Analyse statistique de l'impacts de choix des héros et des lanes

Dahiez Ulysse , Burdy Simon

15/12/2020

Sommaire

Introduction

```
player_team_lol_data <- read.csv("player_team_lol_data.csv", header=TRUE)
champion <- read.csv("champion.csv", header=TRUE)
View(player_team_lol_data)


gameWin <- filter(player_team_lol_data, winTeam1 == " Win")
gameLoose <- filter(player_team_lol_data, winTeam1 == " Fail")
count <- table(unlist(gameWin$championID1))
perc <- 100*count/sum(count)
result <- data.frame(code = sprintf("%03d", as.integer(names(count))),
                     count = as.integer(count), perc = as.numeric(perc))
```

League of legends semble etre un jeu d'équipe de stratégie. Le nombre giganteste de héros et de classes peut peut-être déstabiliser les nouveaux joueurs et il est peut-être statégiquement intérressant de connaitre les meilleurs héros , meilleurs combinaisons de héros, les meilleurs emplacements ,etc...

Nous allons nous intéresser à la classe du personnage, ses caractéristiques et les différents emplacements sur les maps . Nous allons donc à l'aide des données de player_team_lol_data pouvoir determiner l'intérêt des différentes classes et leurs spécificitées ainsi que les différents placements .

Phénomènes

Phénomène numero 1 : Le héros le plus joué

Hypothèse: Quel est la problalité de gagner en jouant dans une équipe avec le héros le plus joué en général.

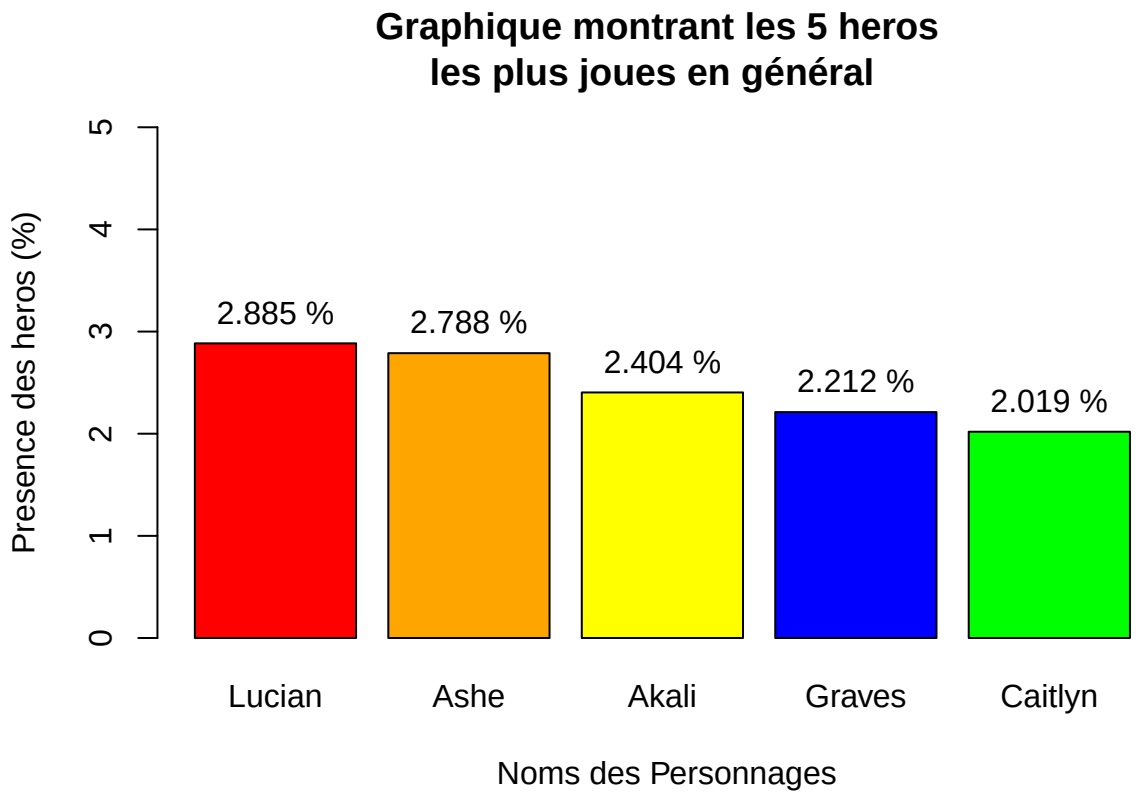
Nous allons d'abord rechercher les 5 héros les plus joués.

```
count <- table(unlist(player_team_lol_data$championID1))
perc <- 100*count/sum(count)
result <- data.frame(code = sprintf("%03d", as.integer(names(count))),
                     count = as.integer(count), perc = as.numeric(perc))



attach(result)
```

```
## The following objects are masked _by_ .GlobalEnv:
##
##     count, perc
```

```
premiers <- result[order(-perc),]

detach(result)

premier5 <-premiers[1:5,]



functionCorrespondanceKeyNom <-
  function(key){
    championNom <- filter(champion , data__.__key == key)
    return (championNom[,4])
  }


functionListeKeyNom <-
  function(listeKey){
    liste <- list(length(listeKey))
    for( i in 1:length(listeKey)){
      liste[[i]] <- functionCorrespondanceKeyNom(listeKey[i])
    }
    return(liste)

  }




bestHero <- barplot(premier5$perc , ylim=c(0,5) , names.arg = c(functionCorrespondanceKeyNom(236) ,
                                                                functionCorrespondanceKeyNom(022),func
                                                                functionCorrespondanceKeyNom(104), fun
```

```
                    xlab = " Noms des Personnages" , ylab="Presence des heros (%)",
                    main = "Graphique montrant les 5 heros \nles plus joues en général  ",
                    col = c("red","orange","yellow","blue","green"))


percValeurs<-as.matrix(premier5$perc)

text(bestHero,percValeurs+0.3,labels=as.character(paste(round(percValeurs,3),"%")) )
```



Les 5 héros les plus joués sont : Lucian , Ashe , Akali , Graves , Caitlyn . On remarque que les pourcentages sont assez peu éloignés.

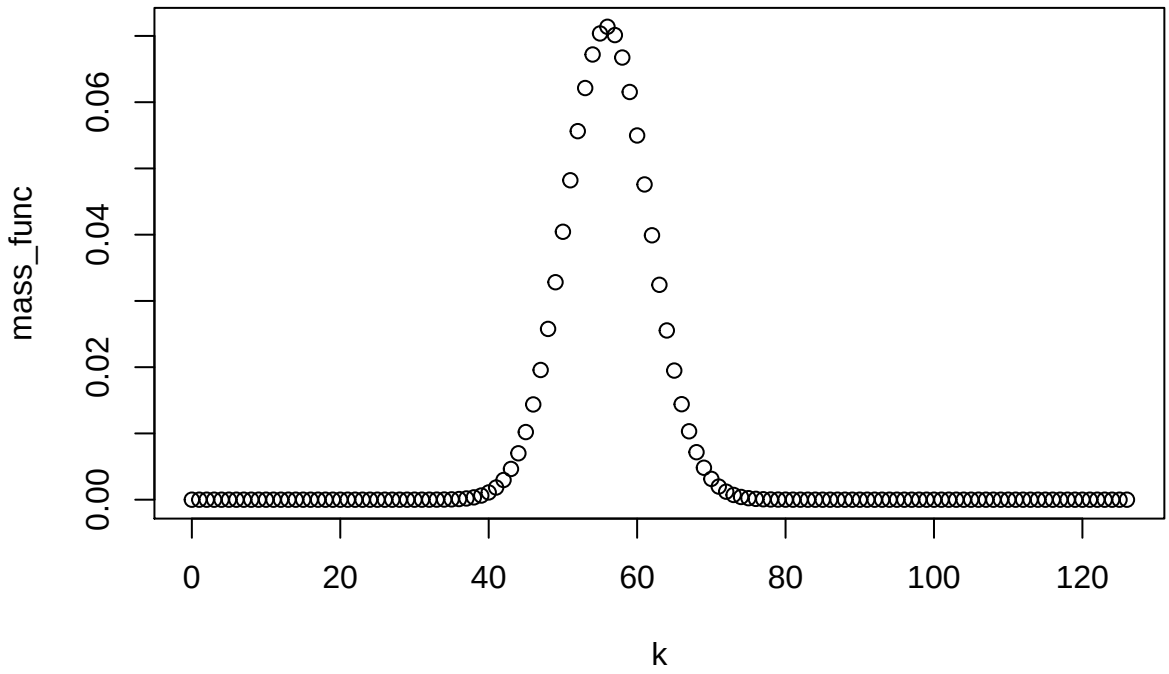
Nous allons maintenant chercher à savoir si Lucian est plus joué parcequ'il a plus d'atouts pour faire des kills.

