

Prédiction avec RandomForest

Thomas MASSÉ

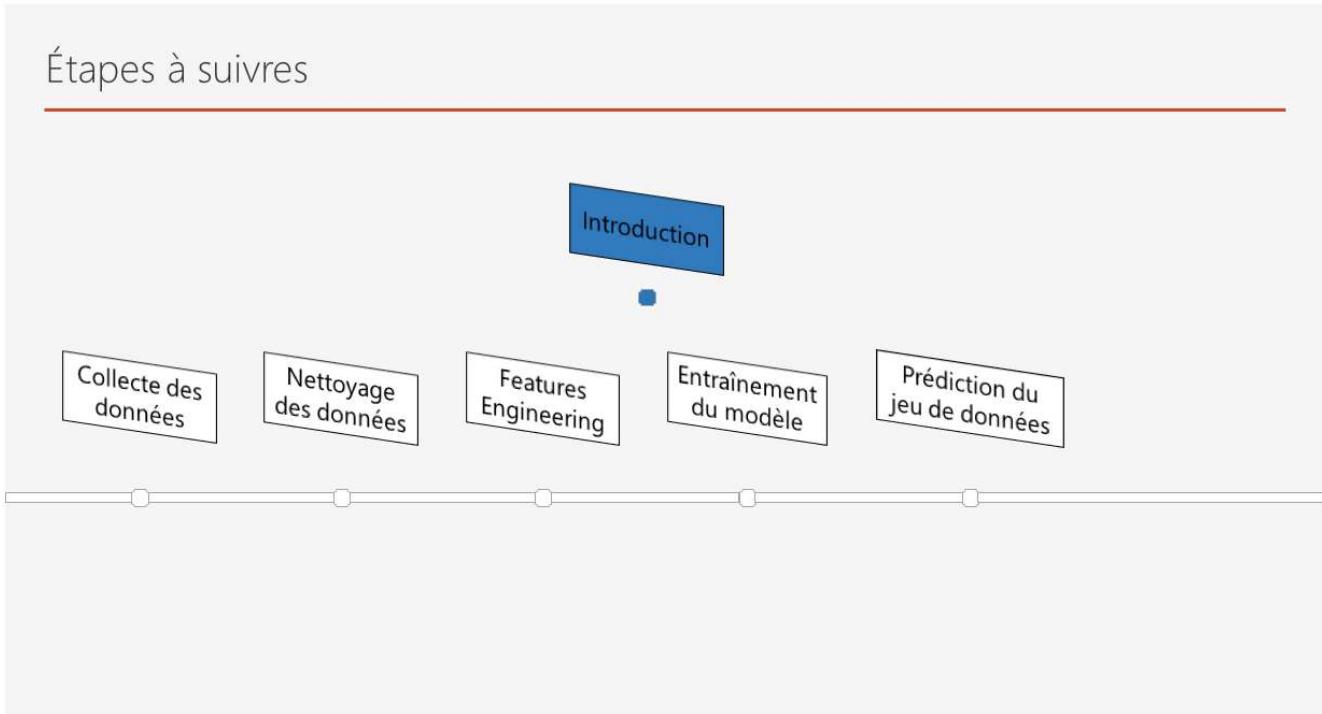
18/12/2020

```
library(randomForest)
library(tidyverse)
library(ggplot2)
library(dplyr)
#setwd("D:/R")
```

INTRODUCTION



Étapes à suivres



Nous allons voir ensemble comment utiliser le package RandomForest en R. Pour se faire on a souhaité réaliser le défi kaggle suivant : <https://www.kaggle.com/c/pubg-finish-placement-prediction> (<https://www.kaggle.com/c/pubg-finish-placement-prediction>), il suffit de s'inscrire sur Kaggle pour pouvoir accéder aux jeux de données.

Dans une première partie nous verrons ce que sont les fôrets d'arbres aléatoires, dans une seconde on parcourera ensemble les différentes données à notre disposition et pour finir on utilisera RandomForest sur tout ceci.

LES FORETS D'ARBRES ALEATOIRES

Avant de comprendre les fôrets d'arbres aléatoires, il faut comprendre ce qu'est un arbre décisionnel.

Un arbre décisionnel c'est tout simplement la représentation graphique sous forme d'arbres d'une suite de possibilités (ou de décisions).

On appelle chaque point un noeud et les liens entre les noeuds des branches.

Ce qui deviens intéressant c'est que l'on va pouvoir donner une probabilité à chaque décision.

On va donc pouvoir utiliser le machine learning pour bâtir un arbre de décisions en se basant sur des données existantes et en donnant à chaque combinaison une probabilité.

Il y a 2 types d'arbres :

Arbres de régression, quand la variable dépendante (celle que l'on souhaite prédire) est continue.

Arbres de classification, quand la variable dépendante est catégorielle. C'est ce que l'on utilisera dans notre exemple

Une fôret d'arbres aléatoires, c'est donc la création d'arbres de décisions de manières aléatoires en utilisant des données, ensuite de nouvelles données vont parcourir tout les arbres de notre modèle qui votent pour la classe de sortie (grâce à une moyenne pour la régression, ou par vote pour une classification).

J'ai essayé de vulgariser mais n'étant pas un expert je vous invite à consulter d'autres tutoriaux plus pointu sur ce sujet, si vous voulez plus entrez plus dans les détails.

QU'EST CE QUE PUBG ?

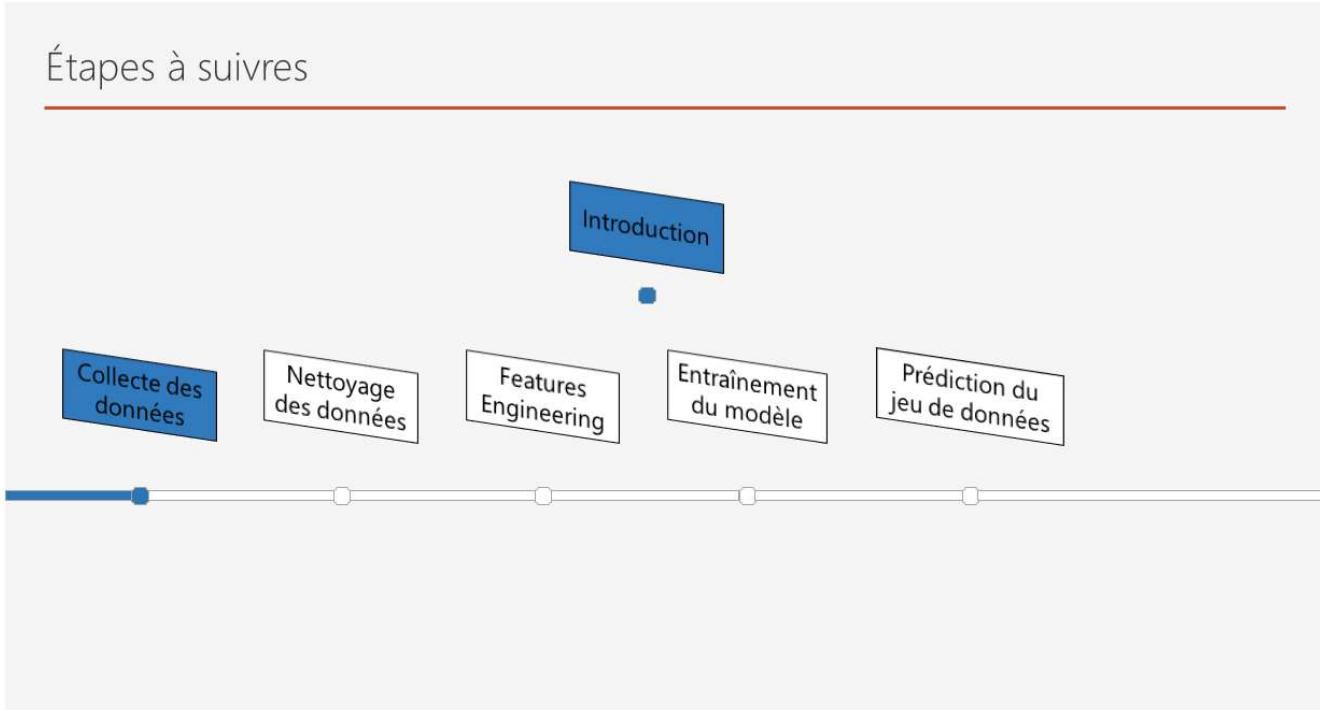
PUBG est l'abréviation de PlayerUnknown's Battlegrounds.

C'est un jeu vidéo de type Battle Royale afin de mieux appréhender ce type de jeu vidéo si vous n'êtes pas familier je vous invite à consulter la vidéo suivante : 

Je vous invite également à consulter du contenu sur twitch pour parfaire votre connaissance, je recommande notamment le stream de MoMaN qui joue régulièrement et est plutôt aguerri au genre ou il a fait ses armes depuis H1 Z1 (pour les plus connaisseurs)

EXPLICATIONS DES DONNEES

Étapes à suivres



Collecte des données

Tout d'abord regardons les données que nous avons.

D'après la description sur Kaggle, nous avons les données de plus de 65000 parties séparés en jeu d'entraînement et jeu de test.

Nous devons prédire les chances de victoires de chaque joueur en fonction des autres données.

Il y a 3 fichiers : - sample_submission_v2.csv : c'est l'exemple du format de soumission pour les concurrents qui ont réalisé le concours.

- train_v2.csv : c'est notre jeu "d'entraînement", c'est ce modèle qui va nous permettre de créer notre modèle de forêt d'arbres aléatoires.

- test_v2.csv : c'est le jeu de données qui va parcourir le modèle et où l'on va chercher à prédire une variable.

Nous allons donc lire les fichiers CSV

```

#Lire le training set
trainsetPUBG <- read.csv2("D:/Dataset/PUBG/train_V2.csv",
                           sep = ",", header = TRUE, fileEncoding = "utf-8")
#Le mettre dans une dataframe
trainsetPUBG <- as.data.frame(trainsetPUBG)
#Afficher le nom des colonnes
colnames(trainsetPUBG)

```

```
## [1] "Id"                 "groupId"           "matchId"           "assists"
## [5] "boosts"             "damageDealt"        "DBNOs"              "headshotKills"
## [9] "heals"               "killPlace"          "killPoints"         "kills"
## [13] "killStreaks"        "longestKill"        "matchDuration"     "matchType"
## [17] "maxPlace"            "numGroups"          "rankPoints"         "revives"
## [21] "rideDistance"        "roadKills"          "swimDistance"      "teamKills"
## [25] "vehicleDestroys"   "walkDistance"       "weaponsAcquired"  "winPoints"
## [29] "winPlacePerc"
```

```
#Lire le test set
testsetPUBG <- read.csv2("D:/Dataset/PUBG/test_V2.csv",
                           sep = ",", header = TRUE, fileEncoding = "utf-8")
#Le mettre dans une dataframe
testsetPUBG <- as.data.frame(testsetPUBG)
#Afficher le nom des colonnes
colnames(testsetPUBG)
```

```
## [1] "Id"                 "groupId"           "matchId"           "assists"
## [5] "boosts"             "damageDealt"        "DBNOs"              "headshotKills"
## [9] "heals"               "killPlace"          "killPoints"         "kills"
## [13] "killStreaks"        "longestKill"        "matchDuration"     "matchType"
## [17] "maxPlace"            "numGroups"          "rankPoints"         "revives"
## [21] "rideDistance"        "roadKills"          "swimDistance"      "teamKills"
## [25] "vehicleDestroys"   "walkDistance"       "weaponsAcquired"  "winPoints"
```

On voit bien qu'entre les 2 jeux de données nous avons les mêmes variables, moyennant bien sûr une seule, notre variable à prédire (`winPlacePerc`) qui est présente uniquement dans le jeu d'entraînement.

