Détecter des faux billets

CME

I- Contexte

Vous êtes consultant Data Analyst dans une entreprise spécialisée dans la data. Votre entreprise a décroché une prestation en régie au sein de l'**Organisation nationale de lutte contre le faux-monnayage (ONCFM)**.

Cette institution a pour objectif de mettre en place des méthodes d'identification des contrefaçons des billets en euros. Ils font donc appel à vous, spécialiste de la data, pour mettre en place une modélisation qui serait capable d'identifier automatiquement les vrais des faux billets. Et ce à partir simplement de certaines dimensions du billet et des éléments qui le composent.

Voici le cahier des charges de l'ONCFM ainsi que le jeu de données

Le client souhaite que vous travailliez directement depuis ses locaux sous la responsabilité de Marie, responsable du projet d'analyse de données à l'ONCFM. Elle vous laissera une grande autonomie pendant votre mission, et vous demande simplement que vous lui présentiez vos résultats une fois la mission terminée. Elle souhaite voir quels sont les traitements et analyses que vous avez réalisés en amont, les différentes pistes explorées pour la construction de l'algorithme, ainsi que le modèle final retenu.

Après avoir lu en détail le cahier des charges, vous vous préparez à vous rendre à l'ONCFM pour prendre vos nouvelles fonctions. Vous notez tout de même un post-it qui se trouve sur le coin de votre bureau, laissé par un de vos collègues :

II- Importation des fichiers

```
options(repos = c(CRAN = "https://cran.rstudio.com/"))
```

```
data <- read.csv("data_raw/billets.csv", sep =";")
```

III- Résumé des datas

```
summary(data)
```

```
 is_genuine           diagonal      height_left     height_right
Length:1500        Min.   :171.0   Min.   :103.1   Min.   :102.8
Class :character   1st Qu.:171.8   1st Qu.:103.8   1st Qu.:103.7
Mode  :character   Median :172.0   Median :104.0   Median :103.9
                   Mean   :172.0   Mean   :104.0   Mean   :103.9
                   3rd Qu.:172.2   3rd Qu.:104.2   3rd Qu.:104.2
                   Max.   :173.0   Max.   :104.9   Max.   :105.0

  margin_low      margin_up         length
Min.   :2.980   Min.   :2.270   Min.   :109.5
1st Qu.:4.015   1st Qu.:2.990   1st Qu.:112.0
Median :4.310   Median :3.140   Median :113.0
Mean   :4.486   Mean   :3.151   Mean   :112.7
3rd Qu.:4.870   3rd Qu.:3.310   3rd Qu.:113.3
Max.   :6.900   Max.   :3.910   Max.   :114.4
NA's   :37
```

Nous avons un dataframe de 7 colonnes et 1 500 lignes 1 colonne de type character 6 colonnes numériques

IV- Description des variables

```
if (!require(skimr)) install.packages("skimr")
```

```
Le chargement a nécessité le package : skimr
```

```
library(skimr)

skim(data)
```

Table 1: Data summary

Name	data
Number of rows	1500
Number of columns	7
_____	
Column type frequency:	
character	1
numeric	6
_____	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
is_genuine	0	1	4	5	0	2	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
diagonal	0	1.00	171.96	0.31	171.04	171.75	171.96	172.17	173.01	
height_left	0	1.00	104.03	0.30	103.14	103.82	104.04	104.23	104.88	
height_right	0	1.00	103.92	0.33	102.82	103.71	103.92	104.15	104.95	
margin_low	37	0.98	4.49	0.66	2.98	4.02	4.31	4.87	6.90	
margin_up	0	1.00	3.15	0.23	2.27	2.99	3.14	3.31	3.91	
length	0	1.00	112.68	0.87	109.49	112.03	112.96	113.34	114.44	

Nous avons 37 valeurs manquantes dans la colonne margin_low

```r
# Afficher les lignes où 'margin_low' est NA
missing_margin_low <- data[is.na(data$margin_low), ]
print(missing_margin_low)
```

```
     is_genuine diagonal height_left height_right margin_low margin_up length
73         True   171.94      103.89       103.45         NA      3.25 112.79
100        True   171.93      104.07       104.18         NA      3.14 113.08
152        True   172.07      103.80       104.38         NA      3.02 112.93
198        True   171.45      103.66       103.80         NA      3.62 113.27
242        True   171.83      104.14       104.06         NA      3.02 112.36
252        True   171.80      103.26       102.82         NA      2.95 113.22
285        True   171.92      103.83       103.76         NA      3.23 113.29
335        True   171.85      103.70       103.96         NA      3.00 113.36
411        True   172.56      103.72       103.51         NA      3.12 112.95
414        True   172.30      103.66       103.50         NA      3.16 112.95
446        True   172.34      104.42       103.22         NA      3.01 112.97
482        True   171.81      103.53       103.96         NA      2.71 113.99
506        True   172.01      103.97       104.05         NA      2.98 113.65
612        True   171.80      103.68       103.49         NA      3.30 112.84
655        True   171.97      103.69       103.54         NA      2.70 112.79
676        True   171.60      103.85       103.91         NA      2.56 113.27
711        True   172.03      103.97       103.86         NA      3.07 112.65
740        True   172.07      103.74       103.76         NA      3.09 112.41
743        True   172.14      104.06       103.96         NA      3.24 113.07
781        True   172.41      103.95       103.79         NA      3.13 113.41
799        True   171.96      103.84       103.62         NA      3.01 114.44
845        True   171.62      104.14       104.49         NA      2.99 113.35
846        True   172.02      104.21       104.05         NA      2.90 113.62
872        True   171.37      104.07       103.75         NA      3.07 113.27
896        True   171.81      103.68       103.80         NA      2.98 113.82
920        True   171.92      103.68       103.45         NA      2.58 113.68
946        True   172.09      103.74       103.52         NA      3.02 112.78
947        True   171.63      103.87       104.66         NA      3.27 112.68
982        True   172.02      104.23       103.72         NA      2.99 113.37
1077      False   171.57      104.27       104.44         NA      3.21 111.87
1122      False   171.40      104.38       104.19         NA      3.17 112.39
1177      False   171.59      104.05       103.94         NA      3.02 111.29
1304      False   172.17      104.49       103.76         NA      2.93 111.21
1316      False   172.08      104.15       104.17         NA      3.40 112.29
1348      False   171.72      104.46       104.12         NA      3.61 110.31
1436      False   172.66      104.33       104.41         NA      3.56 111.47
1439      False   171.90      104.28       104.29         NA      3.24 111.49
```

```r
valeur_unique <- unique(data$is_genuine)
print(valeur_unique)
```

```
[1] "True"  "False"
```

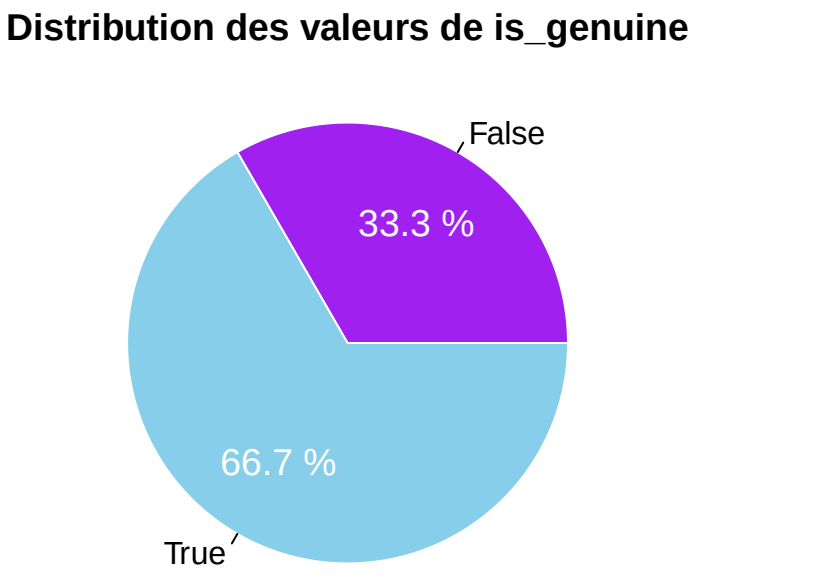
Nous avons 2 valeurs uniques dans la colonne is_genuine => True ou False

```r
valeur_compte <- table(data$is_genuine)
print(valeur_compte)
```

```
False  True
  500  1000
```

Il y a **500 valeurs False** et **1 000 valeurs True**

```r
# Calcul des pourcentages
pourcentage <- round(valeur_compte / sum(valeur_compte) * 100, 1)
pourcentage_labels <- paste(pourcentage, "%")

# Définir les marges du graphique (gauche, droite, bas, haut)
par(mar = c(1, 1, 1, 1))

# Définir le rapport d'aspect pour le graphique circulaire
par(pty = "s")

# Création du pie chart avec les labels (sans les valeurs)
pie(valeur_compte,
    labels = names(valeur_compte),
    main = "Distribution des valeurs de is_genuine",
    col = c("purple", "skyblue"),
    border = "white")

# Calcul des positions des étiquettes
positions <- cumsum(valeur_compte) - valeur_compte / 2
text(x = cos(2 * pi * positions / sum(valeur_compte)) * 0.5,
     y = sin(2 * pi * positions / sum(valeur_compte)) * 0.5,
     labels = pourcentage_labels,
     cex = 1.2, col = "white")
```



En résumé

Nous avons un tableau regroupant les données de 1 500 billets

1 colonne décrivant s'il s'agit de vrais ou faux billets :
- il y a 1 000 vrais billets 66.7% et 500 faux billets 33.3%

6 colonne décrivants le format de ces billets :
- diagonale
- hauteur gauche
- hauteur droite
- marge basse
- marge haute
- longueur

37 billets n'ont pas l'information de la marge basse dans le tableau.

Nous allons aggréger les données sur la colonnne is_genuine
Afficher la valeur moyenne de chaque variable pour les lignes False et True afin de voir si nous pouvons observer des différences significatives sur une ou plusieurs variables.

```r
if (!require(dplyr)) install.packages("dplyr")
```

```
Le chargement a nécessité le package : dplyr
```

```
Attachement du package : 'dplyr'
```

```
Les objets suivants sont masqués depuis 'package:stats':

    filter, lag
```

```
Les objets suivants sont masqués depuis 'package:base':

    intersect, setdiff, setequal, union
```

```r
library(dplyr)
```

```r
resultats <- data %>%
  group_by(is_genuine) %>%
  summarise(across(where(is.numeric), \(x) mean(x, na.rm = TRUE)))

print(resultats)
```

```
# A tibble: 2 x 7
  is_genuine diagonal height_left height_right margin_low margin_up length
  <chr>         <dbl>       <dbl>        <dbl>      <dbl>     <dbl>  <dbl>
1 False          172.        104.         104.       5.22      3.35   112.
2 True           172.        104.         104.       4.12      3.05   113.
```

Il y a bien quelques différences de mesures entre les vrais et faux billets mais il est difficile d'utiliser ces moyennes pour notre objectif final qui sera de déterminer d'après les mesures de billets s'il s'agit de "vrais" ou de "faux".

V- Remlacement des NaN de la colonne “margin_low”

Pour commencer il est nécessaire de traiter les valeurs manquantes de notre jeu de données.

Nous avons la possibilité de retirer ces 37 lignes ou bien de déterminer les valeurs manquantes de la colonne “margin_low” en fonction des autres variables présentes à l’aide d’une régréssion linéaire.