On utilise la loi Binomiale pour calcuer la probalité de gagner en ayant ce hero dans l'équipe.

```
lucian <- filter(player_team_lol_data, championID1 == 236 |  championID2 == 236  |  championID3 == 236
lucianWin <- filter(lucian , winTeam1 == " Win")


n <- nrow(lucian)
p <- nrow(lucianWin)/nrow(lucian)
p
```

```
## [1] 0.4444444
```

```r
k <- 0:n
mass_func <- dbinom(k,n,p)
plot(k , mass_func)
```



La probabilité de gagner avec ce héro dans l'équipe est seulement de 0.44, cela nous montre donc que ce héro est très populaire mais il n'est pas forcément très bien joué par les joueurs de lol ou très fort. Plusieurs paramètres peuvent impacter cette statistique mais cela reste intéréssant.

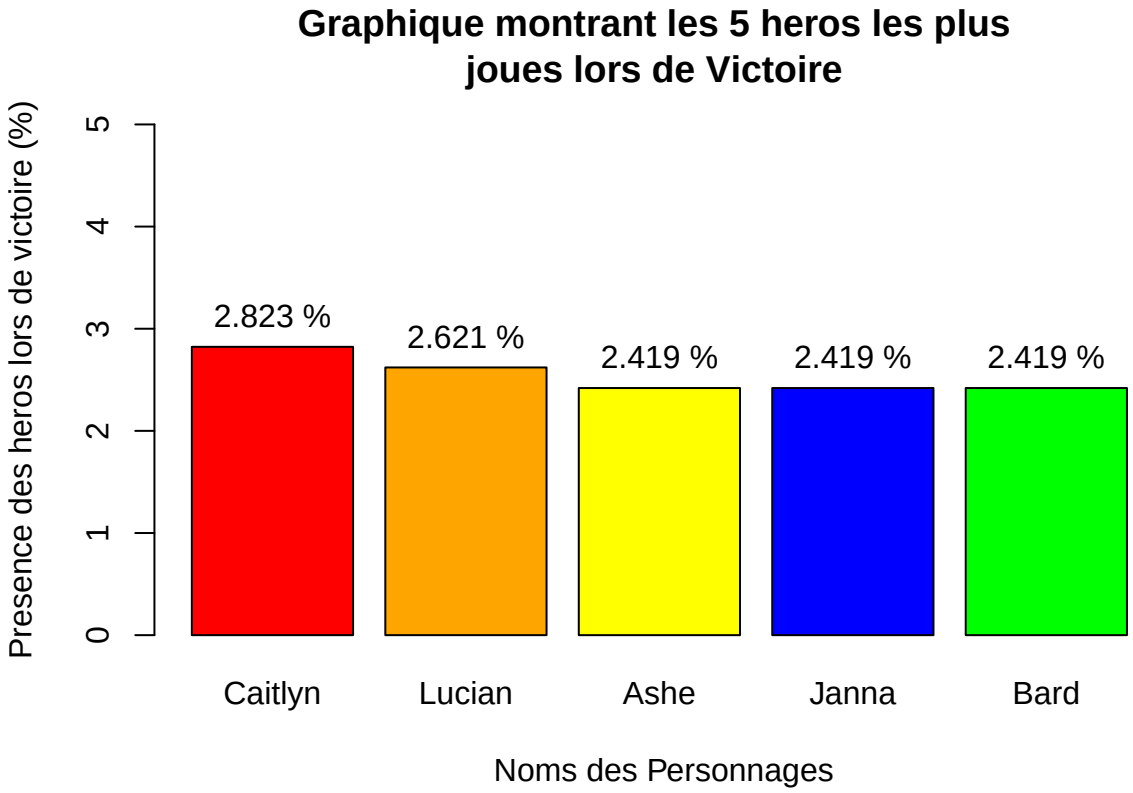
Pour comparer avec les reste des héros on va chercher le héro le plus joué lors de victoire.

```r
count <- table(unlist(gameWin$championID1))
perc <- 100*count/sum(count)
result <- data.frame(code = sprintf("%03d", as.integer(names(count))),
                     count = as.integer(count), perc = as.numeric(perc))



attach(result)
```

```
## The following objects are masked _by_ .GlobalEnv:
##
##     count, perc
```

```r
premiers <- result[order(-perc),]
```

```r
detach(result)

premier5 <-premiers[1:5,]



functionCorrespondanceKeyNom <-
  function(key){
    championNom <- filter(champion , data__.__key == key)
    return (championNom[,4])
  }


functionListeKeyNom <-
  function(listeKey){
    liste <- list(length(listeKey))
    for( i in 1:length(listeKey)){
      liste[[i]] <- functionCorrespondanceKeyNom(listeKey[i])
    }
    return(liste)

  }




bestHero <- barplot(premier5$perc , ylim=c(0,5) , names.arg = c(functionCorrespondanceKeyNom(051) ,
        functionCorrespondanceKeyNom(236),functionCorrespondanceKeyNom(022),
        functionCorrespondanceKeyNom(040), functionCorrespondanceKeyNom(432)),
        xlab = " Noms des Personnages" , ylab="Presence des heros lors de victoire (%)",
        main = "Graphique montrant les 5 heros les plus \n joues lors de Victoire  ",
        col = c("red","orange","yellow","blue","green"))


percValeurs<-as.matrix(premier5$perc)

text(bestHero,percValeurs+0.3,labels=as.character(paste(round(percValeurs,3),"%")) )
```