Explication des variables

Id = C'est l'identifiant unique du joueur

groupId = C'est l'identifiant du groupe du joueur, selon le type de partie nous pouvons jouer tout seul, à 2 ou à 4.

matchId = C'est l'identifiant unique du match

assists = Le nombre d'assistance, une assistance c'est quand l'on touche un autre joueur mais que c'est un autre joueur (ami ou ennemi) qui mets la dernière balle pour le finir.

boosts = Le nombres de **boosts** pris par le joueur. Les **boosts** sont des éléments qui permettent de se régénérer un petit peu de vie et de se déplacer plus vite.

damageDealt = Le nombre de dégâts réalisé, 1 point de vie enlevé à un ennemi correspondent à 1 point de dégat.

DBNOs = Le nombre d'ennemi que l'on a mis K.O, dans les parties en équipes avant de mourrir on a un état supplémentaire qui est K.O, les alliés ont alors quelques secondes pour réanimer leurs allié ou les ennemis pour le finir.

headshotKills = Le nombre de personne tués d'un tir dans la tête

heals = Le nombre de soins utilisés

killPlace = C'est le placement dans la partie du joueur d'un point de vue de **kills**

killPoints = Un peu comme un ELO (score aux échecs par exemple) d'un point de vue des **kills** uniquement

kills = Le nombre de personnes tués

killStreaks = Nombres de personnes tués lors d'un enchaînement (quelques secondes pour enchaîner)

longestKill = Distance en m, c'est la distance à laquelle un joueur a réalisé un kill du plus loin

matchDuration = Durée du match

matchType = Type du match, solo duo ou squad ainsi que la carte

maxPlace = Plus mauvais placement d'un joueur

numGroups = Nombre de groupes

rankPoints = Un peu comme un ELO d'un point de vue des points de classement uniquement.

revives = Nombres de fois où on a relevé (de l'état de KO) des équipiers.

rideDistance = Distance (en m) parcouru en véhicule

roadKills = Nombres de personnes tués en véhicule

swimDistance = Distance (en m) parcouru à la nage

teamKills = Nombre d'alliés que le joueur a tué

vehicleDestroys = Nombre de véhicule détruit

walkDistance = Distance (en m) parcouru à pieds

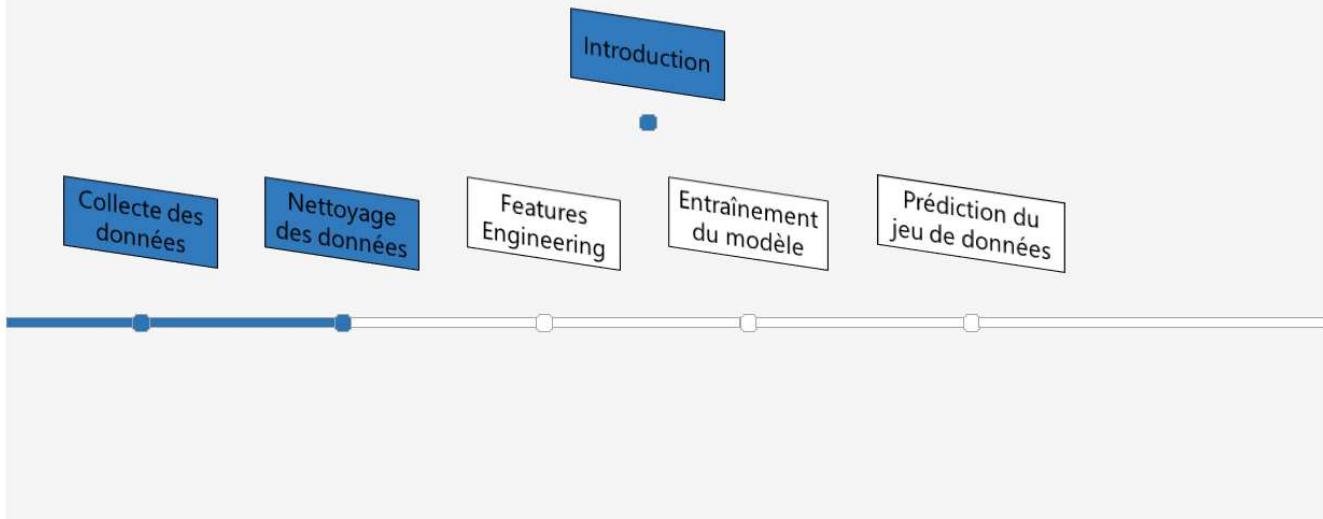
weaponsAcquired = Nombre d'armes ramassées

winPoints = Un peu comme un ELO d'un point de vue des victoires uniquement

winPlacePerc = C'est le pourcentage de chance d'un joueur de remporter la partie, c'est également la variable que l'on cherche à prédire

Analyse et nettoyage des données

Étapes à suivre



```
summary(trainsetPUBG)
```

```

##      Id          groupId        matchId        assists
## Length:4446966 Length:4446966 Length:4446966 Min.   : 0.0000
## Class :character Class :character Class :character 1st Qu.: 0.0000
## Mode  :character Mode  :character Mode  :character Median : 0.0000
##                                         Mean   : 0.2338
##                                         3rd Qu.: 0.0000
##                                         Max.   :22.0000
##      boosts       damageDealt        DBNOs        headshotKills
## Min.   : 0.000 Length:4446966 Min.   : 0.0000 Min.   : 0.0000
## 1st Qu.: 0.000 Class :character 1st Qu.: 0.0000 1st Qu.: 0.0000
## Median : 0.000 Mode  :character Median : 0.0000 Median : 0.0000
## Mean   : 1.107                               Mean   : 0.6579 Mean   : 0.2268
## 3rd Qu.: 2.000                               3rd Qu.: 1.0000 3rd Qu.: 0.0000
## Max.   :33.000                               Max.   :53.0000 Max.   :64.0000
##      heals         killPlace        killPoints        kills
## Min.   : 0.00 Min.   : 1.0 Min.   : 0 Min.   : 0.0000
## 1st Qu.: 0.00 1st Qu.: 24.0 1st Qu.: 0 1st Qu.: 0.0000
## Median : 0.00 Median : 47.0 Median : 0 Median : 0.0000
## Mean   : 1.37 Mean   : 47.6 Mean   : 505 Mean   : 0.9248
## 3rd Qu.: 2.00 3rd Qu.: 71.0 3rd Qu.:1172 3rd Qu.: 1.0000
## Max.   :80.00 Max.   :101.0 Max.   :2170 Max.   :72.0000
##      killStreaks      longestKill        matchDuration        matchType
## Min.   : 0.000 Length:4446966 Min.   : 9 Length:4446966
## 1st Qu.: 0.000 Class :character 1st Qu.:1367 Class :character
## Median : 0.000 Mode  :character Median :1438 Mode  :character
## Mean   : 0.544                               Mean   :1580
## 3rd Qu.: 1.000                               3rd Qu.:1851
## Max.   :20.000                               Max.   :2237
##      maxPlace      numGroups        rankPoints        revives
## Min.   : 1.0 Min.   : 1.00 Min.   : -1 Min.   : 0.0000
## 1st Qu.: 28.0 1st Qu.: 27.00 1st Qu.: -1 1st Qu.: 0.0000
## Median : 30.0 Median : 30.00 Median :1443 Median : 0.0000
## Mean   : 44.5 Mean   : 43.01 Mean   : 892 Mean   : 0.1647
## 3rd Qu.: 49.0 3rd Qu.: 47.00 3rd Qu.:1500 3rd Qu.: 0.0000
## Max.   :100.0 Max.   :100.00 Max.   :5910 Max.   :39.0000
##      rideDistance      roadkills        swimDistance        teamKills
## Length:4446966 Min.   : 0.000000 Length:4446966 Min.   : 0.00000
## Class :character 1st Qu.: 0.000000 Class :character 1st Qu.: 0.00000
## Mode  :character Median : 0.000000 Mode  :character Median : 0.00000
##                                         Mean   : 0.003496 Mean   : 0.02387
##                                         3rd Qu.: 0.000000 3rd Qu.: 0.00000
##                                         Max.   :18.000000 Max.   :12.00000
##      vehicleDestroys      walkDistance        weaponsAcquired        winPoints
## Min.   :0.000000 Length:4446966 Min.   : 0.00 Min.   : 0.0
## 1st Qu.:0.000000 Class :character 1st Qu.: 2.00 1st Qu.: 0.0
## Median :0.000000 Mode  :character Median : 3.00 Median : 0.0
## Mean   :0.007918                               Mean   : 3.66 Mean   : 606.5
## 3rd Qu.:0.000000                               3rd Qu.: 5.00 3rd Qu.:1495.0
## Max.   :5.000000                               Max.   :236.00 Max.   :2013.0
##      winPlacePerc
## Length:4446966
## Class :character
## Mode  :character
## 
## 
## 
```

Première constatation il y a des colonnes qui sont en **character** alors qu'elles sont constitués de valeurs numérique : - DamageDealt

- longestKill
- rideDistance
- swimDistance
- walkDistance
- winPlacePerc

Nous allons donc transformer ces colonnes en valeurs numériques.

```
trainsetPUBG$winPlacePerc <- as.numeric(trainsetPUBG$winPlacePerc)
trainsetPUBG$damageDealt <- as.numeric(trainsetPUBG$damageDealt)
trainsetPUBG$longestKill <- as.numeric(trainsetPUBG$longestKill)
trainsetPUBG$rideDistance <- as.numeric(trainsetPUBG$rideDistance)
trainsetPUBG$swimDistance <- as.numeric(trainsetPUBG$swimDistance)
trainsetPUBG$walkDistance <- as.numeric(trainsetPUBG$walkDistance)
```

Afin que notre jeu de données soit le plus précis possible regardons de plus près chaque variable et essayons de détecter d'éventuelles problèmes ou tout simplement valeur que l'on souhaite regarder de plus près.