Regréssion linéaire multiple : Nous utilisons toutes les variables pour commencer et nous allons voir si elles sont toutes significatives pour la détermination de “margin_low”. Nous fixons notre seuil de test à 5% Toutes les p-valeur inférieure à 0.05 seront donc considérée comme significatives. Si toutes les variables ne sont pas significatives, nous utiliserons la méthode backward et retirerons une à une toutes les variables non significatives.

```
# Supprimer les lignes où 'margin_low' est NaN
data_without_na <- data[!is.na(data$margin_low), ]
```

```
reg_multi <- lm(margin_low~diagonal+height_left+height_right+margin_up+length, data=data_wit
summary(reg_multi)
```

```
Call:
lm(formula = margin_low ~ diagonal + height_left + height_right +
    margin_up + length, data = data_without_na)

Residuals:
     Min       1Q   Median       3Q      Max
-1.47234 -0.31707 -0.04168  0.27353  1.97084

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  22.99484    9.65555   2.382  0.01737 *
diagonal     -0.11106    0.04144  -2.680  0.00744 **
height_left   0.18412    0.04477   4.113 4.13e-05 ***
height_right  0.25714    0.04301   5.978 2.84e-09 ***
margin_up     0.25619    0.06437   3.980 7.23e-05 ***
length       -0.40910    0.01808 -22.627  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4807 on 1457 degrees of freedom
Multiple R-squared:  0.4773,    Adjusted R-squared:  0.4755
F-statistic: 266.1 on 5 and 1457 DF,  p-value: < 2.2e-16
```

Les variables sont toutes significatives car leur p-value est inférieure à 5% (0.05) le niveau de test que nous souhaitons. Nous conservons donc toutes les variables pour la détermination de 'margin_low'

Nous pouvons tester le modèle sous forme de prédiction en renseignant toutes les variables pour déterminer une valeur à margin_low

```
prev <- data.frame(diagonal=169, height_left=105, height_right=105, margin_up=3, length=115)
marg_low_prev <- predict(reg_multi, prev)
round(marg_low_prev,digits=2)
```

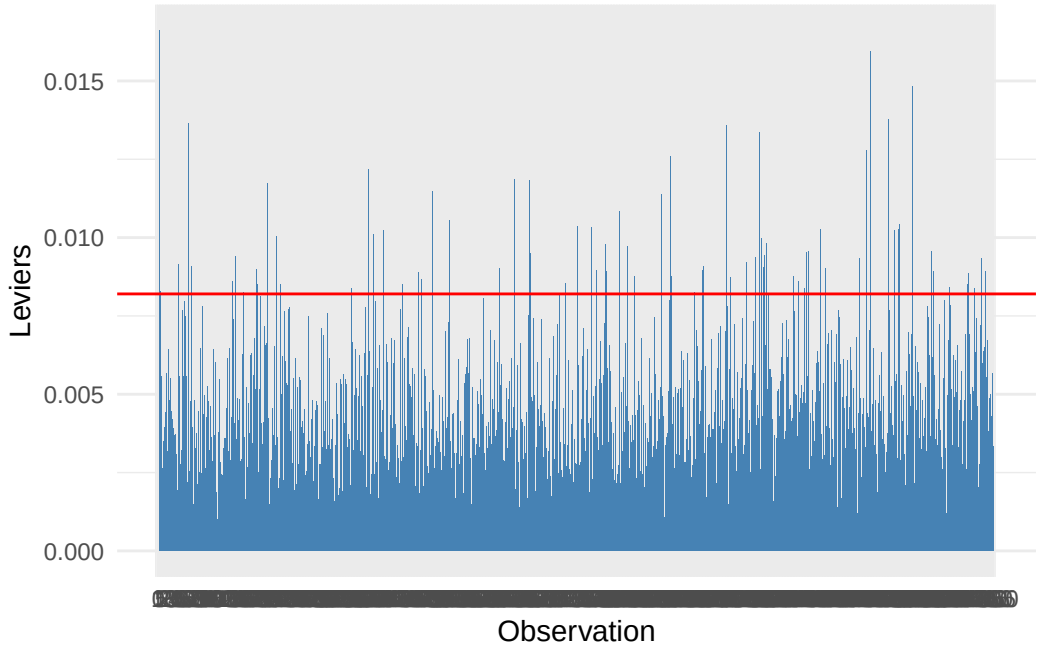
```
   1
4.28
```

```
#niveau de test
alpha <- 0.05
#nombre d'individus
n <- dim(data_without_na)[1]
#nombre de variables
p <- 6
```

```
#analyse sur les valeurs atypique ou influantes
analyses <- data.frame(obs=1:n)
```

```
#calcul des leviers
analyses$levier <- hat(model.matrix(reg_multi))
seuil_levier <- 2*p/n
```

```
library(ggplot2)
ggplot(data=analyses, aes(x=obs,y=levier))+
  geom_bar(stat='identity', fill="steelblue")+
  geom_hline(yintercept=seuil_levier, col="red")+
  theme_minimal()+
  xlab("Observation")+
  ylab("Leviers")+
  scale_x_continuous(breaks=seq(0,n,by=5))
```




```
#pour récupérer les numéros des observations pour lesquels les leviers sont supérieurs au seu
idl <- analyses$levier>seuil_levier
head(idl)
```

```
[1]  TRUE FALSE  TRUE FALSE FALSE FALSE
```

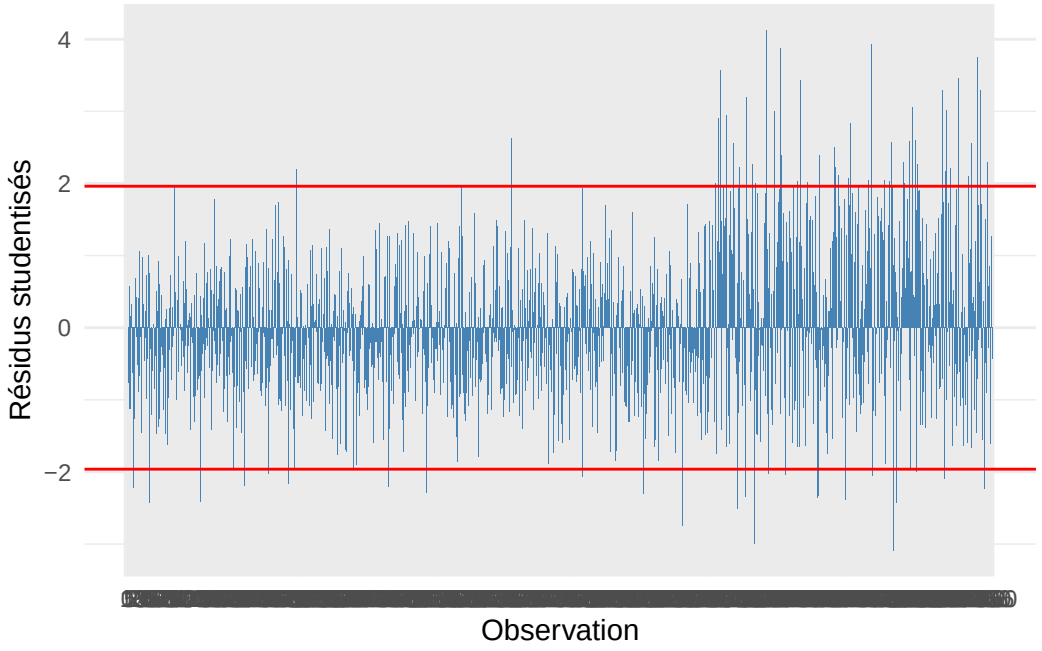
```
analyses$levier[idl]
```

```
 [1] 0.016606433 0.008297956 0.009156405 0.013635471 0.009087284 0.008615883
 [7] 0.008222240 0.009397300 0.008239495 0.008980313 0.008506194 0.011723023
[13] 0.010037681 0.008513933 0.008384592 0.012165484 0.010106900 0.010243133
[19] 0.008503534 0.008900042 0.008652842 0.011478525 0.010559686 0.009003567
[25] 0.011867888 0.011824475 0.009480505 0.008553764 0.010368443 0.010316756
[31] 0.008959938 0.009778237 0.009661468 0.008907537 0.010832538 0.009722524
[37] 0.008768476 0.011377525 0.012585764 0.008767460 0.008250039 0.008969260
[43] 0.009082513 0.013582547 0.008740641 0.009200472 0.009380825 0.013372012
[49] 0.009973714 0.009054657 0.009427569 0.009814530 0.008751936 0.008610965
[55] 0.008380906 0.009526687 0.009563757 0.010251487 0.009009819 0.009338419
[61] 0.012772324 0.015927673 0.013778658 0.010241817 0.010269781 0.010408617
[67] 0.014823503 0.009557381 0.008913496 0.008397866 0.008521435 0.008870294
[73] 0.008390446 0.009324645 0.008931108
```

Calculer les résidus studentisés

```
analyses$rstudent <- rstudent(reg_multi)
seuil_rstudent <- qt(1-alpha/2,n-p-1)
```

```
ggplot(data=analyses, aes(x=obs,y=rstudent))+
  geom_bar(stat='identity', fill="steelblue")+
  geom_hline(yintercept=-seuil_rstudent, col="red")+
  geom_hline(yintercept=seuil_rstudent, col="red")+
  theme_minimal()+
  xlab("Observation")+
  ylab("Résidus studentisés")+
  scale_x_continuous(breaks=seq(0,n,by=5))
```



Pour récupérer la distance de Cook

```
influence<-influence.measures(reg_multi)
names(influence)
```

```
[1] "infmat" "is.inf" "call"
```

```r
colnames(influence$infmat)
```

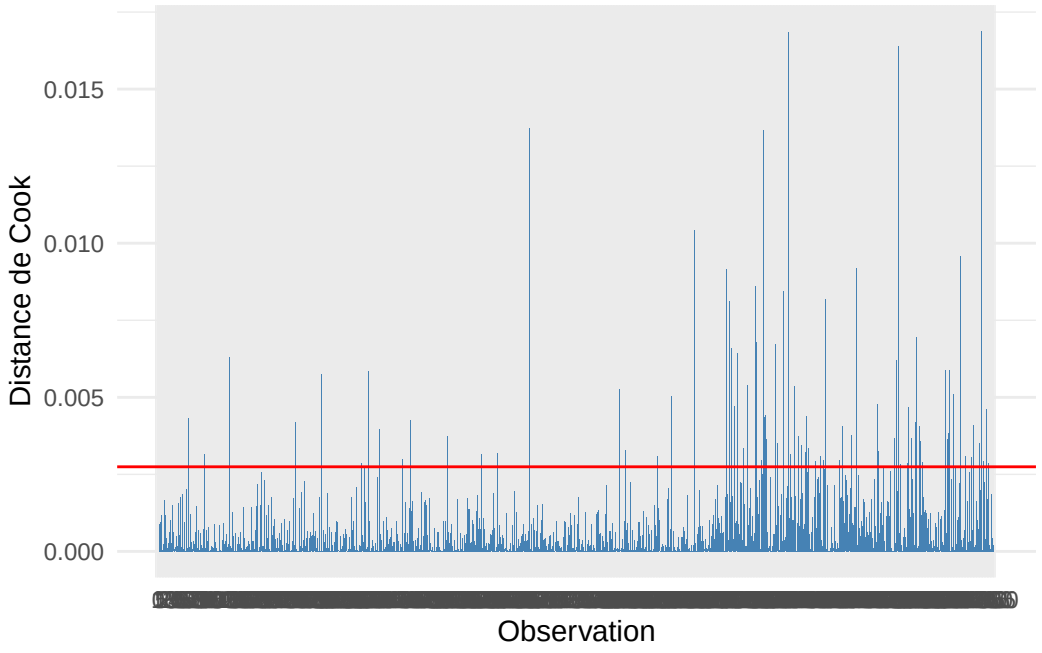
```
[1] "dfb.1_"     "dfb.dgnl"   "dfb.hght_l" "dfb.hght_r" "dfb.mrg_"
[6] "dfb.lngt"   "dffit"      "cov.r"      "cook.d"     "hat"
```

```r
analyses$dcook <- influence$infmat[,"cook.d"]
seuil_dcook <- 4/(n-p)
```

```r
seuil_dcook
```

```
[1] 0.002745367
```

```r
ggplot(data=analyses, aes(x=obs,y=dcook))+
  geom_bar(stat='identity', fill="steelblue")+
  geom_hline(yintercept=seuil_dcook, col="red")+
  theme_minimal()+
  xlab("Observation")+
  ylab("Distance de Cook")+
  scale_x_continuous(breaks=seq(0,n,by=5))
```



