - OBJECTIVES OF THE PROJECT AND PIPELINE PRESENTATION

(A) PROBLEM OVERVIEW AND REFORMULATION

The National Basketball Association (NBA) is the American basketball league, composed of 30 teams – including the only canadian team the Toronto Raptors – who compete all season long to win the championship. The regular season is composed of 82 games per team, which add up to 1230 games in total. The regular season starts around late october and finishes around the 15th of april, therefore there are multiple games each night during this period (around 50 games per week). The 30 teams are divided into 2 conferences, which are composed of three divisions that each contain 5 teams [insert scheme]. Finally, the best 8 teams from each conference participate in the playoffs, which first designates the conferences winner, before the Finals series which designates the NBA champion. Each round of the playoffs is designed as a best-of-7 game series.

The main idea was, from the raw historical data, to create a set of predictive variables $\{Vi\}_{1 \le i \le n}$ and a linear sequence of mathematical functions $\{Ti\}_{1 \le i \le m}$ such that applying the convolution of the mathematical functions to the initial raw variables would result in an accurate estimation of victories for a given NBA game :

$$\left(\prod_{i=1}^{m} T_i\right)(V_1, \dots, V_n) = \left(\widehat{P_{model}}(winner = hometeam\,), \widehat{P_{model}}(winner = awayteam)\right)$$

$$\text{With } \widehat{P_{model}}(winner = hometeam\,) + \widehat{P_{model}}(winner = awayteam\,) = 1$$

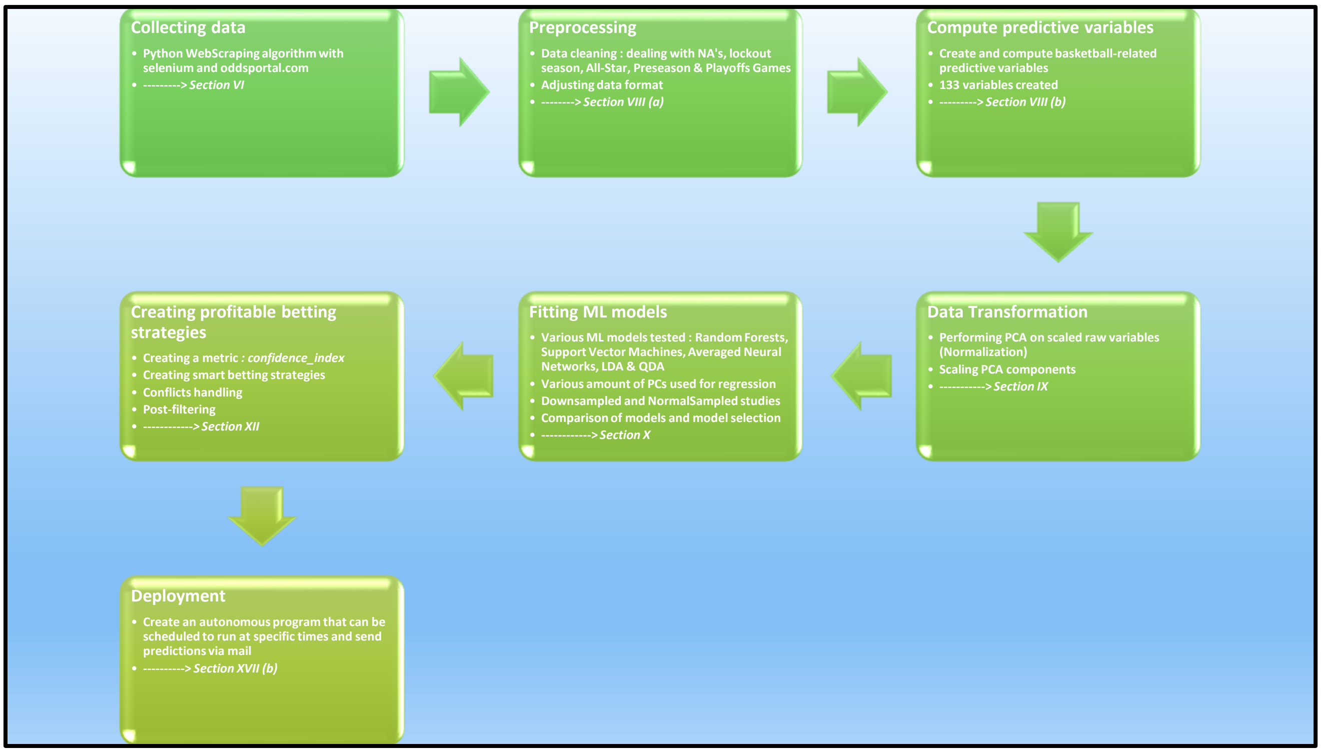
A bookmaker, such as the one I studied during this work, is a company whose function is to take bets from gamblers. Bookmakers propose various betting markets which depend on the bookmaker itself, for the NBA the most usual markets are the *moneyline* (predicting the winning team), the *handicap* (predicting marging of victory, spread) and the *total* (predicting if the number of points is below or above a fixed threshold). This said, when a gambler bets on an event, he stakes an amount S of real currency onto the realization of the event, at a fixed odd C : if the event happens, the gambler takes back a total earnings of $C * S$, which represents a net profit of $(C - 1) * S$, and if the event doesn't happen then he just lost his stake S.

(B) STARTING FRAMEWORK

The initial objectives were to create an ultra precise model, which should have been able to predict accurately 80% of the games. However, I quickly realized my expectations were way too high and out of my range – if even possible. After having documented myself towards the subject, I realized that many of the similar projects only settled for creating the most accurate model but most of the time they couldn't exploit their model to make huge profit in sports betting because the bookmakers margin still compensated for the good accuracy of their models. I then tried to figure out how to counter this, and started creating my own models, then I thought about a way to identify biases in the bookmakers estimation.

The objectives were obviously financial, but the challenge of creating such a software from scratch really interested me. I've always been a huge fan of sports, and this was the opportunity to mix up my skills in both sports knowledge and Data Science fields.

In order to fullfill the many objectives, I created a whole pipeline from scratch, which can be represented in the following way :



Collecting data
Python WebScraping algorithm with selenium and oddsportal.com
---------> Section VI
Preprocessing
Data cleaning : dealing with NA's, lockout season, All-Star, Preseason & Playoffs Games
Adjusting data format
--------> Section VIII (a)
Compute predictive variables
Create and compute basketball-related predictive variables
133 variables created
---------> Section VIII (b)
Data Transformation
Performing PCA on scaled raw variables (Normalization)
Scaling PCA components
-----------> Section IX
Fitting ML models
Various ML models tested : Random Forests, Support Vector Machines, Averaged Neural Networks, LDA & QDA
Various amount of PCs used for regression
Downsampled and NormalSampled studies
Comparison of models and model selection
------------> Section X
Creating profitable betting strategies
Creating a metric : confidence_index
Creating smart betting strategies
Conflicts handling
Post-filtering
------------> Section XII
Deployment
Create an autonomous program that can be scheduled to run at specific times and send predictions via mail
----------> Section XVII (b)

<u>Figure xx :</u> Descriptive pipeline of the project

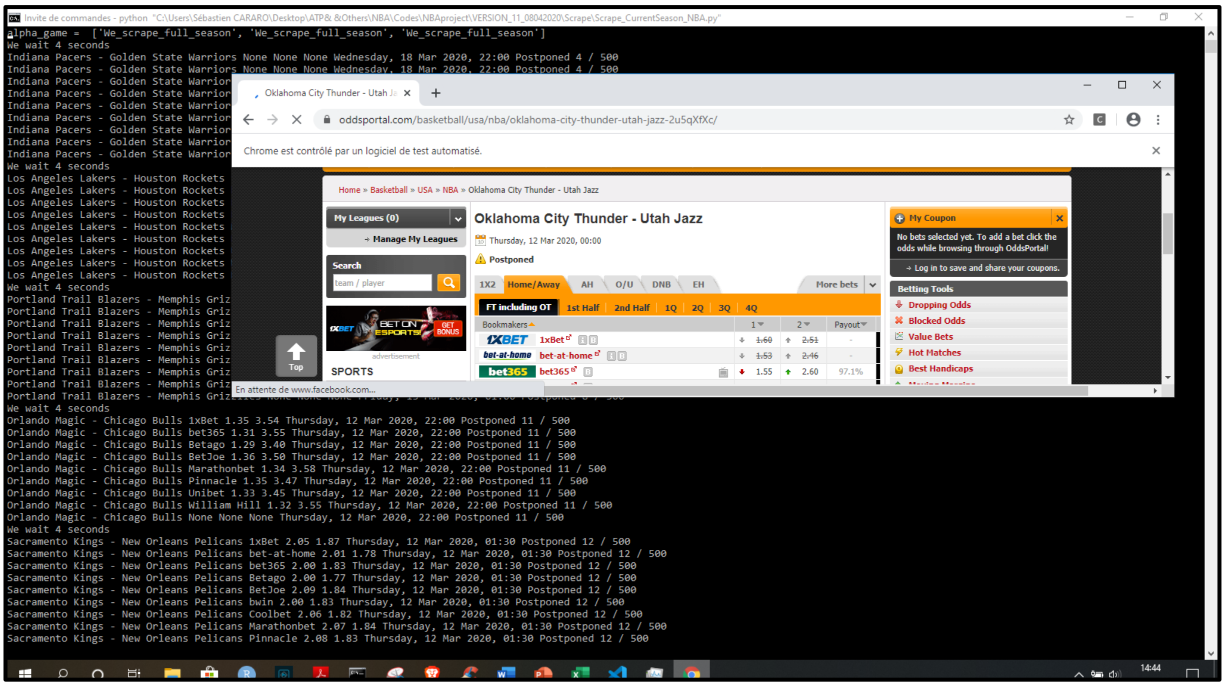
VI - DATA SOURCES AVAILABLE

The only raw data that I needed to start from was an historical dataset which contained the result of each game, along with the teams names and the closing odds proposed by the bookmaker Bet365[xx]. Closing odds are the odds available just before the beginning of the match and it is very important to have this particular data because odds can fluctuate a lot between the opening and the closing time, mainly to cope with players injuries and with gamblers staking one or another team.

In order to create this dataset, I chose to create a web scraping algorithm in Python [xx] to collect historical closing odds displayed on the website oddsportal.com, since season 2009-2010. The program was coded in order to collect the basic information for each game, *i.e.* the following variables :

- Date of the match
- Teams id
- Final score

- Money line odds proposed by the bookmaker bet365.com



```
Invite de commandes - python  "C:\Users\Sébastien CARARO\Desktop\ATP& &Others\NBA\Codes\NBAproject\VERSION_11_08042020\Scrape\Scrape_CurrentSeason_NBA.py"
alpha_game =  ['We_scrape_full_season', 'We_scrape_full_season', 'We_scrape_full_season']
We wait 4 seconds
Indiana Pacers - Golden State Warriors None None None Wednesday, 18 Mar 2020, 22:00 Postponed 4 / 500
Indiana Pacers - Golden State Warriors None None None Wednesday, 18 Mar 2020, 22:00 Postponed 4 / 500
Indiana Pacers - Golden State Warrior
Indiana Pacers - Golden State Warrior
Indiana Pacers - Golden State Warrior
Indiana Pacers - Golden State Warrior
Indiana Pacers - Golden State Warrior
Indiana Pacers - Golden State Warrior
Indiana Pacers - Golden State Warrior
We wait 4 seconds
Los Angeles Lakers - Houston Rockets
Los Angeles Lakers - Houston Rockets
Los Angeles Lakers - Houston Rockets
Los Angeles Lakers - Houston Rockets
Los Angeles Lakers - Houston Rockets
Los Angeles Lakers - Houston Rockets
Los Angeles Lakers - Houston Rockets
Los Angeles Lakers - Houston Rockets
Los Angeles Lakers - Houston Rockets
We wait 4 seconds
Portland Trail Blazers - Memphis Griz
Portland Trail Blazers - Memphis Griz
Portland Trail Blazers - Memphis Griz
Portland Trail Blazers - Memphis Griz
Portland Trail Blazers - Memphis Griz
Portland Trail Blazers - Memphis Griz
Portland Trail Blazers - Memphis Griz
Portland Trail Blazers - Memphis Griz
Portland Trail Blazers - Memphis Griz
We wait 4 seconds
Orlando Magic - Chicago Bulls 1xBet 1.35 3.54 Thursday, 12 Mar 2020, 22:00 Postponed 11 / 500
Orlando Magic - Chicago Bulls bet365 1.31 3.55 Thursday, 12 Mar 2020, 22:00 Postponed 11 / 500
Orlando Magic - Chicago Bulls Betago 1.29 3.40 Thursday, 12 Mar 2020, 22:00 Postponed 11 / 500
Orlando Magic - Chicago Bulls BetJoe 1.36 3.50 Thursday, 12 Mar 2020, 22:00 Postponed 11 / 500
Orlando Magic - Chicago Bulls Marathonbet 1.34 3.58 Thursday, 12 Mar 2020, 22:00 Postponed 11 / 500
Orlando Magic - Chicago Bulls Pinnacle 1.35 3.47 Thursday, 12 Mar 2020, 22:00 Postponed 11 / 500
Orlando Magic - Chicago Bulls Unibet 1.33 3.45 Thursday, 12 Mar 2020, 22:00 Postponed 11 / 500
Orlando Magic - Chicago Bulls William Hill 1.32 3.55 Thursday, 12 Mar 2020, 22:00 Postponed 11 / 500
Orlando Magic - Chicago Bulls None None None Thursday, 12 Mar 2020, 22:00 Postponed 11 / 500
We wait 4 seconds
Sacramento Kings - New Orleans Pelicans 1xBet 2.05 1.87 Thursday, 12 Mar 2020, 01:30 Postponed 12 / 500
Sacramento Kings - New Orleans Pelicans bet-at-home 2.01 1.78 Thursday, 12 Mar 2020, 01:30 Postponed 12 / 500
Sacramento Kings - New Orleans Pelicans bet365 2.00 1.83 Thursday, 12 Mar 2020, 01:30 Postponed 12 / 500
Sacramento Kings - New Orleans Pelicans Betago 2.00 1.77 Thursday, 12 Mar 2020, 01:30 Postponed 12 / 500
Sacramento Kings - New Orleans Pelicans BetJoe 2.09 1.84 Thursday, 12 Mar 2020, 01:30 Postponed 12 / 500
Sacramento Kings - New Orleans Pelicans bwin 2.00 1.83 Thursday, 12 Mar 2020, 01:30 Postponed 12 / 500
Sacramento Kings - New Orleans Pelicans Coolbet 2.06 1.82 Thursday, 12 Mar 2020, 01:30 Postponed 12 / 500
Sacramento Kings - New Orleans Pelicans Marathonbet 2.07 1.84 Thursday, 12 Mar 2020, 01:30 Postponed 12 / 500
Sacramento Kings - New Orleans Pelicans Pinnacle 2.08 1.83 Thursday, 12 Mar 2020, 01:30 Postponed 12 / 500
```

Oklahoma City Thunder - Utah Ja
oddsportal.com/basketball/usa/nba/oklahoma-city-thunder-utah-jazz-2u5qXfXc/
Chrome est contrôlé par un logiciel de test automatisé.
Home » Basketball » USA » NBA » Oklahoma City Thunder - Utah Jazz
My Leagues (0)
Manage My Leagues
Search
team / player
Oklahoma City Thunder - Utah Jazz
Thursday, 12 Mar 2020, 00:00
Postponed
1X2 Home/Away AH O/U DNB EH
FT including OT 1st Half 2nd Half 1Q 2Q 3Q 4Q
Bookmakers 1 2 Payout
1xBet 1.60 2.51
bet-at-home 1.53 2.46
bet365 1.55 2.60 97.1%
My Coupon
No bets selected yet. To add a bet click the odds while browsing through OddsPortal!
Log in to save and share your coupons.
Betting Tools
Dropping Odds
Blocked Odds
Value Bets
Hot Matches
Best Handicaps
SPORTS
Top
En attente de www.facebook.com...
14:44

Figure xx : *Screenshot of the web scraping algorithm running*

Home_id	Away_id	B365_OT_H	B365_OT_A	Score_home	Score_away	Seasons	Date	x	MatchId
CLE	BOS	1.5	2.7	89	95	2009/2010	27/10/2009	1	1
DAL	WAS	1.25	4.2	91	102	2009/2010	27/10/2009	1	2
POR	HOU	1.18	5.25	96	87	2009/2010	28/10/2009	1	3
LAL	LAC	1.15	5.5	99	92	2009/2010	28/10/2009	1	4
TOR	CLE	3	1.41	101	91	2009/2010	28/10/2009	1	5
ORL	PHI	1.22	4.5	120	106	2009/2010	28/10/2009	1	6

Figure xx : *Scraped table overview*

I decided not to use in-game statistics neither player injuries information, mostly because I didn't find any reliable source to access this data on a daily basis. Furthermore, creating a player-based predictive model might have been too much of a charge for my computer and I, given the fact that the study would then have been tremendously heavier.

VII - SOFTWARES USED

I coded everything from scratch in R 3.6.0, a sample of the librairies that I used is :

- *ggplot2* for plotting results
- *MASS* for Machine Learning models and tools
- *e1071* for the Support Vector Machines models
- *nnet* for the averaged Neural Networks
- *dplyr* for better code clarity
- *taskscheduleR* to schedule R tasks from R interface (automation process – *Section XVII (b)*)

Considering the lack of publicly-available tools - especially in R - I had to create the vast majority of the functions by myself. The web scraping program was developed in Python 3.6.2 using the Pyzo environment, and a few usual and web scraping librairies were used such as *selenium*, *pandas*, *numpy* and *urllib*. The webscraping process is later used to automatically scrape odds right before the game, in order not to have to do it by hand.

VIII - PREPROCESSING STEPS

(A) EXPLORATORY ANALYSIS AND DATA CLEANING

Once I had scraped the raw source dataset, I first proceeded to an exploratory analysis in order to be familiar with the dataset and anticipate the next steps. I found out that there were a couple variations in the data, and that I had to proceed to data cleaning before any further step. First I excluded the playoffs games from the study, given the fact that for the moment I solely focused on the regular season, plus the dynamics during playoffs is certainly different from the regular season dynamics. Apart from this, a few games had odds missing : I decided to replace the missing value by the median value of the home/away odd in the dataset (9 matchs over the whole dataset).

After inspecting the amount of games per season, I noticed that the 2011-2012 season contained only 990 games : this is due to the fact this season was a *lockout* season, *i.e.* there was a players strike during this season which resulted in a number of cancelled games. This season was then only composed of 66 games per team.

(B) COMPUTE PREDICTIVE VARIABLES

The next step was to create predictive variables in order to fit the differents Machine Learning models on. I coded a total number of 133 features for this program, which are noted down in the appendix n°1 at the end of this report.

I chose to encode common variables used in sport analytics such as the offensive & defensive performance (number of points scored/against), the season record, the number of rest days since last game, along with multiple other measures. Furthermore, I also used odds-derived metrics as predictive variables : I computed the Return-On-Investment (*ROI*) for each team, when we bet them winning or losing, when they're playing home or away, the average odd when winning/losing, the biggest odd passed this season, etc… These metrics give precious information about teams strength evaluation by bookmakers, as we can then deduce whether a team is overperforming this season, or on the other case if the team is underperforming according to the bookmakers' estimation.

Advancing towards my study, I discovered the existence of a rating system for sports teams called Elo [xx] which is known for being a good estimation of a team power. At first I decided to directly use the historical Elo rating dataset proposed by the website fivethirtyeight [xx], which is one of the most broadly followed website concerning sports predictions and analytics-based analysis. …). The open-source ELO table can be found in their github page [xx] and is updated on a daily basis, which fitted my daily utilization of the NBA algorithm.

However, I later decided to compute the variable by myself, in order to being able to generalize the calculation on the future leagues and sports (which don't have similar publicly available datasets

I then implemented this metrics as a predictive variable, following the downwritten recursive simplified formula :

$$Elo_{teamA\ match1} = 1500$$

- ✓ <u>If teamA won match n°i against team B:</u>

$$Elo_{teamA\ match\ i+1} = Elo_{teamA\ match\ i} + K * \frac{1}{1 - 10^{\frac{Elo_{teamB\ match\ i} - Elo_{teamA\ match\ i}}{400}}}$$

- ✓ <u>Else if teamA lost the match n°I gainst team B:</u>

$$Elo_{teamA\ match\ i+1} = Elo_{teamA\ match\ i} - K * \frac{1}{1 - 10^{\frac{Elo_{teamA\ match\ i} - Elo_{teamB\ match\ i}}{400}}}$$

With K being a dynamics constant which I set to 32, and

$$\frac{1}{1 - 10^{\frac{Elo_{teamA\ match\ i} - Elo_{teamB\ match\ i}}{400}}}$$

being the probability that teamB wins the game, according to this Elo ranking system.

The Elo rating is a zero-sum system, *i.e.* the total amount of ELO points stays constant over the season as a consequence of the mathematical formulation. As one can expect from the formula, this rating puts the emphasis on strength difference for points earnings : an unexpected win from a weaker team against a supposedly stronger one will make the team win more points than a win against a supposedly weaker opponent. ([xx] – FiveThirthyEight Elo description).