- 53 **DBNOs** alors que la moyenne est de 0.6579
- 64 **headshotKills** alors que la moyenne est de 0.2268
- 80 **heals** alors que 3/4 des valeurs sont en dessous de 2
- 101 **killPlace** alors que le maximum de joueurs est de 100, le maximum de killPlace devrait aussi être de 100. On regardera cette valeur de plus près pour essayer de voir si c'est un bug ou si il y a bien eu une partie à 101 joueurs.
- 72 **kills** quand la moyenne est égale à 0.9248
- 20 **killStreaks**, de ma connaissance du jeu il me paraît impossible de réaliser un enchaînement de 20 joueurs la aussi on va regarder de plus près pour voir si on détecte une anomalie.
- 39 **revives** pour une moyenne de 0.1647
- 40710 **rideDistance** pour une moyenne de 606.12, de plus la aussi la zone se réduisant tout de même rapidement il me paraît difficile de parcourir 40km sans avoir croisé un alpha.
- 18 **roadKills** pour une moyenne de 0.003496, également ici il me paraît difficile d'écraser 18 joueurs sans tomber sur plus fort ou plus malin que soi.
- 25780 **walkDistance** comme pour la rideDistance 25.7 km a pieds semble beaucoup trop. Surtout le comportement d'un bon joueur même si très mobile reste de très bien se placer et de profiter de placement avantageux pour prendre le dessus sur ces adversaires.
- 236 **weaponsAcquired**, la encore comment ramasser 236 armes en une seule partie?
- **winPlacePerc**, une valeur N/A dans la colonne winPlacePerc n'est pas souhaitable, nous allons donc supprimer cette ligne de la dataframe.

Certaines des valeurs plus haut mène à réflexion, afin de pouvoir mieux visualiser si ce sont des valeurs vraiment extrême (liés à des cheateurs par exemple) ou si ce sont juste des très bon joueurs.

Pour cela on va imprimer les graphes de ses variables afin de mieux le visualiser.

```
#plot(trainsetPUBG$DBNOs)
#plot(trainsetPUBG$headshotKills)
#plot(trainsetPUBG$heals)
#plot(trainsetPUBG$kills)
#plot(trainsetPUBG$killStreaks)
#plot(trainsetPUBG$rideDistance)
#plot(trainsetPUBG$roadKills)
#plot(trainsetPUBG$walkDistance)
#plot(trainsetPUBG$weaponsAcquired)
```

Avec un graphique rapide bien que sommaire on voit très bien la distribution, la valeur max de 53 paraît encore plus étonnant, on va étudier ça de plus près.
Tout d'abord en regardant les lignes de la dataframe pour lequel le **DBNOs** est supérieur à 40.

```
DBNOs = trainsetPUBG[trainsetPUBG$DBNOs > 40, ]
print(DBNOs)
```

```
##                               Id      groupId      matchId assists boosts damageDealt
## 3673966 f83f0bfaafb7d8 22d80bf00f56b7 fc0bbecba8db99      2     1      5330
##           DBNOs headshotKills heals killPlace killPoints kills killStreaks
## 3673966    53          41     21         1      1000     55        14
##           longestKill matchDuration      matchType maxPlace numGroups
## 3673966     310.3        1796 normal-squad-fpp       11       11
##           rankPoints revives rideDistance roadKills swimDistance teamKills
## 3673966      -1         0         0         0         0         0
##           vehicleDestroys walkDistance weaponsAcquired winPoints winPlacePerc
## 3673966          0      12.19        17      1500        1
```

La encore en regardant la ligne dans son ensemble, c'est étrange le joueur **f83f0bfaafb7d8** à réaliser 55 **kills** en parcourant seulement 12.19m.

Intéressant nous maintenant aux **headshotKills**, regardons également les valeurs au dessus de 40 pour voir si on a à faire à d'excellent joueur ou des tricheurs.

```
HSkills = trainsetPUBG[trainsetPUBG$headshotKills > 40, ]
print(HSkills)
```

```

##           Id      groupId      matchId assists boosts damageDealt
## 1454066 c47bd86daa3de6 4df3e348b910d8 3ebf1bf8bc6bae      2      2     4495
## 2020832 9b5bcc3a3dd42a bdd2cf09863501 6bf647ecee30da      2      0     4889
## 3431248 06308c988bf0c2 4c4ee1e9eb8b5e 6680c7c3d17d48      7      4     5990
## 3673966 f83f0bfaafb7d8 22d80bf00f56b7 fc0bbecba8db99      2      1     5330
##          DBNOs headshotKills heals killPlace killPoints kills killStreaks
## 1454066    0            42      2       1     1000     50       7
## 2020832    0            46      2       1       0     53      10
## 3431248    0            64     10       1       0     72       7
## 3673966   53            41     21       1     1000     55      14
##          longestKill matchDuration      matchType maxPlace numGroups
## 1454066   324.20        1136 normal-solo-fpp      30      17
## 2020832   690.40        833  normal-squad-fpp     13      13
## 3431248   78.23        1800  normal-squad      15      15
## 3673966   310.30        1796 normal-squad-fpp     11      11
##          rankPoints revives rideDistance roadKills swimDistance teamKills
## 1454066    -1            0       0       0       0       0       0
## 2020832   1500           0       0       0       0       0       0
## 3431248   1500           0       0       0       0       0       0
## 3673966    -1            0       0       0       0       0       0
##          vehicleDestroys walkDistance weaponsAcquired winPoints winPlacePerc
## 1454066      0      1021.00      15     1500       1
## 2020832      0      272.30      14       0       1
## 3431248      0      728.10      35       0       1
## 3673966      0      12.19      17     1500       1

```

Sans surprise on retrouve notre ami **f83f0bfaafb7d8** mais ce n'est pas lui qui à le plus de **headshotKills**.

Difficile de trancher ici, il est possible que ces 3 joueurs ne soit pas des tricheurs.

Jettons un coup d'oeil aux autres valeurs maximum étrange en utilisant la même méthode

```

heals = trainsetPUBG[trainsetPUBG$heals > 60, ]
print(heals)

```

```

##           Id      groupId      matchId assists boosts damageDealt
## 700616 512ca0f7c6729b de2c3988154e5e 03a24daa0f3e65      0     3    0.000
## 996710 6737eed1cf8dfc 12cc81f7d2eb4d 5f9a59a3ef190d      0     6    1.824
## 3419346 4d6c412604232f ff7daf14a68de4 fea1354b6e5089      1     8   146.700
## 3476344 1ac3c7d24475c8 ea7008368b4c5d fcc78beee326e3      1     4   207.200
## 4262663 3be1ded892f443 5c6c4e66418c25 ce09de37cf3f5a      0     1   230.000
## 4397806 703ce5805efed7 b84c7dd65f03cd d8fa2223c98979      0     1   44.510
##          DBNOs headshotKills heals killPlace killPoints kills killStreaks
## 700616      3            0    61      48        0     0     0
## 996710      0            0    63      37     1025     0     0
## 3419346      0            0    61      30     1402     1     1
## 3476344      0            0    73      26        0     1     1
## 4262663      5            0    80      41     1440     0     0
## 4397806      0            0    62      44        0     0     0
##          longestKill matchDuration matchType maxPlace numGroups rankPoints
## 700616      0.00        1876      duo       45      44   1519
## 996710      0.00        1925  solo-fpp      80      76    -1
## 3419346      2.36        1975      duo       45      45    -1
## 3476344     79.82        1888  squad-fpp      28      28   1555
## 4262663      0.00        1866      squad      26      25    -1
## 4397806      0.00        1867      duo       50      47   1497
##          revives rideDistance roadKills swimDistance teamKills vehicleDestroys
## 700616      3      662.8        0        0      1     0
## 996710      0      3906.0        0        0      0     0
## 3419346      1      7320.0        0        0      0     0
## 3476344      2      4677.0        0        0      0     0
## 4262663      4      3367.0        0        0      0     0
## 4397806      0      0.0         0        0      0     0
##          walkDistance weaponsAcquired winPoints winPlacePerc
## 700616      963.8        2        0    0.6364
## 996710     1752.0        4     1516    0.8481
## 3419346     1762.0        6     1620    0.8409
## 3476344     3078.0        4        0    0.8889
## 4262663     932.7        4     1527    0.7600
## 4397806     2629.0        6        0    0.8367

```

```

kills = trainsetPUBG[trainsetPUBG$kills > 60, ]
print(kills)

```

```

##           Id      groupId      matchId assists boosts damageDealt
## 334401 810f2379261545 7f3e493ee71534 f900de1ec39fa5      20      0     6616
## 1248349 80ac0bbf58bfaf 1e54ab4540a337 08e4c9e6c033e2       5      0     6375
## 3431248 06308c988bf0c2 4c4ee1e9eb8b5e 6680c7c3d17d48       7      4     5990
##          DBNOs headshotKills heals killPlace killPoints kills killStreaks
## 334401      0            13      5      1      0    65      7
## 1248349      0            21      4      1      0    66      8
## 3431248      0            64     10      1      0    72      7
##          longestKill matchDuration      matchType maxPlace numGroups rankPoints
## 334401      73.90        1798 normal-solo-fpp      11      11    1500
## 1248349      319.90       1390 normal-solo-fpp      18      12    1500
## 3431248      78.23        1800 normal-squad      15      15    1500
##          revives rideDistance roadKills swimDistance teamKills vehicleDestroys
## 334401      0            0      0      0      0      0      0
## 1248349      0            0      0      0      0      0      0
## 3431248      0            0      0      0      0      0      0
##          walkDistance weaponsAcquired winPoints winPlacePerc
## 334401      1036.0        60      0      1
## 1248349      1740.0        23      0      1
## 3431248      728.1         35      0      1

```

```

killstreak = trainsetPUBG[trainsetPUBG$killStreaks > 15, ]
print(killstreak)

```

```

##           Id      groupId      matchId assists boosts damageDealt
## 2714365 df578079868527 c9dab2739e98a5 367dd0033fae50      0      0     1683
## 2890741 a3438934e3e535 1081c315a80d14 fe744430ac0070      0      8     2074
## 3924730 579949f753978c 108f4c00d80882 ff9cd80c0d8fb7      1      2     3680
##          DBNOs headshotKills heals killPlace killPoints kills killStreaks
## 2714365      0            5      0      1    1000     16      16
## 2890741      0            1     11      1    1114     20      18
## 3924730      0            35      0      1    1000     40      20
##          longestKill matchDuration      matchType maxPlace numGroups
## 2714365      7.395        1367      solo      94      74
## 2890741      64.290       1970      solo      38      20
## 3924730      621.800       799 normal-squad-fpp      10      10
##          rankPoints revives rideDistance roadKills swimDistance teamKills
## 2714365      -1            0      0      0      0      0      0
## 2890741      -1            0     2726      18      0      0
## 3924730      -1            0      0      0      0      0      0
##          vehicleDestroys walkDistance weaponsAcquired winPoints winPlacePerc
## 2714365      0            258      2    1500   0.4624
## 2890741      0            3150      4    1568   1.0000
## 3924730      0            1180      2    1500   1.0000

```

```

rideD = trainsetPUBG[trainsetPUBG$rideDistance > 30000, ]
print(rideD)