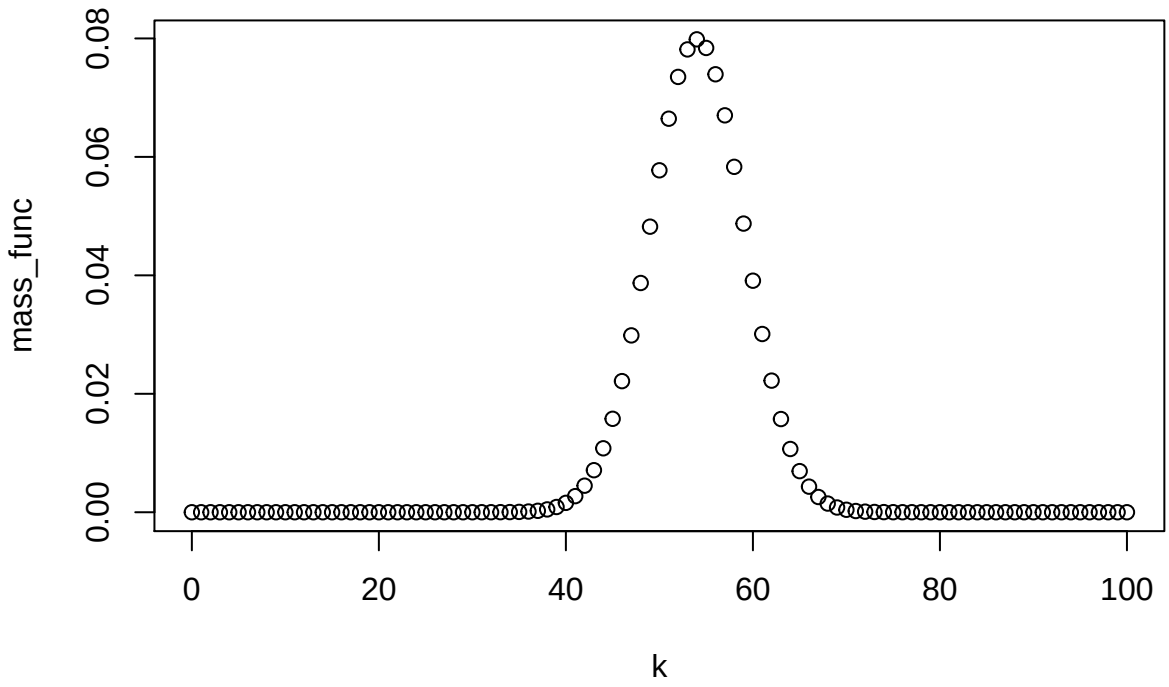


Caitlyn semble être le héro le plus joué lors de victoires , il fait lui aussi parti des héros les plus populaires.

```
caitlyn <- filter(player_team_lol_data, championID1 == 051 |  championID2 == 051  |  championID3 == 051
caitlynWin <- filter(caitlyn , winTeam1 == " Win")


n <- nrow(caitlyn)
p <- nrow(caitlynWin)/nrow(caitlyn)
p
```

```
## [1] 0.54
```

```
k <- 0:n
mass_func <- dbinom(k,n,p)
plot(k , mass_func)
```

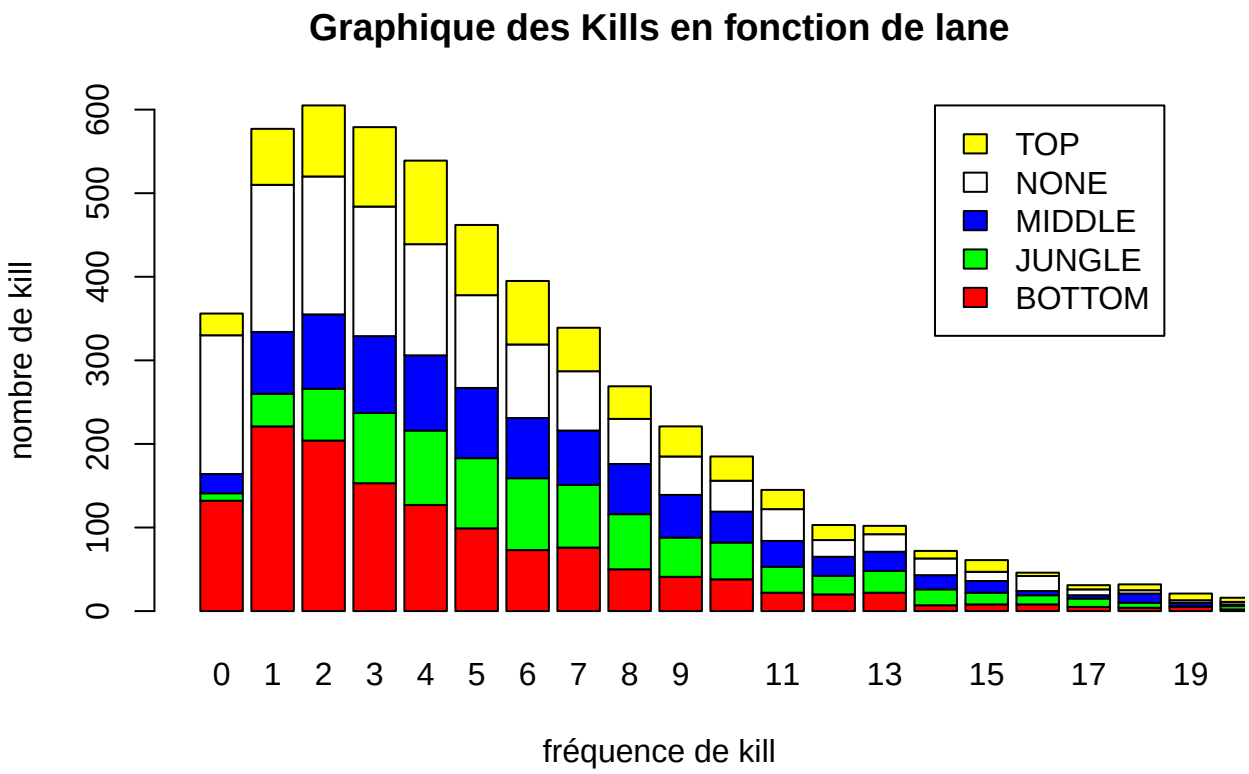


Une équipe contenant Caitlyn a une probabilité de 0.54 de gagner . Cette probabilité est bien supérieure à celle de Lucian. Les héros ont bien une influence sur la victoire .

Phenomène numero 2 : Lane et kills

Dans quelle lane faut-il faire le plus de kills pour gagner ? Hypothèse : C'est dans la lane Middle qu'il faut faire le plus de kills pour gagner.

```
barplot(table(( c(player_team_lol_data$lane1,
                  player_team_lol_data$lane2,
                  player_team_lol_data$lane3,
                  player_team_lol_data$lane4,
                  player_team_lol_data$lane5)),
                c(player_team_lol_data$kills1,
                 player_team_lol_data$kills2,
                  player_team_lol_data$kills3,
                  player_team_lol_data$kills4,
                  player_team_lol_data$kills5)
                ),
        col = c("red", "green", "blue", "white", "yellow"),
        legend.text = c('BOTTOM' , 'JUNGLE', 'MIDDLE', 'NONE', 'TOP'),
        args.legend = list(x = "topright"), xlim = c(0, 22),
      main = "Graphique des Kills en fonction de lane",
      xlab = "fréquence de kill", ylab = "nombre de kill")
```



Nous pouvons constater que dans la bottom, les kills sont plus nombreux. Mais faut-il se concentrer sur cette lane pour gagner une partie ?

Nous allons donc voir si plus le nombre de kills par lane est suppérieure à la moyenne, plus la chance de gagné est élevé.

Nous allons observer le nombre de kills par lane lors de victoires et comparer avec le nombre de kills moyens en espérant remarquer une lane plus significative.

Nous allons voir dans quelle lane :

```
bottom <- filter(player_team_lol_data, lane1 == " BOTTOM")
bottomWin <- filter(bottom, winTeam1 == " Win")
nbMoyKillBottom = mean(bottom$kills1)
nbMoyKillBottomWin = mean(bottomWin$kills1)
ecartTypeNbKillbottom <- sqrt(mean(bottom$kills1^2)-mean(bottom$kills1)^2)
pnorm(nbMoyKillBottomWin , nbMoyKillBottom, ecartTypeNbKillbottom)
```

```
## [1] 0.6080247
```

```
jungle <- filter(player_team_lol_data, lane1 == " JUNGLE")
jungleWin <- filter(jungle, winTeam1 == " Win")
nbMoyKilljungle = mean(jungle$kills1)
nbMoyKilljungleWin = mean(jungleWin$kills1)
ecartTypeNbKilljungle <- sqrt(mean(jungle$kills1^2)-mean(jungle$kills1)^2)
pnorm(nbMoyKilljungleWin , nbMoyKilljungle, ecartTypeNbKilljungle)
```

```
## [1] 0.6675635
```

```
middle <- filter(player_team_lol_data, lane1 == " MIDDLE")
middleWin <- filter(middle, winTeam1 == " Win")
nbMoyKillmiddle = mean(middle$kills1)
nbMoyKillmiddleWin = mean(middleWin$kills1)
ecartTypeNbKillmiddle <- sqrt(mean(middle$kills1^2)-mean(middle$kills1)^2)
pnorm(nbMoyKillmiddleWin , nbMoyKillmiddle, ecartTypeNbKillmiddle)
```

```
## [1] 0.6107388
```

```
top <- filter(player_team_lol_data, lane1 == " TOP")
topWin <- filter(top, winTeam1 == " Win")
nbMoyKilltop = mean(top$kills1)
nbMoyKilltopWin = mean(topWin$kills1)
ecartTypeNbKilltop <- sqrt(mean(top$kills1^2)-mean(top$kills1)^2)
pnorm(nbMoyKilltopWin , nbMoyKilltop, ecartTypeNbKilltop)
```

```
## [1] 0.6006214
```

Après ces quatre observations, nous pouvons constater que dans les parties gagnées, le nombre de kills dans la jungle par rapport à la moyenne est plus élevé que dans les autres lanes, Donc il est stratégiquement plus intéligent de bien observé les kills de la jungle.