In the end, the ELO repartion histogram has the following bell curve shape – centered in 1500 - over the course of the 10 years period :

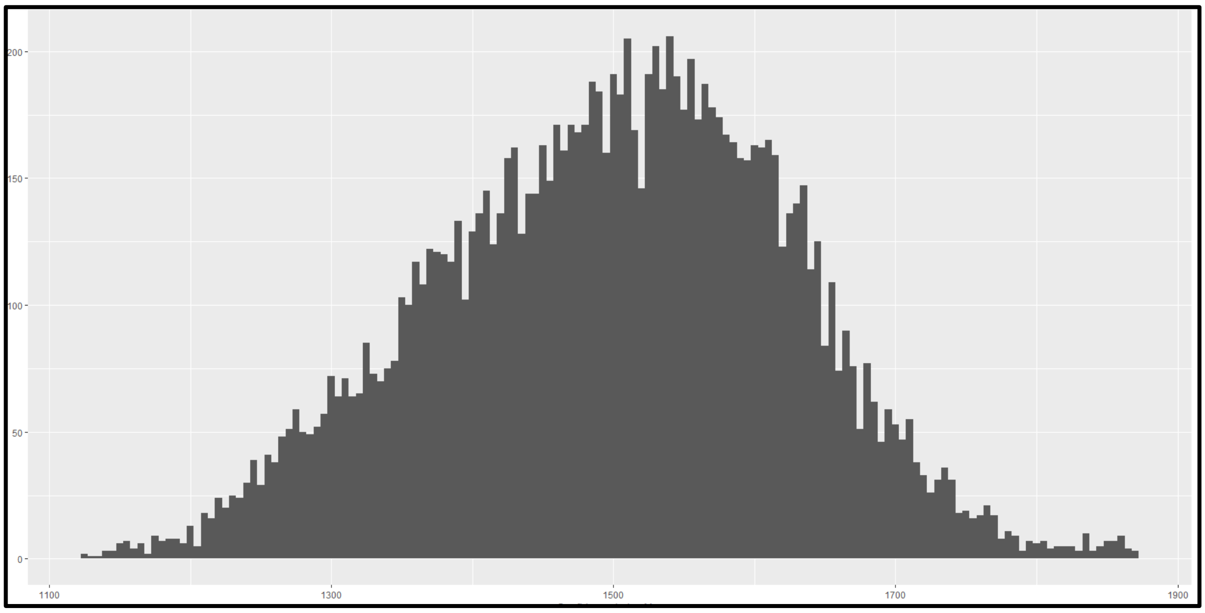


Figure xx : ELO repartition in the NBA dataset

For each match, I computed the statistics concerning all matches this season, but also concerning home games for the home team, away games for the away team, the last 5 and last 10 games, the head-to-head record, along with some other composite variables, considering multiple combinations of the listed attributes. I also computed some percentages associated with the stats, *e.g.* once I had computed the number of wins and the number of loss this season, I could easily compute the percentage of victory this season.

I also computed statistics who took into account the previous seasons, so that the machine learning models would also learn from long-term dynamics to make predictions aswell.

IX - PCA

(A) VARIABLES OVERVIEW

After having correctly computed the 133 predictive variables, we can first have a look at the correlation matrices, which we can compute according to different metrics. I chose to represent the correlation matrices with Spearman and Pearson methods, which I represent there down – the matrix are very similar, as expected :

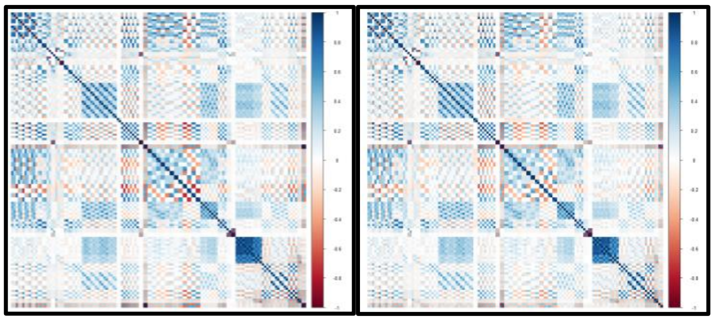
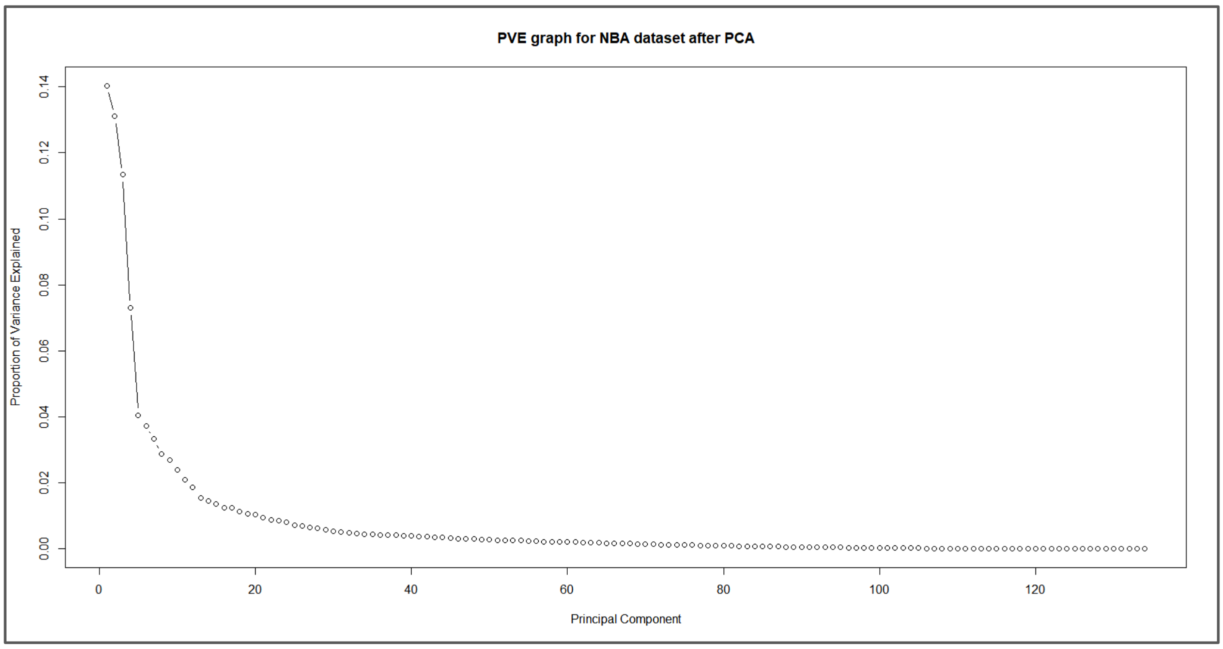


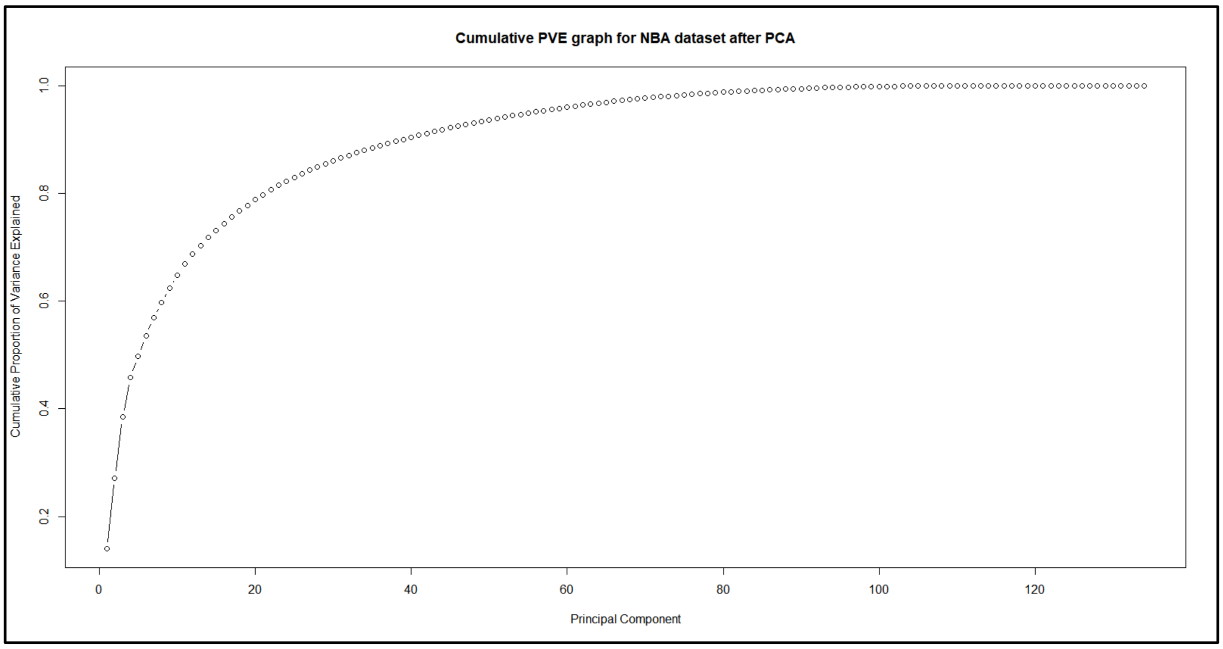
Figure xx : Correlation matrices between predictive variables

- ✓ Comments : As a general statement, we can notice that a lot of the variables are totally uncorrelated, which might be a good sign as this would mean that we can capture different statistical patterns in the dataset. This could also mean that we have noise in the dataset, however because of their definition I would think that most of the variables are related to the outcome. The next section, Principal Component Analysis, is a great tool for filtering out the possible noisy information in the dataset.

(B) APPLYING PCA

After having computed all variables and before fitting my models, I decided to apply Principal Component Analysis through the Eigen Vectors Decomposition (EVD PCA - [xx]). This widely used technique creates a new set of components from the initial 133 variables, in order to create a set of orthogonal components who concentrate the variance in the first components. This is a very powerful technique to enhance the Signal-To-Noise ratio (SNR – [xx]) ratio because we can then select the first components in order to capture most of the information. It can typically be a huge benefit to our study because many variables are highly correlated, hence we will remove the repeated correlation with this step. I chose to scale and normalize the original variables before applying PCA.





Figures xx & xx : PVE graph and Cumulative PVE graph

- ✓ Comments : We can identify a clear "elbow" pattern in the Cumulative Proportion of Variance Explained (Cumulative PVE) graph, which might indicate that from the 80th Principal Component the additional information contained in not that important (*i.e.* that there is not so much variance in the last PCs). We will perform cross validation when building the models, in order to compare the results we obtain with differents sets of PCs. The final goal will be to determine the optimal number of PCs such that the bias/variance tradeoff is as good as possible in terms of accuracy.

(C) SCALING THE PRINCIPAL COMPONENTS

In order to gain time during the Machine Learning models training, I decided to scale the output of the PCA, such that the final variables are all centered to 0 and scaled to have *standard deviation* = 1. Such transformation is usually recognized as being a gain of training time, plus it may also put the emphasis on using not only the very first PCs to train the models. The models would then adjust the relative PCs importance during the training process.

X - FITTING MODELS : THE DIFFERENT TYPES OF MODELS, PARAMETERS

I chose to fit a couple models over the newly created principal components, and to fit them with different hyperparameters settings. The four types of models that I used were Random Forests [xx], Support Vector Machines [xx], averaged Neural Networks [xx], and Discrimination Analysis (Linear/Quadratic) [xx]. These models were trained to predict the outcome of the game in a binary way : either predict the home team ('H') or the away team ('A') as winner. I also fitted the models to output estimated probabilities of victory. All of the aforementioned models are supervised models, *i.e.* we train the models to predict Y given x, as opposed to unsupervised methods which are only trained to cluster the observations, using the predictive variables x. All models were trained using "*accuracy*" as a metric, even though various criterions such as *gini*, *entropy* or *Area-Under-the-Curve (AUC)* could have been used in this case of binary classification. Such new implementations could be the subject of future work (*hyperparameters tuning*).

I also performed two types of study : the first one considering the downsampled training set, *i.e.* removing training samples from the majority class in order to train the models with a balanced dataset, and a second study where I used the whole training set to train the models. The class repartition on each study is presented there down :

Training set :	**Downsampled**	**Normal Sampled**
Home Victories	4445	6379
Away Victories	4445	4445

Table xx : Training sets class repartition

Downsampling is a widely used method to cope with imbalanced training sets, which can cause some models to overpredict the majority class in the case of future predictions. Some models such as Neural Networks and SVM are known to cope well with such cases, but for instance Random Forests can diverge fastly when the offset is too wide. In our case the raw training set is composed of 58.93 % of home victories, therefore the coherence of such study. On the other hand, downsampling forces us to let aside 1934 training observations, reducing the size of the training set and hence causing us to lose information. We will see in the *section XI* which set of models performed better.

Other methods can be used to deal with imbalanced datasets, such as upsampling and upweighting ([xx]).

(A) RANDOM FORESTS

Random Forests is a powerful tree-based algorithm, that enhances the original classification tree thanks to the use of heuristics, which allow the training process to be more robust to new observations : deeply grown classification trees are known for not being very robust when confronting to new observations, this phenomenon is known as overfitting [xx] and is a common problem in the Machine Learning field.

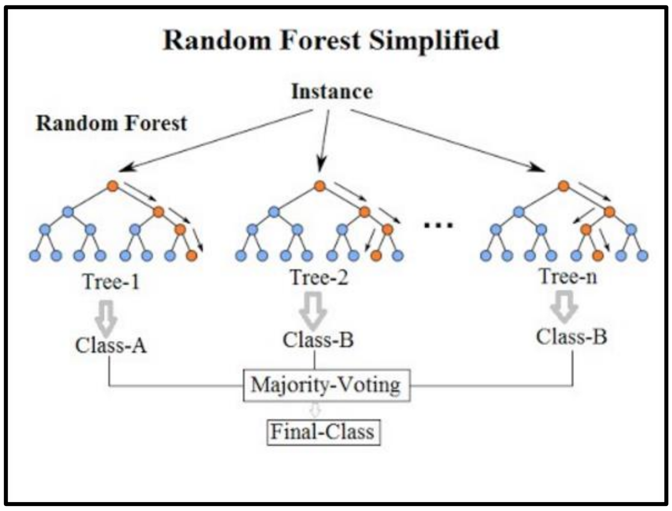
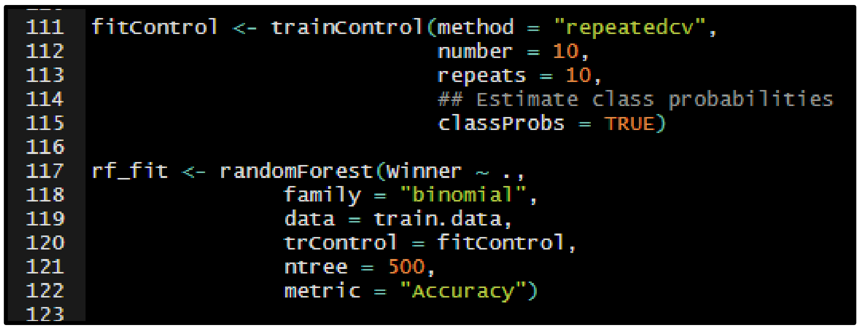


Figure xx : Random Forest construction illustrated ([xx])

The models were fitted using 10-fold cross validation, and I used a total of 500 trees to construct the model.



```
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 10,
                           ## Estimate class probabilities
                           classProbs = TRUE)

rf_fit <- randomForest(winner ~ .,
               family = "binomial",
               data = train.data,
               trControl = fitControl,
               ntree = 500,
               metric = "Accuracy")
```

Figure xx : Hyperparameters for Random Forest training

(B) SUPPORT VECTOR MACHINES

Support Vector Machines (SVM - [xx]) is a supervised machine learning that, given a set of observations described by a total of m predictive variables in the training set , seeks for the best hyperplane H in the space $\mathbb{R}^m$ that maximizes the separation between the two classes. Alternatively, the $\mathbb{R}^m$ decision space can also be transformed such that the decision hyperplane can have another shape (radial, polynomial kernels, personalized functions).

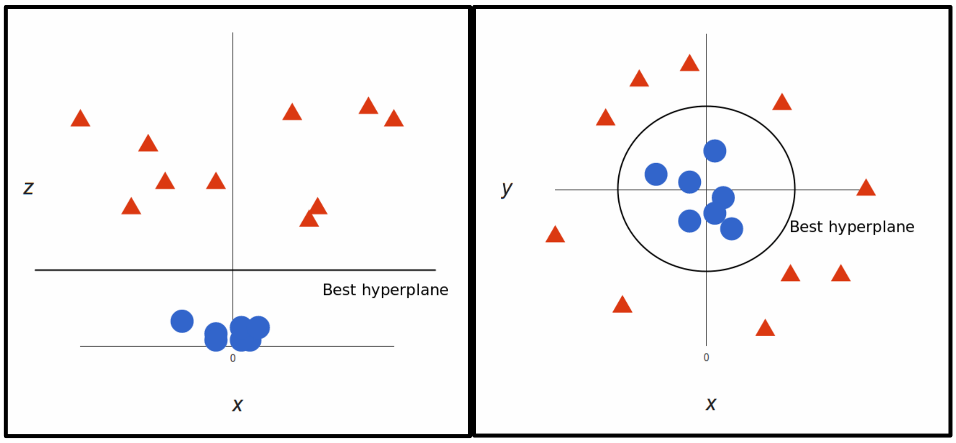
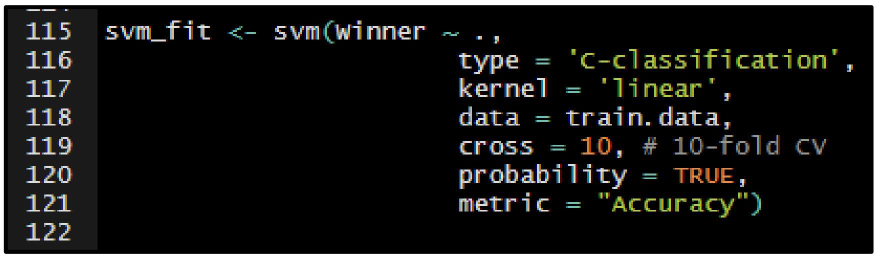


Figure xx : SVM decision process illustrated ([xx])



```
svm_fit <- svm(winner ~ .,
               type = 'C-classification',
               kernel = 'linear',
               data = train.data,
               cross = 10, # 10-fold CV
               probability = TRUE,
               metric = "Accuracy")
```

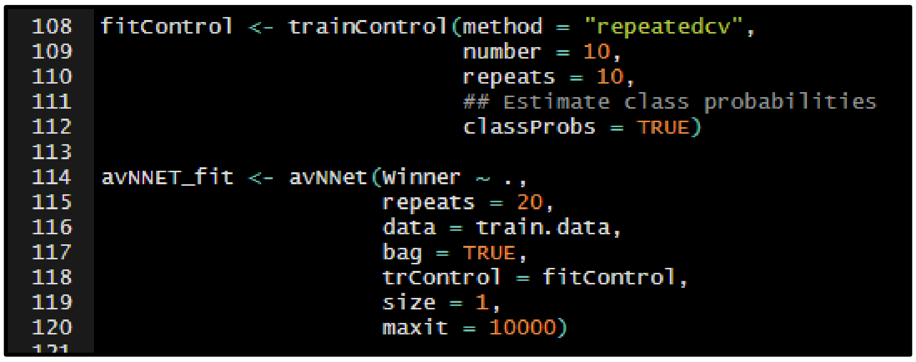
Figure xx : Hyperparameters for linear SVM training

(C) AVERAGED NEURAL NETWORKS

Neural Networks are considered a state-of-the-art technique for building powerful models that can cope with a huge number of variables and observations. Averaged Neural Networks is a construction process that, as during the Random Forest building process, builds and train a large number of Neural Networks with different initial random number seeds and then process to majority voting in order to reduce the variance related to building a single architecture.

Thanks to the *nnet* [xx] package implemented in R, I was able to train feedforward neural networks without having to get too deep into the architecture. This packages offers a very simple way to build and call such models, hence I decided to fit multiple sizes of them and integrates the various models into my set of models.

The models were built using 10-fold cross validation :



```r
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 10,
                           ## Estimate class probabilities
                           classProbs = TRUE)

avNNET_fit <- avNNet(Winner ~ .,
                     repeats = 20,
                     data = train.data,
                     bag = TRUE,
                     trControl = fitControl,
                     size = 1,
                     maxit = 10000)
```

Figure xx : Hyperparameters for avNNet size 1 training

(D) LDA AND QDA

Linear Discriminant Analysis (LDA – [xx]) is a Machine Learning model that is often compared to Principal Component Analysis, in the sense that the model seeks for the best linear combination of the predictive variables in order to differenciate between the classes. However, LDA is a supervised technique, which allows it to be way more precise in the case of binary classification.