```

```

##          Id      groupId      matchId assists boosts damageDealt
## 426709 149e224a2330ae 6d8cb80b3de8ff f8b8e2643f60ee    0     2     0.000
## 605624 8dee54600c67ca e77f3e175b5959 26bc606cdae6bf    0     0    11.210
## 1214638 470dca70fbef4d d44161e5d29c1a c7d52d5aaff270    0     0   127.300
## 1232005 2c5add87b29a8c 1d9d7faa352311 ef2ae297b170f3    0     7   196.900
## 1232363 4bf27787223351 b0140dd17f3238 5ed9c405f7e71d    0     1     0.000
## 1361248 f09d0c99127850 4ca387960d74a7 93aa204e9c193e    0     2   265.600
## 1713774 239905dd1f3787 da3d922a92ace5 692bd6a11b86c8    0     0   141.200
## 1725338 9f14f77b6dd9b6 8a2a8ae2506323 904b5fa7f59bc5    0     2     2.389
## 2137064 73b793b4e7523f f4444641af7e5c 6385f7641268c3    0     0     0.000
## 2700692 3428324a089608 fa77b4ac736dbf 9b1c87f4016f47    0     1   31.270
## 2927409 260efc0cf6b36c b112f8d21e8185 3e977510ce1d3c    0     0   11.220
## 3404579 8e84f4228e5436 c5ca5892142cbd 56775920f704e8    0     0     3.298
## 3567266 99f342c6022411 9158efebffecc7 ddbe41fe165c7f    2     2   113.200
## 4099939 afe34618b04d77 c14f532b283844 732c3b752e055f    0     0     0.000
## 4150058 a74c1acff09c4f a1488a2d301bc4 98f74c2a19a67e    0     0   18.760
## 4270944 7a9ef7be64e37e 4873e2b0d83c69 653dc89930fdc2    0     7   560.100
## 4440262 9a7c448670f199 04b0e435abc817 a4063f40804b24    0     0     0.000
##          DBNOs headshotKills heals killPlace killPoints kills killStreaks
## 426709     0         0     2      43     1120     0     0
## 605624     0         0     0      27     1179     0     0
## 1214638     0         0     8      24      0     1     1
## 1232005     0         0     4      24     1517     1     1
## 1232363     0         0    10      47      0     0     0
## 1361248     0         0     4      10     1568     2     1
## 1713774     0         0     1      21     1216     1     1
## 1725338     0         0     5      19     1193     0     0
## 2137064     0         0     0      52      0     0     0
## 2700692     0         0     0      46     1211     0     0
## 2927409     0         0     0      44     1281     0     0
## 3404579     0         0     0      40     1060     0     0
## 3567266     0         0     3      34      0     0     0
## 4099939     0         0     0      42      0     0     0
## 4150058     0         0     0      51     1236     0     0
## 4270944     0         3     3      2     1602     6     2
## 4440262     0         0     0      45      0     0     0
##          longestKill matchDuration matchType maxPlace numGroups rankPoints
## 426709     0.00       2007      solo      83      83     -1
## 605624     0.00       1851      solo      51      50     -1
## 1214638     6.30       1855  solo-fpp      95      92   1541
## 1232005    85.44       1862  solo-fpp      98      96     -1
## 1232363     0.00       1896      solo      95      91   1501
## 1361248    26.68       1804  solo-fpp      92      88     -1
## 1713774    87.71       1719  solo-fpp      96      93     -1
## 1725338     0.00       1918  solo-fpp      37      35     -1
## 2137064     0.00       1794  solo-fpp      94      92   1512
## 2700692     0.00       1886  solo-fpp      98      95     -1
## 2927409     0.00       1865      solo      92      90     -1
## 3404579     0.00       1877  solo-fpp      98      96     -1
## 3567266     0.00       1958 duo-fpp      48      47   1729
## 4099939     0.00       1939  solo-fpp      93      90   1486
## 4150058     0.00       1969  solo-fpp      98      96     -1
## 4270944    250.50       1740      solo      99      97     -1
## 4440262     0.00       2055      solo      96      90   1500
##          revives rideDistance roadKills swimDistance teamKills vehicleDestroys
## 426709     0       31960     0      0.00      0     0
## 605624     0       33970     0      0.00      0     0

```

## 1214638	0	31960	1	0.00	0	0
## 1232005	0	30030	0	53.38	0	0
## 1232363	0	31010	0	0.00	1	0
## 1361248	0	35400	0	0.00	0	0
## 1713774	0	31500	0	0.00	0	0
## 1725338	0	37670	0	0.00	0	0
## 2137064	0	31370	0	0.00	0	0
## 2700692	0	31290	0	0.00	0	0
## 2927409	0	30660	0	0.00	0	0
## 3404579	0	32320	0	0.00	0	0
## 3567266	1	31020	0	0.00	0	0
## 4099939	0	40710	0	0.00	0	0
## 4150058	0	30110	0	0.00	0	0
## 4270944	0	32450	0	0.00	0	1
## 4440262	0	31690	0	0.00	0	0
<hr/>						
## walkDistance	weaponsAcquired	winPoints	winPlacePerc			
## 426709	402.10	3	1449	0.8171		
## 605624	1641.00	0	1498	0.6800		
## 1214638	642.00	1	0	0.9787		
## 1232005	1328.00	2	1565	0.8866		
## 1232363	1473.00	3	0	0.7979		
## 1361248	1912.00	7	1538	1.0000		
## 1713774	448.20	3	1537	0.9474		
## 1725338	542.10	2	1565	0.6944		
## 2137064	28.76	0	0	0.7204		
## 2700692	174.10	1	1530	0.8557		
## 2927409	426.60	2	1619	0.8132		
## 3404579	233.70	5	1497	0.9175		
## 3567266	616.20	4	0	1.0000		
## 4099939	560.10	0	0	0.8478		
## 4150058	27.30	0	1505	0.6392		
## 4270944	969.00	5	1576	0.9898		
## 4440262	23.77	0	0	0.7474		

```
roadK = trainsetPUBG[trainsetPUBG$roadKills > 10, ]
print(roadK)
```

```

##           Id      groupId      matchId assists boosts damageDealt
## 2733927 c3e444f7d1289f 489dd6d1f2b3bb 4797482205aaa4      0      0     1246
## 2768000 34193085975338 bd7d50fa305700 a22354d036b3d6      0      0     1102
## 2890741 a3438934e3e535 1081c315a80d14 fe744430ac0070      0      8     2074
## 3524414 9d9d044f81de72 8be97e1ba792e3 859e2c2db5b125      0      3     1866
##          DBNOs headshotKills heals killPlace killPoints kills killStreaks
## 2733927      0            0      0      1      1403     14      13
## 2768000      0            0      0      1      1497     11      11
## 2890741      0            1     11      1      1114     20      18
## 3524414      0            5      7      1      1520     18      11
##          longestKill matchDuration matchType maxPlace numGroups rankPoints
## 2733927      7.181        1456      solo      92      80      -1
## 2768000      0.000        1946      solo      88      77      -1
## 2890741      64.290       1970      solo      38      20      -1
## 3524414      341.400       1925      solo      84      70      -1
##          revives rideDistance roadKills swimDistance teamKills vehicleDestroys
## 2733927      0      0.005      14      5.297      0      0
## 2768000      0    4118.000      11      0.000      0      0
## 2890741      0    2726.000      18      0.000      0      0
## 3524414      0    6812.000      11      0.000      0      0
##          walkDistance weaponsAcquired winPoints winPlacePerc
## 2733927      1277.0        0      1371      0.4286
## 2768000      816.6        5      1533      0.4713
## 2890741      3150.0        4      1568      1.0000
## 3524414      1041.0       10      1606      0.9398

```

```

walkD = trainsetPUBG[trainsetPUBG$walkDistance > 15000, ]
print(walkD)

```

```

##           Id      groupId      matchId assists boosts damageDealt
## 497966 3304d0c68e27f5 503757159d22dd ce88f9c8c35bdf      0      0     0.0
## 1797163 7b9a750b17e9c6 2d00192e4029c5 6fe77402ddb0e1      0      0     0.0
## 2238901 861d1e5a63f3e6 1da6a90addir96d 16f2c51578f323      0      1     0.0
## 2395010 d6d37216c02c17 56dd8821cacb32 f27850950d5597      0      3     273.3
##          DBNOs headshotKills heals killPlace killPoints kills killStreaks
## 497966      0            0      0      46      0      0      0
## 1797163      0            0      0      55      0      0      0
## 2238901      0            0      5      46      0      0      0
## 2395010      0            0      1      11     1300      2      1
##          longestKill matchDuration matchType maxPlace numGroups rankPoints
## 497966      0.0        1889      solo-fpp      95      93     1520
## 1797163      0.0        1977      squad-fpp      26      25     1482
## 2238901      0.0        1828      solo      97      95     1651
## 2395010     136.8        1932      solo-fpp      97      93      -1
##          revives rideDistance roadKills swimDistance teamKills vehicleDestroys
## 497966      0      0.0078      0      381.90      0      0
## 1797163      0      0.0401      0      129.40      0      0
## 2238901      0      0.0111      0      389.50      0      0
## 2395010      0    1575.0000      0      59.05      0      0
##          walkDistance weaponsAcquired winPoints winPlacePerc
## 497966      25780        0      0      0.9894
## 1797163      16250        3      0      0.6000
## 2238901      15370        7      0      0.8125
## 2395010      15130        3     1583      1.0000

```

```
wA = trainsetPUBG[trainsetPUBG$weaponsAcquired > 150, ]
print(wA)
```

```
##          Id      groupId      matchId assists boosts damageDealt
## 2743409 afcb46681b909f 3715298b7eea9c 4dd4d42772464f      0     4    186.90
## 2749694 940b52bf12805a 4757d102ca4fbf 7ecae2cedc064f      1     3    378.90
## 2797868 da68d2812229a8 536975608768ed fbbdc018f4b771      0     0     0.00
## 2982526 3f2bcf53b108c4 2dbc013e849f5a 1df2560f0937ab      0     1    85.85
##          DBNOs headshotKills heals killPlace killPoints kills killStreaks
## 2743409    2           0     2       18     1288     2       2
## 2749694    2           0     5       7     1791     3       1
## 2797868    0           0     0       58       0     0       0
## 2982526    0           0     2       55     1664     0       0
##          longestKill matchDuration matchType maxPlace numGroups rankPoints
## 2743409    33.43        1511      duo      45       45      -1
## 2749694    140.50        1400  squad-fpp      27       26      -1
## 2797868    0.00        1793  solo-fpp      98       95   1527
## 2982526    0.00        1928  squad-fpp      29       28      -1
##          revives rideDistance roadKills swimDistance teamKills vehicleDestroys
## 2743409    0      1675.0       0       0       1       0
## 2749694    0      303.7       0       0       0       0
## 2797868    0       0.0       0       0       0       0
## 2982526    0      4778.0       0       0       0       0
##          walkDistance weaponsAcquired winPoints winPlacePerc
## 2743409    1643         167     1452    0.7955
## 2749694    3338         177     1711    0.8846
## 2797868    1235         153       0    0.5464
## 2982526    2112         236     1553    0.6786
```

Difficile de trancher mais tout paraît bien légitime.