Les Tests

Test hypotèse du chi2 :

Peut-on affirmer, au risque 1%, que le nombre de kills dépend de la classe ?

```
functionPourcentagePerso<-function(typeChampionKey , testChampionId){
    conteur <- 0
   for (i in 1:length(typeChampionKey)) {
    b <-typeChampionKey[i]
    for(j in 1:length(testChampionId)){
      c <- testChampionId[j]
      if(b == c ){
        conteur<- conteur + 1
      }
    }
   }
    return ((conteur/length(testChampionId))*100)
}



functionCountPerso<-function(typeChampionKey , testChampionId){
    conteur <- 0
   for (i in 1:length(typeChampionKey)) {
    b <-typeChampionKey[i]
    for(j in 1:length(testChampionId)){
      c <- testChampionId[j]
      if(b == c ){
        conteur<- conteur + 1
      }
    }
   }
    return (conteur)
}




gameDurationTrue <- filter (player_team_lol_data  , gameDuration > 1000 )

quantkills25 <- mean(quantile(gameDurationTrue$kills1 , prob = 0.25),
quantile(gameDurationTrue$kills2 , prob = 0.25),
quantile(gameDurationTrue$kills3 , prob = 0.25),
quantile(gameDurationTrue$kills4 , prob = 0.25),
quantile(gameDurationTrue$kills5 , prob = 0.25))

quantkills50 <- mean(quantile(gameDurationTrue$kills1 , prob = 0.55),
quantile(gameDurationTrue$kills2 , prob = 0.50),
quantile(gameDurationTrue$kills3 , prob = 0.50),
quantile(gameDurationTrue$kills4 , prob = 0.50),
quantile(gameDurationTrue$kills5 , prob = 0.50))
```

```
quantkills75 <- mean(quantile(gameDurationTrue$kills1 , prob = 0.75),
quantile(gameDurationTrue$kills2 , prob = 0.75),
quantile(gameDurationTrue$kills3 , prob = 0.75),
quantile(gameDurationTrue$kills4 , prob = 0.75),
quantile(gameDurationTrue$kills5 , prob = 0.75))



quantAssits25 <- quantile(gameDurationTrue$assists1 , prob = 0.25) #4
quantAssits50 <- quantile(gameDurationTrue$assists1 , prob = 0.50) #7
quantAssits75 <- quantile(gameDurationTrue$assists1 , prob = 0.75) # 12


championSupport <- filter(champion ,data__.__tags__001 == "Support" )
championAssassin <- filter(champion ,data__.__tags__001 == "Assassin")
championFighter <- filter(champion ,data__.__tags__001 == "Fighter" )
championMage <- filter(champion ,data__.__tags__001 == "Mage"  )
championTank <- filter(champion ,data__.__tags__001 == "Tank" )
championMarksman <- filter(champion ,data__.__tags__001 == "Marksman"  )




lowkill1 <- filter(gameDurationTrue ,   kills1 <= quantkills25 , assists1 >= quantAssits25 )
lowkill2 <- filter(gameDurationTrue ,  kills2 <= quantkills25 , assists2 >= quantAssits25 )
lowkill3 <- filter(gameDurationTrue  , kills3 <= quantkills25 , assists3 >= quantAssits25 )
lowkill4 <- filter(gameDurationTrue , kills4 <= quantkills25 , assists4 >= quantAssits25 )
lowkill5 <- filter(gameDurationTrue  , kills5 <= quantkills25 , assists5 >= quantAssits25 )


supportWin <-sum(functionCountPerso(championSupport$data__.__key,lowkill1$championID1),
                 functionCountPerso(championSupport$data__.__key,lowkill2$championID2),
                 functionCountPerso(championSupport$data__.__key,lowkill3$championID3),
                 functionCountPerso(championSupport$data__.__key,lowkill4$championID4),
                 functionCountPerso(championSupport$data__.__key,lowkill5$championID5))

assassinWin <- sum(functionCountPerso(championAssassin$data__.__key,lowkill1$championID1),
functionCountPerso(championAssassin$data__.__key,lowkill2$championID2),
functionCountPerso(championAssassin$data__.__key,lowkill3$championID3),
functionCountPerso(championAssassin$data__.__key,lowkill4$championID4),
functionCountPerso(championAssassin$data__.__key,lowkill5$championID5))


fighterWin <- sum(functionCountPerso(championFighter$data__.__key,lowkill1$championID1),
                  functionCountPerso(championFighter$data__.__key,lowkill2$championID2),
                   functionCountPerso(championFighter$data__.__key,lowkill3$championID3),
                   functionCountPerso(championFighter$data__.__key,lowkill4$championID4),
                   functionCountPerso(championFighter$data__.__key,lowkill5$championID5))


mageWin <- sum(functionCountPerso(championMage$data__.__key,lowkill1$championID1),
                  functionCountPerso(championMage$data__.__key,lowkill2$championID2),
                  functionCountPerso(championMage$data__.__key,lowkill3$championID3),
```

```
                    functionCountPerso(championMage$data__.__key,lowkill4$championID4),
                    functionCountPerso(championMage$data__.__key,lowkill5$championID5))


tankWin <- sum(functionCountPerso(championTank$data__.__key,lowkill1$championID1),
       functionCountPerso(championTank$data__.__key,lowkill2$championID2),
       functionCountPerso(championTank$data__.__key,lowkill3$championID3),
       functionCountPerso(championTank$data__.__key,lowkill4$championID4),
      functionCountPerso(championTank$data__.__key,lowkill5$championID5))

marksmanWin <- sum(functionCountPerso(championMarksman$data__.__key,lowkill1$championID1),
                   functionCountPerso(championMarksman$data__.__key,lowkill2$championID2),
                   functionCountPerso(championMarksman$data__.__key,lowkill3$championID3),
                   functionCountPerso(championMarksman$data__.__key,lowkill4$championID4),
                   functionCountPerso(championMarksman$data__.__key,lowkill5$championID5))









lowkillsup1 <- filter(gameDurationTrue ,   kills1 >= quantkills25 , assists1 <= quantAssits25 )
lowkillsup2 <- filter(gameDurationTrue ,  kills2 >= quantkills25 , assists2 <= quantAssits25 )
lowkillsup3 <- filter(gameDurationTrue  , kills3 >= quantkills25 , assists3 <= quantAssits25 )
lowkillsup4 <- filter(gameDurationTrue , kills4 >= quantkills25 , assists4 <= quantAssits25 )
lowkillsup5 <- filter(gameDurationTrue  , kills5 >= quantkills25 , assists5 <= quantAssits25 )









supportWinsup <- sum(functionCountPerso(championSupport$data__.__key,lowkillsup1$championID1),
                     functionCountPerso(championSupport$data__.__key,lowkillsup2$championID2),
                     functionCountPerso(championSupport$data__.__key,lowkillsup3$championID3),
                     functionCountPerso(championSupport$data__.__key,lowkillsup4$championID4),
                     functionCountPerso(championSupport$data__.__key,lowkillsup5$championID5))

assassinWinsup <- sum(functionCountPerso(championAssassin$data__.__key,lowkillsup1$championID1),
functionCountPerso(championAssassin$data__.__key,lowkillsup2$championID2),
functionCountPerso(championAssassin$data__.__key,lowkillsup3$championID3),
functionCountPerso(championAssassin$data__.__key,lowkillsup4$championID4),
functionCountPerso(championAssassin$data__.__key,lowkillsup5$championID5))


fighterWinsup <- sum(functionCountPerso(championFighter$data__.__key,lowkillsup1$championID1),
                     functionCountPerso(championFighter$data__.__key,lowkillsup2$championID2),
                     functionCountPerso(championFighter$data__.__key,lowkillsup3$championID3),
                     functionCountPerso(championFighter$data__.__key,lowkillsup4$championID4),
                     functionCountPerso(championFighter$data__.__key,lowkillsup5$championID5))


mageWinsup <- sum(functionCountPerso(championMage$data__.__key,lowkillsup1$championID1),
```

```
                    functionCountPerso(championMage$data__.__key,lowkillsup2$championID2),
                    functionCountPerso(championMage$data__.__key,lowkillsup3$championID3),
                    functionCountPerso(championMage$data__.__key,lowkillsup4$championID4),
                    functionCountPerso(championMage$data__.__key,lowkillsup5$championID5))


tankWinsup <- sum(functionCountPerso(championTank$data__.__key,lowkillsup1$championID1),
      functionCountPerso(championTank$data__.__key,lowkillsup2$championID2),
       functionCountPerso(championTank$data__.__key,lowkillsup3$championID3),
       functionCountPerso(championTank$data__.__key,lowkillsup4$championID4),
       functionCountPerso(championTank$data__.__key,lowkillsup5$championID5))

marksmanWinsup <- sum(functionCountPerso(championMarksman$data__.__key,lowkillsup1$championID1),
                  functionCountPerso(championMarksman$data__.__key,lowkillsup2$championID2),
                  functionCountPerso(championMarksman$data__.__key,lowkillsup3$championID3),
                  functionCountPerso(championMarksman$data__.__key,lowkillsup4$championID4),
                  functionCountPerso(championMarksman$data__.__key,lowkillsup5$championID5))
```