Quadratic Discriminant Analysis (QDA – [xx]) is based on the same principle as LDA, nonetheless the discriminative criterion is different because QDA models assume a quadratic function as boundary between classes :

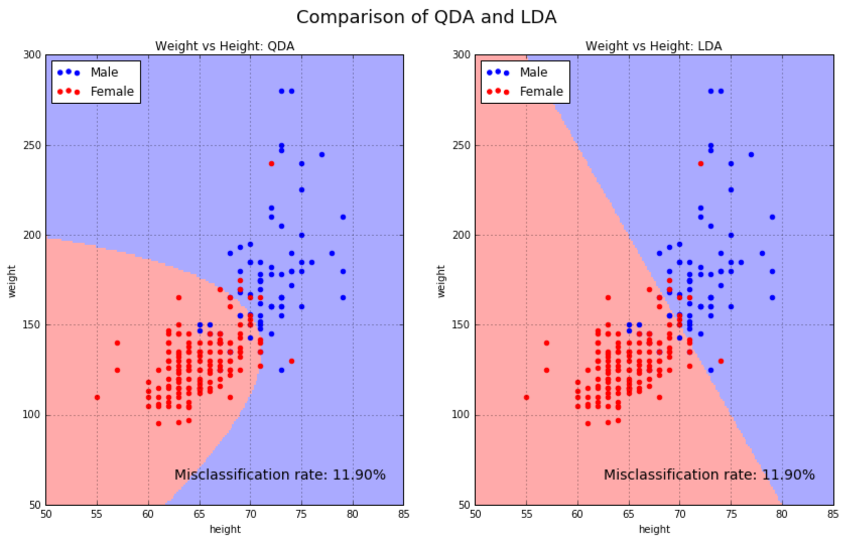
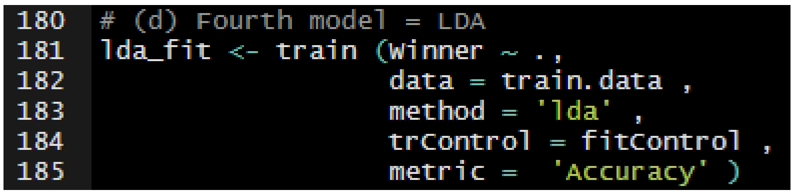


Figure xx : QDA and LDA illustration ([xx])

I fitted the two models using 10-fold cross validation :

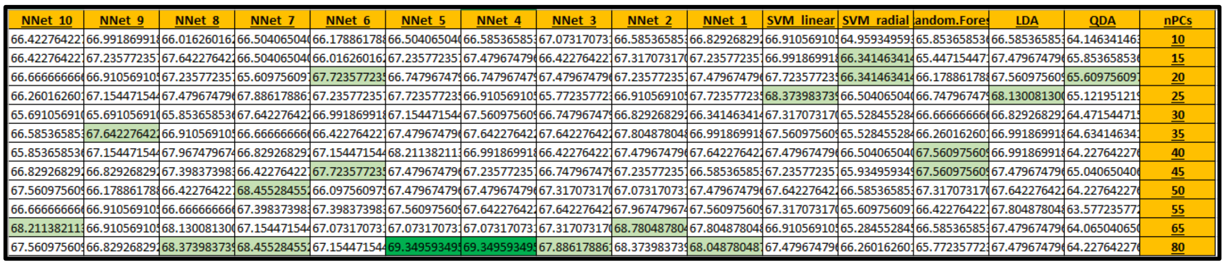


```r
# (d) Fourth model = LDA
lda_fit <- train (Winner ~ .,
                  data = train.data ,
                  method = 'lda' ,
                  trControl = fitControl ,
                  metric =  'Accuracy' )
```

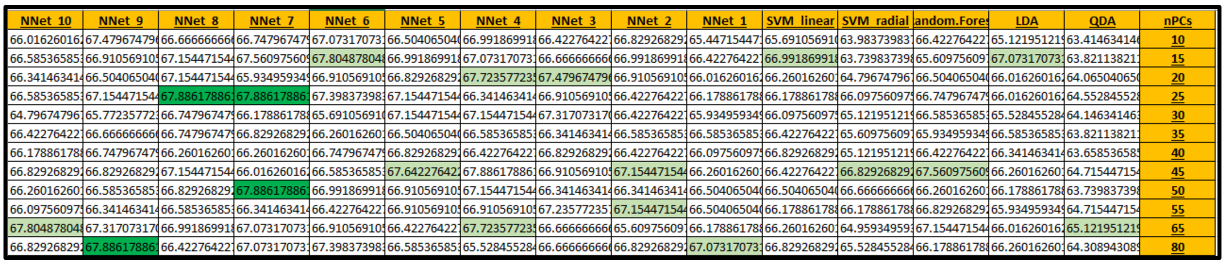
<u>Figure xx :</u> Hyperparameters for LDA

XI - COMPARISON BETWEEN MODELS

The test set for all models was composed of the very last season, hence I kept 9 seasons for training the models. The results, presented hereby, show a slight domination of the avNNet models concerning forecasting accuracy [include training times] :



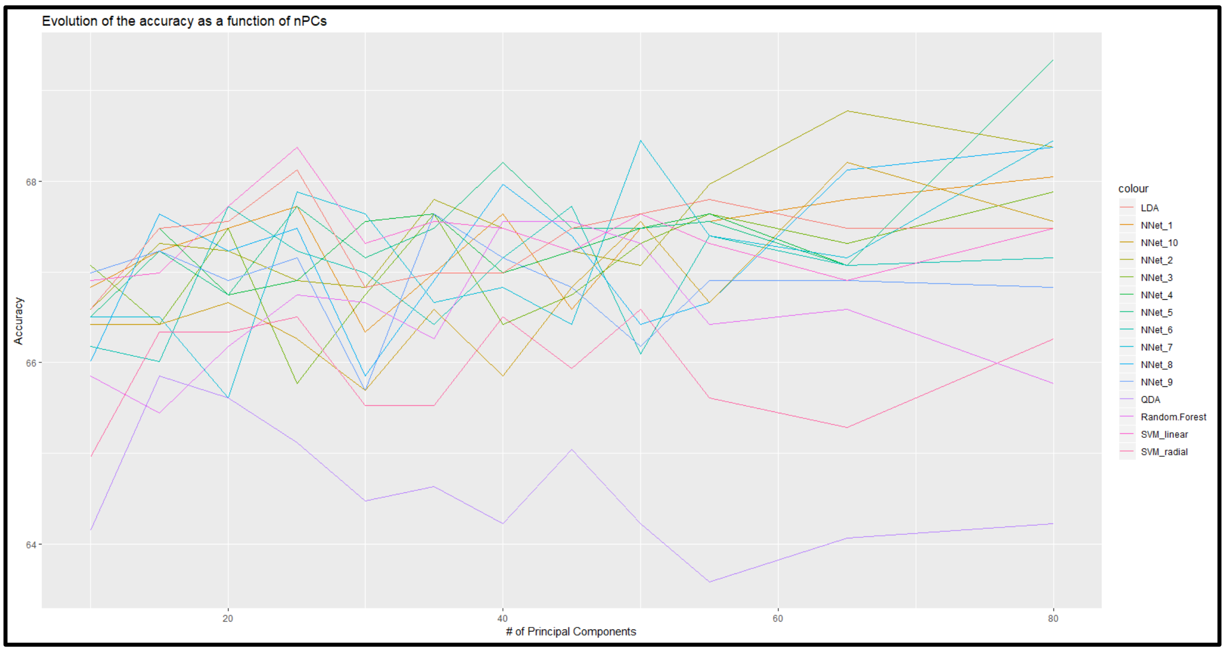
NNet 10	NNet 9	NNet 8	NNet 7	NNet 6	NNet 5	NNet 4	NNet 3	NNet 2	NNet 1	SVM linear	SVM radial	andom.Fores	LDA	QDA	nPCs
66.422764227	66.991869918	66.016260162	66.504065040	66.178861788	66.504065040	66.585365853	67.073170731	66.585365853	66.829268292	66.910569105	64.959349593	65.853658536	66.585365853	64.146341463	10
66.422764227	67.235772357	67.642276422	66.504065040	66.016260162	67.235772357	67.479674796	66.422764227	67.317073170	67.235772357	66.991869918	66.341463414	65.447154471	67.479674796	65.853658536	15
66.666666666	66.910569105	67.235772357	65.609756097	67.723577235	66.747967479	66.747967479	67.479674796	67.235772357	67.479674796	67.723577235	66.341463414	66.178861788	67.560975609	65.609756097	20
66.260162601	67.154471544	67.479674796	67.886178861	67.235772357	67.723577235	66.910569105	65.772357723	66.910569105	67.723577235	68.373983739	66.504065040	66.747967479	68.130081300	65.121951219	25
65.691056910	65.691056910	65.853658536	67.642276422	66.991869918	67.154471544	67.560975609	66.747967479	66.829268292	66.341463414	67.317073170	65.528455284	66.666666666	66.829268292	64.471544715	30
66.585365853	67.642276422	66.910569105	66.666666666	66.422764227	67.479674796	67.642276422	67.642276422	67.804878048	66.991869918	67.560975609	65.528455284	66.260162601	66.991869918	64.634146341	35
65.853658536	67.154471544	67.967479674	66.829268292	67.154471544	68.211382113	66.991869918	66.422764227	67.479674796	67.642276422	67.479674796	66.504065040	67.560975609	66.991869918	64.227642276	40
66.829268292	66.829268292	67.398373983	66.422764227	67.723577235	67.479674796	67.235772357	66.747967479	67.235772357	66.585365853	67.235772357	65.934959349	67.560975609	67.479674796	65.040650406	45
67.560975609	66.178861788	66.422764227	68.455284552	66.097560975	67.479674796	67.479674796	67.317073170	67.073170731	67.479674796	67.642276422	66.585365853	67.317073170	67.642276422	64.227642276	50
66.666666666	66.910569105	66.666666666	67.398373983	67.398373983	67.560975609	67.642276422	67.642276422	67.967479674	67.560975609	67.317073170	65.609756097	66.422764227	67.804878048	63.577235772	55
68.211382113	66.910569105	68.130081300	67.154471544	67.073170731	67.073170731	67.073170731	67.317073170	68.780487804	67.804878048	66.910569105	65.284552845	66.585365853	67.479674796	64.065040650	65
67.560975609	66.829268292	68.373983739	68.455284552	67.154471544	69.349593495	69.349593495	67.886178861	68.373983739	68.048780487	67.479674796	66.260162601	65.772357723	67.479674796	64.227642276	80



NNet 10	NNet 9	NNet 8	NNet 7	NNet 6	NNet 5	NNet 4	NNet 3	NNet 2	NNet 1	SVM linear	SVM radial	andom.Fores	LDA	QDA	nPCs
66.016260162	67.479674796	66.666666666	66.747967479	67.073170731	66.504065040	66.991869918	66.422764227	66.829268292	65.447154471	65.691056910	63.983739837	66.422764227	65.121951219	63.414634146	10
66.585365853	66.910569105	67.154471544	67.560975609	67.804878048	66.991869918	67.073170731	66.666666666	66.991869918	66.422764227	66.991869918	63.739837398	65.609756097	67.073170731	63.821138211	15
66.341463414	66.504065040	67.154471544	65.934959349	66.910569105	66.829268292	67.723577235	67.479674796	66.910569105	66.016260162	66.260162601	64.796747967	66.504065040	66.016260162	64.065040650	20
66.585365853	67.154471544	67.886178861	67.886178861	67.398373983	67.154471544	66.341463414	66.910569105	66.422764227	66.178861788	66.178861788	66.097560975	66.747967479	66.016260162	64.552845528	25
64.796747967	65.772357723	66.747967479	66.178861788	65.691056910	67.154471544	67.154471544	67.317073170	66.422764227	65.934959349	66.097560975	65.121951219	66.585365853	65.528455284	64.146341463	30
66.422764227	66.666666666	66.747967479	66.829268292	66.260162601	66.504065040	66.585365853	66.341463414	66.585365853	66.585365853	66.422764227	65.609756097	65.934959349	66.585365853	63.821138211	35
66.178861788	66.747967479	66.260162601	66.260162601	66.747967479	66.829268292	66.422764227	66.829268292	66.422764227	66.097560975	66.829268292	65.121951219	66.422764227	66.341463414	63.658536585	40
66.829268292	66.829268292	67.154471544	66.016260162	66.585365853	67.642276422	67.886178861	66.910569105	67.154471544	66.260162601	66.422764227	66.829268292	67.560975609	66.260162601	64.715447154	45
66.260162601	66.585365853	66.829268292	67.886178861	66.991869918	66.910569105	67.154471544	66.341463414	66.341463414	66.504065040	66.504065040	66.666666666	66.260162601	66.178861788	63.739837398	50
66.097560975	66.341463414	66.585365853	66.341463414	66.422764227	66.910569105	66.910569105	67.235772357	67.154471544	66.504065040	66.178861788	66.178861788	66.829268292	65.934959349	64.715447154	55
67.804878048	67.317073170	66.991869918	67.073170731	66.910569105	66.422764227	67.723577235	66.666666666	65.609756097	66.178861788	66.260162601	64.959349593	67.154471544	66.016260162	65.121951219	65
66.829268292	67.886178861	66.422764227	67.073170731	67.398373983	66.585365853	65.528455284	66.666666666	66.829268292	67.073170731	66.829268292	65.528455284	66.178861788	66.260162601	64.308943089	80

<u>Table xx and table xx :</u> Comparison between models accuracies over 2018-2019 season (up : raw dataset, down : downsampled dataset)

We can also plot this table for visualization purposes :



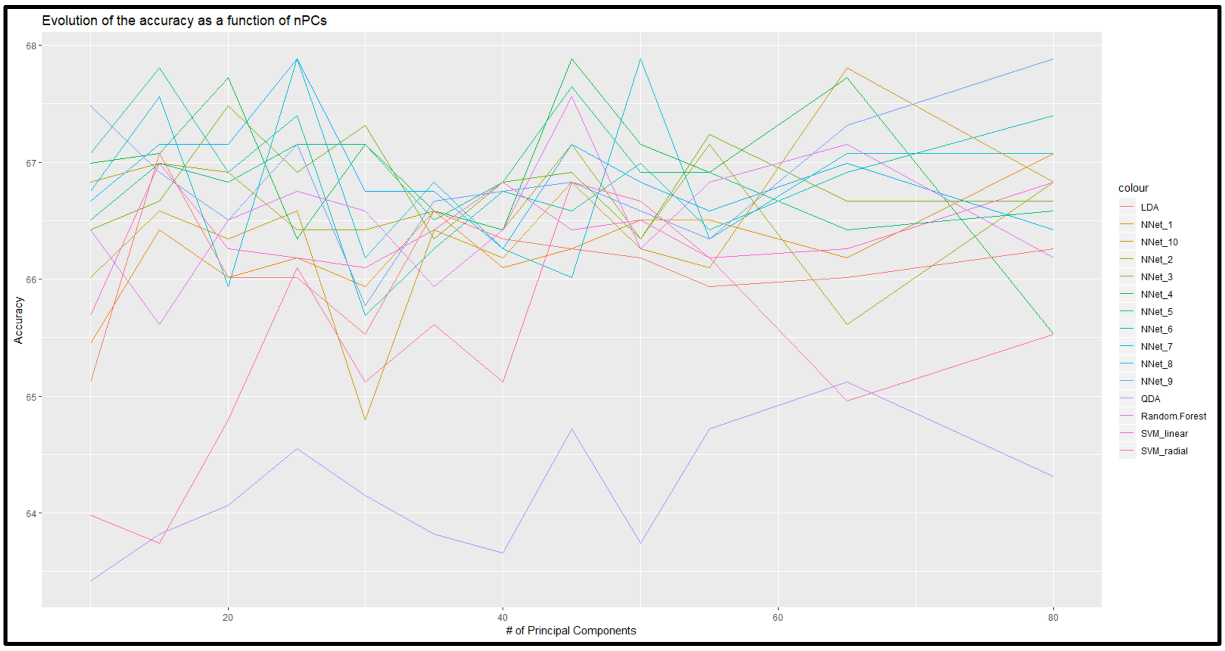


Figure xx and xx : Models comparative accuracies display (up : raw dataset, down : downsampled)

✓ Comments :

We can observe that the normal sampled training set (*i.e.* not downsampling the training set) yields slightly better results than the downsampled study, mainly because we remove observation samples during downsampling. The overall best model was able to correctly guess the winner of the game in **69,34 %** of the time over the 1230 games of the 2018/2019 NBA season (*Nnet 5 – 80 PC – NormalSampled*). The Neural Networks models are clearly dominating the accuracy tables, hence we could imagine their typical structure is able to perceive statistical patterns that the other classical models can not identify. Furthermore, we can also notice that the SVM radial models are overperforming the SVM linear models, and that the QDA models are also outperforming the LDA models : from this clear observation we can assume that the decision boundary between classes is likely to have a quadratic shape, and is surely not a linear boundary.

XII - CREATING PROFITABLE BETTING STRATEGIES

(A) STEP-BY-STEP PROCESS

As described earlier, the litterature shows a lot of very powerful models which take many parameters as input and employ a lot of computation power to compute the variables and the models. However, few of them are capable of yielding a great betting strategy, because even if the estimator accurately estimates chances of victory, one also has to cope with the bookmakers margin. Actually, this margin is settled around 5%, which means one should outperform the bookmakers models by more than 5%, which is practicaly impossible most of the time because the bookmakers models are state-of-the-art forecasting models, and they have way more information than I have.

The key point for understanding my strategy is to conceive we don't aim to create the perfect model, which is too difficult given the fact I don't have in-game data for recent matches, but instead I try to identify clear biases in the bookmaker's estimated chances of victory. To do this, I created a metric which represents the relative confidence offset between my estimated probabilities and the bookmaker's estimated probabilities.

The main innovation is the creation of a metric called confidence index - often referred to as *conf_index* in my scripts - which is the result, for a given game and a predictive model, of the odd multiplied by the estimated winning probability :

$$\begin{cases} ConfidenceIndex_{hometeam} = Odd_{home} * \widehat{P_{model}}(winner = hometeam) \\ ConfidenceIndex_{awayteam} = Odd_{away} * \widehat{P_{model}}(winner = awayteam) \end{cases}$$

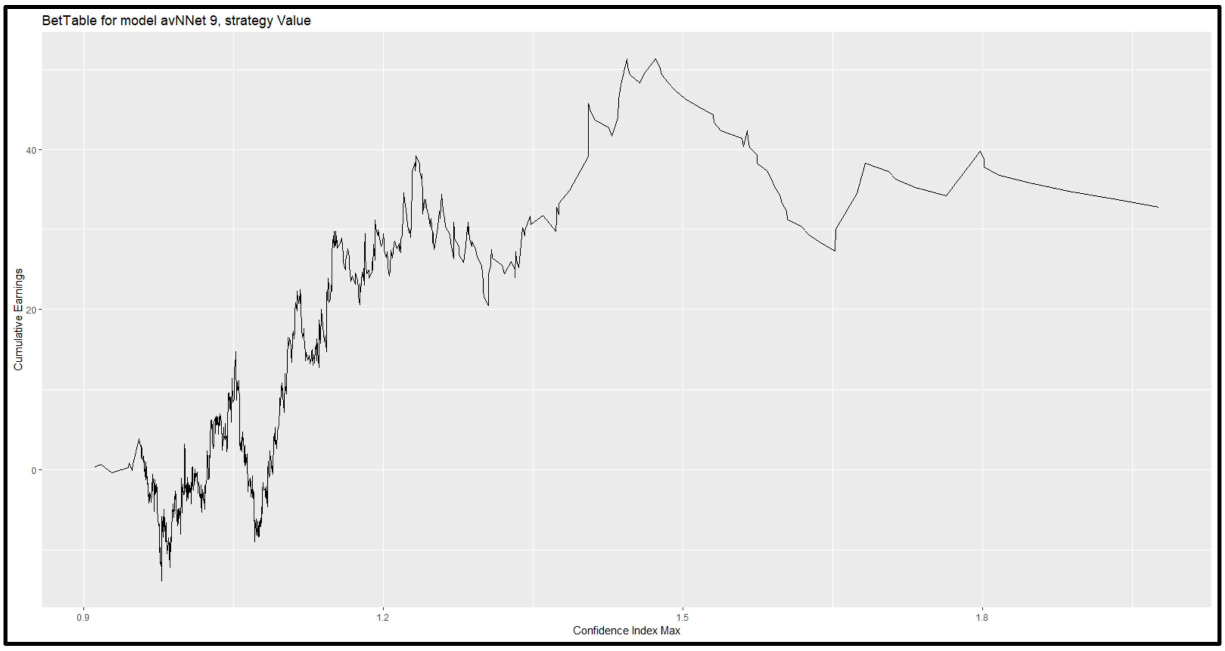
Eventually we compute the maximum of these two quantities to create a unique metric called *conf_index_max* :

$$ConfidenceIndexMax = \max\left(ConfidenceIndex_{hometeam}, ConfidenceIndex_{awayteam}\right)$$

This metric is supposed to represent the offset between my estimated probabilities of a team winning the game, and the bookmakers' estimated probability (we assume there that the odd should equals the inverse of the bookmakers' estimated probability). As theorically the odd should be equal to $\frac{1}{p}$, then the higher the confidence index, the higher the odd is considered a value bet. On the other hand, the lower the confidence index, the lower the bet is a value bet.

For each model, we can then sort the current season's games by order of $conf\_index\_max$, and display the cumulative earnings depending of the strategy of betting.

For example, for the LDA model with 65 PCs, the $ConfidenceIndexMax$ associated cumulative earnings when we bet on the value bet look like this :



- ✓ Interpretation : We can clearly distinguish at least two zones, according to the value of the $ConfidenceIndexMax$: when the metric is lower than 1.1 we would be losing money betting on the value bet, however the trend is straight opposed when the metric is between 1.1 and 1.47. By selecting games where the metric is between 1.1 and 1.47, we would then select profitable games, therefore earning money. In a sense it is quite understable, given the definition of the metric : the higher the ratio, the higher we estimate the offset between our estimated probability and the bookmakers' probability is higher, hence this looks like an opportunity to make money. We can still notice than when the metric is above 1.47, we are losing money, therefore we might want to select only games between 1.1 and 1.47, as mentioned above.