Recherche de colinéarité (commande VIF)

```r
# Installation du package
install.packages("car")
```

```
Installation du package dans 'C:/Users/Utilisateur/AppData/Local/R/win-library/4.4'
(car 'lib' n'est pas spécifié)
```

```
le package 'car' a été décompressé et les sommes MD5 ont été vérifiées avec succés
```

```
Les packages binaires téléchargés sont dans
    C:\Users\Utilisateur\AppData\Local\Temp\RtmpENZ3ol\downloaded_packages
```

```r
# Charger le package
library(car)
```

```
Le chargement a nécessité le package : carData
```

```
Attachement du package : 'car'
```

```
L'objet suivant est masqué depuis 'package:dplyr':
```

```
    recode
```

```r
vif(reg_multi)
```

```
    diagonal  height_left height_right    margin_up       length
    1.013613     1.138261     1.230115     1.404404     1.576950
```

Tous les VIF étant inférieur à 10, il ne semble pas y avoir de problème de colinéarité

On peut également tester l'homoscédasticité, c'est à dire la constance de la variance des erreurs
On peut donc réaliser le test de Breusch-Pagan

```r
install.packages("lmtest")
```

```
Installation du package dans 'C:/Users/Utilisateur/AppData/Local/R/win-library/4.4'
(car 'lib' n'est pas spécifié)
```

```
le package 'lmtest' a été décompressé et les sommes MD5 ont été vérifiées avec succés

Warning: impossible de supprimer l'installation précédente du package 'lmtest'

Warning in file.copy(savedcopy, lib, recursive = TRUE): problème lors de la
copie de
C:\Users\Utilisateur\AppData\Local\R\win-library\4.4\00LOCK\lmtest\libs\x64\lmtest.dll
vers
C:\Users\Utilisateur\AppData\Local\R\win-library\4.4\lmtest\libs\x64\lmtest.dll :
Permission denied

Warning: 'lmtest' restauré


Les packages binaires téléchargés sont dans
    C:\Users\Utilisateur\AppData\Local\Temp\RtmpENZ3ol\downloaded_packages
```

```
library(lmtest)
```

```
Le chargement a nécessité le package : zoo


Attachement du package : 'zoo'

Les objets suivants sont masqués depuis 'package:base':

    as.Date, as.Date.numeric
```

```
bptest(reg_multi)
```

```


    studentized Breusch-Pagan test

data:  reg_multi
BP = 80.163, df = 5, p-value = 7.76e-16
```

La p-valeur est inférieur à 0.05 et proche de 0 On rejete donc l'hypothèse nulle H0.
L'hypothèse nulle de ce test est que les erreurs ont une variance constante, c'est-à-dire que l'hétéroscédasticité n'est pas présente dans les résidus du modèle.
En d'autres termes, l'hypothèse nulle affirme que les résidus sont homoscédastiques.

Hypothèse nulle (H0) : Les résidus du modèle de régression sont homoscédastiques (la variance des erreurs est constante).
Hypothèse alternative (H1) : Les résidus du modèle de régression ne sont pas homoscédastiques (il y a de l'hétéroscédasticité).

La p-value obtenue est extrêmement petite (7.76e-16), bien en dessous du seuil de 0,05, qui est généralement utilisé pour les tests statistiques.

il y a des preuves significatives d'hétéroscédasticité dans le modèle de régression.
En d'autres termes, la variance des erreurs n'est pas constante et cela peut biaiser les estimations de l'erreur standard des coefficients de régression, ce qui peut à son tour affecter la validité des tests d'hypothèses et des intervalles de confiance.

Autre point Pour tester la normalité des résidus, nous pouvons utiliser le test de shapiro

```
shapiro.test(reg_multi$residuals)
```

```
    Shapiro-Wilk normality test

data:  reg_multi$residuals
W = 0.98579, p-value = 8.54e-11
```

L'hypothèse de normalité est remise en cause car la p-value est inférieure à 0.05 et proche de 0

Nous aurions pu vouloir sélectionner automatiquement un modèle avec l'ensemble des variables avec le package stat

```
reg_null <- lm(margin_low~1, data=data_without_na)
reg_tot <- lm(margin_low~diagonal+height_left+height_right+margin_up+length, data=data)
```

```
#Méthode backward
reg_backward <- step(reg_tot, direction="backward")
```

```
Start:  AIC=-2137.16
margin_low ~ diagonal + height_left + height_right + margin_up +
    length
```

```
              Df Sum of Sq    RSS     AIC
<none>                     336.71 -2137.2
- diagonal     1     1.660 338.37 -2132.0
- margin_up    1     3.661 340.37 -2123.3
- height_left  1     3.909 340.62 -2122.3
- height_right 1     8.259 344.97 -2103.7
- length       1   118.316 455.03 -1698.6
```

Le résultat nous montre que toutes les variables explicatives sont importantes dans le modèle pour prédire margin_low. la suppression de length à l'impact le plus négatif sur le modèle augmentant considérablement le RSS (somme des carrés des résidus) chaque suppression rend le modèle moins performant. Le modèle final sera le modèle complet

ANCIENNE PARTIE A REVOIR

V- Remplacement des données manquantes

```r
# Prédire les valeurs manquantes
predicted_values <- predict(reg_multi, newdata = missing_margin_low)
```

```r
predicted_values
```

```
      73       100       152       198       242       252       285       335
4.318525 4.393668 4.410457 4.319014 4.650617 3.803308 4.179736 4.127442
     411       414       446       482       506       612       655       676
4.135034 4.160539 4.177420 3.768554 4.058764 4.298047 4.160607 4.094065
     711       740       743       781       799       845       846       872
4.439846 4.470650 4.341643 4.080414 3.614306 4.371811 4.093621 4.249629
     896       920       946       947       982      1077      1122      1177
3.893748 3.746333 4.237415 4.710533 4.137780 5.050277 4.802145 5.067584
    1304      1316      1348      1436      1439
5.047570 4.778967 5.726993 5.185862 5.140043
```

```r
# Étape 3: Remplacer les NA par les valeurs prédites
data$margin_low[is.na(data$margin_low)] <- predicted_values
```

```r
# Voir le résultat
summary(data$margin_low)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  2.980   4.020   4.310   4.483   4.870   6.900
```

```
missing_margin_low
```

	is_genuine	diagonal	height_left	height_right	margin_low	margin_up	length
73	True	171.94	103.89	103.45	NA	3.25	112.79
100	True	171.93	104.07	104.18	NA	3.14	113.08
152	True	172.07	103.80	104.38	NA	3.02	112.93
198	True	171.45	103.66	103.80	NA	3.62	113.27
242	True	171.83	104.14	104.06	NA	3.02	112.36
252	True	171.80	103.26	102.82	NA	2.95	113.22
285	True	171.92	103.83	103.76	NA	3.23	113.29
335	True	171.85	103.70	103.96	NA	3.00	113.36
411	True	172.56	103.72	103.51	NA	3.12	112.95
414	True	172.30	103.66	103.50	NA	3.16	112.95
446	True	172.34	104.42	103.22	NA	3.01	112.97
482	True	171.81	103.53	103.96	NA	2.71	113.99
506	True	172.01	103.97	104.05	NA	2.98	113.65
612	True	171.80	103.68	103.49	NA	3.30	112.84
655	True	171.97	103.69	103.54	NA	2.70	112.79
676	True	171.60	103.85	103.91	NA	2.56	113.27
711	True	172.03	103.97	103.86	NA	3.07	112.65
740	True	172.07	103.74	103.76	NA	3.09	112.41
743	True	172.14	104.06	103.96	NA	3.24	113.07
781	True	172.41	103.95	103.79	NA	3.13	113.41
799	True	171.96	103.84	103.62	NA	3.01	114.44
845	True	171.62	104.14	104.49	NA	2.99	113.35
846	True	172.02	104.21	104.05	NA	2.90	113.62
872	True	171.37	104.07	103.75	NA	3.07	113.27
896	True	171.81	103.68	103.80	NA	2.98	113.82
920	True	171.92	103.68	103.45	NA	2.58	113.68
946	True	172.09	103.74	103.52	NA	3.02	112.78
947	True	171.63	103.87	104.66	NA	3.27	112.68
982	True	172.02	104.23	103.72	NA	2.99	113.37
1077	False	171.57	104.27	104.44	NA	3.21	111.87
1122	False	171.40	104.38	104.19	NA	3.17	112.39
1177	False	171.59	104.05	103.94	NA	3.02	111.29
1304	False	172.17	104.49	103.76	NA	2.93	111.21
1316	False	172.08	104.15	104.17	NA	3.40	112.29
1348	False	171.72	104.46	104.12	NA	3.61	110.31
1436	False	172.66	104.33	104.41	NA	3.56	111.47

```
1439      False   171.90      104.28       104.29         NA      3.24 111.49
```

On contrôle ici que les valeurs déterminées par le modèle ont bien remplacées les NA de notre jeu de donnée “data”

```
indices <- rownames(missing_margin_low)
lignes_data <- data[indices, ]
print(lignes_data)
```

```
     is_genuine diagonal height_left height_right margin_low margin_up length
73         True   171.94      103.89       103.45   4.318525      3.25 112.79
100        True   171.93      104.07       104.18   4.393668      3.14 113.08
152        True   172.07      103.80       104.38   4.410457      3.02 112.93
198        True   171.45      103.66       103.80   4.319014      3.62 113.27
242        True   171.83      104.14       104.06   4.650617      3.02 112.36
252        True   171.80      103.26       102.82   3.803308      2.95 113.22
285        True   171.92      103.83       103.76   4.179736      3.23 113.29
335        True   171.85      103.70       103.96   4.127442      3.00 113.36
411        True   172.56      103.72       103.51   4.135034      3.12 112.95
414        True   172.30      103.66       103.50   4.160539      3.16 112.95
446        True   172.34      104.42       103.22   4.177420      3.01 112.97
482        True   171.81      103.53       103.96   3.768554      2.71 113.99
506        True   172.01      103.97       104.05   4.058764      2.98 113.65
612        True   171.80      103.68       103.49   4.298047      3.30 112.84
655        True   171.97      103.69       103.54   4.160607      2.70 112.79
676        True   171.60      103.85       103.91   4.094065      2.56 113.27
711        True   172.03      103.97       103.86   4.439846      3.07 112.65
740        True   172.07      103.74       103.76   4.470650      3.09 112.41
743        True   172.14      104.06       103.96   4.341643      3.24 113.07
781        True   172.41      103.95       103.79   4.080414      3.13 113.41
799        True   171.96      103.84       103.62   3.614306      3.01 114.44
845        True   171.62      104.14       104.49   4.371811      2.99 113.35
846        True   172.02      104.21       104.05   4.093621      2.90 113.62
872        True   171.37      104.07       103.75   4.249629      3.07 113.27
896        True   171.81      103.68       103.80   3.893748      2.98 113.82
920        True   171.92      103.68       103.45   3.746333      2.58 113.68
946        True   172.09      103.74       103.52   4.237415      3.02 112.78
947        True   171.63      103.87       104.66   4.710533      3.27 112.68
982        True   172.02      104.23       103.72   4.137780      2.99 113.37
1077      False   171.57      104.27       104.44   5.050277      3.21 111.87
1122      False   171.40      104.38       104.19   4.802145      3.17 112.39
1177      False   171.59      104.05       103.94   5.067584      3.02 111.29
```

```
1304      False   172.17      104.49      103.76   5.047570      2.93 111.21
1316      False   172.08      104.15      104.17   4.778967      3.40 112.29
1348      False   171.72      104.46      104.12   5.726993      3.61 110.31
1436      False   172.66      104.33      104.41   5.185862      3.56 111.47
1439      False   171.90      104.28      104.29   5.140043      3.24 111.49
```

```r
skim(data)
```