Soient X le caractère qualitatif “kills et assists” et Y le caractère qualitatif “Classes”. Les modalités de X sont “Inférieure à 1 kill et plus de 4 assists” et “Supérieure à 1 kill et moins de 4 assists”, et les modalités de Y sont “support”,“assassin”,“fighter”,“mage”,“tank” et “marksman” (on a k = 2 et h = 6). Par l’énoncé, on observe la valeur de (X, Y) pour chacun des n individus (kills et assists) d’un échantillon avec n = 180. On considère les hypothèses : H0 : “les caractères X et Y sont indépendants” contre H1 : “les caractères X et Y ne sont pas indépendants”.

```
A = matrix(c(supportWin  ,
             assassinWin ,
             fighterWin ,
             mageWin ,
             tankWin ,
             marksmanWin ,
             supportWinsup  ,
             assassinWinsup ,
             fighterWinsup ,
             mageWinsup ,
             tankWinsup ,
             marksmanWinsup
         ), nrow = 2, byrow = T)
A
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]  312   33  107  135  152  120
## [2,]   18  191  299  133   39  266
```

```
chisq.test(A)$p.value
```

```
## [1] 7.707578e-124
```

Comme la p-valeur est < 0.05 , on rejette H0 . Les données nous permettent de rejetter l’indépendence de X et Y .

Test de nullité :

Peut-on affirmer qu'il y a une liaison significative entre le nombre de kills et les assists ?

Soient X le caractère quantitatif "kills" et Y le caractère quantitatif "assists". Par l'énoncé, on observe la valeur de (X, Y) pour chacun des n individus (parties) d'un échantillon avec n = 1040. On considère les hypothèses : H0 : "les caractères X et Y sont indépendants" contre H1 : "les caractères X et Y ne sont pas indépendants". On peut utiliser le test de nullité du coefficient de corrélation. Dans un premier temps, on considère les

```
kill1 <- c(player_team_lol_data$kills1)
assists1 <- c(player_team_lol_data$assists1)
cor.test(kill1 ,assists1)$p.value
```

```
## [1] 6.524986e-11
```

p-valeur = 6.524986e-11

On peut admètre l'hypothèse de normalité sur la loi de (X, Y). Comme p-valeur n'est pas compris entre]0.01, 0.05], on ne rejette pas l'hypothèse H0 . Ainsi, on peut affirmer qu'il y a pas de liaisons significative entre le nombre de kills et le nombre d'assists.

Test de conformité :

```
gameDurationTrue <-filter(player_team_lol_data  , gameDuration > 1000  , winTeam1 == "   Win")


moyenne <- c(gameDurationTrue$kills1,gameDurationTrue$kills2,gameDurationTrue$kills3,gameDurationTrue$k

mean(moyenne)
```

```
## [1] NaN
```

Observe que généralement la moyenne des kills toutes classes confondu lors d'une partie gagnante se trouve aux alentours de 5,49 et le troisième quartile se trouve aux alentours de 7 .

On va observer si jouer une classe a une influence sur le nombre kills . On tire 30 résultats de chaques nombres de kills pour la classe Fighter :

```
functionCorrespondanceKeyClasse <-
  function(key){
    championClasse <- filter(champion , data__.__key == key)
    return (championClasse[,22])
  }




res = c(15,  4 , 2,  5, 10 , 7 , 1 , 8 , 4,  3, 11 , 5 , 3,  4 , 5 , 4  ,4 , 8 , 4 ,13 ,11  ,6  ,2  ,5
mean(res)
```

```
## [1] 6.466667
```

Nous avons une liste de 30 valeurs de kills efféctué par des membres de la classe fighter lors de victoire .

Calcul de l'écart type et variance :

```
res = c(15,  4 , 2,  5, 10 , 7 , 1 , 8 , 4,  3, 11 , 5 , 3,  4 , 5 , 4  ,4 , 8 , 4 ,13 ,11  ,6  ,2  ,5
mean(res)
```

```
## [1] 6.466667
```

```
median(res)
```

```
## [1] 5
```

```
sd(res)
```

```
## [1] 3.794127
```

```
var(res)
```

```
## [1] 14.3954
```

On suppose que le nombre de kills d'une classe peut être modélisée par une var X N(µ, (1.947)^2), avec µ inconnu. Peut-on affirmer, avec un faible risque de se tromper, que les fighters font plus de kills que la moyenne?

Par l'énoncé, on observe la valeur de X N(µ, 2) pour chacun des n individus (kills) d'un échantillon avec n = 24, µ inconnu et = 1.947. On veut affirmer, avec un faible risque de se tromper, que le fournisseur ne respecte pas ses engagements. Cela est le cas si l'épaisseur moyenne de ses composants est différente de 7.3 millimètres, soit µ != 5.49.

Par conséquent, l'hypothèse H1 est : H1 : µ >= 5.49. On considère alors les hypothèses : H0 : µ <= 5.49 contre H1 : µ > 5.49. Comme est connu, on utilise un Z-Test. Il est bilatéral. On considère les commandes :

```
library(OneTwoSamples)
```

```
## Warning: package 'OneTwoSamples' was built under R version 4.0.3
```

```
mean_test1(res,5.49, 1.947 , side=1)$p_value
```

```
## [1] 0.003002382
```

p-valeur = 0.003002382

p-valeurn'est pas entre < 0.05 , on rejette H0 .