Furthermore, to complete the study, I identified 6 types of strategies, which are respectively betting of our favorite, on our value (*i.e.* the team which has the highest $conf\_index\_max$), on the hometeam when the Value is on betting home, or on the awayteam when the Value is on betting away, on the favorite when the Value is on the Favorite and finally on the outsider when the Value is on the outsider.

However, the identified region might aswell yield an increasing trend by pure chance, without being a revelator of any bias. But we will see in the results part at the end of the report that this technique works in the long term, hence we can be confident about this process for finding biases.

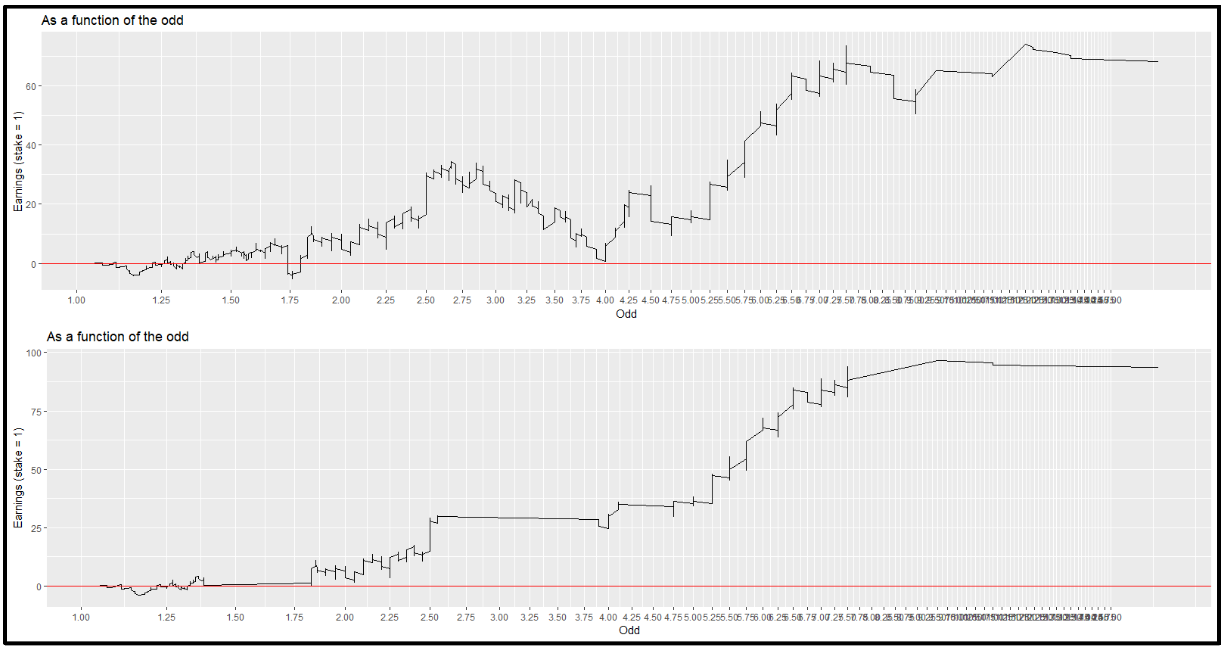
Finally, I repeated this technique of identifying potential biases over the six strategies, and over the different models. I then obtained a set of "Methods", which I define as the tuple $[Strategy, Model, region\ of\ potential\ bias]$.

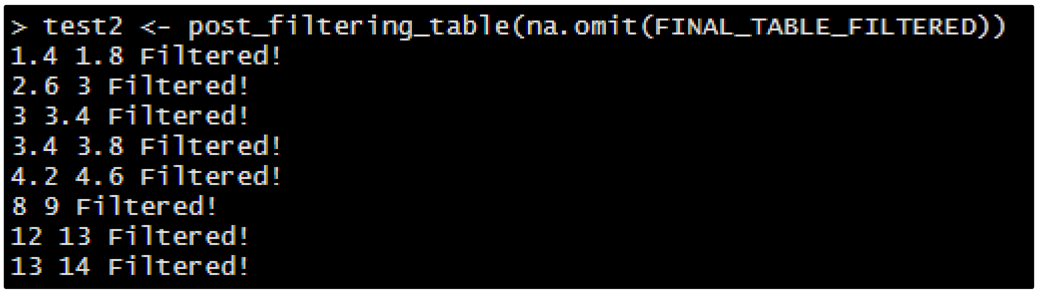
In our example it is then :

$$[Value, LDA, (1.34,1.56)]$$

Once I builded this set of methods, I then had to construct a conflict handling strategy, as sometimes the methods were yielding different bets for the same match. In such cases, I simply decided to choose the method yielding the best ROI over the season's matches at first, but I later evolved toward another metric to deal with conflicts (developed in next section *Automatic Implementation*)

Once this step done, I also filtered the predictions by having a look at badly predicted areas, depending on the odd :





```
> test2 <- post_filtering_table(na.omit(FINAL_TABLE_FILTERED))
1.4 1.8 Filtered!
2.6 3 Filtered!
3 3.4 Filtered!
3.4 3.8 Filtered!
4.2 4.6 Filtered!
8 9 Filtered!
12 13 Filtered!
13 14 Filtered!
```

After having done the selection steps described in the previous paragraph, there we are, the final process is finally over ! I will now describe how I automated the whole process, such that the program is now capable is updating itself and choosing the best strategies all season long, the latter evolving all season long, depending on the season's dynamics.

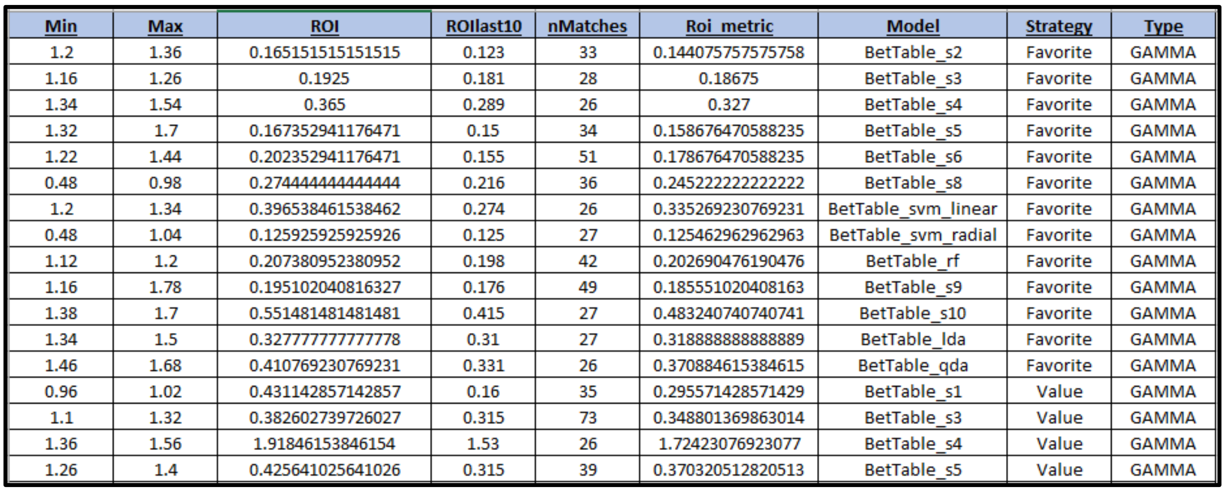
(B) AUTOMATIC IMPLEMENTATION

After having done by hand all the previously described steps, *i.e.* noting down the profitable segments and implementing them by hand on the code every two weeks, I decided to start coding an implementation of these steps so that the program would update itself automatically. The only way I could achieve that was by coding the following process :

1. Given one model, compute all probabilities for the m current season's games, then compute $ConfidenceIndexMax$ for each game, hence for the m first games we obtain a set $\{conf\_index\_max_i, \forall\ i\ \in\ [\![1, m]\!]\}$ which is ordered such that $i\ \leq j$ implies $conf\_index\_max_i\ \leq\ conf\_index\_max_j$
2. Compute all possible segments $\left\{[\![conf\_index\_max_i, conf\_index\_max_j]\!], \forall\ (i,j)\ \in\ [\![1, m]\!]^2, i\ < j\right\} = \{S_l\}_{1\ \leq\ l\ \leq\ \frac{m(m-1)}{2}}$
3. For every possible segment S_l, create a table T_l containing all games where $conf\_index\_\max \in S_l$. These tables are our starting set of possible betting methods.
4. Select only the tables T_l which contain at least 25 games, whose ROI is above 10 % and whose ROI over last 10 matches is above 10% aswell, creating a subset of profitable methods $\left\{T_{profitable_j}\right\}$
5. Select <u>only the highest ROI's method</u> from this set of selected methods (the metric can be changed). This step allows a great computation gain, however we might miss the other methods for this model but it is not a huge loss of information compared to the time gain.
6. Repeat steps 1 to 5 for every model in our set of models, and regroup all profitable methods.
7. These methods might eventually yield contradictory predictions for future games, hence to cope with these conflicts, for each predicted game we select only the highest ROI method (metric can be changed) among our profitable set of methods $\left\{T_{profitable_j}\right\}$

As the process computes all possible segments, we compute a total of $\frac{m(m-1)}{2}$ objects, hence this method has a complexity of $O(m^2)$. However, this exponential complexity is not a limitation to running the algorithm on a daily basis, as the running time is around 30 min and needs to be done at least twice in a week to be efficient.

The methods are then saved onto a csv file which is later used for predictions. The file looks like this :



Min	Max	ROI	ROIlast10	nMatches	Roi_metric	Model	Strategy	Type
1.2	1.36	0.165151515151515	0.123	33	0.144075757575758	BetTable_s2	Favorite	GAMMA
1.16	1.26	0.1925	0.181	28	0.18675	BetTable_s3	Favorite	GAMMA
1.34	1.54	0.365	0.289	26	0.327	BetTable_s4	Favorite	GAMMA
1.32	1.7	0.167352941176471	0.15	34	0.158676470588235	BetTable_s5	Favorite	GAMMA
1.22	1.44	0.202352941176471	0.155	51	0.178676470588235	BetTable_s6	Favorite	GAMMA
0.48	0.98	0.274444444444444	0.216	36	0.245222222222222	BetTable_s8	Favorite	GAMMA
1.2	1.34	0.396538461538462	0.274	26	0.335269230769231	BetTable_svm_linear	Favorite	GAMMA
0.48	1.04	0.125925925925926	0.125	27	0.125462962962963	BetTable_svm_radial	Favorite	GAMMA
1.12	1.2	0.207380952380952	0.198	42	0.202690476190476	BetTable_rf	Favorite	GAMMA
1.16	1.78	0.195102040816327	0.176	49	0.185551020408163	BetTable_s9	Favorite	GAMMA
1.38	1.7	0.551481481481481	0.415	27	0.483240740740741	BetTable_s10	Favorite	GAMMA
1.34	1.5	0.327777777777778	0.31	27	0.318888888888889	BetTable_lda	Favorite	GAMMA
1.46	1.68	0.410769230769231	0.331	26	0.370884615384615	BetTable_qda	Favorite	GAMMA
0.96	1.02	0.431142857142857	0.16	35	0.295571428571429	BetTable_s1	Value	GAMMA
1.1	1.32	0.382602739726027	0.315	73	0.348801369863014	BetTable_s3	Value	GAMMA
1.36	1.56	1.91846153846154	1.53	26	1.72423076923077	BetTable_s4	Value	GAMMA
1.26	1.4	0.425641025641026	0.315	39	0.370320512820513	BetTable_s5	Value	GAMMA

Figure xx : Output from the methods selection algorithm

I chose to select the methods according to another metric, instead of the ROI : I created the metric

$$ROI_{metric} = \frac{Roi + Roi\_last10}{2}$$

The choice of this metric is justified by the fact that then we can then select methods that are both working on the long-run and on the last matches, so that we select methods that are supposedly in great form aswell.

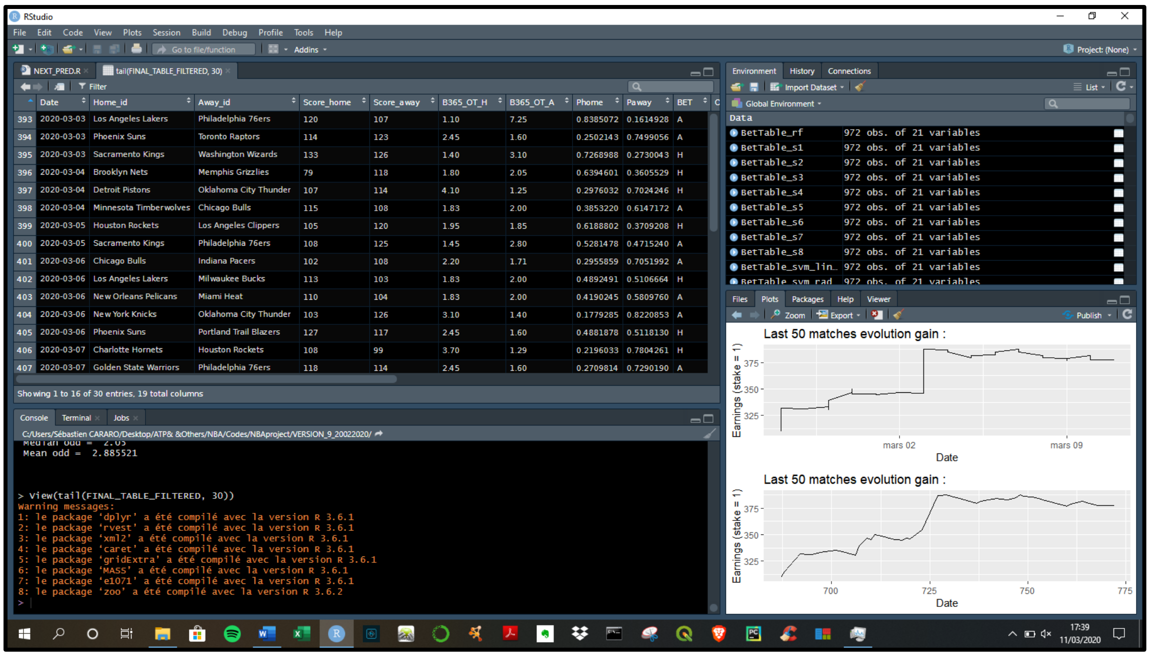
XIII - THE MATH BETWEEN THE BIASES FOUNDING THANKS TO THE TECHNIQUE

Looking forward to meeting a statistics researcher in order to ask questions about the mathematical background [September 2020]

XIV - CREATE A USER INTERFACE SUITABLE FOR EVERYBODY

Most of the tasks have now been automatically. As a result, the process to make predictions over the script is :

- 1 – Run NEXT_PRED.R : the code will automatically call the python webscraping function in order to collect the very last odds, then it will compute winning probabilities and predicted bets.
- 2 – The output is a table which displays in Rstudio – and is also sent by mail via an SMTP server directly from R - where we can read the predictions since the beginning of the season and the future games



	Date	Home_id	Away_id	Score_home	Score_away	B365_OT_H	B365_OT_A	Phome	Paway	BET
393	2020-03-03	Los Angeles Lakers	Philadelphia 76ers	120	107	1.10	7.25	0.8385072	0.1614928	A
394	2020-03-03	Phoenix Suns	Toronto Raptors	114	123	2.45	1.60	0.2502143	0.7499056	A
395	2020-03-03	Sacramento Kings	Washington Wizards	133	126	1.40	3.10	0.7268988	0.2730043	H
396	2020-03-04	Brooklyn Nets	Memphis Grizzlies	79	118	1.80	2.05	0.6394601	0.3605529	H
397	2020-03-04	Detroit Pistons	Oklahoma City Thunder	107	114	4.10	1.25	0.2976032	0.7024246	H
398	2020-03-04	Minnesota Timberwolves	Chicago Bulls	115	108	1.83	2.00	0.3853220	0.6147172	A
399	2020-03-05	Houston Rockets	Los Angeles Clippers	105	120	1.95	1.85	0.6188802	0.3709208	H
400	2020-03-05	Sacramento Kings	Philadelphia 76ers	108	125	1.45	2.80	0.5281478	0.4715240	A
401	2020-03-06	Chicago Bulls	Indiana Pacers	102	108	2.20	1.71	0.2955859	0.7051992	A
402	2020-03-06	Los Angeles Lakers	Milwaukee Bucks	113	103	1.83	2.00	0.4892491	0.5106664	H
403	2020-03-06	New Orleans Pelicans	Miami Heat	110	104	1.83	2.00	0.4190245	0.5809760	A
404	2020-03-06	New York Knicks	Oklahoma City Thunder	103	126	3.10	1.40	0.1779285	0.8220853	A
405	2020-03-06	Phoenix Suns	Portland Trail Blazers	127	117	2.45	1.60	0.4881878	0.5118130	H
406	2020-03-07	Charlotte Hornets	Houston Rockets	108	99	3.70	1.29	0.2196033	0.7804261	H
407	2020-03-07	Golden State Warriors	Philadelphia 76ers	118	114	2.45	1.60	0.2709814	0.7290190	A

Showing 1 to 16 of 30 entries, 19 total columns

```
C:/Users/Sébastien CARARO/Desktop/ATP& &Others/NBA/Codes/NBAproject/VERSION_9_20022020/
Median odd =  2.05
Mean odd =  2.885521



> View(tail(FINAL_TABLE_FILTERED, 30))
Warning messages:
1: le package ‘dplyr’ a été compilé avec la version R 3.6.1
2: le package ‘rvest’ a été compilé avec la version R 3.6.1
3: le package ‘xml2’ a été compilé avec la version R 3.6.1
4: le package ‘caret’ a été compilé avec la version R 3.6.1
5: le package ‘gridExtra’ a été compilé avec la version R 3.6.1
6: le package ‘MASS’ a été compilé avec la version R 3.6.1
7: le package ‘e1071’ a été compilé avec la version R 3.6.1
8: le package ‘zoo’ a été compilé avec la version R 3.6.2
>
```

```
Data
BetTable_rf        972 obs. of 21 variables
BetTable_s1        972 obs. of 21 variables
BetTable_s2        972 obs. of 21 variables
BetTable_s3        972 obs. of 21 variables
BetTable_s4        972 obs. of 21 variables
BetTable_s5        972 obs. of 21 variables
BetTable_s6        972 obs. of 21 variables
BetTable_s7        972 obs. of 21 variables
BetTable_s8        972 obs. of 21 variables
BetTable_svm_lin…  972 obs. of 21 variables
BetTable_svm_rad   972 obs. of 21 variables
```

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
NEXT_PRED.R
tail(FINAL_TABLE_FILTERED, 30)
Console Terminal Jobs
Environment History Connections
Global Environment
Files Plots Packages Help Viewer
Last 50 matches evolution gain :
Earnings (stake = 1)
375
350
325
mars 02
mars 09
Date
Last 50 matches evolution gain :
Earnings (stake = 1)
375
350
325
700
725
750
775
Date
17:39
11/03/2020

<u>Figure xx :</u> Final Rstudio rendering

XV - DESCRIBE THE PRINCIPLE WITH ATP, FOOTBALL, ESPORTS AND RUGBY ASWELL

Satisfied with the NBA program and the profits earned thanks to it, I then decided to apply the same pipeline process to tennis and football forecasting. As I was more at ease with the principle, I could quickly code the algorithms and proceed to betting on the proposed games.

The creation of the two programs was facilitated by the two source websites which contain an impressive amount of data for both sports([xx] & [xx]).

The latter programs are more evolved, for example I was able to compute a lot more variables in both sports (respectively 191 and 330 variables for football and tennis – [appendix n° 1]). I repeated the same process, I just had to adapt the computed variables based on the sport and proceed to a few modifications. Also, the betting strategies were differents, according to the sport format : for football I studied 5 different betting strategies (Value, Favorite, Home, Draw, Away) and for tennis I only had 2 strategies available (Favorite and Value).