Table 4: Data summary

Name	data
Number of rows	1500
Number of columns	7
_____	
Column type frequency:	
character	1
numeric	6
_____	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
is_genuine	0	1	4	5	0	2	0

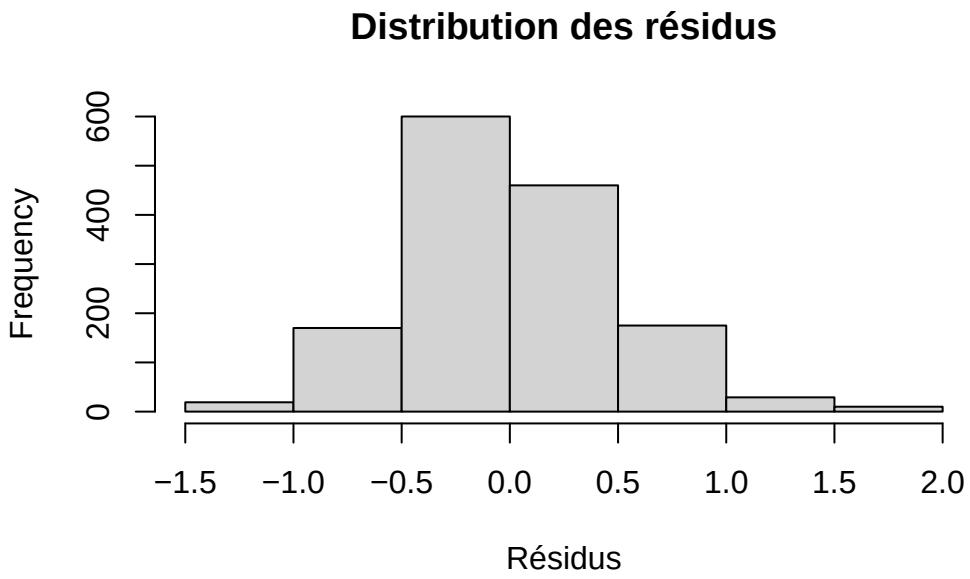
Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
diagonal	0	1	171.96	0.31	171.04	171.75	171.96	172.17	173.01	
height_left	0	1	104.03	0.30	103.14	103.82	104.04	104.23	104.88	
height_right	0	1	103.92	0.33	102.82	103.71	103.92	104.15	104.95	
margin_low	0	1	4.48	0.66	2.98	4.02	4.31	4.87	6.90	
margin_up	0	1	3.15	0.23	2.27	2.99	3.14	3.31	3.91	
length	0	1	112.68	0.87	109.49	112.03	112.96	113.34	114.44	

```r
# Obtenir les résidus du modèle
residus <- residuals(reg_multi)

# Résumé des résidus
summary(residus)
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-1.47234 -0.31707 -0.04168  0.00000  0.27353  1.97084
```

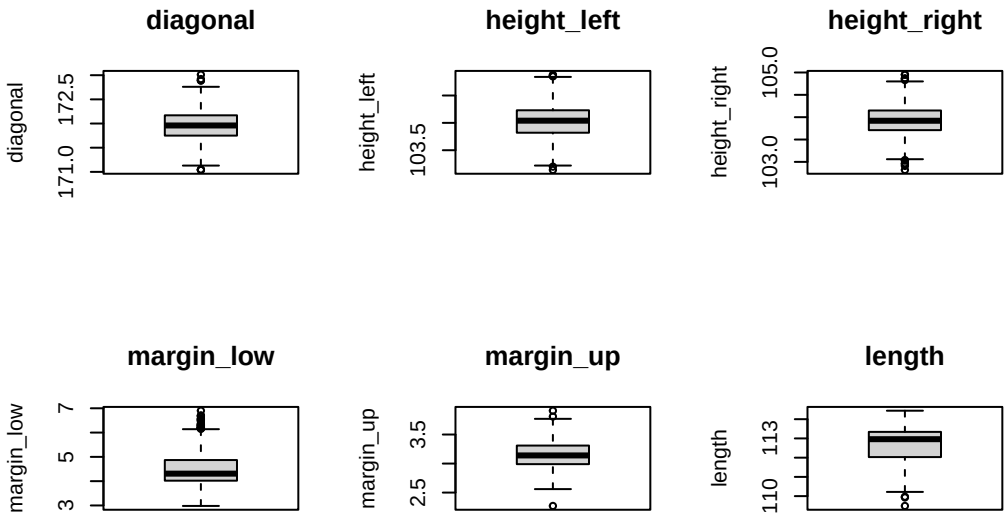
```r
# Visualiser les résidus
hist(residus, main="Distribution des résidus", xlab="Résidus")
```



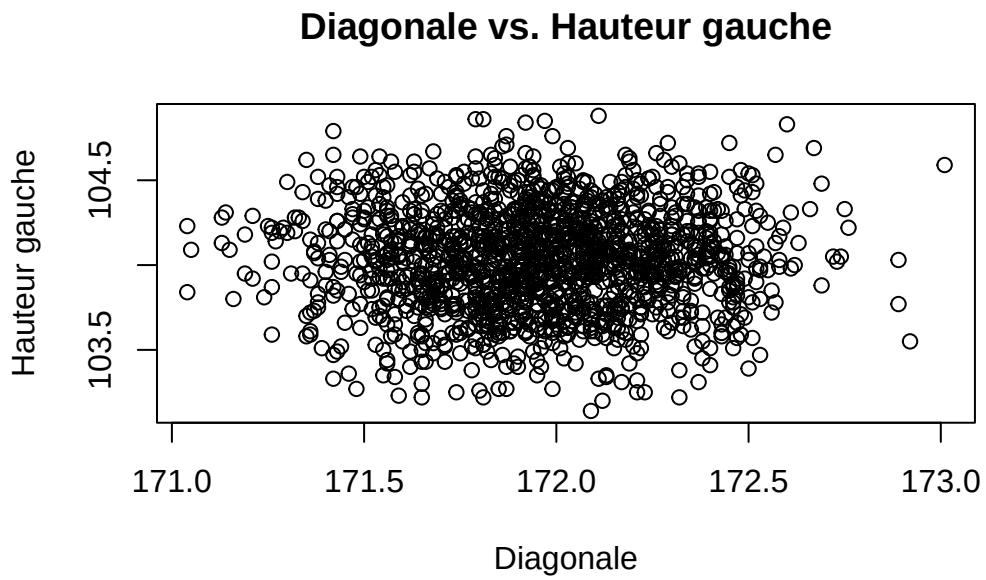
VI- ACP et Clustering

1- Recherche de valeurs aberrantes

```r
# Boxplot pour chaque variable numérique
data_numeric <- data %>% select(where(is.numeric))

# Afficher boxplots
par(mfrow = c(2, 3))  # Adapter le nombre de graphiques selon le nombre de variables
for (col in colnames(data_numeric)) {
  boxplot(data_numeric[[col]], main = col, ylab = col)
}
```




```
# Scatter plot entre deux variables
plot(data$diagonal, data$height_left, main = "Diagonale vs. Hauteur gauche", xlab = "Diagonal
```



a- Méthode IQR

```
# Calculer l'IQR pour chaque colonne
iqr <- function(x) {
  q3 <- quantile(x, 0.75)
  q1 <- quantile(x, 0.25)
  q3 - q1
}

# Détecter les outliers pour chaque colonne
outliers <- function(x) {
  q1 <- quantile(x, 0.25)
  q3 <- quantile(x, 0.75)
  iqr_value <- iqr(x)
  lower_bound <- q1 - 1.5 * iqr_value
  upper_bound <- q3 + 1.5 * iqr_value
  x < lower_bound | x > upper_bound
}
```

```
# Appliquer la fonction pour détecter les outliers
iqr_outliers_data <- data_numeric %>%
  mutate(across(everything(), ~ outliers(.)))

# Afficher les outliers
head(iqr_outliers_data)
```

```
  diagonal height_left height_right margin_low margin_up length
1    FALSE        TRUE         TRUE      FALSE     FALSE  FALSE
2    FALSE       FALSE        FALSE      FALSE     FALSE  FALSE
3    FALSE       FALSE        FALSE      FALSE     FALSE  FALSE
4    FALSE       FALSE        FALSE      FALSE     FALSE  FALSE
5    FALSE       FALSE        FALSE      FALSE     FALSE  FALSE
6    FALSE       FALSE        FALSE      FALSE     FALSE  FALSE
```

```
# Afficher les indices des outliers
iqr_outliers_indices <- which(rowSums(iqr_outliers_data) > 0)
print(iqr_outliers_indices)
```

```
 [1]    1   78  177  194  225  252  293  523  665  730  762  829  843 1023 1024
[16] 1028 1030 1032 1042 1054 1076 1083 1091 1093 1111 1125 1134 1135 1143 1151
[31] 1170 1200 1255 1271 1278 1291 1322 1323 1332 1346 1349 1354 1356 1383 1389
[46] 1421 1427 1442 1454 1460 1465 1474 1485
```

```r
# Afficher les lignes du dataframe original contenant des outliers
data_with_iqr_outliers <- data[iqr_outliers_indices, ]

print(data_with_iqr_outliers)
```

```
     is_genuine diagonal height_left height_right margin_low margin_up length
1          True   171.81      104.86       104.95   4.520000      2.89 112.83
78         True   171.84      104.09       103.03   4.110000      2.77 113.18
177        True   171.75      103.63       102.97   4.460000      2.77 113.22
194        True   172.35      103.73       102.95   4.490000      3.37 112.49
225        True   172.12      103.20       103.92   4.460000      3.26 113.44
252        True   171.80      103.26       102.82   3.803308      2.95 113.22
293        True   172.09      103.14       103.81   4.880000      3.01 113.69
523        True   172.02      104.42       102.91   3.860000      3.12 113.43
665        True   172.05      103.70       103.75   5.040000      2.27 113.55
730        True   171.04      103.84       103.64   4.220000      3.36 112.70
762        True   172.16      103.93       103.04   4.140000      2.99 113.26
829        True   172.92      103.55       103.94   4.780000      3.27 113.55
843        True   172.89      103.77       104.24   4.120000      3.01 113.72
1023      False   172.89      104.03       104.03   6.030000      3.00 110.95
1024      False   172.02      104.26       104.20   6.200000      3.58 111.25
1028      False   171.63      104.02       104.66   6.700000      3.28 111.28
1030      False   171.96      104.29       104.03   6.010000      3.91 110.83
1032      False   172.40      104.00       103.82   6.330000      3.10 112.11
1042      False   171.77      104.12       104.42   6.650000      3.63 111.53
1054      False   171.85      104.52       104.05   6.210000      3.43 111.96
1076      False   172.02      104.51       103.69   6.230000      3.39 112.35
1083      False   171.75      103.96       103.83   5.390000      3.54 109.49
1091      False   172.11      104.88       104.10   4.800000      3.73 110.78
1093      False   171.87      104.76       104.02   6.300000      3.61 111.29
1111      False   171.73      104.32       104.07   6.560000      3.30 112.80
1125      False   171.88      103.92       104.27   6.700000      3.11 110.93
1134      False   171.79      103.99       103.67   6.160000      3.52 110.93
1135      False   171.91      103.70       104.41   6.340000      3.50 113.05
1143      False   171.04      104.23       104.22   4.870000      3.56 111.54
1151      False   171.79      104.86       104.34   5.390000      3.14 113.02
1170      False   171.99      104.14       104.15   6.480000      3.42 112.16
1200      False   172.03      104.32       104.87   4.490000      3.77 111.04
1255      False   171.15      104.09       104.30   6.490000      3.20 111.61
1271      False   171.26      104.22       104.07   4.780000      3.81 112.88
1278      False   173.01      104.59       104.31   5.040000      3.05 110.91
1291      False   171.94      104.06       104.22   6.900000      3.36 111.70
```

```
1322      False   172.29      104.72      104.86   5.710000      3.16 112.15
1323      False   172.07      104.50      104.23   6.190000      3.07 111.21
1332      False   172.32      104.60      104.83   4.840000      3.51 112.55
1346      False   171.56      104.17      103.87   6.160000      3.38 111.55
1349      False   171.84      104.32      104.50   6.280000      3.00 111.06
1354      False   171.61      104.04      104.06   6.190000      3.08 110.73
1356      False   171.68      103.89      103.70   5.970000      3.03 109.97
1383      False   171.97      104.85      104.52   5.870000      3.56 110.98
1389      False   171.05      104.09      104.50   4.720000      3.10 112.44
1421      False   171.56      104.47      104.04   6.380000      3.43 112.12
1427      False   172.22      103.92      104.03   6.250000      3.14 110.89
1442      False   171.63      104.55      103.81   6.560000      3.10 111.87
1454      False   171.55      104.20      104.49   5.420000      3.54 109.93
1460      False   171.78      104.31      103.82   6.190000      3.25 111.14
1465      False   172.07      104.17      104.37   6.540000      3.54 111.20
1474      False   171.76      104.04      104.12   6.290000      3.20 112.87
1485      False   172.08      103.96      104.95   5.220000      3.45 112.07
```

```r
library(dplyr)

# Comptage du nombre de lignes par groupe dans la colonne is_genuine
iqr_count_by_is_genuine <- data_with_iqr_outliers %>%
  group_by(is_genuine) %>%
  summarise(count = n())

# Affichage du résultat
print(iqr_count_by_is_genuine)
```

```
# A tibble: 2 x 2
  is_genuine count
  <chr>      <int>
1 False         40
2 True          13
```