Test d' homogéinité

Le joueur ne joue pas de la même manière selon la lane

Dans l'énoncé, on observe :
• la valeur de BOTTOM) pour chacun des n individus d'un échantillon avec n = 261, µ inconnu et = 4.031703,
• la valeur de TOP) pour chacun des n individus d'un échantillon avec n = 186, µ inconnu et = 4.445639
• la valeur de MIDDLE) pour chacun des n individus d'un échantillon avec n = 167, µ inconnu et = 4.445639
• la valeur de JUNGLE) pour chacun des n individus d'un échantillon avec n = 153, µ inconnu et = 4.003492

Les échantillons sont indépendants car les individus considérés sont tous différents. On veut affirmer, avec un faible risque de se tromper, que les parties sont joué de manière différente selon la lane. Cela est le cas si le nombre de kill par lane est different, soit. Par conséquent, l'hypothèse H1 est : H1 : µ1 =6 µ2. On considère alors les hypothèses : H0 : µ1 = µ2 contre H1 : µ1 != µ2. Comme 1, 2 sont connus, on utilise un 2-Comp-Z-Test. Il est bilatéral.

```
laneBottom = filter(player_team_lol_data, lane1 == " BOTTOM")
laneTop =  filter(player_team_lol_data, lane1 == " TOP")
laneJungle <- filter(player_team_lol_data, lane1 == " JUNGLE")
laneMiddle <- filter(player_team_lol_data, lane1 == " MIDDLE")

mean_test2(laneBottom$kills1,
            laneTop$kills1,

           sigma = c(sd(laneBottom$kills1),
                     sd(laneTop$kills1)
          ))$p_value
```

```
## [1] 0.0002760607
```

```
mean_test2(
           laneBottom$kills1,
           laneJungle$kills1,
           sigma = c(sd(laneBottom$kills1),
                     sd(laneJungle$kills1)

          ))$p_value
```

```
## [1] 1.495909e-08
```

```
mean_test2(
           laneMiddle$kills1,
           laneBottom$kills1,
           sigma = c(sd(laneMiddle$kills1),
                     sd(laneBottom$kills1)
          ))$p_value
```

```
## [1] 3.86212e-07
```

```r
mean_test2(
           laneTop$kills1,
           laneJungle$kills1,
           sigma = c(sd(laneTop$kills1),
                     sd(laneJungle$kills1)

          ))$p_value
```

```
## [1] 0.07408161
```

```r
mean_test2(
           laneTop$kills1,
           laneMiddle$kills1,
           sigma = c(sd(laneTop$kills1),
                     sd(laneMiddle$kills1)

          ))$p_value
```

```
## [1] 0.1920777
```

```r
mean_test2(
           laneMiddle$kills1,
           laneJungle$kills1,
           sigma = c(sd(laneMiddle$kills1),
                     sd(laneJungle$kills1)

          ))$p_value
```

```
## [1] 0.6402834
```

p-valeur > 0.05, on ne rejette pas H0. . Ces tests nous permettent d'assurer l'homogénéité des parties entre les lanes ; TOP - JUNGLE, TOP - MIDDLE, MIDDLE - JUNGLE

Et la non-homogénéité entre les lanes ; BOTTOM - TOP, BOTTOM - JUNGLE, BOTTOM - MIDDLE

Nous pouvons donc en conclure que les parties dans la lane BOTTOM sont jouées différement par le joueur 1.

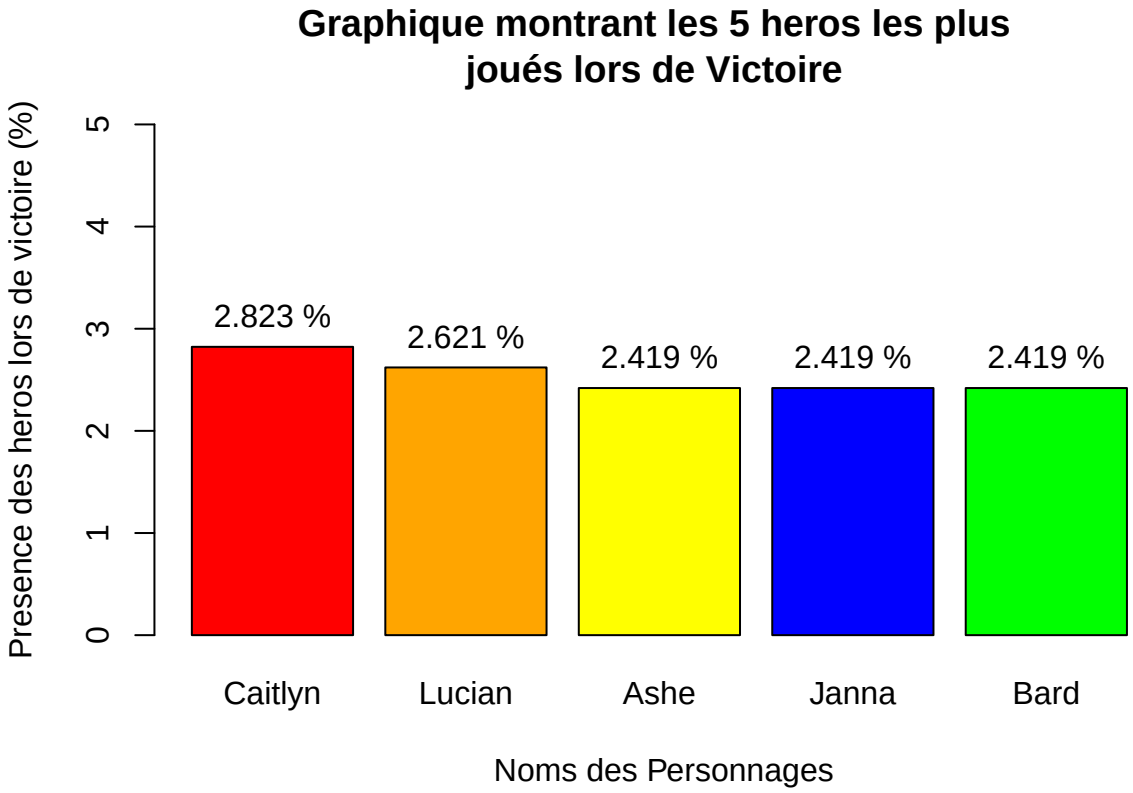
Heros

Hero le plus joué

```
attach(result)
```

```
## The following objects are masked _by_ .GlobalEnv:
##
##     count, perc
```

```
premiers <- result[order(-perc),]

detach(result)

premier5 <-premiers[1:5,]

functionCorrespondanceKeyNom <-
  function(key){
    championNom <- filter(champion , data__.__key == key)
    return (championNom[,4])
  }


functionListeKeyNom <-
  function(listeKey){
    liste <- list(length(listeKey))
    for( i in 1:length(listeKey)){
      liste[[i]] <- functionCorrespondanceKeyNom(listeKey[i])
    }
    return(liste)

  }
bestHero <- barplot(premier5$perc , ylim=c(0,5) , names.arg = c(functionCorrespondanceKeyNom(051) ,
        functionCorrespondanceKeyNom(236),functionCorrespondanceKeyNom(022),
        functionCorrespondanceKeyNom(040), functionCorrespondanceKeyNom(432)),
        xlab = " Noms des Personnages" , ylab="Presence des heros lors de victoire (%)",
        main = "Graphique montrant les 5 heros les plus \n joués lors de Victoire  ",
        col = c("red","orange","yellow","blue","green"))


percValeurs<-as.matrix(premier5$perc)

text(bestHero,percValeurs+0.3,labels=as.character(paste(round(percValeurs,3),"%")) )
```



Dans cette première visualisation nous pouvons constater que le héro le plus joué lors de victoire est Caitlyn.

Composition lors de Victoire :

Regardons mainenant si stratégiquement il est plus malin de placer certaines classes à cetains endroits.

```
bottomWin <- filter(gameWin , gameDuration > 300 , lane1 == " BOTTOM")
TopWin <- filter(gameWin , gameDuration > 300 , lane1 == " TOP")
jungleWin<- filter(gameWin , gameDuration > 300 , lane1 == " JUNGLE")
middleWin <- filter(gameWin , gameDuration > 300 , lane1 == " MIDDLE")

##############Win################

topWin <- filter(gameWin , gameDuration > 300 , lane1 == " TOP")



supportWinTop <- functionPourcentagePerso(championSupport$data__.__key,topWin$championID1) # 4.39
assassinWinTop<- functionPourcentagePerso(championAssassin$data__.__key,topWin$championID1) # 7.62
fighterWinTop <- functionPourcentagePerso(championFighter$data__.__key,topWin$championID1)# 53.85
mageWinTop <- functionPourcentagePerso(championMage$data__.__key,topWin$championID1) # 12.08
tankWinTop<- functionPourcentagePerso(championTank$data__.__key,topWin$championID1) # 15.38
marksmanWinTop <- functionPourcentagePerso(championMarksman$data__.__key,topWin$championID1) # 6.59

supportWinTop
```