Once I had finished the first football program for the british Premier League, as my source datasets were in the same format for each league I simply had to run the pipeline onto the other leagues to create a program. In total I created 15 football scripts ([appendix n°xx])

XVI - RESULTS IN TERMS OF SPORT BETTING

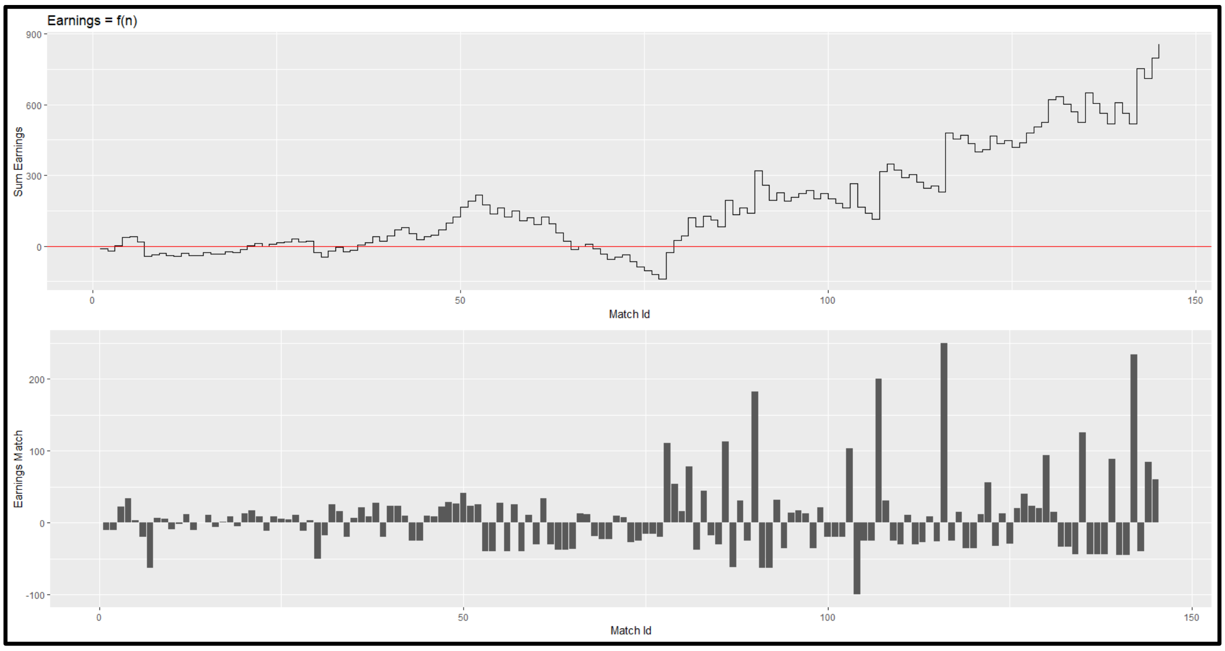
<u>NB :</u> The models and techniques developed during this project have been evolving for the best all way through, hence at the day I am writing this paper I consider having my best pipeline ever, but when I started to bet with the first algorithms I used models and techniques that I now consider way less efficient than the more recent ones. In particular, the automatic implementation of strategies was a huge step forward, as before this, I had to do it by hand (~2h) hence I could only do it once a couple weeks.

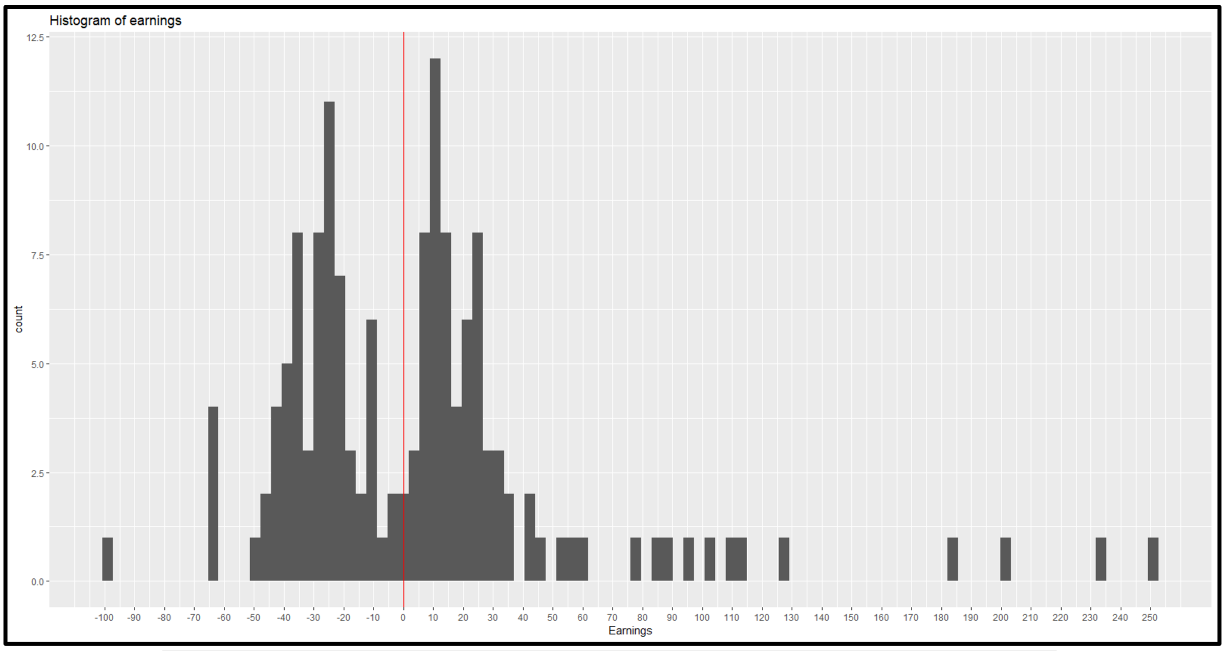
As a consequence, the newly constructed models are likely to give way better results in the future than the ones presented in the next section. Nonetheless, one might appreciate the already high-performing results obtained from January 15th to March 13rd, presented there down :

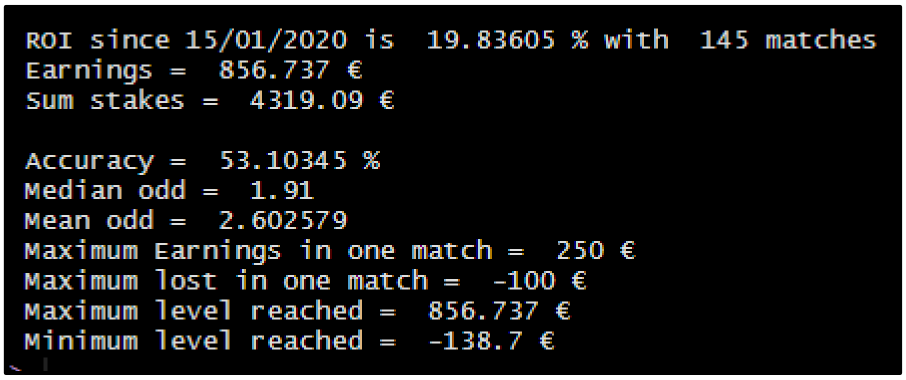
(A) NBA BETTING

Thanks to this pipeline, I've been able to apply betting instructions derived from this algorithm, and was able to earn 856€ thanks to the NBA games. The NBA league is by far my best program, which is delightful considering the fact that there are a lot a games over the 6-months season. Here are a few plots that display the path to earning this amount :

<u>Figure xx & figure xx :</u> Cumulative earnings and marginal earnings (for each game)
<u>Figure xx :</u> Histogram of earnings
<u>Figure xx :</u> Detailed NBA betting statistics







```
ROI since 15/01/2020 is  19.83605 % with  145 matches
Earnings =  856.737 €
Sum stakes =  4319.09 €

Accuracy =  53.10345 %
Median odd =  1.91
Mean odd =  2.602579
Maximum Earnings in one match =  250 €
Maximum lost in one match =  -100 €
Maximum level reached =  856.737 €
Minimum level reached =  -138.7 €
```

✓ Commentary : We can notice that the NBA has been a very profitable league, with a double-digit ROI percentage. Even though there had been a dark series between game n°50 and game n°80 – which might be due to the fact that this period was around the All-Star break, hence the dynamics could have been different (star player can rest to avoid injuries before the All-Star + some teams feel exhausted after 4 straight months) – the general trend is clearly profitable, and during the last dozens of games it was very profitable. We can also notice that the program advices to bet on outsider (mean odd = 2.6) and that we could eventually make huge bets : Charlotte Hornets @ Toronto Raptors on February, 29th (Odd 9) and Golden State Warriors @ Denver Nuggets on March, the 4th (Odd 12).

(B) ALL RESULTS

As of the 13rd of march, 2020, the application of the described strategy, coupled with a betting strategy along which I staked more on well-performing leagues such as the NBA and the italian football league *Serie A,* I realised a net profit of 1266,49 € with an intial bankroll of 250€ between the 1st of January and the 13rd of March, with a return of investment of 7,30%, which perfectly fits my expectations :

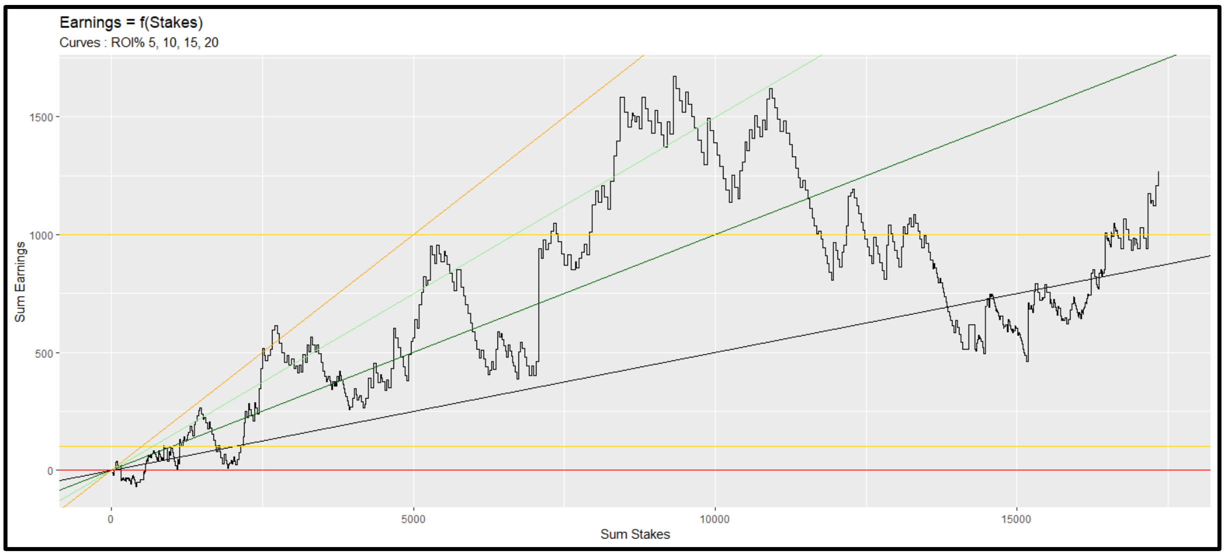


Figure xx : *Profit evolution since January 2020 (€)*

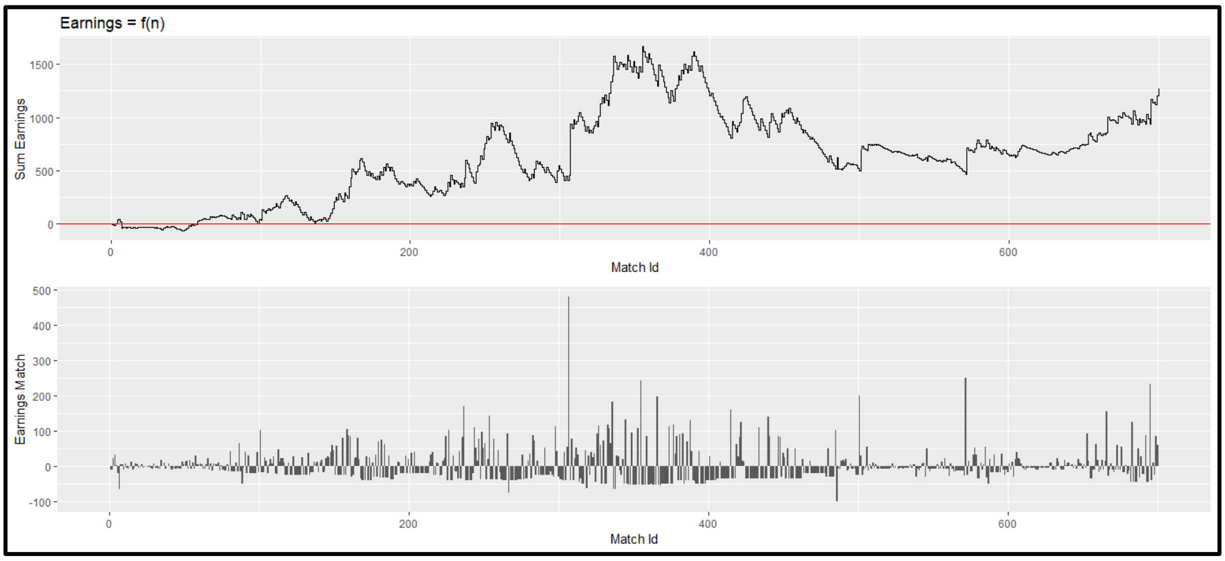
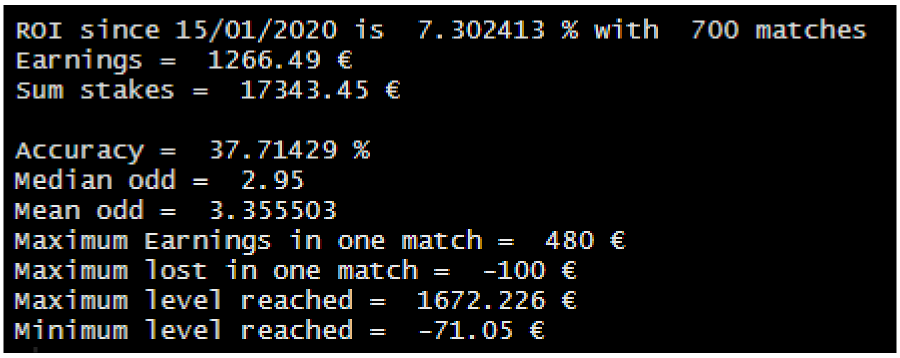
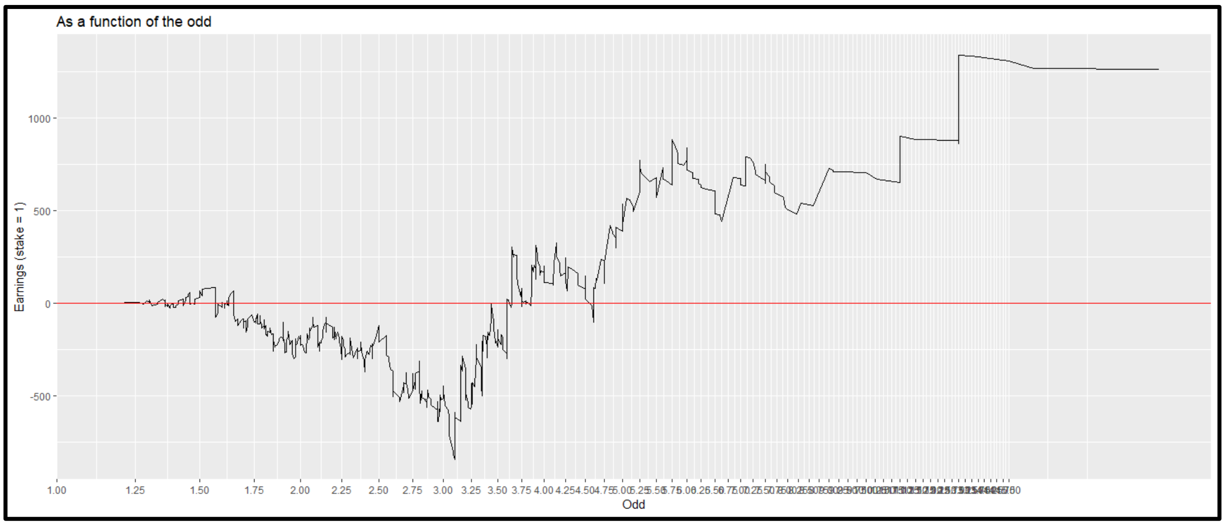


Figure xx : *Profit evolution as a function of the number of bets*


```
ROI since 15/01/2020 is  7.302413 % with  700 matches
Earnings =  1266.49 €
Sum stakes =  17343.45 €

Accuracy =  37.71429 %
Median odd =  2.95
Mean odd =  3.355503
Maximum Earnings in one match =  480 €
Maximum lost in one match =  -100 €
Maximum level reached =  1672.226 €
Minimum level reached =  -71.05 €
```

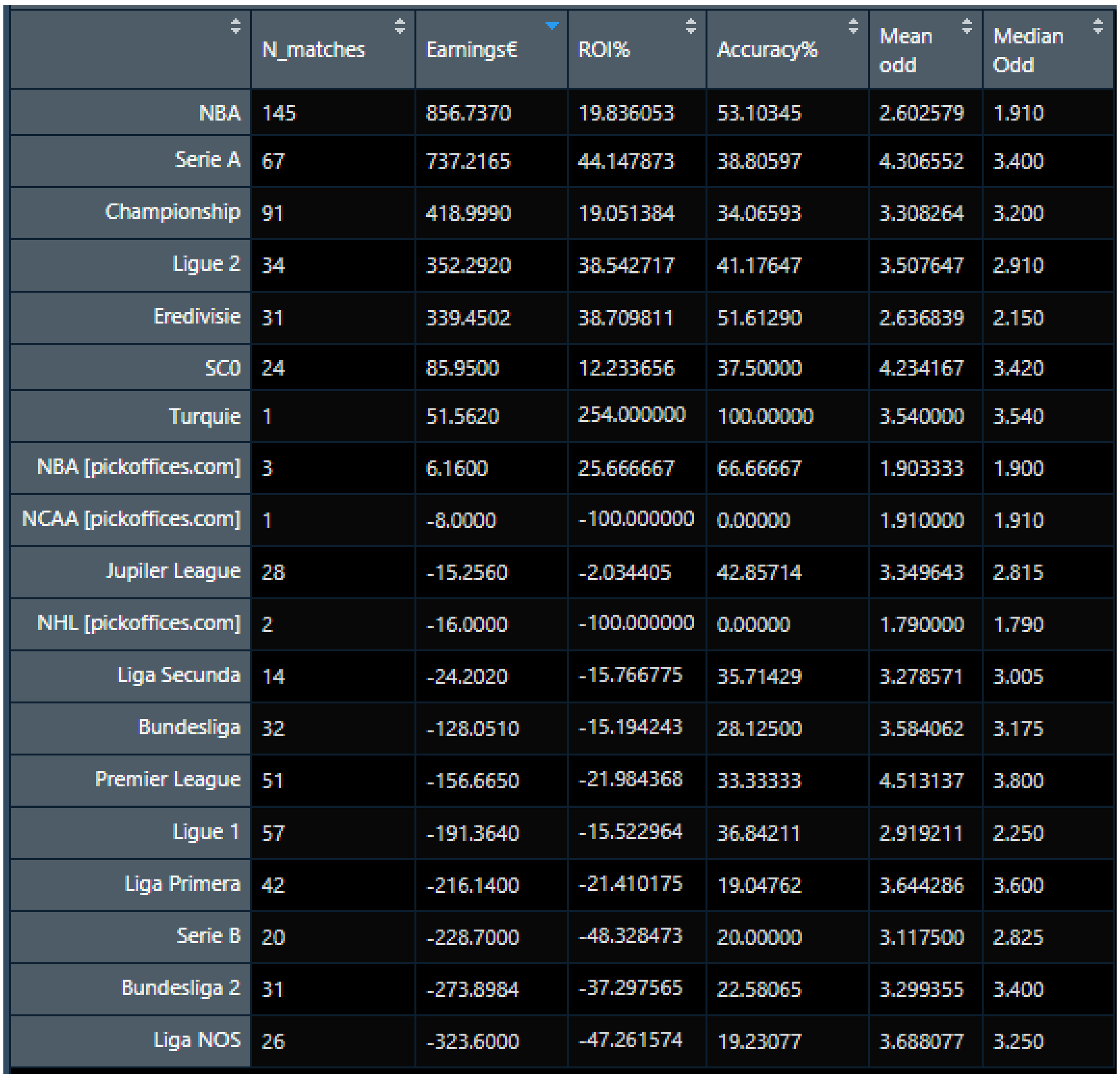
Figure xx : *Betting statistics since january 2020*



<u>Figure xx :</u> Profit evolution as a function of the odd

- ✓ <u>Commentary :</u> The shape of the profit curve was at first very promising, as the ROI was oscillating between the 10% and the 20% projection, however at some point there was a *really* dark series which made me lose half of my profit. I was a bit disapointed at this time, hence I chose to drastically review my staking plan. We can see that during the last two weeks of exploitation the algorithm was once again yielding great results, which were brutally stopped by the raise of the Covid-19 pandemic, forcing all championships to stop the season from then on.
  We can also notice, from the last plot, that we are losing money on odds <3.15 and earning our money on great odds (odd > 3.15).

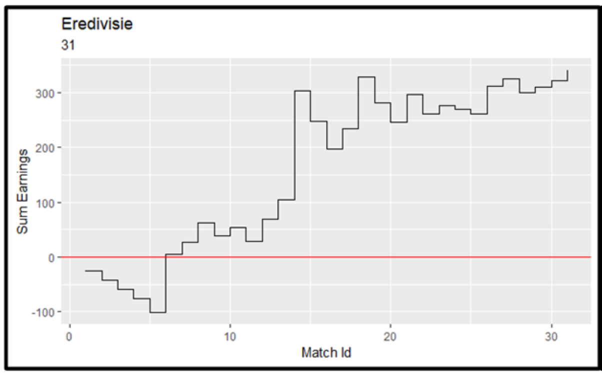
There down is displayed the results according to the league, with some leagues performing way better than others. In the future one could think about testing the leagues with paper betting before betting real money on them, which would have had allowed me to exclude some leagues such as Bundesliga, Bundesliga 2 and Liga NOS. However, I did not have time to do such task and preffered strating betting because the season was already coming to an end.