Il y a 53 lignes d'outliers avec la méthode IQR
40 lignes concernent des faux billets
13 lignes concernent de vrais billets

b- Méthode Z_score

```r
if (!require(tidyverse)) install.packages("tidyverse")
```

```
Le chargement a nécessité le package : tidyverse
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v forcats   1.0.0     v stringr   1.5.1
v lubridate 1.9.3     v tibble    3.2.1
v purrr     1.0.2     v tidyr     1.3.1
v readr     2.1.5
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
x car::recode()   masks dplyr::recode()
x purrr::some()   masks car::some()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```r
# Charger les bibliothèques nécessaires
library(tidyverse)

# Sélectionner les colonnes numériques
data_numeric <- data %>% select(where(is.numeric))

# Calculer les scores Z
data_z <- scale(data_numeric)

# Vérifier les scores Z
head(data_z)
```

```
       diagonal height_left height_right  margin_low  margin_up     length
[1,] -0.4863774   2.7731984   3.16218582  0.05537133 -1.1279488  0.1735932
[2,] -1.6331847  -2.2357896  -0.79940117 -1.08162671 -0.6965669  0.4715090
[3,]  2.3970239   1.5042548  -1.29076079 -0.12654835 -0.9122578  0.5517171
[4,] -1.9608439  -0.3991607   0.06047818 -1.30902632 -0.6102905  0.9527576
[5,] -0.7485048   0.8363897  -1.41360070 -0.67230742  1.4172048 -0.1586975
[6,]  0.6931959  -0.9668460   0.49041785 -0.09622841 -0.8691196  0.1506766
```

```r
summary(data_z)
```

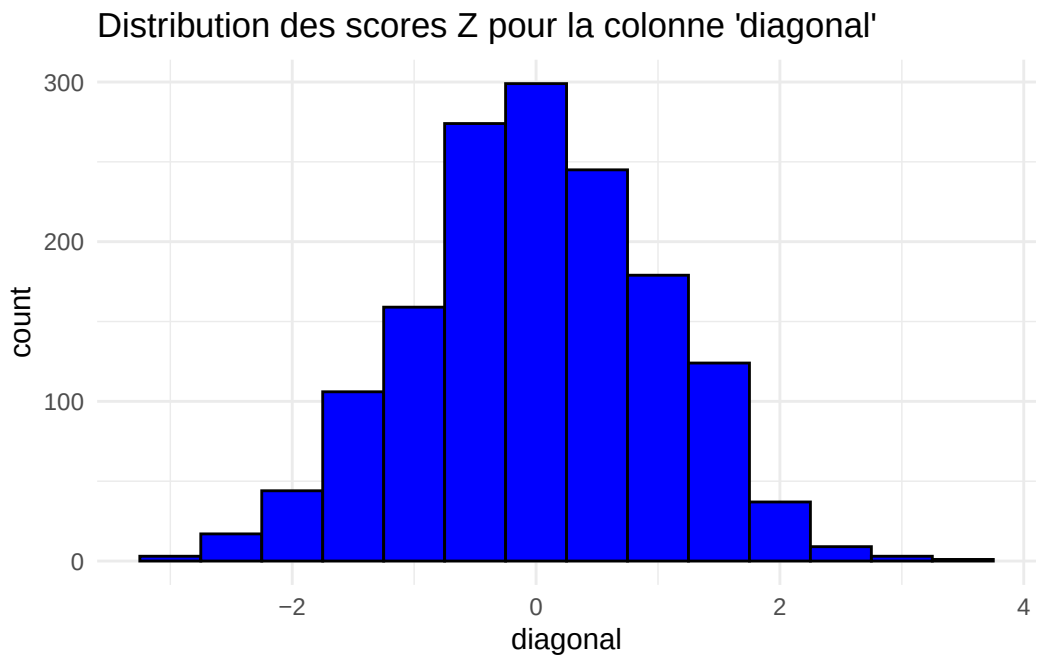
```
    diagonal          height_left        height_right         margin_low
 Min.   :-3.009353   Min.   :-2.97044   Min.   :-3.379039   Min.   :-2.2793
```

```
 1st Qu.:-0.682973   1st Qu.:-0.69970   1st Qu.:-0.645851   1st Qu.:-0.7026
 Median : 0.005111   Median : 0.03495   Median :-0.000942   Median :-0.2630
 Mean   : 0.000000   Mean   : 0.00000   Mean   : 0.000000   Mean   : 0.0000
 3rd Qu.: 0.693196   3rd Qu.: 0.66942   3rd Qu.: 0.705388   3rd Qu.: 0.5860
 Max.   : 3.445533   Max.   : 2.83998   Max.   : 3.162186   Max.   : 3.6634
   margin_up           length
 Min.   :-3.80252   Min.   :-3.6535
 1st Qu.:-0.69657   1st Qu.:-0.7431
 Median :-0.04949   Median : 0.3226
 Mean   : 0.00000   Mean   : 0.0000
 3rd Qu.: 0.68385   3rd Qu.: 0.7580
 Max.   : 3.27215   Max.   : 2.0184
```

```r
# Ajouter les scores Z au dataframe original en conservant la colonne de caractères
data_z_full <- cbind(Row_Index = 1:nrow(data), data %>% select(where(is.character)), as.data

# Afficher les premières lignes du nouveau dataframe
head(data_z_full)
```

```
  Row_Index is_genuine   diagonal height_left height_right  margin_low
1         1       True -0.4863774   2.7731984   3.16218582  0.05537133
2         2       True -1.6331847  -2.2357896  -0.79940117 -1.08162671
3         3       True  2.3970239   1.5042548  -1.29076079 -0.12654835
4         4       True -1.9608439  -0.3991607   0.06047818 -1.30902632
5         5       True -0.7485048   0.8363897  -1.41360070 -0.67230742
6         6       True  0.6931959  -0.9668460   0.49041785 -0.09622841
   margin_up     length
1 -1.1279488  0.1735932
2 -0.6965669  0.4715090
3 -0.9122578  0.5517171
4 -0.6102905  0.9527576
5  1.4172048 -0.1586975
6 -0.8691196  0.1506766
```

```r
# Visualiser la distribution des scores Z pour la colonne 'diagonal'
ggplot(as.data.frame(data_z), aes(x=diagonal)) +
  geom_histogram(binwidth=0.5, fill="blue", color="black") +
  theme_minimal() +
  labs(title="Distribution des scores Z pour la colonne 'diagonal'")
```