En Top

```
## [1] 4.395604
```

```
assassinWinTop
```

```
## [1] 7.692308
```

```
fighterWinTop
```

```
## [1] 53.84615
```

```
tankWinTop
```

```
## [1] 12.08791
```

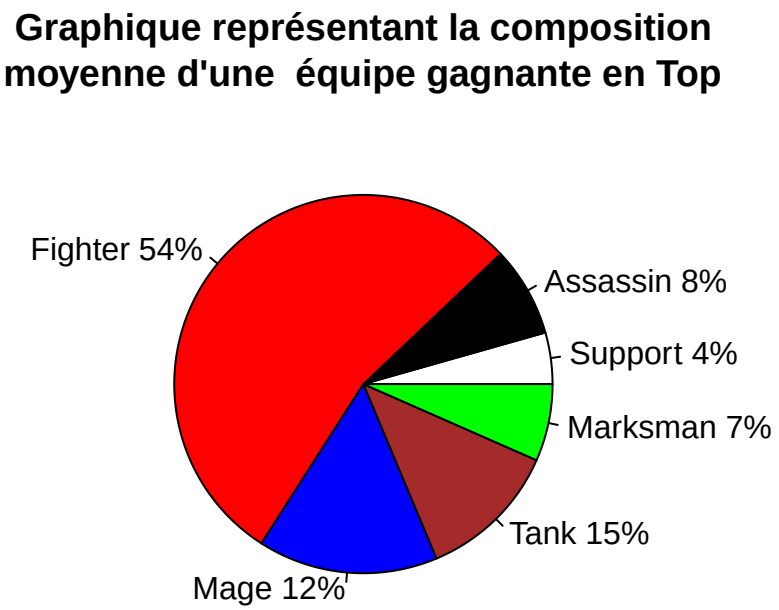
```
mageWinTop
```

```
## [1] 15.38462
```

```
marksmanWinTop
```

```
## [1] 6.593407
```

```
slices <-c(4.39,7.62,53.85,12.08,15.38,6.59)
lbls <- c("Support","Assassin","Fighter","Mage","Tank","Marksman")
pct <- pct <- round(slices/sum(slices)*100)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels

championsNames = c("support","assassin","fighter","mage","tank","marksman")
pie(c(supportWinTop,assassinWinTop,fighterWinTop,mageWinTop,tankWinTop,marksmanWinTop) , labels = lbls,
    ,main="Graphique représentant la composition\n moyenne d'une  équipe gagnante en Top ")
```



En top on observe une explosion de Fighter (54%). Les fighter doivent certainement avoir une facilitée de jeu dans cette lane.

```
bottomWin <- filter(gameWin , gameDuration > 300 , lane1 == " BOTTOM")
TopWin <- filter(gameWin , gameDuration > 300 , lane1 == " TOP")
jungleWin<- filter(gameWin , gameDuration > 300 , lane1 == " JUNGLE")
middleWin <- filter(gameWin , gameDuration > 300 , lane1 == " MIDDLE")

##############Win################

jungleWin <- filter(gameWin , gameDuration > 300 , lane1 == " TOP")
```

```
supportWinMiddle <- functionPourcentagePerso(championSupport$data__.__key,middleWin$championID1) # 4.39
assassinWinMiddle <- functionPourcentagePerso(championAssassin$data__.__key,middleWin$championID1) # 7.
fighterWinMiddle  <- functionPourcentagePerso(championFighter$data__.__key,middleWin$championID1)# 53.8
mageWinMiddle  <- functionPourcentagePerso(championMage$data__.__key,middleWin$championID1) # 12.08
tankWinMiddle <- functionPourcentagePerso(championTank$data__.__key,middleWin$championID1) # 15.38
marksmanWinMiddle  <- functionPourcentagePerso(championMarksman$data__.__key,middleWin$championID1) # 6

supportWinMiddle
```

En Middle

```
## [1] 12.34568
```

```
assassinWinMiddle
```

```
## [1] 28.39506
```

```
fighterWinMiddle
```

```
## [1] 12.34568
```

```
tankWinMiddle
```

```
## [1] 4.938272
```

```
mageWinMiddle
```

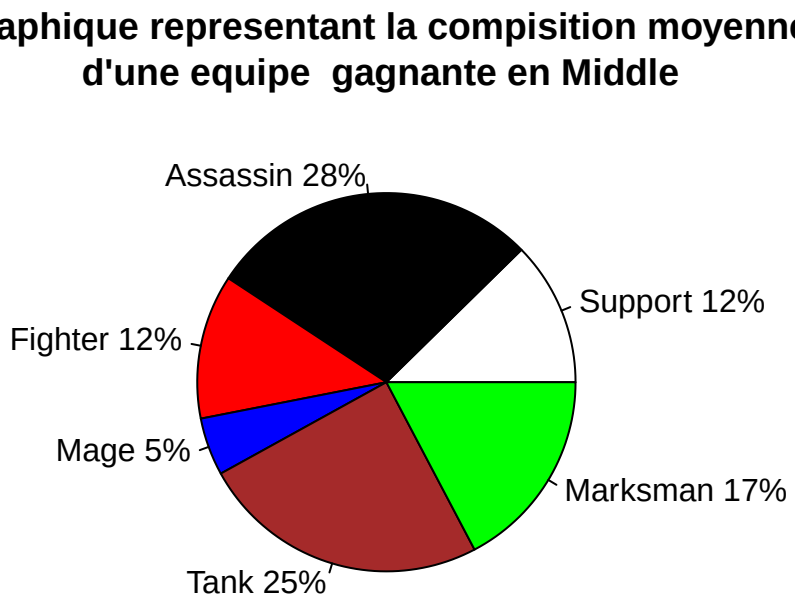
```
## [1] 24.69136
```

```
marksmanWinMiddle
```

```
## [1] 17.28395
```

```
slices <-c(
  supportWinMiddle,
  assassinWinMiddle,
  fighterWinMiddle,
  tankWinMiddle,
  mageWinMiddle,
  marksmanWinMiddle)
lbls <- c("Support","Assassin","Fighter","Mage","Tank","Marksman")
pct <- pct <- round(slices/sum(slices)*100)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels

championsNames = c("support","assassin","fighter","mage","tank","marksman")
pie(c(
  supportWinMiddle,
  assassinWinMiddle,
```

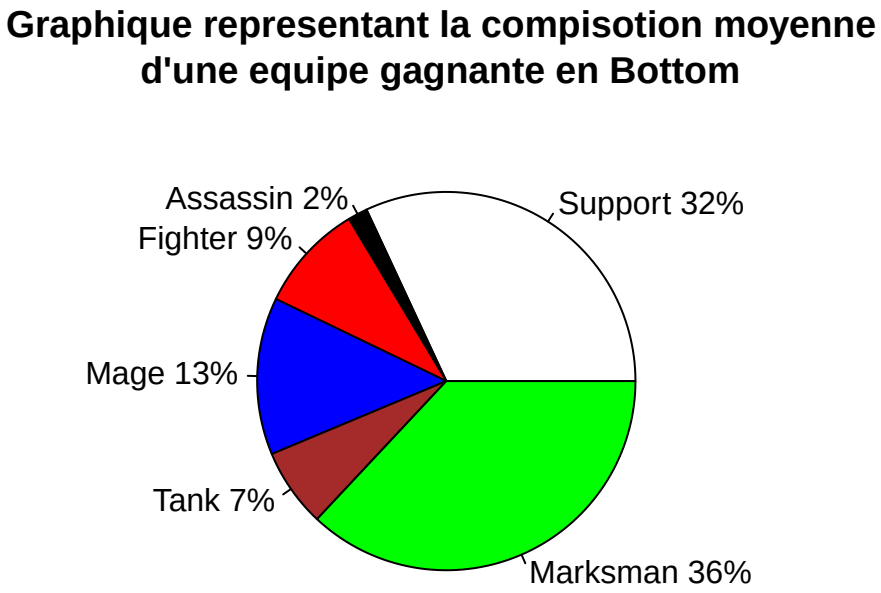
```
  fighterWinMiddle,
  tankWinMiddle,
  mageWinMiddle,
  marksmanWinMiddle) , labels = lbls,  col =c("white","black","red","blue","brown","green")
    ,main="Graphique representant la compisition moyenne \nd'une equipe  gagnante en Middle ")
```