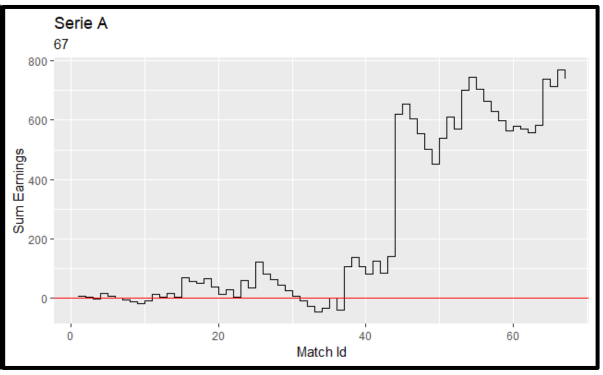


	N_matches	Earnings€	ROI%	Accuracy%	Mean odd	Median Odd
NBA	145	856.7370	19.836053	53.10345	2.602579	1.910
Serie A	67	737.2165	44.147873	38.80597	4.306552	3.400
Championship	91	418.9990	19.051384	34.06593	3.308264	3.200
Ligue 2	34	352.2920	38.542717	41.17647	3.507647	2.910
Eredivisie	31	339.4502	38.709811	51.61290	2.636839	2.150
SC0	24	85.9500	12.233656	37.50000	4.234167	3.420
Turquie	1	51.5620	254.000000	100.00000	3.540000	3.540
NBA [pickoffices.com]	3	6.1600	25.666667	66.66667	1.903333	1.900
NCAA [pickoffices.com]	1	-8.0000	-100.000000	0.00000	1.910000	1.910
Jupiler League	28	-15.2560	-2.034405	42.85714	3.349643	2.815
NHL [pickoffices.com]	2	-16.0000	-100.000000	0.00000	1.790000	1.790
Liga Secunda	14	-24.2020	-15.766775	35.71429	3.278571	3.005
Bundesliga	32	-128.0510	-15.194243	28.12500	3.584062	3.175
Premier League	51	-156.6650	-21.984368	33.33333	4.513137	3.800
Ligue 1	57	-191.3640	-15.522964	36.84211	2.919211	2.250
Liga Primera	42	-216.1400	-21.410175	19.04762	3.644286	3.600
Serie B	20	-228.7000	-48.328473	20.00000	3.117500	2.825
Bundesliga 2	31	-273.8984	-37.297565	22.58065	3.299355	3.400
Liga NOS	26	-323.6000	-47.261574	19.23077	3.688077	3.250

✓ Commentary: We can notice that the results are very polarized, in the sense that for each league, the related algorithm is either yielding tremendous profits (double-digit ROI's) or yielding dramatic losses. The programs were created with the same pipeline tho, therefore we could explain these differences by the fact that there are relatively few matches per league, hence it could be a matter of variance, or that our method is simply working with some leagues, and isn't working for some other leagues. However, those results are very satisfying because overall we made profits, and that if, for the losing leagues, the losing trend continues then we would just need to stop betting on these leagues and simply continue with the profitable leagues.

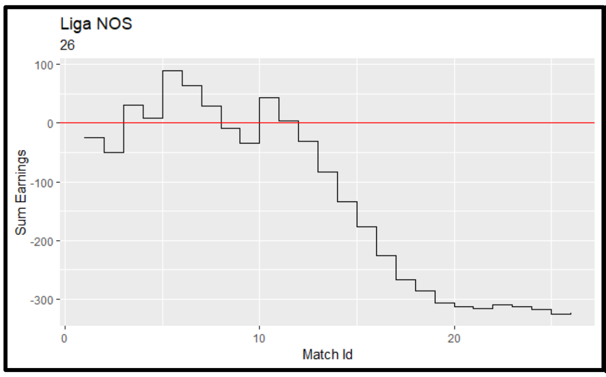
As examples, we can have a look at well performing leagues such as Eredivisie and Serie A profiles, which display monotonous high-profitable figures and very enjoyable to look at. For these leagues the process is clearly validated :

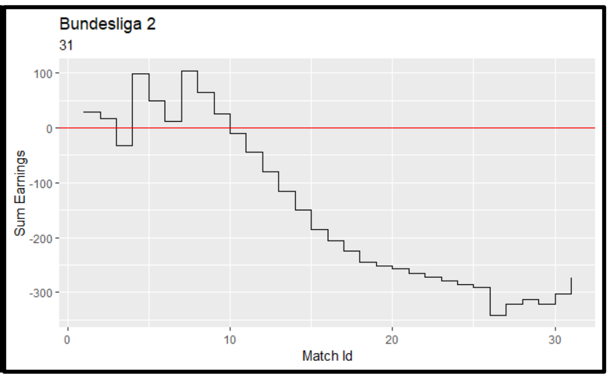




<u>Figures xx & xx :</u> Eredivisie and Serie A cumulative earnings

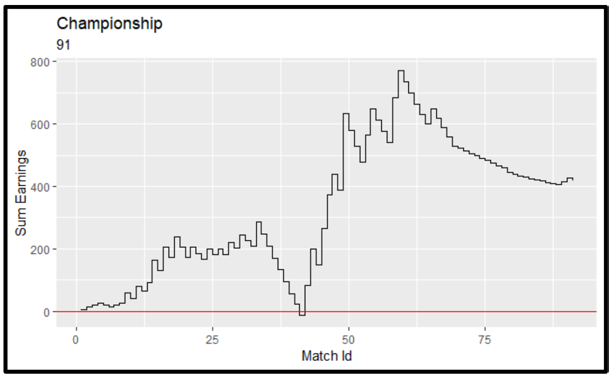
In the other hand, some leagues display clear opposite trends, which - as I explained – may be attributed either to bad luck or to an invalidation of the process for these leagues :

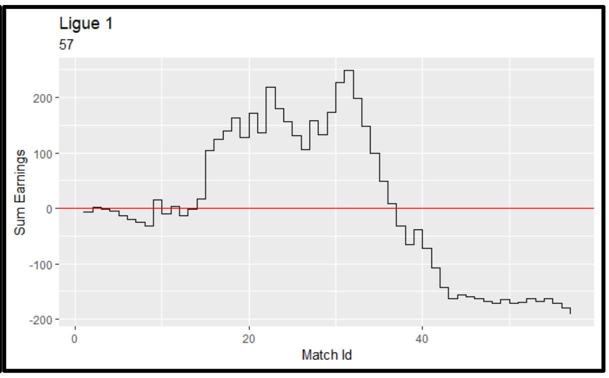


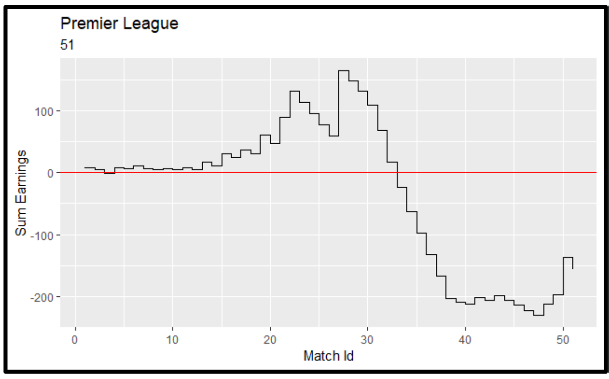


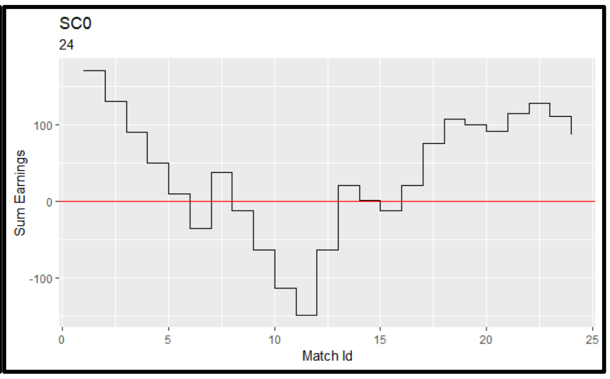
<u>Figures xx & xx :</u> Liga NOS and Bundesliga 2 cumulative earnings

Finally, there are multiple leagues for which the trend is changing over time, with great periods alternating with dark series :









<u>Figures xx, xx, xx, xx :</u> Championship, Ligue1, Premier League and Scottish Premiership cumulative earnings

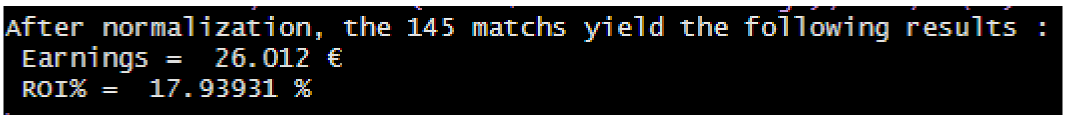
(C) FURTHER OBSERVATIONS ON THE RESULT, MONTE-CARLO APPROACH

Having all these results, we may now want to evaluate the degree at which these results are satisfying. To do this, we will proceed to a Monte-Carlo [xx] set of simulations in order to benchmark our results with multiple basic strategies, which are respectively :

- Random Betting
- Home Betting
- Away Betting
- Favorite Betting (*lowest odd*)
- Outsider Betting (*biggest odd*)

For each strategy, we will draw 145 matches out of the 970 finished games this season, and bet accordingly to each strategy. I decided to draw a total of $n = 10\ 000$ matches to make the simulations. I then had a look at the cumulative earnings in order to compare them with my selection of bets.

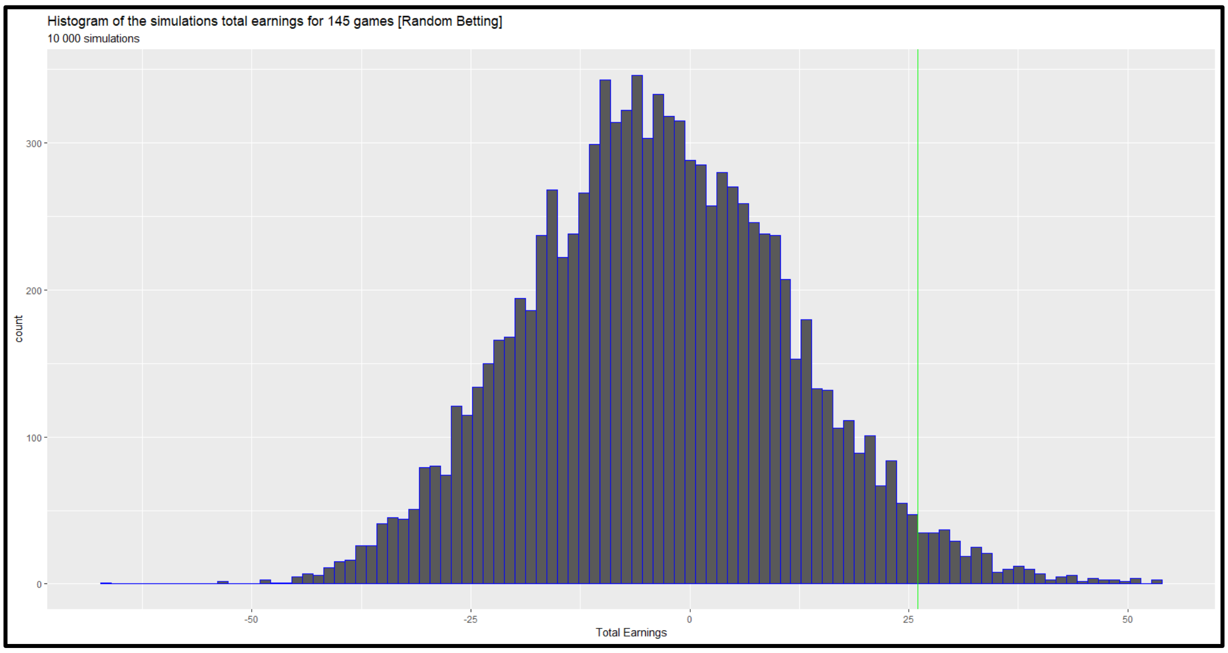
In order to account for the fact that, during my real-life bets, I did not stake the same amount of money for each game, I had to normalize my record of betting first (simply set the stake of each bet to 1€, and update the theoretical earnings for each match). After having done this step, my set of 145 matches yields the following results :

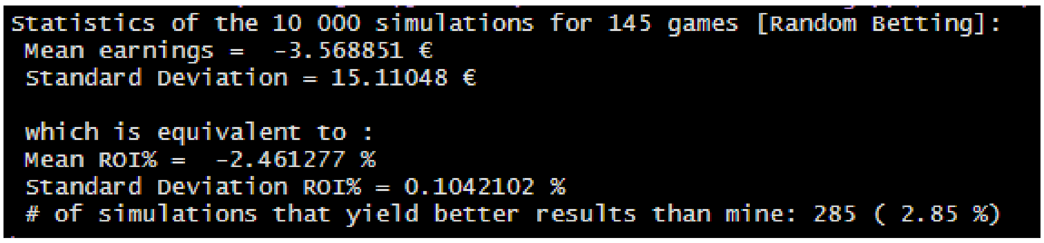


```
After normalization, the 145 matchs yield the following results :
 Earnings =  26.012 €
 ROI% =  17.93931 %
```

From the simulations – whose results are presented here after – we can draw the conclusion that our real-life bets are very unlikely to have happened by pure luck , and that therefore the profits reflect a real profitable strategy. The best dummy strategy over this season was the away betting strategy, even though – of course – in the long-run such strategy would yield losses.

[I] – RANDOM BETTING

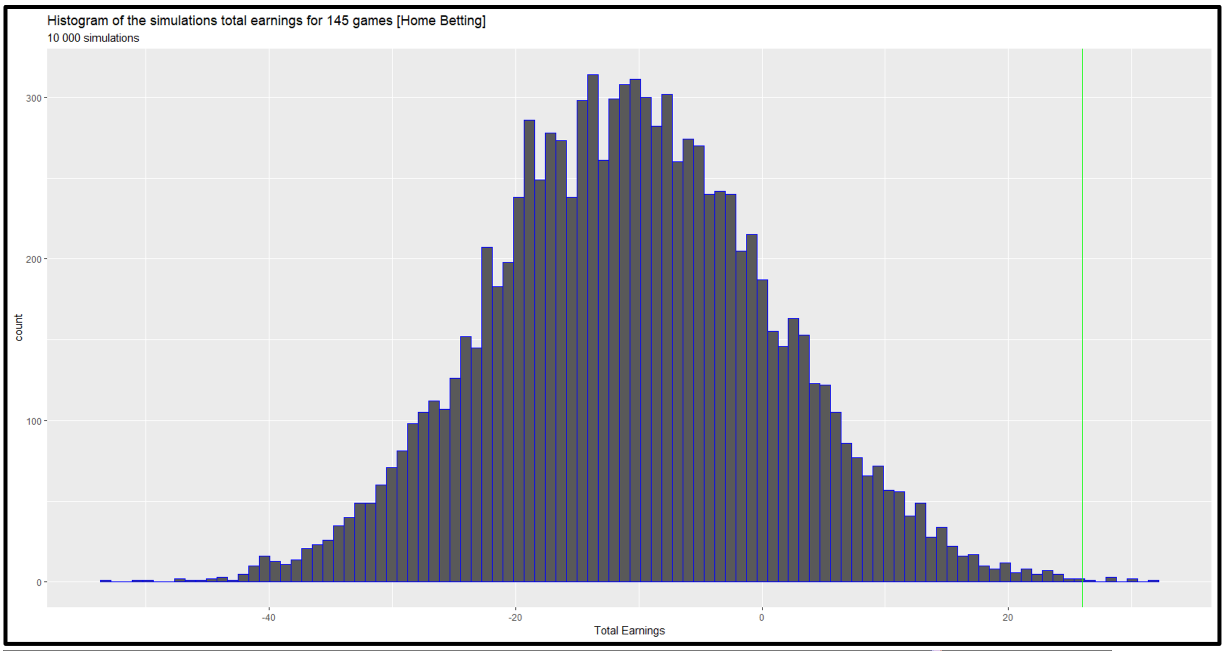


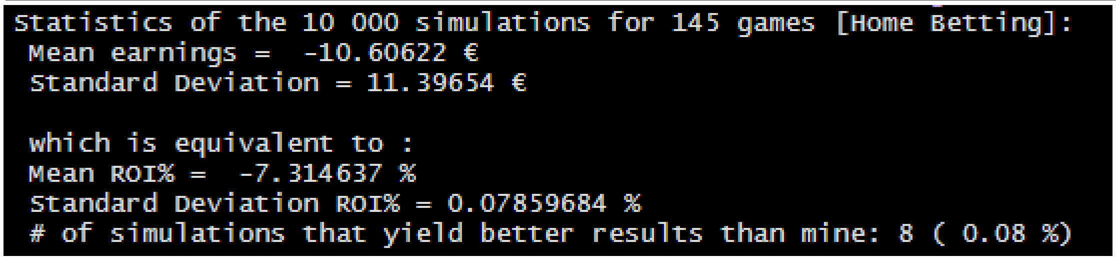


```
Statistics of the 10 000 simulations for 145 games [Random Betting]:
 Mean earnings =  -3.568851 €
 Standard Deviation = 15.11048 €

 which is equivalent to :
 Mean ROI% =  -2.461277 %
 Standard Deviation ROI% = 0.1042102 %
 # of simulations that yield better results than mine: 285 ( 2.85 %)
```

[II] – HOME BETTING

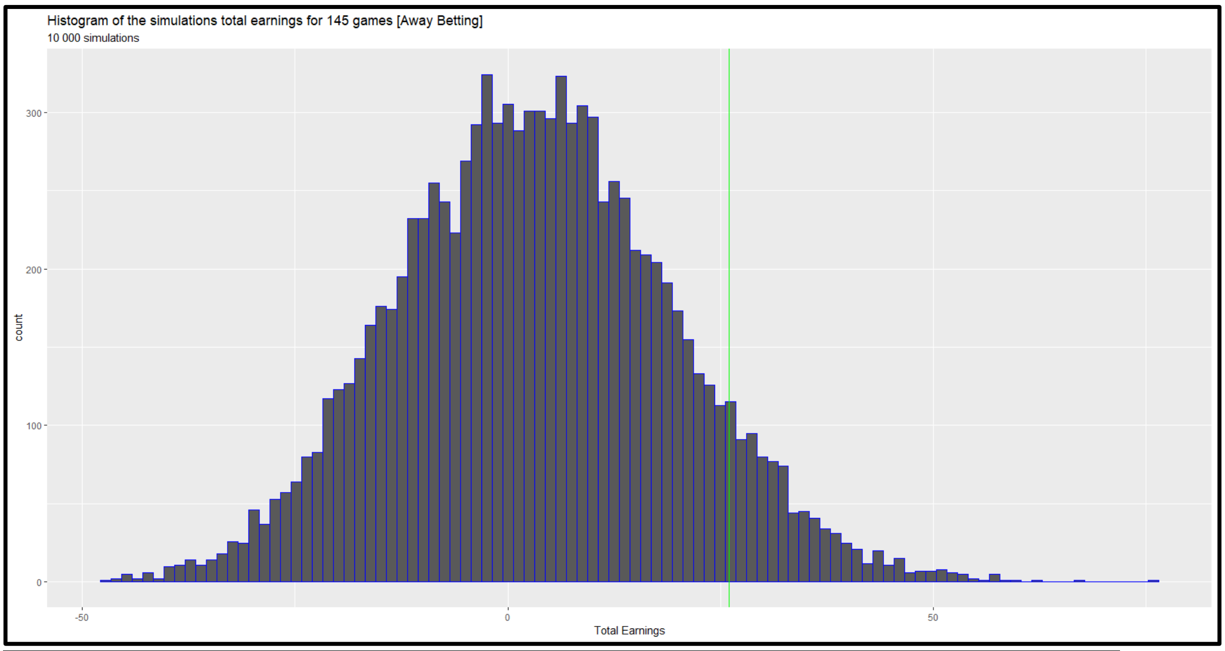


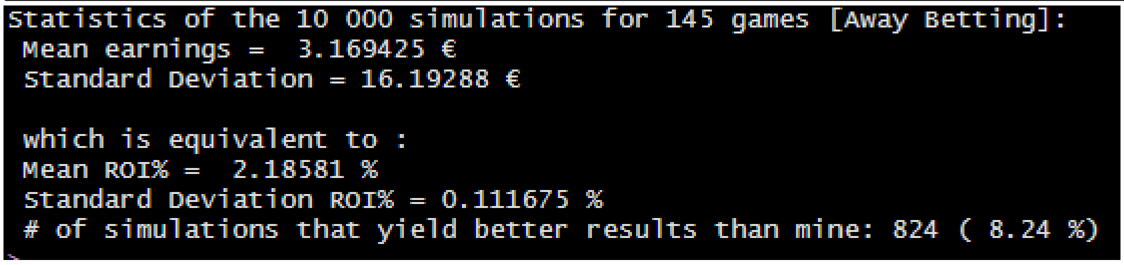


```
Statistics of the 10 000 simulations for 145 games [Home Betting]:
 Mean earnings =  -10.60622 €
 Standard Deviation = 11.39654 €

 which is equivalent to :
 Mean ROI% =  -7.314637 %
 Standard Deviation ROI% = 0.07859684 %
 # of simulations that yield better results than mine: 8 ( 0.08 %)
```