On fixe à +3 et -3 la limite de z_score pour identifier les outliers

Puis on affiche les lignes pour lesquels il y a au moins une variables considérée comme outlier

```
# Identifier les lignes avec des outliers (Z > 3 ou Z < -3)
z_outliers <- data_z_full %>%
  filter(apply(data_z, 1, function(row) any(row > 3 | row < -3)))

# Afficher les outliers
print(z_outliers)
```

```
   Row_Index is_genuine     diagonal  height_left height_right  margin_low
1          1       True -0.486377368  2.773198409   3.16218582  0.05537133
2        252       True -0.519143291 -2.569722183  -3.37903921 -1.03113141
3        523       True  0.201707025  1.303895246  -3.10264942 -0.94518695
4        665       True  0.300004795 -1.100419020  -0.52301138  0.84368998
5        730       True -3.009353473 -0.632913468  -0.86082112 -0.39942789
6        829       True  3.150640135 -1.601317826   0.06047818  0.44953066
7        843       True  3.052342364 -0.866666244   0.98177748 -0.55102762
8       1023      False  3.052342364  0.001558352   0.33686797  2.34452740
9       1028      False -1.076163990 -0.031834902   2.27159650  3.36024565
10      1030      False  0.005111484  0.869782948   0.33686797  2.31420745
11      1042      False -0.617441061  0.302097635   1.53455706  3.28444578
```

```
12      1083      False -0.682972908 -0.232194424  -0.27733156  1.37428907
13      1111      False -0.748504755  0.969962709   0.45970788  3.14800602
14      1125      False -0.257015903 -0.365767439   1.07390741  3.36024565
15      1143      False -3.009353473  0.669423426   0.92035753  0.58597042
16      1170      False  0.103409254  0.368884143   0.70538769  3.02672623
17      1255      False -2.648928315  0.201917874   1.16603734  3.04188620
18      1278      False  3.445533446  1.871580559   1.19674732  0.84368998
19      1291      False -0.060420363  0.101738113   0.92035753  3.66344513
20      1356      False -0.912334372 -0.465947200  -0.67656126  2.25356755
21      1442      False -1.076163990  1.738007544  -0.33875152  3.14800602
22      1454      False -1.338291377  0.569243665   1.74952690  1.41976899
23      1465      False  0.365536642  0.469063904   1.38100718  3.11768607
24      1485      False  0.398302565 -0.232194424   3.16218582  1.11656951
    margin_up      length
1  -1.1279488  0.17359325
2  -0.8691196  0.62046695
3  -0.1357703  0.86109125
4  -3.8025171  0.99859085
5   0.8995465  0.02463535
6   0.5113027  0.99859085
7  -0.6102905  1.19338195
8  -0.6534287 -1.98056716
9   0.5544409 -1.60244326
10  3.2721473 -2.11806676
11  2.0642778 -1.31598576
12  1.6760340 -3.65347896
13  0.6407173  0.13921835
14 -0.1789085 -2.00348376
15  1.7623104 -1.30452746
16  1.1583756 -0.59411286
17  0.2093353 -1.22431936
18 -0.4377377 -2.02640036
19  0.8995465 -1.12119466
20 -0.5240141 -3.10348056
21 -0.2220467 -0.92640356
22  1.6760340 -3.14931376
23  1.6760340 -1.69410966
24  1.2877902 -0.69723756
```

```r
# Afficher les indices des outliers
z_outliers_indices <- z_outliers$Row_Index
print(z_outliers_indices)
```

```
 [1]    1  252  523  665  730  829  843 1023 1028 1030 1042 1083 1111 1125 1143
[16] 1170 1255 1278 1291 1356 1442 1454 1465 1485
```

```r
# Afficher les lignes du dataframe original contenant des outliers avec le méthode du z-score
data_with_z_outliers <- data[z_outliers_indices, ]

print(data_with_z_outliers)
```

```
     is_genuine diagonal height_left height_right margin_low margin_up length
1          True   171.81      104.86       104.95   4.520000      2.89 112.83
252        True   171.80      103.26       102.82   3.803308      2.95 113.22
523        True   172.02      104.42       102.91   3.860000      3.12 113.43
665        True   172.05      103.70       103.75   5.040000      2.27 113.55
730        True   171.04      103.84       103.64   4.220000      3.36 112.70
829        True   172.92      103.55       103.94   4.780000      3.27 113.55
843        True   172.89      103.77       104.24   4.120000      3.01 113.72
1023      False   172.89      104.03       104.03   6.030000      3.00 110.95
1028      False   171.63      104.02       104.66   6.700000      3.28 111.28
1030      False   171.96      104.29       104.03   6.010000      3.91 110.83
1042      False   171.77      104.12       104.42   6.650000      3.63 111.53
1083      False   171.75      103.96       103.83   5.390000      3.54 109.49
1111      False   171.73      104.32       104.07   6.560000      3.30 112.80
1125      False   171.88      103.92       104.27   6.700000      3.11 110.93
1143      False   171.04      104.23       104.22   4.870000      3.56 111.54
1170      False   171.99      104.14       104.15   6.480000      3.42 112.16
1255      False   171.15      104.09       104.30   6.490000      3.20 111.61
1278      False   173.01      104.59       104.31   5.040000      3.05 110.91
1291      False   171.94      104.06       104.22   6.900000      3.36 111.70
1356      False   171.68      103.89       103.70   5.970000      3.03 109.97
1442      False   171.63      104.55       103.81   6.560000      3.10 111.87
1454      False   171.55      104.20       104.49   5.420000      3.54 109.93
1465      False   172.07      104.17       104.37   6.540000      3.54 111.20
1485      False   172.08      103.96       104.95   5.220000      3.45 112.07
```

```r
library(dplyr)

# Comptage du nombre de lignes par groupe dans la colonne is_genuine
z_count_by_is_genuine <- z_outliers %>%
  group_by(is_genuine) %>%
  summarise(count = n())
```

```
# Affichage du résultat
print(z_count_by_is_genuine)
```

```
# A tibble: 2 x 2
  is_genuine count
  <chr>      <int>
1 False         17
2 True           7
```

Il y a 24 lignes d'outliers avec la méthode du Z_score
17 lignes concernent des faux billets
7 lignes concernent de vrais billets

2- Suppression des lignes d'outliers

Je vais commencer mon analyse en supprimant les outliers identifiés par la méthode du Z_Score
Je vais donc supprimer 24 lignes du dataframe original "data"
je le nommerai "data_without_ouliers_z"

```
data_without_ouliers_z <- data[-z_outliers_indices, ]
data_without_ouliers_z
```

	is_genuine	diagonal	height_left	height_right	margin_low	margin_up	length
2	True	171.46	103.36	103.66	3.770000	2.99	113.09
3	True	172.69	104.48	103.50	4.400000	2.94	113.16
4	True	171.36	103.91	103.94	3.620000	3.01	113.51
5	True	171.73	104.28	103.46	4.040000	3.48	112.54
6	True	172.17	103.74	104.08	4.420000	2.95	112.81
7	True	172.34	104.18	103.85	4.580000	3.26	112.81
8	True	171.88	103.76	104.08	3.980000	2.92	113.08
9	True	172.47	103.92	103.67	4.000000	3.25	112.85
10	True	172.47	104.07	104.02	4.040000	3.25	113.45
11	True	171.83	104.14	103.62	3.160000	3.18	113.22
12	True	171.84	104.59	104.00	3.880000	3.27	113.08
13	True	171.89	103.89	103.40	4.110000	2.96	113.24
14	True	172.20	104.35	103.67	4.440000	3.38	113.65
15	True	172.06	103.87	103.83	4.090000	2.92	113.19
16	True	171.73	103.92	103.74	4.430000	2.78	112.98
17	True	171.30	104.19	103.70	4.120000	2.82	112.87
18	True	171.88	104.47	103.45	4.560000	3.33	113.01

19	True	172.47	103.89	104.14	3.740000	3.28	113.47
20	True	171.90	103.67	103.15	4.550000	3.15	113.12
21	True	171.87	103.91	103.96	3.740000	2.97	113.59
22	True	171.82	103.65	103.64	4.250000	2.80	112.57
23	True	172.10	103.96	103.71	3.930000	2.89	112.68
24	True	172.56	103.85	103.62	3.320000	3.13	113.37
25	True	172.22	103.75	103.89	4.180000	3.19	113.34
26	True	172.09	103.63	103.33	4.320000	2.88	113.46
27	True	171.83	103.44	103.70	4.150000	3.10	113.47
28	True	171.88	103.92	103.74	4.020000	3.17	113.76
29	True	172.02	104.18	104.16	3.970000	3.23	113.64
30	True	171.58	103.59	103.79	4.100000	2.92	113.56
31	True	171.71	103.95	103.97	4.000000	2.83	113.55
32	True	171.98	104.33	103.61	3.680000	3.14	113.41
33	True	171.99	103.79	103.90	4.620000	2.90	112.92
34	True	171.85	104.10	103.88	4.130000	2.93	112.93
35	True	172.29	103.61	104.44	4.540000	2.69	113.48
36	True	172.01	103.81	103.65	4.360000	3.03	113.26
37	True	171.80	104.28	104.06	4.470000	3.29	113.93
38	True	171.73	104.10	104.19	3.390000	3.19	112.79
39	True	172.00	103.76	104.27	4.420000	3.29	112.67
40	True	172.28	104.11	104.19	3.930000	3.08	113.51
41	True	172.44	103.96	104.55	3.730000	2.86	113.16
42	True	172.08	104.19	103.82	3.990000	3.21	113.20
43	True	172.10	103.73	103.33	3.720000	3.02	113.50
44	True	172.40	103.41	103.85	4.010000	3.15	113.45
45	True	171.79	103.83	103.76	3.990000	3.18	112.90
46	True	171.42	104.65	103.63	3.990000	3.05	113.08
47	True	171.47	103.74	104.42	4.190000	2.92	113.53
48	True	172.13	104.15	103.93	3.970000	2.87	113.31
49	True	172.07	104.05	103.82	3.710000	3.61	113.05
50	True	171.82	103.82	103.61	4.420000	2.90	113.02
51	True	171.95	104.02	103.82	4.280000	3.00	113.28
52	True	172.03	103.59	104.09	4.490000	2.85	113.44
53	True	171.64	104.18	103.78	3.490000	3.74	113.94
54	True	171.71	103.99	103.47	4.490000	3.04	113.07
55	True	172.02	103.58	103.80	3.950000	3.46	112.40
56	True	171.56	103.42	103.64	3.770000	3.04	113.30
57	True	171.25	104.23	103.93	4.360000	2.68	113.50
58	True	171.66	103.75	104.07	4.010000	2.98	113.47
59	True	171.88	104.12	103.86	4.070000	3.41	113.50
60	True	171.88	103.91	103.88	4.050000	3.16	113.24
61	True	172.24	104.03	103.86	3.710000	3.01	113.03

62	True	171.90	103.46	104.22	3.990000	3.00	113.08
63	True	171.42	103.82	103.84	4.010000	2.96	113.16
64	True	171.98	103.92	103.57	4.140000	2.75	113.13
65	True	171.79	103.91	103.94	3.440000	2.96	113.39
66	True	172.08	104.49	103.96	4.380000	3.14	113.42
67	True	172.41	104.10	104.27	3.800000	3.31	112.68
68	True	172.33	103.97	103.66	3.970000	3.07	112.97
69	True	171.60	103.55	103.74	3.790000	2.86	112.82
70	True	171.98	104.16	103.76	4.080000	2.94	113.08
71	True	171.81	103.84	103.66	4.350000	2.93	112.97
72	True	171.84	103.59	103.98	4.090000	3.05	113.09
73	True	171.94	103.89	103.45	4.318525	3.25	112.79
74	True	171.35	103.58	103.59	4.080000	2.85	114.09
75	True	172.49	103.69	103.79	3.810000	2.77	113.09
76	True	172.26	103.85	103.70	4.160000	3.20	112.35
77	True	171.65	103.30	103.55	4.140000	3.37	112.87
78	True	171.84	104.09	103.03	4.110000	2.77	113.18
79	True	171.71	103.43	103.61	4.110000	2.97	112.95
80	True	171.58	103.71	103.57	4.490000	3.14	113.65
81	True	171.89	103.42	103.35	4.920000	3.02	113.20
82	True	171.85	103.87	103.27	4.090000	2.95	113.09
83	True	172.08	103.93	103.85	4.300000	3.26	113.65
84	True	172.52	104.32	103.86	3.700000	3.14	113.36
85	True	171.85	103.91	103.91	4.500000	3.21	113.09
86	True	171.98	104.32	103.71	4.250000	3.06	113.46
87	True	172.01	104.58	103.89	3.960000	2.87	113.28
88	True	171.70	103.75	103.40	4.470000	3.28	113.57
89	True	171.93	103.65	103.56	4.120000	2.77	112.95
90	True	171.94	103.49	103.82	4.240000	3.19	113.05
91	True	171.81	103.76	103.61	4.310000	3.08	112.74
92	True	172.08	104.01	104.50	4.300000	2.97	113.05
93	True	171.81	104.14	103.35	4.030000	3.03	112.83
94	True	172.05	103.75	103.98	4.100000	3.12	113.27
95	True	172.34	104.26	103.76	4.010000	3.17	113.28
96	True	172.00	103.83	103.82	4.640000	3.32	113.00
97	True	171.57	104.61	103.91	4.490000	3.16	113.42
98	True	171.86	103.60	103.99	3.650000	3.15	113.56
99	True	171.54	103.75	103.62	4.590000	2.85	113.44
100	True	171.93	104.07	104.18	4.393668	3.14	113.08
101	True	171.53	104.02	103.87	4.100000	3.03	113.32
102	True	171.55	103.35	103.69	4.310000	3.19	113.42
103	True	172.05	103.69	103.89	4.420000	3.15	112.67
104	True	172.00	104.14	103.98	3.870000	3.12	113.50