On va juste supprimer la ligne de notre ami tricheur et la ligne avec la valeur **N/A**.

```
#Nombres de Lignes dans La DataFrame
nrow(trainsetPUBG)
```

```
## [1] 4446966
```

```
trainsetPUBG <- subset(trainsetPUBG, Id!="f83f0bfaafb7d8")
#Nombres de Lignes dans La DataFrame
nrow(trainsetPUBG)
```

```
## [1] 4446965
```

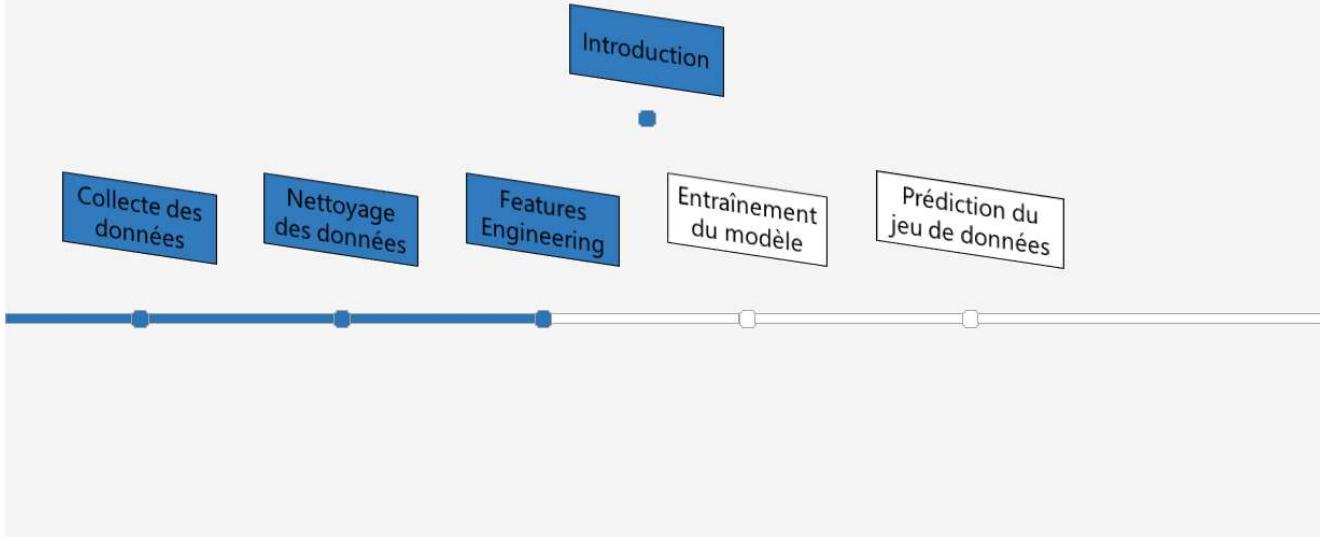
```
trainsetPUBG <- trainsetPUBG[ !is.na(trainsetPUBG$winPlacePerc),]
#Nombres de Lignes dans La DataFrame
nrow(trainsetPUBG)
```

```
## [1] 4446964
```

Tout s'est bien déroulé on a bien supprimer seulement les 2 lignes qu'on souhaitait.

Un peu de features engineering

Étapes à suivres



Ici nous allons le faire plus dans un soucis de réduire le nombres de variables que dans un réel soucis d'efficacité, mais on va au maximum essayer de faire les 2.

Il y a pas mal de variables rédundantes, on va donc créer une colonne **distanceParcouru** qui sera la distance totale parcouru par le joueur.

On va également créer une colonne **healsBoosts** qui sera la somme de ces 2 consommables.

Les tirs à la tête nécessitant beaucoup plus de skills que les **kills** classique on va rajouter cette colonne, on ne supprimera pas les colonnes car on veux quand même pouvoir étudier les **kills** et **headshotKills** indépendamment.

```

trainsetPUBG$DistanceParcouru = trainsetPUBG$rideDistance + trainsetPUBG$swimDistance + trainsetPUBG$walkDistance
trainsetPUBG$healsBoosts = trainsetPUBG$heals + trainsetPUBG$boosts
trainsetPUBG$HSratio = trainsetPUBG$headshotKills / trainsetPUBG$kills
trainsetPUBG$GlobalPoints = trainsetPUBG$winPoints + trainsetPUBG$killPoints + trainsetPUBG$rankPoints
trainsetPUBG$KDA = trainsetPUBG$kills + trainsetPUBG$assists/3
#suppression des colonnes désormais inutile
trainsetPUBG = trainsetPUBG[, !(colnames(trainsetPUBG) %in% c("rideDistance","swimDistance","walkDistance","heals","boosts","Id","groupId","matchId","kills","assists","winPoints","killPoints","rankPoints"))]
trainsetPUBG[is.na(trainsetPUBG)] <- 0
head(trainsetPUBG, 10)
  
```

```

##   damageDealt DBNOS headshotKills killPlace killStreaks longestKill
## 1      0.000    0        0       60        0      0.00
## 2     91.470    0        0       57        0      0.00
## 3    68.000    0        0       47        0      0.00
## 4    32.900    0        0       75        0      0.00
## 5   100.000    0        0       45        1    58.53
## 6  100.000    1        1       44        1    18.44
## 7      0.000    0        0       96        0      0.00
## 8     8.538    0        0       48        0      0.00
## 9    51.600    0        0       64        0      0.00
## 10   37.270    0        0       74        0      0.00
##   matchDuration matchType maxPlace numGroups revives roadkills teamKills
## 1          1306 squad-fpp     28       26        0        0        0
## 2          1777 squad-fpp     26       25        0        0        0
## 3          1318 duo         50       47        0        0        0
## 4          1436 squad-fpp     31       30        0        0        0
## 5          1424 solo-fpp     97       95        0        0        0
## 6          1395 squad-fpp     28       28        0        0        0
## 7          1316 squad-fpp     28       28        0        0        0
## 8          1967 solo-fpp     96       92        0        0        0
## 9          1375 squad        28       27        0        0        0
## 10         1930 squad        29       27        0        0        0
##   vehicleDestroys weaponsAcquired winPlacePerc DistanceParcouru healsBoosts
## 1            0                1      0.4444    244.800        0
## 2            0                5      0.6400   1445.044        0
## 3            0                2      0.7755   161.800        0
## 4            0                3      0.1667   202.700        0
## 5            0                2      0.1875    49.750        0
## 6            0                1      0.0370    34.700        0
## 7            0                1      0.0000    13.500        0
## 8            0                6      0.7368  3093.000        0
## 9            0                4      0.3704   799.900        0
## 10           0                1      0.2143    65.670        0
##   HSratio GlobalPoints      KDA
## 1      0  2706 0.0000000
## 2      0  1484 0.0000000
## 3      0  1491 0.3333333
## 4      0  1408 0.0000000
## 5      0  1560 1.0000000
## 6      1  1418 1.0000000
## 7      0  2758 0.0000000
## 8      0  2499 0.0000000
## 9      0  1493 0.0000000
## 10     0  1349 0.0000000

```

Nous allons séparer le jeu de données en 3, en effet je pense que le jeux en solo ou en groupes (duo et squad) est différent, on va donc essayer de séparer nos jeux de données en 3 un pour la solo, un pour le duo et un pour les groupes.

```
vSolo <- c('flarefpp','flaretpp','crashfpp','crashtpp','normal-solo','normal-solo-fpp','solo-fpp')
trainsetSolo <- filter(trainsetPUBG, matchType %in% vSolo)
vDuo <- c('duo-fpp','normal-duo','normal-duo-fpp')
trainsetDuo <- filter(trainsetPUBG, matchType %in% vDuo)
vSquad <- c('normal-squad','normal-squad-fpp','squad-fpp')
trainsetSquad <- filter(trainsetPUBG, matchType %in% vSquad)
nrow(trainsetSolo)
```

```
## [1] 548650
```

```
nrow(trainsetDuo)
```

```
## [1] 1002379
```

```
nrow(trainsetSquad)
```

```
## [1] 1773875
```

Tout s'est bien déroulé, il y a des variables que l'on utilisera pas pour les joueurs solo. **groupId DBNOs matchType numGroups revives teamKills**

```
trainsetSolo = trainsetSolo[, !(colnames(trainsetSolo) %in% c("groupId","DBNOs","matchType",
"numGroups","revives","teamKills"))]
head(trainsetSolo, 10)
```

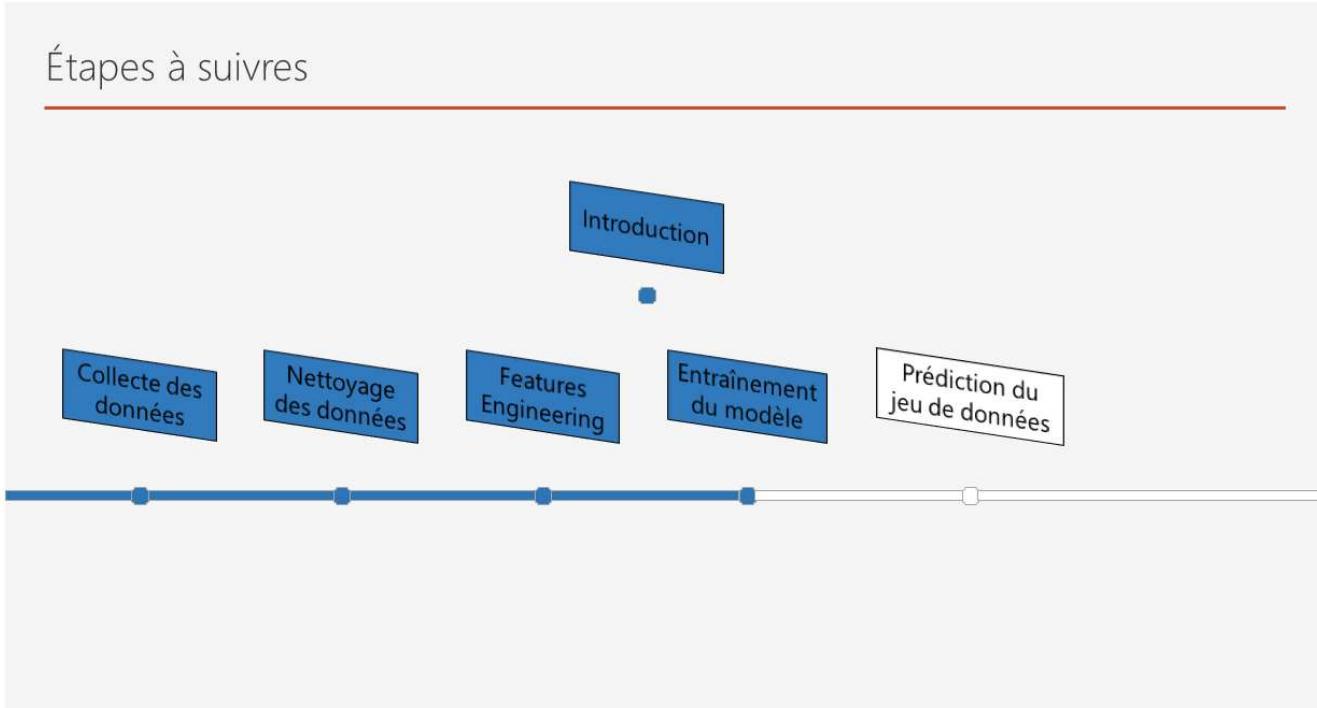
```

##   damageDealt headshotKills killPlace killStreaks longestKill matchDuration
## 1    100.000          0        45         1      58.53       1424
## 2     8.538          0        48         0       0.00       1967
## 3    324.200          1        5         1     49.83       1886
## 4    254.300          0       13         1     36.00       1371
## 5    136.900          0       37         1     22.83       1425
## 6    194.500          1       19         1     29.08       2067
## 7   100.000          1       29         1     21.86       1959
## 8     46.410          0       81         0       0.00       1493
## 9    604.700          2        4         1     99.23       1465
## 10   76.440          0       50         0       0.00       1441
##   maxPlace roadkills vehicleDestroys weaponsAcquired winPlacePerc
## 1      97          0            0           2     0.1875
## 2      96          0            0           6     0.7368
## 3      97          0            0           6     0.8750
## 4      96          0            0           3     0.8211
## 5      96          0            0           1     0.3474
## 6      96          0            0           5     0.2947
## 7      95          0            0           5     0.6809
## 8      95          0            0           1     0.1489
## 9      93          0            0           2     0.9891
## 10     95          0            0           2     0.8830
##   DistanceParcouru healsBoosts HSratio GlobalPoints      KDA
## 1        49.75          0    0.00      1560 1.000000
## 2     3093.00          0    0.00      2499 0.000000
## 3     3354.84          6    0.25      2447 4.000000
## 4     4169.29         15    0.00      1536 2.000000
## 5     270.70          0    0.00      1500 1.000000
## 6     248.60          1    0.50      1460 2.000000
## 7    1606.00          4    1.00      1698 1.000000
## 8      50.82          0    0.00      1499 0.000000
## 9    1766.00         13    0.40      1543 5.333333
## 10   2596.00          5    0.00      1560 0.000000