[III] – AWAY BETTING

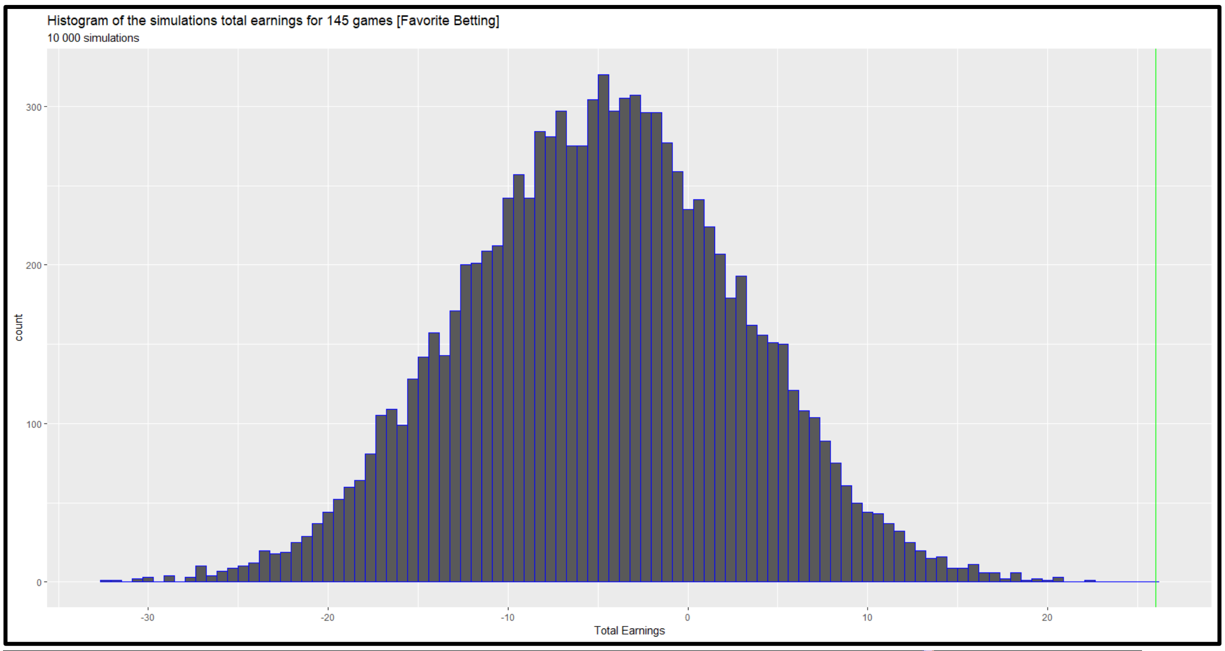


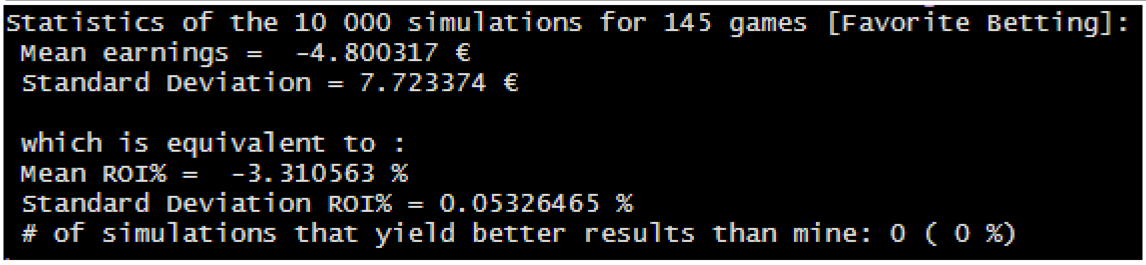


```
Statistics of the 10 000 simulations for 145 games [Away Betting]:
 Mean earnings =   3.169425 €
 Standard Deviation = 16.19288 €

 which is equivalent to :
 Mean ROI% =   2.18581 %
 Standard Deviation ROI% = 0.111675 %
 # of simulations that yield better results than mine: 824 ( 8.24 %)
```

[IV] – FAVORITE BETTING

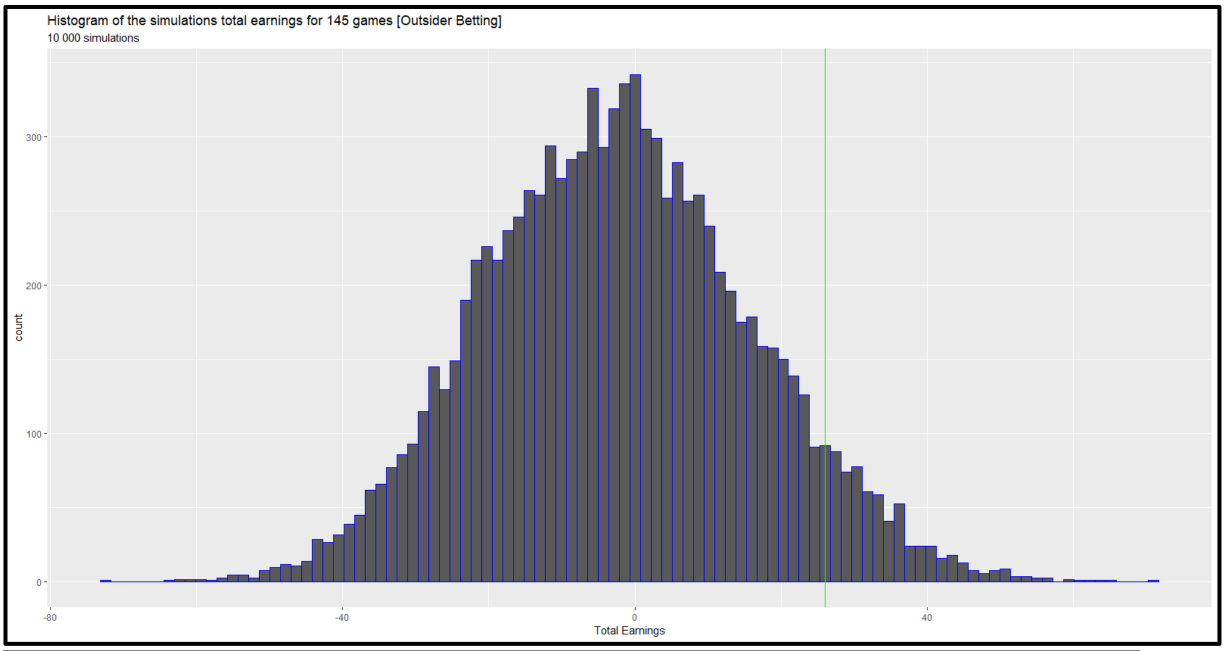


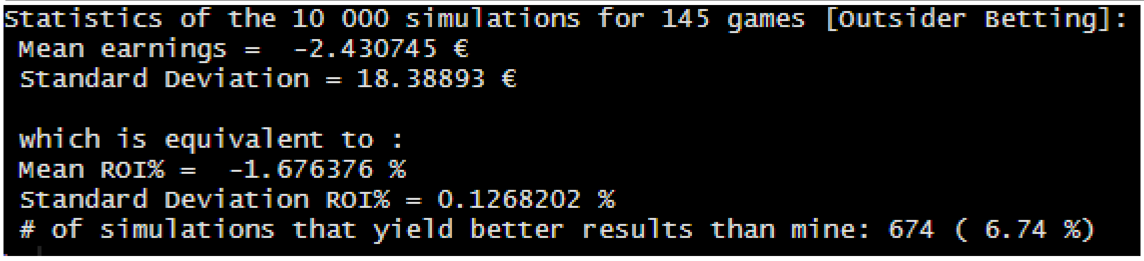


```
Statistics of the 10 000 simulations for 145 games [Favorite Betting]:
 Mean earnings =  -4.800317 €
 Standard Deviation = 7.723374 €

 which is equivalent to :
 Mean ROI% =  -3.310563 %
 Standard Deviation ROI% = 0.05326465 %
 # of simulations that yield better results than mine: 0 ( 0 %)
```

[V] – OUTSIDER BETTING





```
Statistics of the 10 000 simulations for 145 games [Outsider Betting]:
 Mean earnings =  -2.430745 €
 Standard Deviation = 18.38893 €

 which is equivalent to :
 Mean ROI% =  -1.676376 %
 Standard Deviation ROI% = 0.1268202 %
 # of simulations that yield better results than mine: 674 ( 6.74 %)
```

XVII - LIMITATIONS AND FUTURE WORK

(A) LIMITATIONS

In order to complexify the models, could encode way variables to get more precise models. We could also take the players performances into account, along with in-game statistics such as the field goal %, the 3-point shots %, free throws %, rebounds statistics, etc… . We could also look forward to taking injuries into account.

Concerning the organization side, as at the beginning of the project I had to manually update odds before each game, at some point – especially during week-ends where there was about 60 games on Saturday/Sunday – I would make a few staking errors. Especially at some point I staked 100 euros and 67 euros (and lost both times!).

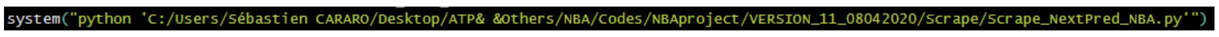
(B) DEPLOYMENT, CURRENT AND FUTURE WORK

As of the time I am currently writing this report, I am actively working on applying the same process to various leagues and sports ([Appendix xx – projection table of next leagues]) in order to obtain as many games as possible on a weekly basis.

I was also interested into enhancing the user interface and the general program, so that the program would automatically detect when future games are played, then scrape the odds 30 minutes before the games and run the program to display future bets.

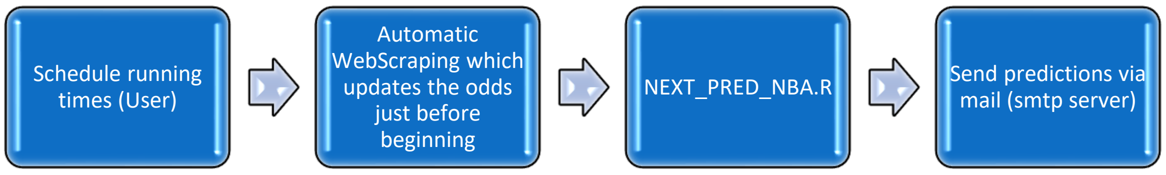
This is now the case, as I finally completed the many necessary code parts. Thanks to windows task scheduler and various personnally created webscraping programs, at this moment the program is practicaly autonomous, in the sense that I only need to schedule the times at which I want each program to run, and then the algorithm scrapes the next games on oddsportal.com, then computes predictions and it finally sends the predictions by mail directly onto my email address. Thanks to this organization I now only need to schedule running times and updates of the programs – which I simply need to launch for night computing. Considering the Covid-19 crisis among sports championships, I also had to adapt myself and move towards new leagues, especially eSports leagues. I re-used many of the variables that I previously built during the basketball program, as the dynamics metrics are similar. Apart from computation times optimization, the program is fully completed and is fully deployed right now, I am waiting for the start of next season in order to – hopefully – make great profits !

For information, this is the current deployment process, the user only needs to schedule the time of running with Windows Task Scheduler and then the whole process is done autonomously, as the python webscraper is directly called at the beginning of the R script :



```
system("python 'C:/Users/Sébastien CARARO/Desktop/ATP& &Others/NBA/Codes/NBAproject/VERSION_11_08042020/Scrape/Scrape_NextPred_NBA.py'")
```

Figure xx : Calling a Python script from R



Schedule running times (User)
Automatic WebScraping which updates the odds just before beginning
NEXT_PRED_NBA.R
Send predictions via mail (smtp server)

XVIII - CONCLUSION

I consider this project a great success, as I fullfilled many objectives that I had at the beginning of the project. This made me learn about various technical fields, from learning to conceive a whole pipeline, to building it : collecting the data, merging the different data sources, imaginating the predictive variables, comparing different Machine Learning models along with hyper-parameters tuning, creating a metric to handle the many different strategies, and finally building a user-friendly interface. Along with my different researches, I also learnt how to scrape data on various websites such as oddsportal.com and oddsshark.com, in order to access historical results and closing odds for certain leagues.

XIX - ACKNOWLEDGEMENTS

Me, Myself and I

XX - SOURCES

[xx] NBA dataset
https://www.indatabet.com/
[xx] FiveThirtyEight's GitHub page
https://github.com/fivethirtyeight/data/tree/master/nba-forecasts
[xx] Random Forests
https://towardsdatascience.com/understanding-random-forest-58381e0602d2
[xx] Support Vector Machines
https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/
[xx] Downsampling
https://developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data
[xx] Football data source
http://www.football-data.co.uk/
[xx] Tennis data source
http://www.tennis-data.co.uk/
[xx] nnet R package
https://cran.r-project.org/web/packages/nnet/nnet.pdf
[xx] Beating bookmakers with their own odds – and how the online sports betting market is rigged, Lisandro Kaunitz,Shenjun Zhong and Javier Kreiner
https://www.researchgate.net/publication/320296375_Beating_the_bookies_with_their_own_numbers_-_and_how_the_online_sports_betting_market_is_rigged
[xx] Non-Linear classification as a tool for predicting tennis matches (Jakub Hostacny, 2018)
https://www.semanticscholar.org/paper/Non-Linear-Classification-as-a-Tool-for-Predicting-Hosta%C4%8Dn%C3%BD/f6e5c9cffa57e29f77f4d4c2a5a073b47943cdd6
[xx] A statistical approach to sports betting (Anton Altmann, 2004)
https://1library.net/document/6qm6pe4y-a-statistical-approach-to-sports-betting.html
[xx] Ruberry, Michael Edward. 2013. Prediction Markets: Theory and Applications. Doctoral dissertation, Harvard University.
http://nrs.harvard.edu/urn-3:HUL.InstRepos:11181140
[xx] Beating the bookmakers on tennis matches
https://github.com/edouardthom/ATPBetting
[xx] Bet on Sibyl application code

https://github.com/jrbadiabo/Bet-on-Sibyl
[xx] Soccer betting
https://github.com/jiawei90/Soccer-Betting
[xx] Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA). Introduction to Statistical Learning, chapter 4.6. James et al.
http://faculty.marshall.usc.edu/gareth-james/ISL/ISLR%20Seventh%20Printing.pdf
[xx] Elo calculation process
https://en.wikipedia.org/wiki/Elo_rating_system#Mathematical_details
[xx] FiveThirtyEigth ELO description
https://fivethirtyeight.com/features/how-we-calculate-nba-elo-ratings/
[xx] SVM decision process illustrated
https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/
[xx] Random Forest construction illustrated
https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377895a60d2d
[xx] QDA and LDA illustration
http://www.phillypham.com/projects/Machine%20Learning%3A%20a%20Probabilistic%20Perspective%20in%20Python
[xx] Predicting football results using Machine Learning techniques. Corentin Herbinet, 2018.
https://www.imperial.ac.uk/media/imperial-college/faculty-of-engineering/computing/public/1718-ug-projects/Corentin-Herbinet-Using-Machine-Learning-techniques-to-predict-the-outcome-of-profressional-football-matches.pdf
[xx] Upsampling and upweighting a dataset
https://towardsdatascience.com/handling-imbalanced-datasets-in-deep-learning-f48407a0e758
[xx] Creative Commons licenses
https://creativecommons.org/licenses/
[xx] Description of overfitting
http://cv.znu.ac.ir/afsharchim/AI/lectures/Decision%20Trees%203.pdf
[xx] Monte-Carlo Simulations
https://www.palisade.com/risk/monte_carlo_simulation.asp
[xx] Signal To Noise ratio explanation
https://www.sciencedirect.com/topics/computer-science/noise-to-signal-ratio
[xx] Favorite-longshot bias in European Football betting market : Differences between popular and non-popular football competitions (Daniël van Raaij, 2019, [xx])
https://www.semanticscholar.org/paper/%E2%80%98%E2%80%99Favorite-longshot-bias-in-European-Football-and-van/d782c15add5ee1a60a55e38776f481da74cf765b

XXI - APPENDICES

- <u>Appendix n°1 : NBA, football and tennis variables</u>
- <u>NBA variables (133)</u>

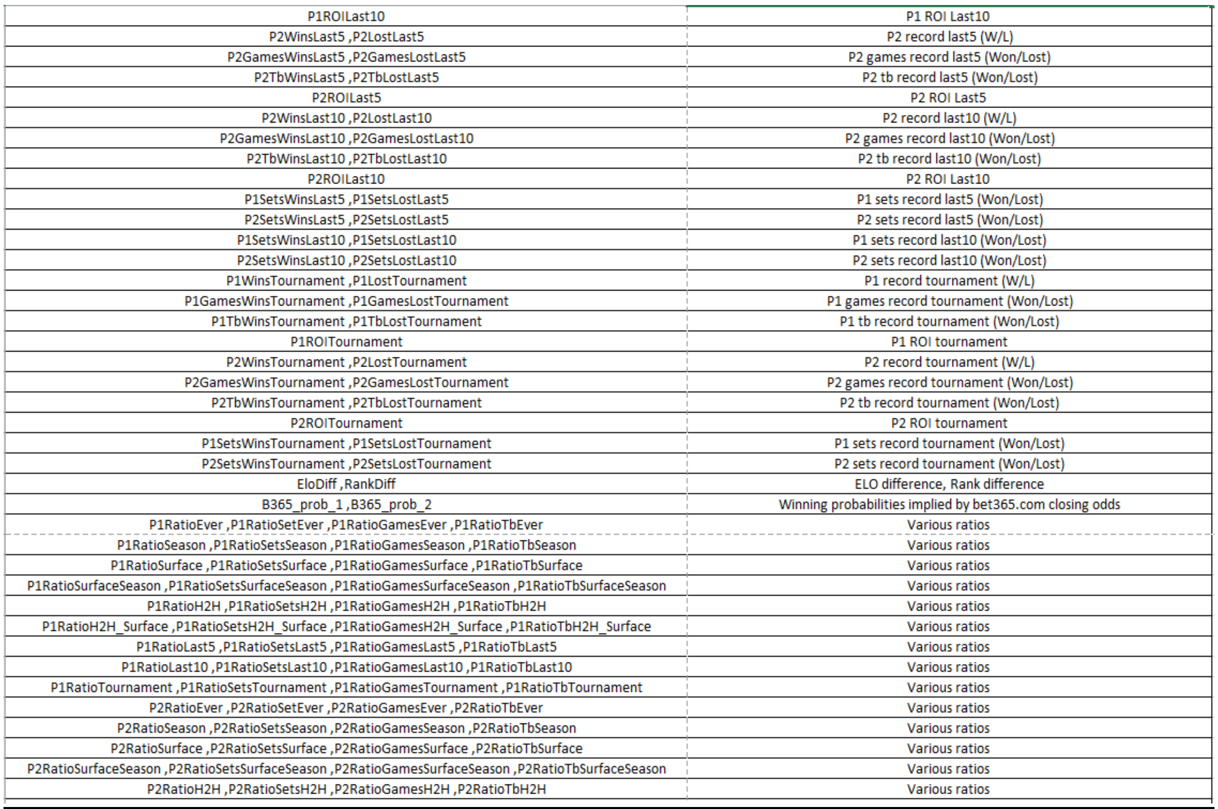
List of predictive variables used in the NBA program :	Meaning (Statistics only take the current season into account)
B365_OT_H ,B365_OT_A	Hometeam and awayteam closing odds (bet365.com)
TotWins_home ,TotLose_home ,Pct_home	Hometeam overall record this season : Wins, Loss, % of wins
TotWins_away ,TotLose_away ,Pct_away	Awayteam overall record this season : Wins, Loss, % of wins
TotWinsHome_home ,TotLoseHome_home ,PctHome_home	Hometeam overall record when playing home this season : Wins, Loss, % of wins
TotWinsAway_away ,TotLoseAway_away ,PctAway_away	Awayteam overall record when playing away this season : Wins, Loss, % of wins
Last5_home, Last10_home	Number of wins for hometeam during last 5 and 10 games
Last5_away ,Last10_away	Number of wins for awayteam during last 5 and 10 games
H2Hv_home ,H2Hv_away , H2H_ratio	Heah-to-head record : games won by hometeam, games won by awayteam, ratio
Elo_home ,Elo_away	Elo rating before the game for hometeam and for awayteam
H2Hv_home_home ,H2Hv_away_away , H2H_home_away_ratio	Heah-to-head record when hometeam plays home : games won by hometeam, games won by awayteam, ratio
Serie_home ,Serie_home_home	Current streak for hometeam and when playing home (>0 means streak of victories, <0 means streak of defeats)
Serie_away ,Serie_away_away	Current streak for awayteam and when playing away (>0 means streak of victories, <0 means streak of defeats)
ROI_home ,ROI_away ,ROI_home_home ,ROI_away_away	Return on Investment if we always bet respectively on hometeam or on awayteam
Moy_scored_season_home_home ,Moy_against_season_home_home	Mean points scored/against for hometeam when playing home
Moy_scored_season_away_away ,Moy_against_season_away_away	Mean points scored/against for awayteam when playing away
Moy_scored_last5_home_home ,Moy_against_last5_home_home	Mean points scored/against for hometeam over last 5 games when playing home
Moy_scored_last5_away_away ,Moy_against_last5_away_away	Mean points scored/against for awayteam over last 5 games when playing away
Moy_scored_season_home ,Moy_against_season_home	Mean points scored/against for hometeam
Moy_scored_season_away ,Moy_against_season_away	Mean points scored/against for awayteam
Moy_scored_last5_home ,Moy_against_last5_home	Mean points scored/against for hometeam over last 5 games
Moy_scored_last5_away ,Moy_against_last5_away	Mean points scored/against for awayteam over last 5 games
rest_days_home ,rest_days_away	Number of rest days for each team
mean_spread_season_home ,mean_spread_season_away	Mean spread (=pts scored - pts against)
mean_spread_season_home_home ,mean_spread_season_away_away	Mean spread (=pts scored - pts against) when playing home/away respectively
mean_spread_last5_home ,mean_spread_last5_away	Mean spread (=pts scored - pts against) over last 5 games
mean_spread_last5_home_home ,mean_spread_last5_away_away	Mean spread (=pts scored - pts against) over last 5 games when playing home/away respectively
Elo_prob_home ,Elo_prob_away	Probabilities of victory implied by the ELO rating (see documentation for formula)
B365_prob_home ,B365_prob_away	Probabilities of victory implied by the bet365 odds