105	True	171.96	103.95	103.81	4.230000	3.09	113.27
106	True	171.85	103.66	103.90	4.160000	2.98	112.91
107	True	172.27	103.95	103.75	3.940000	3.19	114.04
108	True	172.69	103.88	104.06	4.270000	3.20	113.22
109	True	171.91	104.23	104.14	3.590000	3.11	113.42
110	True	171.65	103.77	103.95	4.180000	3.16	113.26
111	True	172.07	103.72	103.44	4.030000	2.90	113.55
112	True	172.03	103.96	103.58	4.060000	2.85	113.02
113	True	171.80	104.22	103.74	3.660000	3.08	113.53
114	True	171.82	104.23	104.05	4.210000	3.20	112.89
115	True	172.24	103.97	103.69	4.320000	2.98	113.26
116	True	171.88	103.61	103.46	3.520000	2.99	112.94
117	True	172.23	104.18	103.71	3.930000	3.32	113.48
118	True	172.11	104.36	104.01	4.140000	3.17	112.59
119	True	171.92	103.60	103.36	4.230000	2.93	113.51
120	True	171.89	103.66	103.28	3.790000	3.03	112.74
121	True	171.92	103.78	103.61	4.020000	3.31	112.91
122	True	172.23	104.12	103.75	3.910000	3.02	113.02
123	True	171.81	103.22	103.92	3.990000	3.06	113.32
124	True	171.62	104.24	103.85	4.080000	2.90	112.87
125	True	171.89	103.63	104.12	4.370000	3.06	113.38
126	True	171.50	104.03	103.97	3.260000	3.54	113.24
127	True	172.22	104.07	103.98	4.010000	2.83	112.92
128	True	171.93	103.98	104.07	4.410000	2.96	113.02
129	True	171.96	104.34	103.85	3.950000	3.28	113.73
130	True	172.00	104.05	103.67	4.020000	3.18	113.40
131	True	171.66	103.43	103.54	4.380000	3.51	113.20
132	True	171.27	104.14	103.78	3.910000	2.69	112.81
133	True	171.84	103.77	103.98	4.610000	2.99	113.59
134	True	171.54	104.36	103.28	3.990000	3.21	113.43
135	True	172.27	104.12	103.96	4.120000	2.74	113.12
136	True	171.87	104.71	103.85	4.360000	2.71	113.71
137	True	172.06	104.19	103.59	4.390000	3.22	113.69
138	True	171.60	104.02	103.90	4.200000	3.15	113.15
139	True	172.43	104.05	104.02	4.090000	2.93	113.60
140	True	172.46	103.81	103.52	4.050000	3.16	112.60
141	True	172.47	103.82	103.59	3.850000	2.77	113.17
142	True	171.52	104.11	104.23	4.730000	3.08	113.33
143	True	172.18	104.46	103.80	4.310000	2.84	113.58
144	True	172.04	103.82	103.79	4.070000	3.20	114.13
145	True	171.80	104.31	103.78	3.790000	3.12	112.78
146	True	171.65	103.59	103.83	3.970000	3.12	113.04
147	True	172.38	103.72	103.88	4.360000	3.05	113.19

148	True	172.35	103.72	103.67	4.200000	3.07	112.87
149	True	172.46	103.99	104.04	4.140000	2.68	112.93
150	True	171.91	103.97	103.60	4.660000	3.03	114.04
151	True	171.72	103.91	104.57	4.080000	2.79	113.44
152	True	172.07	103.80	104.38	4.410457	3.02	112.93
153	True	171.97	104.29	104.04	4.460000	2.97	113.81
154	True	171.96	104.16	103.67	3.980000	3.23	112.97
155	True	172.08	103.95	103.70	3.760000	2.97	113.52
156	True	171.75	104.03	103.85	4.360000	2.96	113.22
157	True	171.57	103.98	103.40	4.050000	2.74	113.70
158	True	171.88	104.04	104.08	3.800000	3.01	113.60
159	True	172.05	103.88	103.76	4.200000	3.15	113.94
160	True	172.15	103.74	103.92	4.640000	3.35	113.19
161	True	172.26	104.46	103.60	4.180000	3.31	113.00
162	True	172.31	103.89	103.78	4.490000	3.06	113.34
163	True	172.09	104.05	104.26	4.100000	2.95	113.68
164	True	171.90	103.40	103.45	4.090000	3.33	112.97
165	True	172.38	103.98	104.40	3.810000	2.80	112.90
166	True	171.81	103.53	103.76	3.910000	2.87	113.52
167	True	172.31	104.05	103.51	4.430000	2.86	113.18
168	True	172.20	104.06	103.97	4.160000	3.17	112.47
169	True	172.21	103.32	103.99	4.110000	3.04	113.46
170	True	171.28	104.20	103.89	4.290000	3.06	112.56
171	True	171.62	104.41	103.71	4.130000	3.21	114.03
172	True	172.40	103.51	103.52	3.680000	3.16	113.35
173	True	172.26	104.09	103.56	3.530000	2.99	113.10
174	True	172.58	104.03	103.90	3.790000	2.92	113.57
175	True	171.55	104.40	103.17	4.540000	2.83	113.30
176	True	172.13	104.26	103.88	3.970000	3.00	113.11
177	True	171.75	103.63	102.97	4.460000	2.77	113.22
178	True	172.09	104.19	103.50	4.160000	3.04	112.94
179	True	171.79	104.57	104.04	4.260000	3.15	113.46
180	True	171.66	104.16	103.94	4.050000	3.04	113.09
181	True	171.80	104.17	103.74	4.030000	2.92	113.04
182	True	172.37	103.31	103.50	3.790000	3.13	112.43
183	True	171.94	104.44	104.13	3.320000	3.02	113.49
184	True	171.94	103.97	103.84	3.950000	3.05	112.88
185	True	171.84	103.71	103.90	4.610000	2.82	112.68
186	True	172.40	104.13	103.84	4.250000	3.09	113.29
187	True	172.21	103.48	104.04	4.400000	3.00	113.15
188	True	172.74	104.05	104.17	3.680000	3.38	113.12
189	True	172.10	104.05	103.84	3.760000	3.17	113.44
190	True	172.18	104.02	103.54	4.170000	3.49	113.64

191	True	171.95	103.73	103.77	4.090000	3.02	113.60
192	True	171.86	104.70	103.62	4.010000	3.34	112.89
193	True	171.78	103.74	103.55	3.860000	2.87	113.41
194	True	172.35	103.73	102.95	4.490000	3.37	112.49
195	True	171.94	104.34	103.99	4.270000	2.87	113.14
196	True	172.22	103.59	103.81	3.470000	3.34	113.23
197	True	171.99	103.98	103.73	4.340000	3.20	113.05
198	True	171.45	103.66	103.80	4.319014	3.62	113.27
199	True	171.90	104.14	103.75	4.560000	2.97	113.00
200	True	171.90	104.42	104.02	4.380000	3.10	112.87
201	True	171.97	104.01	103.41	3.800000	2.96	112.85
202	True	171.71	104.27	104.19	3.560000	3.15	112.71
203	True	172.11	103.98	104.02	4.170000	3.24	112.96
204	True	172.33	104.18	104.04	4.290000	2.99	114.08
205	True	171.88	103.83	103.31	4.360000	3.07	113.83
206	True	172.16	104.04	104.02	4.060000	3.08	113.02
207	True	172.18	103.68	103.14	4.160000	3.18	113.93
208	True	172.30	104.34	103.93	3.980000	3.33	113.08
209	True	171.59	103.80	104.08	4.180000	3.17	113.04
210	True	172.23	103.25	103.97	4.010000	3.38	113.90
211	True	172.12	103.76	103.66	4.270000	2.95	113.40
212	True	171.78	103.52	103.69	4.310000	2.85	113.59
213	True	171.59	104.06	103.85	4.410000	3.29	113.49
214	True	172.26	103.75	103.87	3.950000	3.10	113.08
215	True	172.10	103.61	103.79	4.860000	3.14	112.89
216	True	172.05	104.08	103.66	4.490000	3.24	113.23
217	True	171.74	104.43	103.09	4.230000	3.15	113.19
218	True	172.37	104.32	103.52	3.960000	3.11	112.84
219	True	172.19	104.07	103.71	4.000000	2.82	112.71
220	True	172.11	103.33	103.80	4.550000	2.91	113.17
221	True	171.26	103.59	103.80	4.110000	3.25	113.03
222	True	172.04	103.67	103.60	4.090000	3.14	112.99
223	True	171.79	103.88	103.70	4.100000	3.12	113.42
224	True	171.97	104.05	103.82	4.090000	2.88	112.67
225	True	172.12	103.20	103.92	4.460000	3.26	113.44
226	True	171.29	104.22	104.16	4.370000	3.14	113.30
227	True	171.84	103.81	103.83	4.330000	2.86	113.14
228	True	171.69	103.80	104.10	4.320000	2.69	113.52
229	True	172.50	104.54	103.78	3.990000	3.10	112.82
230	True	172.28	103.63	103.97	3.790000	2.94	112.90
231	True	172.32	103.22	104.00	4.010000	3.08	112.87
232	True	172.76	104.22	104.00	4.200000	3.27	113.69
233	True	171.34	104.26	104.06	3.770000	3.20	113.91

234	True	172.26	104.16	103.20	4.550000	3.35	113.23
235	True	171.97	103.55	104.20	4.210000	3.12	113.20
236	True	171.77	103.62	103.32	3.900000	3.17	113.45
237	True	171.85	104.10	104.16	4.180000	3.06	113.19
238	True	172.19	103.96	104.03	3.950000	3.20	112.71
239	True	172.23	104.21	103.97	3.740000	2.90	113.40
240	True	171.85	104.07	103.90	4.800000	3.46	113.73
241	True	171.86	103.60	104.09	3.900000	2.80	113.18
242	True	171.83	104.14	104.06	4.650617	3.02	112.36
243	True	171.74	103.95	103.78	4.740000	3.29	113.04
244	True	172.17	103.61	104.33	3.390000	2.82	112.79
245	True	171.57	104.09	103.58	3.990000	3.23	112.98
246	True	171.99	103.70	103.59	4.370000	3.12	112.85
247	True	171.94	103.91	103.99	4.020000	2.94	113.06
248	True	172.45	103.97	104.32	4.150000	3.23	113.45
249	True	172.12	104.03	104.02	4.380000	2.83	113.03
250	True	171.67	103.74	103.70	4.010000	2.87	113.29
251	True	172.38	103.45	104.11	4.720000	3.01	113.19
253	True	171.68	104.27	104.07	4.080000	3.34	113.57
254	True	172.07	104.33	104.33	4.410000	3.39	113.39
255	True	172.17	104.21	103.87	4.350000	2.98	113.66
256	True	171.82	103.49	103.47	4.790000	3.11	113.41
257	True	171.85	103.65	103.55	4.480000	3.31	112.89
258	True	172.19	104.03	104.25	4.170000	2.81	112.94
259	True	171.71	103.53	103.78	3.890000	3.03	113.51
260	True	172.35	103.62	103.78	4.380000	2.86	113.28
261	True	172.27	103.71	103.64	4.640000	2.67	113.63
262	True	171.99	103.75	103.53	3.790000	2.81	112.52
263	True	171.91	103.97	103.81	3.960000	3.03	112.64
264	True	172.06	103.92	103.42	4.280000	3.00	113.41
265	True	171.75	104.12	104.10	3.850000	3.24	113.30
266	True	172.15	104.02	103.94	3.960000	2.87	112.38
267	True	171.95	103.94	103.80	4.100000	3.41	113.38
268	True	171.48	103.27	103.42	3.840000	3.04	113.56
269	True	171.47	104.30	104.01	4.320000	2.68	113.07
270	True	171.88	103.93	103.72	4.880000	3.26	113.11
271	True	172.11	103.98	104.33	4.300000	2.91	112.72
272	True	172.06	104.21	103.88	4.630000	3.22	113.70
273	True	171.60	104.22	104.11	4.250000	3.29	113.40
274	True	172.19	104.05	103.54	3.800000	3.24	112.84
275	True	171.52	103.74	104.10	4.470000	2.87	113.58
276	True	171.81	104.17	103.72	4.170000	3.05	112.49
277	True	171.62	104.11	103.71	4.330000	3.14	113.14