```

Cross Validation et entraînement du modèle

Étapes à suivres



Comme expliqué dans la documentation officielle de Breiman, avec randomForest il n'y a pas besoin de cross-validation ou de test séparé pour avoir une estimation des erreurs dans le jeu de test.

Pour se faire une idée nous avons l'OOB error estimate.

Vous pouvez retrouver l'explication complète ici
https://www.stat.berkeley.edu/%7Ebreiman/RandomForests/cc_home.htm#ooberr
(https://www.stat.berkeley.edu/%7Ebreiman/RandomForests/cc_home.htm#ooberr)

Passons donc à l'étape d'entraînement du modèle, pour commencer on va le faire avec les 10000 premières lignes pour que les temps de calculs soit convenable et en utilisant les paramètres par défaut.

```

#Entraînement du modèle
system.time({
  set.seed(123)

  solo <- randomForest(winPlacePerc ~ ., data = head(trainsetSolo, 10000), na.action = na.omit,
                        importance=T, keep.inbag = T)})

##      user  system elapsed
##     64.08    0.06   64.23

```

```
solo
```

```

## 
## Call:
##   randomForest(formula = winPlacePerc ~ ., data = head(trainsetSolo,      10000), importanc
e = T, keep.inbag = T, na.action = na.omit)
##           Type of random forest: regression
##           Number of trees: 500
## No. of variables tried at each split: 5
##
##           Mean of squared residuals: 0.005133088
##           % Var explained: 94.12

```

Number of trees c'est le nombre d'arbres que l'algorithme construit, ici avec la valeur par défaut 500, on peut le modifier dans randomForest en utilisant l'argument ntree.

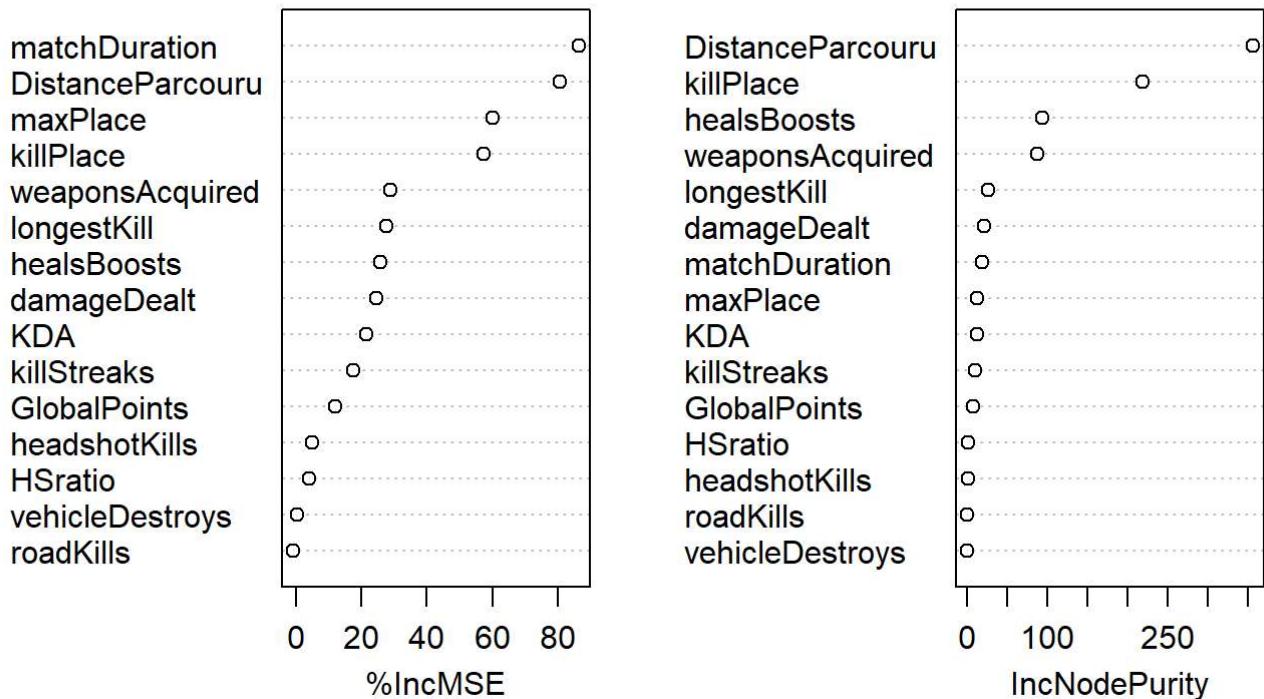
No. of variables tried at each split c'est le nombres de variables utilisés à chaque séparation dans notre cas 7. A noter que par défaut la valeur est égale à la racine carré du nombres de prédicteurs. On peut le modifier dans randomForest en utilisant l'argument mtry.

Mean of squared residuals nous permet d'évaluer les cas que le modèle ne peut pas expliquer, c'est la somme des différences au carré entre la valeur à prédire et la valeur prédictée. Plus cette valeur est basse plus le modèle de prédiction colle à la donnée, une valeur de 0 correspond à un modèle qui colle exactement à la donnée (overfitting). 0.005133088.

%Var explained: 94.12

Maintenant nous pouvons voir quelles variables ont le plus d'importance dans notre prédiction.

```
varImpPlot(solo)
```

solo

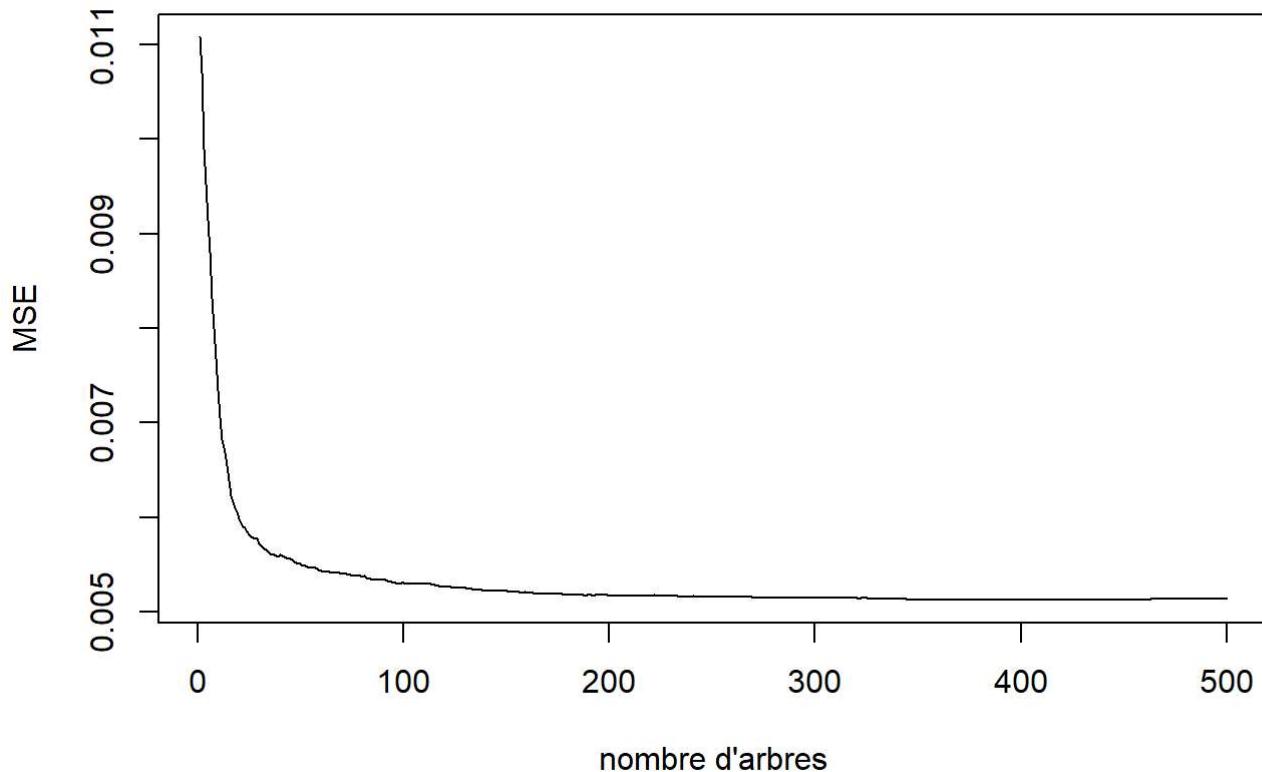
Ce qu'il faut interpréter ici c'est que plus une valeur est en haut de la liste plus elle est importante pour la prediction.

La variable la plus importante semble être la DistanceParcouru, et c'est logique étant donné les compétences nécessaires pour le jeu.

Essayons maintenant de voir si on peut améliorer légèrement tout ça en optimisant quelques valeurs comme le mtry et le ntree en utilisant la commande tuneRF.