En middle on observe un graphique plutôt equilibré. Avec une légére absence de mage dans cette lane. La lane middle semble être praticable par toutes les classes de héros.

```
bottomWin <- filter(gameWin , gameDuration > 300 , lane1 == " BOTTOM")


supportWinBottom <- functionPourcentagePerso(championSupport$data__.__key,bottomWin$championID1) # 31.9
assassinWinBottom <- functionPourcentagePerso(championAssassin$data__.__key,bottomWin$championID1) # 1.
fighterWinBottom <- functionPourcentagePerso(championFighter$data__.__key,bottomWin$championID1)# 9.24
mageWinBottom <- functionPourcentagePerso(championMage$data__.__key,bottomWin$championID1) # 13.44
tankWinBottom<- functionPourcentagePerso(championTank$data__.__key,bottomWin$championID1) # 6.72
marksmanWinBottom <- functionPourcentagePerso(championMarksman$data__.__key,bottomWin$championID1) # 36
```

```r
slices <-c(31.93,1.68,9.24,13.24,6.72,36)
lbls <- c("Support","Assassin","Fighter","Mage","Tank","Marksman")
pct <- pct <- round(slices/sum(slices)*100)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels

championsNames = c("support","assassin","fighter","mage","tank","marksman")
pie(c(supportWinBottom,assassinWinBottom,fighterWinBottom,mageWinBottom,tankWinBottom,marksmanWinBottom
    ,main="Graphique representant la compisotion moyenne \nd'une equipe gagnante en Bottom ")
```



En Bottom

En bottom on observe un graphique contenant majoritairement des Supports(32 %) et des Marksman (36%). Cette lane doit donc subir de nombreux dégats pour que les Supports soient aussi présents, ou l'alliance Marskmam et Support est peut-être très efficasse.

```r
bottomWin <- filter(gameWin , gameDuration > 300 , lane1 == " BOTTOM")
TopWin <- filter(gameWin , gameDuration > 300 , lane1 == " TOP")
jungleWin<- filter(gameWin , gameDuration > 300 , lane1 == " JUNGLE")
middleWin <- filter(gameWin , gameDuration > 300 , lane1 == " MIDDLE")

##############Win################
jungleWin <- filter(gameWin , gameDuration > 300 , lane1 == " TOP")
```

```
supportWinJungle <- functionPourcentagePerso(championSupport$data__.__key,jungleWin$championID1) # 4.39
assassinWinJungle<- functionPourcentagePerso(championAssassin$data__.__key,jungleWin$championID1) # 7.6
fighterWinJungle <- functionPourcentagePerso(championFighter$data__.__key,jungleWin$championID1)# 53.85
mageWinJungle <- functionPourcentagePerso(championMage$data__.__key,jungleWin$championID1) # 12.08
tankWinJungle<- functionPourcentagePerso(championTank$data__.__key,jungleWin$championID1) # 15.38
marksmanWinJungle <- functionPourcentagePerso(championMarksman$data__.__key,jungleWin$championID1) # 6.

supportWinJungle
```

En Jungle

```
## [1] 4.395604
```

```
assassinWinJungle
```

```
## [1] 7.692308
```

```
fighterWinJungle
```

```
## [1] 53.84615
```

```
tankWinJungle
```

```
## [1] 12.08791
```

```
mageWinJungle
```

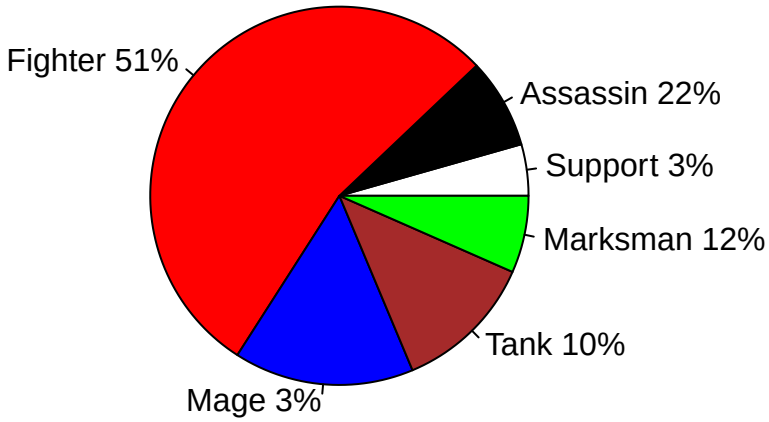
```
## [1] 15.38462
```

```
marksmanWinJungle
```

```
## [1] 6.593407
```

```
slices <-c(2.89,21.73,50.72,2.89,10.14,11.59)
lbls <- c("Support","Assassin","Fighter","Mage","Tank","Marksman")
pct <- pct <- round(slices/sum(slices)*100)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels

championsNames = c("support","assassin","fighter","mage","tank","marksman")
pie(c(supportWinJungle,assassinWinJungle,fighterWinJungle,mageWinJungle,tankWinJungle,marksmanWinJungle
    ,main="Graphique representant la compisition moyenne\n d'une equipe
    gagnante en Jungle ")
```

Graphique representant la compisition moyenne d'une equipe gagnante en Jungle



En top on observe une explosion de Fighter (51%) . Les fighter doivent certainement avoir une facilitée de jeu dans cette lane ou il est peut-être important de bien controler cette lane avec des personnages offensifs .

Conclusion général :

Grace à nos different tests d'hypothèses ,analyses des données et des lois de probabilités on peut conclure que la classe, le héro et la position à laquelle joue un joueur influent sur certains résultats .
Par exemple nous avons remarqué que le fait de jouer un Support limit gégéralement le nombre de kills mais augmente le nombre d'assits. Alors que jouer Fighter permet de faire plus de kills que la moyenne .
On a aussi remarqué que tout les personages ,même au sein d'une même classe , ne sont pas égaux . Caitlyn reste une héros statistiquement plus fort que les autres. On a également pus observer que la lane Bottom est une lane de forte activité, beaucoup de kills y sont fait .
Le fait de reussir à battre l'équipe adverse est lié à un grand nombre de facteurs tels que la compositions de l'équipe, la position, les lvls, etc… mais aussi d'autres facteurs sont à prendre en compte tel que la communication des joueurs etre eux , leurs niveaux personnels , etc…

Implication personnelle

Dahiez Ulysse

Nous avons travaillé sur les données de League of Legende (LOL), même si nous avons tout les deux travaillé sur l'intégralité du projet, ma part d'implication s'est plus porté sur la partie du placement du joueur et des kills efféctué. il etait agréable de travailler sur RStudio.

La chose que j'aurais aimée, c'est de choisir le sujet, je ne connaissais ce jeu que de nom, mais je ne m'y étais jamais trop intéressé, je pense avoir passé tout de même assez de temps à comprendre ce qu'étaient les différentes colonnes. Les problèmes que j'ai rencontrés lors du développement de ce rapport sont plus théorique que pratique, j'arrivais par exemple très bien à lancer les tests, mais j'avais plus de mal comprendre quel test fallait-il utiliser.

Burdy Simon

Nous avons bien partagé le projet avec Ulysse , nous avons fait la moitié des Phénomènes et la moitié des tests d'hypothèses chacuns. J'ai plus travaillé sur la partie concerant les classes et les héros .

Connaissant assez peu le jeu, il fut parfois compliqué de determiner quelles données étaient vraiment utiles pour mes calculs .Le fait d'avoir travaillé sur 2 tables en même temps nous a fait écrire quelques fonctions.

J'ai vraiment aimé travailler sur ce projet car nous avons reussi à bien nous investir dans la compréhension du gamplay de ce jeu et de ses actuces. Nous avons reussi nos calculs basés sur les aspects les plus attirants du jeu (les personnages et les deplacements) et j'ai vraiment aimé y participer.