- Football variables (191)

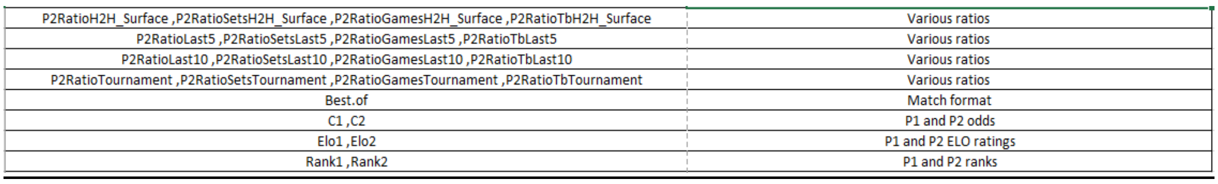
List of predictive variables used in the football programs :	Meaning
B365_OT_H ,B365_OT_D ,B365_OT_A	Bet365 closing odds (Home/Draw/Away)
TotWins_home ,TotDraw_home ,TotLose_home	Hometeam record over the season (Wins/Draws/Loss)
Pct_wins_home ,Pct_draw_home ,Pct_lose_home	Percentages of Wins/Draws/loss for hometeam
TotWins_away ,TotDraw_away ,TotLose_away	Awayteam record over the season (Wins/Draws/Loss)
Pct_wins_away ,Pct_draw_away ,Pct_lose_away	Percentages of Wins/Draws/loss for awayteam
TotWinsHome_home ,TotDrawHome_home ,TotLoseHome_home	Hometeam record over the season when playing home (Wins/Draws/Loss)
PctWinsHome_home ,PctDrawHome_home ,PctLoseHome_home	Percentages of Wins/Draws/loss for hometeam when playing home
TotWinsAway_away ,TotDrawAway_away ,TotLoseAway_away	Awayteam record over the season (Wins/Draws/Loss) when playing away
PctWinsAway_away ,PctDrawAway_away ,PctLoseAway_away	Percentages of Wins/Draws/loss for awayteam when playing away
Last3_home_wins ,Last3_home_draw ,Last3_home_lose	Hometeam record over the last 3 games (Wins/Draws/Loss)
Last6_home_wins ,Last6_home_draw ,Last6_home_lose	Hometeam record over the last 6 games (Wins/Draws/Loss)
Last3_away_wins ,Last3_away_draw ,Last3_away_lose	Awayteam record over the last 3 games (Wins/Draws/Loss)
Last6_away_wins ,Last6_away_draw ,Last6_away_lose	Awayteam record over the last 6 games (Wins/Draws/Loss)
Serie_home_wins ,Serie_home_home_wins	Victory/Defeats streak for home team, + when playing home
Serie_away_wins ,Serie_away_away_wins	Victory/Defeats streak for away team, + when playing away
Moy_scored_season_home ,Moy_against_season_home	Average scored goals and goals taken by hometeam
Moy_scored_season_away ,Moy_against_season_away	Average scored goals and goals taken by awayteam
Moy_scored_last3_home ,Moy_against_last3_home	Average scored goals and goals taken by hometeam over last 3 games
Moy_scored_last3_away ,Moy_against_last3_away	Average scored goals and goals taken by awayteam over last 3 games
Moy_scored_season_home_home ,Moy_against_season_home_home	Average scored goals and goals taken by hometeam when playing home
Moy_scored_season_away_away ,Moy_against_season_away_away	Average scored goals and goals taken by awayteam when playing away
Moy_scored_last3_home_home ,Moy_against_last3_home_home	Average scored goals and goals taken by hometeam over last 3 home games
Moy_scored_last3_away_away ,Moy_against_last3_away_away	Average scored goals and goals taken by awayteam over last 3 away games
mean_spread_season_home ,mean_spread_season_away	Mean spread (goals scored - goals taken) for hometeam and for awayteam
mean_spread_season_home_home ,mean_spread_season_away_away	Mean spread (goals scored - goals taken) for hometeam and for awayteam, when playing home/away
mean_spread_last3_home ,mean_spread_last3_away	Mean spread (goals scored - goals taken) for hometeam and for awayteam over their last 3 games
mean_spread_last3_home_home ,mean_spread_last3_away_away	Mean spread (goals scored - goals taken) for hometeam and for awayteam over their last 3, when playing home/away
B365_prob_home ,B365_prob_away ,B365_prob_draw	Probabilities implied by the closing odds
RestDaysHome ,RestDaysAway	Rest days for home and away team
ROI_home_Wins ,ROI_away_Wins	Return On Investment if we always bet hometeam, if we always bet awayteam
ROI_home_home_wins ,ROI_away_away_wins	Return On Investment if we always bet hometeam when playing home, if we always bet awayteam when playing away
H2H_wins_home ,H2H_draws ,H2H_wins_away	Head-To-Head record
H2H_wins_home_home ,H2H_draws_homeaway ,H2H_wins_away_away	Head-To-Head record when hometeam is playing home
Pct_H2H_wins_home ,Pct_H2H_draws ,Pct_H2H_wins_away	Percentages of home wins/Draws/away wins during Head-To-Head
Pct_H2H_wins_home_home ,Pct_H2H_draws_homeaway ,Pct_H2H_wins_away_away	Percentages of home wins/Draws/away wins during Head-To-Head when hometeam is playing home
PctPointsHome ,PctPointsAway	Percentage of points earned by each team throughout this season (Victory = 3 points, Draw = 1 point)
PctPointsHome_home ,PctPointsAway_away	Percentage of points earned by each team, respectively playing home/away throughout this season

- <u>Tennis variables (330)</u>

List of predictive variables used in the tennis program :	Meaning
P1WinsEver ,P1LostEver	P1 record ever (W/L)
P1GamesWinsEver ,P1GamesLostEver	P1 games record ever (Won/Lost)
P1TbWinsEver ,P1TbLostEver	P1 tb record ever (Won/Lost)
P1ROIEver	P1 ROI ever
P2WinsEver ,P2LostEver	P2 record ever (W/L)
P2GamesWinsEver ,P2GamesLostEver	P2 games record ever (Won/Lost)
P2TbWinsEver ,P2TbLostEver	P2 tb record ever (Won/Lost)
P2ROIEver	P2 ROI ever
Serie_p1_Ever ,Serie_p2_Ever	W/L streaks for P1 & P2
P1SetsWinsEver ,P1SetsLostEver	P1 sets record ever (Won/Lost)
P2SetsWinsEver ,P2SetsLostEver	P2 sets record ever (Won/Lost)
P1WinsSeason ,P1LostSeason	P1 record season (W/L)
P1GamesWinsSeason ,P1GamesLostSeason	P1 games record season (Won/Lost)
P1TbWinsSeason ,P1TbLostSeason	P1 tb record season (Won/Lost)
P1ROISeason	P1 ROI season
P2WinsSeason ,P2LostSeason	P2 record season (W/L)
P2GamesWinsSeason ,P2GamesLostSeason	P2 games record season (Won/Lost)
P2TbWinsSeason ,P2TbLostSeason	P2 tb record season (Won/Lost)
P2ROISeason	P2 ROI season
Serie_p1_Season ,Serie_p2_Season	W/L streaks season for P1 & P2
P1SetsWinsSeason ,P1SetsLostSeason	P1 sets record season (Won/Lost)
P2SetsWinsSeason ,P2SetsLostSeason	P2 sets record season (Won/Lost)
P1WinsSurface ,P1LostSurface	P1 record surface (W/L)
P1GamesWinsSurface ,P1GamesLostSurface	P1 games record surface (Won/Lost)
P1TbWinsSurface ,P1TbLostSurface	P1 tb record surface (Won/Lost)
P1ROISurface	P1 ROI surface
P2WinsSurface ,P2LostSurface	P2 record surface (W/L)
P2GamesWinsSurface ,P2GamesLostSurface	P2 games record surface (Won/Lost)
P2TbWinsSurface ,P2TbLostSurface	P2 tb record surface (Won/Lost)
P2ROISurface	P2 ROI surface
Serie_p1_Surface ,Serie_p2_Surface	W/L streaks for P1 & P2
P1SetsWinsSurface ,P1SetsLostSurface	P1 sets record surface (Won/Lost)
P2SetsWinsSurface ,P2SetsLostSurface	P2 sets record surface (Won/Lost)
P1WinsSurfaceSeason ,P1LostSurfaceSeason	P1 wins record surface season (Won/Lost)
P1GamesWinsSurfaceSeason ,P1GamesLostSurfaceSeason	P1 games record Surface Season (Won/Lost)
P1TbWinsSurfaceSeason ,P1TbLostSurfaceSeason	P1 Tb record surface season (Won/Lost)
P1ROISurfaceSeason	P1 ROI surfaceseason
P2WinsSurfaceSeason ,P2LostSurfaceSeason	P2 wins record surface season (Won/Lost)
P1GamesWinsSurfaceSeason ,P1GamesLostSurfaceSeason	P1 games record Surface Season (Won/Lost)
P1TbWinsSurfaceSeason ,P1TbLostSurfaceSeason	P1 Tb record surface season (Won/Lost)
P1ROISurfaceSeason	P1 ROI surfaceseason
P2WinsSurfaceSeason ,P2LostSurfaceSeason	P2 wins record surface season (Won/Lost)
P2GamesWinsSurfaceSeason ,P2GamesLostSurfaceSeason	P2 games record Surface Season (Won/Lost)
P2TbWinsSurfaceSeason ,P2TbLostSurfaceSeason	P2 Tb record surface season (Won/Lost)
P2ROISurfaceSeason	P2 ROI surfaceseason
Serie_p1_SurfaceSeason ,Serie_p2_SurfaceSeason	W/L streaks surface season
P1SetsWinsSurfaceSeason ,P1SetsLostSurfaceSeason	P1 set record surface season
P2SetsWinsSurfaceSeason ,P2SetsLostSurfaceSeason	P2 set record surface season
P1WinsH2H ,P1LostH2H	P1 record Head to Head(Won/Lost)
P1GamesWinsH2H ,P1GamesLostH2H	P1 games record h2h (Won/Lost)
P1TbWinsH2H ,P1TbLostH2H	P1 tb record h2h (Won/Lost)
P1ROIH2H	P1 ROI H2H
P2WinsH2H ,P2LostH2H	P2 wins record H2H (Won/Lost)
P2GamesWinsH2H ,P2GamesLostH2H	P2 games record h2h (Won/Lost)
P2TbWinsH2H ,P2TbLostH2H	P2 tb record h2h (Won/Lost)
P2ROIH2H	P2 ROI H2H
Serie_p1_H2H ,Serie_p2_H2H	W/L streaks H2H
P1SetsWinsH2H ,P1SetsLostH2H	P1 sets record h2h (Won/Lost)
P2SetsWinsH2H ,P2SetsLostH2H	P2 sets record h2h (Won/Lost)
P1WinsH2H_Surface ,P1LostH2H_Surface	P1 record H2H surface (W/L)
P1GamesWinsH2H_Surface ,P1GamesLostH2H_Surface	P1 games record h2h surface (Won/Lost)
P1TbWinsH2H_Surface ,P1TbLostH2H_Surface	P1 tb record h2h surface (Won/Lost)
P1ROIH2H_Surface	P1 ROI H2H surface
P2WinsH2H_Surface ,P2LostH2H_Surface	P2 record H2H surface (W/L)
P2GamesWinsH2H_Surface ,P2GamesLostH2H_Surface	P2 games record h2h surface (Won/Lost)
P2TbWinsH2H_Surface ,P2TbLostH2H_Surface	P2 tb record h2h surface (Won/Lost)
P2ROIH2H_Surface	P2 ROI H2H surface
Serie_p1_H2H_Surface ,Serie_p2_H2H_Surface	W/L streaks H2H surface
P1SetsWinsH2H_Surface ,P1SetsLostH2H_Surface	P1 set record H2H surface (Won/Lost)
P2SetsWinsH2H_Surface ,P2SetsLostH2H_Surface	P2 set record H2H surface (Won/Lost)
P1WinsLast5 ,P1LostLast5	P1 record last5 (W/L)
P1GamesWinsLast5 ,P1GamesLostLast5	P1 games record last5 (Won/Lost)
P1TbWinsLast5 ,P1TbLostLast5	P1 tb record last5 (Won/Lost)
P1ROILast5	P1 ROI Last5
P1WinsLast10 ,P1LostLast10	P1 record last10 (W/L)
P1GamesWinsLast10 ,P1GamesLostLast10	P1 games record last10 (Won/Lost)
P1TbWinsLast10 ,P1TbLostLast10	P1 tb record last10 (Won/Lost)



P1ROILast10	P1 ROI Last10
P2WinsLast5 ,P2LostLast5	P2 record last5 (W/L)
P2GamesWinsLast5 ,P2GamesLostLast5	P2 games record last5 (Won/Lost)
P2TbWinsLast5 ,P2TbLostLast5	P2 tb record last5 (Won/Lost)
P2ROILast5	P2 ROI Last5
P2WinsLast10 ,P2LostLast10	P2 record last10 (W/L)
P2GamesWinsLast10 ,P2GamesLostLast10	P2 games record last10 (Won/Lost)
P2TbWinsLast10 ,P2TbLostLast10	P2 tb record last10 (Won/Lost)
P2ROILast10	P2 ROI Last10
P1SetsWinsLast5 ,P1SetsLostLast5	P1 sets record last5 (Won/Lost)
P2SetsWinsLast5 ,P2SetsLostLast5	P2 sets record last5 (Won/Lost)
P1SetsWinsLast10 ,P1SetsLostLast10	P1 sets record last10 (Won/Lost)
P2SetsWinsLast10 ,P2SetsLostLast10	P2 sets record last10 (Won/Lost)
P1WinsTournament ,P1LostTournament	P1 record tournament (W/L)
P1GamesWinsTournament ,P1GamesLostTournament	P1 games record tournament (Won/Lost)
P1TbWinsTournament ,P1TbLostTournament	P1 tb record tournament (Won/Lost)
P1ROITournament	P1 ROI tournament
P2WinsTournament ,P2LostTournament	P2 record tournament (W/L)
P2GamesWinsTournament ,P2GamesLostTournament	P2 games record tournament (Won/Lost)
P2TbWinsTournament ,P2TbLostTournament	P2 tb record tournament (Won/Lost)
P2ROITournament	P2 ROI tournament
P1SetsWinsTournament ,P1SetsLostTournament	P1 sets record tournament (Won/Lost)
P2SetsWinsTournament ,P2SetsLostTournament	P2 sets record tournament (Won/Lost)
EloDiff ,RankDiff	ELO difference, Rank difference
B365_prob_1 ,B365_prob_2	Winning probabilities implied by bet365.com closing odds
P1RatioEver ,P1RatioSetEver ,P1RatioGamesEver ,P1RatioTbEver	Various ratios
P1RatioSeason ,P1RatioSetsSeason ,P1RatioGamesSeason ,P1RatioTbSeason	Various ratios
P1RatioSurface ,P1RatioSetsSurface ,P1RatioGamesSurface ,P1RatioTbSurface	Various ratios
P1RatioSurfaceSeason ,P1RatioSetsSurfaceSeason ,P1RatioGamesSurfaceSeason ,P1RatioTbSurfaceSeason	Various ratios
P1RatioH2H ,P1RatioSetsH2H ,P1RatioGamesH2H ,P1RatioTbH2H	Various ratios
P1RatioH2H_Surface ,P1RatioSetsH2H_Surface ,P1RatioGamesH2H_Surface ,P1RatioTbH2H_Surface	Various ratios
P1RatioLast5 ,P1RatioSetsLast5 ,P1RatioGamesLast5 ,P1RatioTbLast5	Various ratios
P1RatioLast10 ,P1RatioSetsLast10 ,P1RatioGamesLast10 ,P1RatioTbLast10	Various ratios
P1RatioTournament ,P1RatioSetsTournament ,P1RatioGamesTournament ,P1RatioTbTournament	Various ratios
P2RatioEver ,P2RatioSetEver ,P2RatioGamesEver ,P2RatioTbEver	Various ratios
P2RatioSeason ,P2RatioSetsSeason ,P2RatioGamesSeason ,P2RatioTbSeason	Various ratios
P2RatioSurface ,P2RatioSetsSurface ,P2RatioGamesSurface ,P2RatioTbSurface	Various ratios
P2RatioSurfaceSeason ,P2RatioSetsSurfaceSeason ,P2RatioGamesSurfaceSeason ,P2RatioTbSurfaceSeason	Various ratios
P2RatioH2H ,P2RatioSetsH2H ,P2RatioGamesH2H ,P2RatioTbH2H	Various ratios



P2RatioH2H_Surface ,P2RatioSetsH2H_Surface ,P2RatioGamesH2H_Surface ,P2RatioTbH2H_Surface	Various ratios
P2RatioLast5 ,P2RatioSetsLast5 ,P2RatioGamesLast5 ,P2RatioTbLast5	Various ratios
P2RatioLast10 ,P2RatioSetsLast10 ,P2RatioGamesLast10 ,P2RatioTbLast10	Various ratios
P2RatioTournament ,P2RatioSetsTournament ,P2RatioGamesTournament ,P2RatioTbTournament	Various ratios
Best.of	Match format
C1 ,C2	P1 and P2 odds
Elo1 ,Elo2	P1 and P2 ELO ratings
Rank1 ,Rank2	P1 and P2 ranks

- <u>Appendix n°2 : Football leagues scripts</u>

Country	League	Script	Script finished ?
Argentina	SuperLiga	NEXT_PRED_ARG.R	✓
Australia	A-League	NEXT_PRED_AUS.R	✓
Austria	Tipico Bundesliga	NEXT_PRED_AUT.R	✓
Belgium	Jupiler Pro League	NEXT_PRED_B1.R	✓
Brasil	Serie A	NEXT_PRED_BRA.R	✓
China	Super League	NEXT_PRED_CHN.R	✓
Denmark	Superliga	NEXT_PRED_DNK.R	✓
Faroe Islands	Premier League	NEXT_PRED_ILF.R	✓
Finland	Veikkausliiga	NEXT_PRED_FIN.R	✓
France	Ligue 1	NEXT_PRED_F1.R	✓
	Ligue 2	NEXT_PRED_F2.R	✓
Germany	Bundesliga	NEXT_PRED_D1.R	✓
	Bundesliga 2	NEXT_PRED_D2.R	✓
Ireland	Premier Division	NEXT_PRED_IRL.R	✓
Italy	Serie A	NEXT_PRED_I1.R	✓
	Serie B	NEXT_PRED_I2.R	✓

Japan	J-League	NEXT_PRED_JPN.R	✓
Mexico	Liga MX	NEXT_PRED_MEX.R	✓
Netherlands	Eredivisie	NEXT_PRED_N1.R	✓
Norway	Eliteserien	NEXT_PRED_NOR.R	✓
Poland	Ekstraklasa	NEXT_PRED_POL.R	✓
Portugal	Liga NOS	NEXT_PRED_P1.R	✓
Romania	Liga 1	NEXT_PRED_ROU.R	✓
Russia	Premier League	NEXT_PRED_RUS.R	✓
Scotland	Scottish Premiership	NEXT_PRED_SC0.R	✓
South Korea	K-League	NEXT_PRED_KLEAGUE.R	✓
Spain	Liga Primera	NEXT_PRED_SP1.R	✓
	Liga Secunda	NEXT_PRED_SP2.R	✓
Sweden	Allsvenskan	NEXT_PRED_SWE.R	✓
Switzerland	Super League	NEXT_PRED_SWZ.R	✓
Turkey	Süper Lig	NEXT_PRED_T1.R	✓
United Kingdom	Premier League	NEXT_PRED_E0.R	✓
	Championship	NEXT_PRED_E1.R	✓
USA	MLS	NEXT_PRED_USA.R	✓

- Appendix n°3 : Table of leagues under development

Sport	**League**	**Data Collected [15/04/2020]**	**Full program working**
Rugby	Top 14	✓	
	English Premiership	✓	
	Champions Cup	✓	
	Challenge Cup	✓	
	NRL	✓	
	Super Rugby	✓	
Basketball	Spain ACB	✓	✓
	French LNB	✓	✓
	Italian Lega A	✓	✓
	Euroleague	✓	✓
	Eurocup	✓	✓
	German BBL	✓	✓
	WNBA	✓	✓
Hockey	NHL		
Football	Champions League	✓	
	Europa League	✓	
American Football	NFL	✓	
Baseball	MLB		
	KBO (South-Korea)	✓	✓
	CPBL (Taiwan)	✓	✓

eSports	LoL Pro League	✓	✓
	LoL Continental League	✓	✓
	LoL Oceanic League	✓	✓
	ESL Pro League	✓	✓
	LoL Brazilian League	✓	✓
	Starcraft 2 South-Korean League	✓	✓
	LoL European Championship	✓	✓
	CS-GO Championship Series	✓	✓
	Overwatch League	✓	✓
	LoL Korean League	✓	✓
Tennis	ATP	✓	✓
	WTA	✓	✓