Tout d'abord traçons le MSE (mean of squared residuals) en fonction du ntree, afin de trouver la valeur optimale pour le nombre d'arbres.

```
plot(solo$mse, type = "l", xlab = "nombre d'arbres", ylab = "MSE")
```

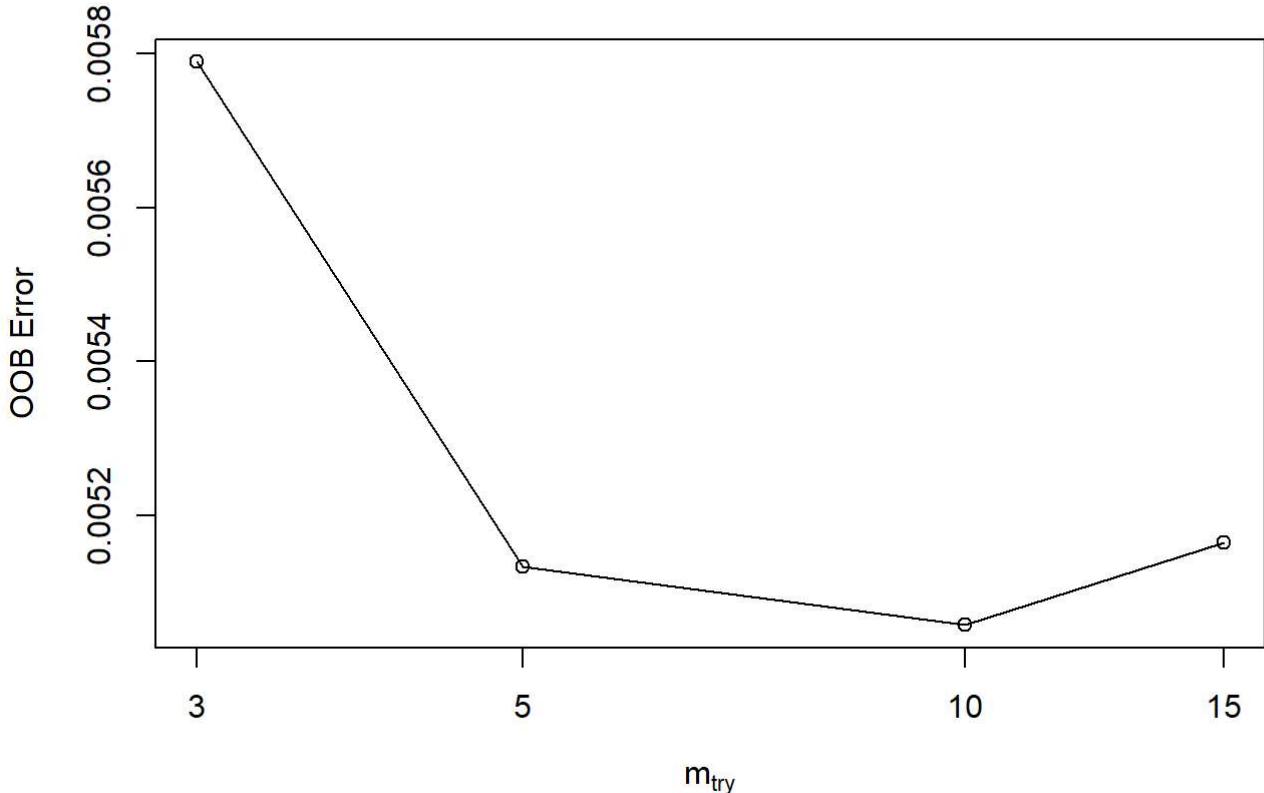


On voit que ça semble se stabiliser vers 300. On va donc utiliser cette valeur pour ntreeTry.

Pour **mtryStart** nous allons rester sur 5. improve, c'est de combien l'OOB error (dans notre cas le **MSE**) doit s'améliorer pour continuer, on va partir ici avec 0.001 car notre **MSE** est de 0.005133088

```
soloPredictors <- trainsetSolo[, !(colnames(trainsetSolo) %in% c("winPlacePerc"))]
tuneRFtest <- tuneRF(head(soloPredictors, 10000), head(trainsetSolo$winPlacePerc, 10000), mtryStart=5, ntreeTry=300, stepFactor=2, improve=0.001,
                      trace=TRUE, plot=TRUE, doBest=FALSE)
```

```
## mtry = 5 OOB error = 0.005132821
## Searching left ...
## mtry = 3      OOB error = 0.005789406
## -0.1279189 0.001
## Searching right ...
## mtry = 10     OOB error = 0.005057051
## 0.01476191 0.001
## mtry = 15     OOB error = 0.005163937
## -0.02113608 0.001
```



Le **mtry** optimal semble être 10.

Relançons notre modèle d'entraînement avec ses paramètres.

```
#Entraînement du modèle
system.time({
set.seed(123)

solo <- randomForest(winPlacePerc ~ ., data = head(trainsetSolo, 10000), na.action = na.omit,
importance=T, mtry=10, ntree=300)})
```

```
##    user  system elapsed
##   62.28    0.13   62.55
```

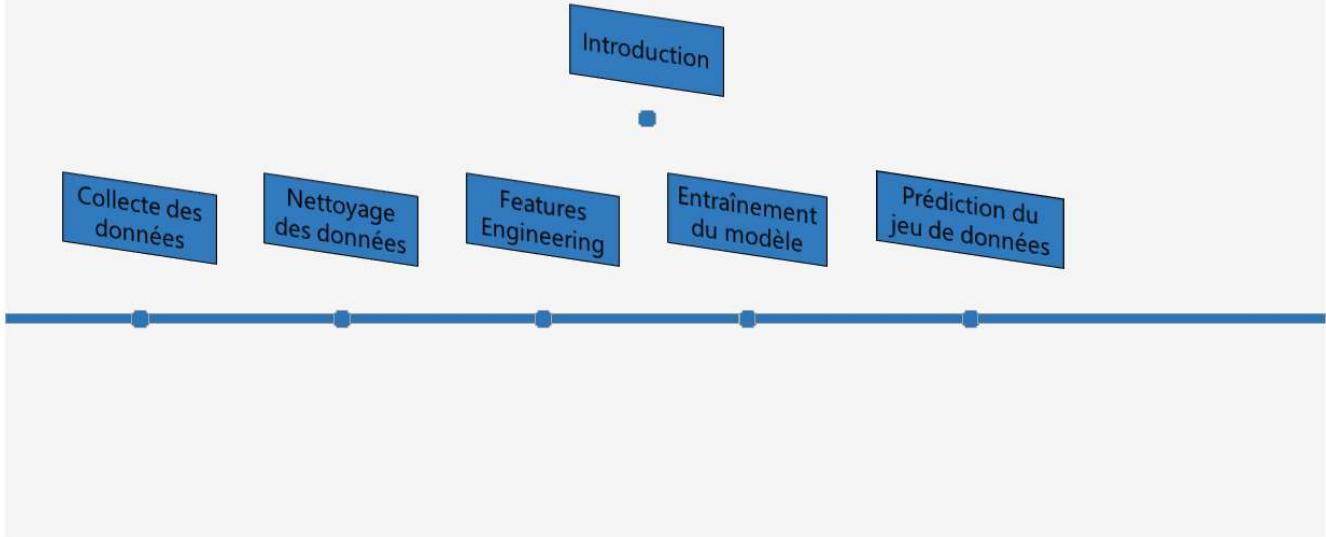
```
solo
```

```
##
## Call:
##  randomForest(formula = winPlacePerc ~ ., data = head(trainsetSolo,      10000), importance = T, mtry = 10, ntree = 300, na.action = na.omit)
##          Type of random forest: regression
##                      Number of trees: 300
## No. of variables tried at each split: 10
##
##          Mean of squared residuals: 0.005049242
##          % Var explained: 94.22
```

C'est un tout petit peu mieux.

Prédiction

Étapes à suivres



On va maintenant passer à la prédiction, on doit tout d'abord appliquer les mêmes modifications sur les variables dans le testSet et sortir les parties solo.

```
#Ici on veut pouvoir retrouver le joueur donc on passe la colonne Id comme index des lignes.  
rownames(testsetPUBG) <- testsetPUBG$Id  
head(testsetPUBG, 10)
```

```

##          Id      groupId      matchId assists boosts
## 9329eb41e215eb 9329eb41e215eb 676b23c24e70d6 45b576ab7daa7f      0      0
## 639bd0dc7bda8 639bd0dc7bda8 430933124148dd 42a9a0b906c928      0      4
## 63d5c8ef8dfe91 63d5c8ef8dfe91 0b45f5db20ba99 87e7e4477a048e      1      0
## cf5b81422591d1 cf5b81422591d1 b7497dbdc77f4a 1b9a94f1af67f1      0      0
## ee6a295187ba21 ee6a295187ba21 6604ce20a1d230 40754a93016066      0      4
## 3e2539b5d78183 3e2539b5d78183 029b5a79e08cd6 10186f5c852f62      0      0
## d812d2f1d88a02 d812d2f1d88a02 6285bb4eec83e4 f185809740a1a7      0      3
## a8a377e4d43bf8 a8a377e4d43bf8 61ec2e7730a3b8 d31adc82a4930e      0      0
## f18301e30d47d3 f18301e30d47d3 e7d609e08f09b6 bcde504ef16743      0      0
## ec7b965ef978b3 ec7b965ef978b3 c4c19ef6d6c6d9 6dc5b34c92d5ff      0      0
##          damageDealt DBNOs headshotKills heals killPlace killPoints kills
## 9329eb41e215eb 51.4600      0      0      0      73      0      0
## 639bd0dc7bda8 179.1000      0      0      2      11      0      2
## 63d5c8ef8dfe91 23.4000      0      0      4      49      0      0
## cf5b81422591d1 65.5200      0      0      0      54      0      0
## ee6a295187ba21 330.2000      1      2      1      7      0      3
## 3e2539b5d78183 0.0000      0      0      0      89      0      0
## d812d2f1d88a02 470.7000      3      2      17      3      0      5
## a8a377e4d43bf8 68.6100      0      0      0      73      0      0
## f18301e30d47d3 0.0000      0      0      0      56      0      0
## ec7b965ef978b3 67.3200      0      0      0      54      1023     0
##          killStreaks longestKill matchDuration matchType maxPlace
## 9329eb41e215eb 0      0.0000      1884 squad-fpp      28
## 639bd0dc7bda8 1      361.9000      1811 duo-fpp      48
## 63d5c8ef8dfe91 0      0.0000      1793 squad-fpp      28
## cf5b81422591d1 0      0.0000      1834 duo-fpp      45
## ee6a295187ba21 1      60.0600      1326 squad-fpp      28
## 3e2539b5d78183 0      0.0000      1775 squad-fpp      29
## d812d2f1d88a02 1      57.6100      1328 duo-fpp      49
## a8a377e4d43bf8 0      0.0000      1870 squad-fpp      29
## f18301e30d47d3 0      0.0000      1815 squad-fpp      28
## ec7b965ef978b3 0      0.0000      1336 squad-fpp      27
##          numGroups rankPoints revives rideDistance roadKills swimDistance
## 9329eb41e215eb 28      1500      0      0.0000      0      0.0000
## 639bd0dc7bda8 47      1503      2      4669.0000      0      0.0000
## 63d5c8ef8dfe91 27      1565      0      0.0000      0      0.0000
## cf5b81422591d1 44      1465      0      0.0000      0      0.0000
## ee6a295187ba21 27      1480      1      0.0000      0      0.0000
## 3e2539b5d78183 29      1490      0      0.0000      0      0.0000
## d812d2f1d88a02 48      1538      0      0.0000      0      0.0000
## a8a377e4d43bf8 27      1487      0      0.0000      0      0.0000
## f18301e30d47d3 27      1640      0      2355.0000      0      0.0000
## ec7b965ef978b3 27      -1      0      0.0000      0      0.0000
##          teamKills vehicleDestroys walkDistance weaponsAcquired winPoints
## 9329eb41e215eb 0      0      588.0000      1      0
## 639bd0dc7bda8 0      0      2017.0000      6      0
## 63d5c8ef8dfe91 0      0      787.8000      4      0
## cf5b81422591d1 0      0      1812.0000      3      0
## ee6a295187ba21 0      0      2963.0000      4      0
## 3e2539b5d78183 0      0      0.0000      0      0
## d812d2f1d88a02 0      0      1000.0000      4      0
## a8a377e4d43bf8 0      0      1217.0000      5      0
## f18301e30d47d3 0      0      1390.0000      7      0
## ec7b965ef978b3 0      0      1634.0000      5      1495

```

```
#Transformer les colonnes character en valeurs numériques
testsetPUBG$damageDealt <- as.numeric(testsetPUBG$damageDealt)
testsetPUBG$longestKill <- as.numeric(testsetPUBG$longestKill)
testsetPUBG$rideDistance <- as.numeric(testsetPUBG$rideDistance)
testsetPUBG$swimDistance <- as.numeric(testsetPUBG$swimDistance)
testsetPUBG$walkDistance <- as.numeric(testsetPUBG$walkDistance)
#Features Engineering
testsetPUBG$DistanceParcouru = testsetPUBG$rideDistance + testsetPUBG$swimDistance + testsetPUBG$walkDistance
testsetPUBG$healsBoosts = testsetPUBG$heals + testsetPUBG$boosts
testsetPUBG$HSratio = testsetPUBG$headshotKills / testsetPUBG$kills
testsetPUBG$GlobalPoints = testsetPUBG$winPoints + testsetPUBG$killPoints + testsetPUBG$rankPoints
testsetPUBG$KDA = testsetPUBG$kills + testsetPUBG$assists/3
#suppression des colonnes désormais inutile
testsetPUBG = testsetPUBG[, !(colnames(testsetPUBG) %in% c("rideDistance", "swimDistance", "walkDistance", "heals", "boosts", "Id", "groupId", "matchId", "kills", "assists", "winPoints", "killPoints", "rankPoints"))]
testsetPUBG[is.na(testsetPUBG)] <- 0
#Récupération des games en solo
vSolo <- c('flarefpp', 'flaretpp', 'crashfpp', 'crashtpp', 'normal-solo', 'normal-solo-fpp', 'solo-fpp')
testsetSolo <- filter(testsetPUBG, matchType %in% vSolo)
vDuo <- c('duo-fpp', 'normal-duo', 'normal-duo-fpp')
testsetDuo <- filter(testsetPUBG, matchType %in% vDuo)
vSquad <- c('normal-squad', 'normal-squad-fpp', 'squad-fpp')
testsetSquad <- filter(testsetPUBG, matchType %in% vSquad)
nrow(testsetSolo)
```

```
## [1] 239585
```

```
nrow(testsetDuo)
```

```
## [1] 443351
```

```
nrow(testsetSquad)
```

```
## [1] 756484
```

```
#Suppression des dernières colonnes inutile
testsetSolo = testsetSolo[, !(colnames(testsetSolo) %in% c("groupId", "DBNOs", "matchType", "numGroups", "revives", "teamKills"))]
```

On va lancer la prédiction de notre variable winPlacePerc dans le testsetSolo (pour les joueurs jouant en solo).

```
predictionSolo <- predict(solo, testsetSolo)
```

```
testsetSolo$winPlacePerc <- predictionSolo
summary(testsetSolo)
```

```
##   damageDealt      headshotKills      killPlace      killStreaks
## Min.   :  0.00   Min.   :0.0000   Min.   : 1.00   Min.   :0.0000
## 1st Qu.:  0.00   1st Qu.:0.0000   1st Qu.:24.00   1st Qu.:0.0000
## Median : 76.63   Median :0.0000   Median :48.00   Median :0.0000
## Mean   :118.77   Mean   :0.2479   Mean   :48.12   Mean   :0.4841
## 3rd Qu.:172.00   3rd Qu.:0.0000   3rd Qu.:72.00   3rd Qu.:1.0000
## Max.  :2193.00   Max.  :10.0000   Max.  :100.00  Max.  :7.0000
##   longestKill     matchDuration     maxPlace      roadkills
## Min.   :  0.00   Min.   : 74   Min.   :  8.00   Min.   :0.000000
## 1st Qu.:  0.00   1st Qu.:1379   1st Qu.: 94.00   1st Qu.:0.000000
## Median :  0.00   Median :1444   Median : 96.00   Median :0.000000
## Mean   :21.57   Mean   :1575   Mean   : 94.52   Mean   :0.007054
## 3rd Qu.:21.75   3rd Qu.:1862   3rd Qu.: 97.00   3rd Qu.:0.000000
## Max.  :888.10   Max.  :2194   Max.  :100.00  Max.  :7.000000
##   vehicleDestroys weaponsAcquired DistanceParcouru   healsBoosts
## Min.  :0.000000   Min.  : 0.000   Min.  :    0.00   Min.  : 0.000
## 1st Qu.:0.000000   1st Qu.: 2.000   1st Qu.: 98.45   1st Qu.: 0.000
## Median :0.000000   Median : 3.000   Median : 512.90   Median : 0.000
## Mean   :0.004262   Mean   : 3.462   Mean   :1373.04   Mean   : 2.076
## 3rd Qu.:0.000000   3rd Qu.: 5.000   3rd Qu.:2065.00   3rd Qu.: 3.000
## Max.  :5.000000   Max.  :41.000   Max.  :40799.92   Max.  :66.000
##   HSratio      GlobalPoints      KDA      winPlacePerc
## Min.  :0.0000   Min.  : 100   Min.  : 0.0000   Min.  :0.0000863
## 1st Qu.:0.0000   1st Qu.:1500   1st Qu.: 0.0000   1st Qu.:0.2435995
## Median :0.0000   Median :1562   Median : 0.0000   Median :0.4934918
## Mean   :0.1149   Mean   :1988   Mean   : 0.9537   Mean   :0.4912969
## 3rd Qu.:0.0000   3rd Qu.:2606   3rd Qu.: 1.0000   3rd Qu.:0.7582587
## Max.  :1.0000   Max.  :3730   Max.  :26.6667   Max.  :0.9929641
```

```
head(testsetSolo, 10)
```

```

##          damageDealt headshotKills killPlace killStreaks longestKill
## 37ae98f31ca542      0.00           0       53        0      0.00
## 494d3d9fad73b2      0.00           0       87        0      0.00
## e59dd1435a2ecf    100.00           0       32        1     25.23
## 3e4413d4780f7c    100.00           0       41        1     36.26
## 7fc13cb6ff72da      0.00           0       60        0      0.00
## b912e5782b1e48      0.00           0       74        0      0.00
## fc6d38774da7b2      0.00           0       95        0      0.00
## 2f70df5da78353   170.60           1       20        1     24.22
## 86efff43fcbbb3c    44.72           0       92        0      0.00
## 2e0304c79f71b4    69.09           0       49        0      0.00
##          matchDuration maxPlace roadkills vehicleDestroys weaponsAcquired
## 37ae98f31ca542      1380         97        0        0        0
## 494d3d9fad73b2      1854         98        0        0        2
## e59dd1435a2ecf     1413         96        0        0        5
## 3e4413d4780f7c     1389         95        0        0        1
## 7fc13cb6ff72da     1404         99        0        0        0
## b912e5782b1e48     1449         98        0        0        1
## fc6d38774da7b2     1980         95        0        0        1
## 2f70df5da78353     1408         96        0        0        4
## 86efff43fcbbb3c     1350         96        0        0        2
## 2e0304c79f71b4     1827         94        0        0        4
##          DistanceParcouru healsBoosts HSratio GlobalPoints KDA
## 37ae98f31ca542      4.894        0        0     2564        0
## 494d3d9fad73b2     82.110        0        0     1499        0
## e59dd1435a2ecf    571.700        1        0     1488        1
## 3e4413d4780f7c     28.490        0        0     1567        1
## 7fc13cb6ff72da     4.330        0        0     2652        0
## b912e5782b1e48    180.900        0        0     2545        0
## fc6d38774da7b2      0.000        0        0     1523        0
## 2f70df5da78353    4810.000        4        1     1584        1
## 86efff43fcbbb3c    37.470        0        0     1492        0
## 2e0304c79f71b4    2429.000        0        0     1497        0
##          winPlacePerc
## 37ae98f31ca542  0.554587317
## 494d3d9fad73b2  0.112032956
## e59dd1435a2ecf  0.575283083
## 3e4413d4780f7c  0.134274061
## 7fc13cb6ff72da  0.513757733
## b912e5782b1e48  0.293013739
## fc6d38774da7b2  0.003045453
## 2f70df5da78353  0.883703417
## 86efff43fcbbb3c 0.040688622
## 2e0304c79f71b4  0.695105500

```

Si on prend par exemple le joueur **2f70df5da78353** qui a de bonnes chances de bien finir dans sa partie avec un **winPlacePerc** de 0.882668153 on voit que sur les variables les plus importante il a de bonnes statistiques avec une **DistanceParcouru** de 4810, 4 **armes ramassés**, et une **killPlace** de 20 la ou pour le joueur **494d3d9fad73b2** ça sera plus compliqué avec une **DistanceParcouru** de 82.1100 et une **killPlace** de 87 qui nous indique sûrement que le joueur est mort tôt.

Sans être exceptionnelle notre modèle à l'air de se comporter correctement avec des résultats attendu.

Un bon moyen d'améliorer un peu les performances seraient d'utiliser plus de données.

Afin de pouvoir comparer l'impact de la quantité de données j'essairai si possible de mettre les résultat obtenu sur la totalité du dataset et des modes de jeu dans une parties à la fin.

REFERENCES

http://mehdikhaneboubi.free.fr/random_forest_r.html

(http://mehdikhaneboubi.free.fr/random_forest_r.html)

<https://www.rdocumentation.org/packages/randomForest/versions/4.6-14/topics/rfcv>

(<https://www.rdocumentation.org/packages/randomForest/versions/4.6-14/topics/rfcv>)

https://www.stat.berkeley.edu/%7Ebreiman/RandomForests/cc_home.htm#ooberr

(https://www.stat.berkeley.edu/%7Ebreiman/RandomForests/cc_home.htm#ooberr)