278	True	172.08	104.16	104.17	3.540000	3.29	112.71
279	True	171.86	104.15	103.50	4.310000	2.92	113.87
280	True	172.29	103.73	103.55	3.840000	3.23	113.90
281	True	171.85	103.64	103.63	3.730000	3.29	112.95
282	True	171.99	103.90	104.21	4.180000	3.07	113.01
283	True	172.02	104.54	103.74	3.960000	2.94	113.51
284	True	172.47	103.57	103.89	4.600000	3.20	112.93
285	True	171.92	103.83	103.76	4.179736	3.23	113.29
286	True	172.27	103.78	103.86	4.250000	2.86	112.73
287	True	172.04	103.99	104.12	4.200000	3.02	113.16
288	True	171.92	104.12	104.15	3.830000	2.88	112.65
289	True	172.21	104.22	103.57	3.570000	3.17	112.43
290	True	172.08	104.40	103.94	4.220000	2.97	113.08
291	True	171.85	103.93	103.81	4.360000	3.12	112.82
292	True	171.77	103.79	104.11	3.920000	3.18	113.33
293	True	172.09	103.14	103.81	4.880000	3.01	113.69
294	True	172.50	103.81	103.83	4.190000	3.01	113.18
295	True	171.52	103.97	103.81	4.080000	3.07	112.74
296	True	172.34	104.54	103.56	4.040000	3.42	112.93
297	True	171.87	103.99	104.03	4.420000	2.89	112.72
298	True	172.12	103.90	104.32	3.830000	3.03	113.56
299	True	172.43	103.83	104.13	3.720000	2.81	113.27
300	True	171.68	103.77	103.82	4.240000	2.86	113.60
301	True	171.54	104.27	104.01	4.630000	3.05	113.09
302	True	171.46	103.83	103.95	4.210000	3.15	113.59
303	True	171.65	103.22	104.11	3.510000	3.04	113.40
304	True	172.02	103.45	103.79	3.910000	3.00	113.12
305	True	172.30	103.88	103.85	4.220000	3.09	113.10
306	True	171.83	104.65	103.66	3.830000	2.95	113.28
307	True	172.10	103.93	103.43	3.960000	3.13	113.38
308	True	172.24	104.22	103.67	4.300000	3.33	113.13
309	True	171.87	103.99	104.02	4.230000	3.02	113.00
310	True	172.25	104.14	103.76	3.840000	2.81	112.72
311	True	172.47	103.87	104.16	4.630000	3.18	112.98
312	True	172.58	103.99	103.63	4.430000	3.07	112.79
313	True	172.07	104.09	103.70	4.110000	3.00	113.19
314	True	172.32	103.99	104.01	4.220000	2.95	113.08
315	True	172.10	104.16	104.18	4.030000	3.08	112.65
316	True	171.99	103.60	103.43	4.570000	2.96	113.09
317	True	171.91	103.99	103.81	4.000000	2.88	113.02
318	True	171.51	104.34	103.49	4.290000	3.03	113.42
319	True	172.22	103.51	103.99	3.670000	3.06	112.86
320	True	172.05	104.11	104.14	4.730000	2.73	113.30

321	True	172.07	104.35	104.43	4.100000	3.02	112.76
322	True	171.74	103.74	103.78	4.430000	3.14	113.02
323	True	171.75	104.13	103.91	4.240000	2.97	113.33
324	True	171.91	103.83	103.68	4.050000	2.86	113.41
325	True	172.10	104.42	103.60	4.180000	2.89	113.32
326	True	171.96	104.17	103.35	4.350000	3.31	113.67
327	True	171.42	103.47	103.82	4.350000	3.05	112.70
328	True	171.85	103.86	104.08	4.230000	3.36	113.01
329	True	172.05	103.79	104.17	4.080000	3.12	112.77
330	True	172.00	103.85	104.04	3.750000	3.45	113.73
331	True	171.99	104.12	104.18	4.410000	2.75	113.18
332	True	172.40	104.17	104.09	4.460000	3.02	113.16
333	True	171.95	103.44	103.80	4.080000	3.19	112.43
334	True	171.53	104.56	103.90	3.990000	3.08	113.15
335	True	171.85	103.70	103.96	4.127442	3.00	113.36
336	True	172.33	103.81	103.67	4.580000	2.87	113.20
337	True	171.80	103.96	103.87	3.970000	2.65	112.68
338	True	171.59	103.80	103.95	4.090000	2.89	113.27
339	True	171.69	104.30	103.94	4.270000	2.96	113.43
340	True	172.26	103.88	104.15	4.300000	2.98	113.24
341	True	172.52	103.90	104.03	3.970000	2.98	113.30
342	True	171.90	104.21	104.21	4.770000	3.38	113.20
343	True	171.78	103.94	103.84	3.550000	2.97	113.13
344	True	171.70	103.77	103.88	4.100000	2.96	112.91
345	True	172.53	103.97	103.28	4.610000	3.41	113.25
346	True	171.54	103.93	103.49	4.080000	3.34	113.13
347	True	172.32	103.38	103.74	4.430000	2.77	112.98
348	True	172.46	103.78	103.54	3.890000	3.20	113.05
349	True	171.87	103.77	104.23	4.100000	3.08	112.44
350	True	172.08	104.17	103.48	4.620000	2.87	113.52
351	True	171.93	104.40	103.40	3.870000	3.15	113.48
352	True	172.06	103.82	103.73	4.180000	2.60	112.64
353	True	171.81	103.64	104.15	4.170000	2.81	113.29
354	True	172.21	103.80	103.23	3.700000	2.98	112.95
355	True	171.96	104.18	104.14	3.350000	2.70	113.56
356	True	172.30	103.82	104.09	4.180000	3.22	113.23
357	True	171.93	103.58	104.02	4.140000	2.86	113.47
358	True	172.33	103.64	104.15	4.390000	2.93	112.43
359	True	172.04	103.95	104.09	4.480000	2.61	112.94
360	True	171.36	103.59	103.68	4.440000	3.32	113.23
361	True	171.61	103.73	104.01	4.290000	2.93	112.76
362	True	172.19	104.09	103.82	3.820000	3.04	113.13
363	True	171.62	103.40	104.05	3.530000	2.96	112.72

364	True	172.40	104.32	103.60	4.130000	2.92	113.40
365	True	172.29	103.83	103.25	3.620000	2.99	113.13
366	True	171.57	103.92	104.03	4.590000	3.14	113.45
367	True	171.72	104.15	103.75	4.100000	2.99	113.24
368	True	171.73	104.17	104.02	3.640000	2.64	112.72
369	True	171.65	103.58	104.21	3.910000	3.30	113.42
370	True	172.46	103.64	103.53	4.410000	3.40	113.75
371	True	172.10	103.99	104.02	4.190000	3.21	113.12
372	True	171.91	104.05	103.97	3.880000	3.12	113.62
373	True	171.43	103.49	103.99	4.070000	2.94	113.07
374	True	171.65	104.29	103.81	4.260000	3.05	113.58
375	True	172.07	104.49	103.52	4.240000	3.00	113.06
376	True	171.54	104.64	103.06	3.540000	3.09	112.82
377	True	171.74	104.53	103.38	4.390000	3.00	112.85
378	True	171.94	104.34	103.71	3.670000	3.17	112.68
379	True	172.06	104.24	103.66	4.240000	3.10	113.02
380	True	172.15	103.63	103.93	4.330000	3.04	113.36
381	True	171.95	104.08	103.91	4.160000	2.91	113.26
382	True	171.75	103.95	103.36	4.570000	3.25	113.08
383	True	172.28	104.62	103.80	4.080000	3.08	113.26
384	True	171.71	104.22	103.72	4.350000	3.29	113.51
385	True	171.65	103.50	103.24	3.820000	2.56	112.98
386	True	171.49	104.02	103.90	4.370000	3.11	112.93
387	True	172.16	103.56	103.64	4.240000	3.01	113.38
388	True	172.37	104.19	104.58	4.250000	2.82	112.88
389	True	172.42	104.07	103.73	4.230000	3.11	113.27
390	True	172.21	104.00	103.86	4.050000	2.82	113.74
391	True	171.66	103.65	103.44	3.260000	2.99	112.89
392	True	171.55	103.68	104.27	4.080000	2.84	112.95
393	True	171.79	103.87	103.95	4.540000	3.01	112.45
394	True	171.96	104.19	104.14	4.300000	3.04	113.48
395	True	171.67	104.57	103.43	3.570000	3.25	112.76
396	True	171.82	104.33	103.83	3.920000	2.73	112.99
397	True	172.47	103.66	103.92	3.590000	3.08	113.45
398	True	172.15	104.18	103.53	3.920000	3.30	113.42
399	True	171.87	104.29	103.53	3.910000	2.89	112.91
400	True	171.52	104.08	104.06	4.410000	3.36	113.32
401	True	171.63	104.30	103.75	3.620000	3.22	113.50
402	True	172.45	104.72	104.17	4.180000	2.83	113.75
403	True	172.30	104.23	104.03	4.120000	3.06	113.23
404	True	171.95	104.03	103.68	4.110000	2.75	113.62
405	True	172.19	103.68	103.93	4.360000	2.92	113.30
406	True	172.40	104.26	103.83	4.310000	3.01	112.82

407	True	171.82	103.72	104.18	4.220000	2.80	113.39
408	True	172.50	103.39	103.99	4.450000	3.06	113.48
409	True	172.18	103.56	104.39	3.750000	3.06	113.46
410	True	172.06	103.96	103.72	4.180000	3.15	113.17
411	True	172.56	103.72	103.51	4.135034	3.12	112.95
412	True	172.27	104.06	103.71	3.740000	3.21	113.48
413	True	172.44	103.96	103.86	4.020000	3.09	112.81
414	True	172.30	103.66	103.50	4.160539	3.16	112.95
415	True	172.30	104.12	103.89	3.550000	2.93	113.68
416	True	172.00	103.68	104.10	3.840000	3.15	112.86
417	True	172.44	104.11	104.25	4.530000	3.18	113.41
418	True	172.73	104.02	104.18	3.740000	3.20	113.52
419	True	172.02	104.23	103.97	4.300000	2.93	113.79
420	True	171.56	103.97	103.96	4.120000	3.32	113.43
421	True	171.89	103.99	103.45	4.120000	3.10	113.00
422	True	171.19	103.95	103.73	3.960000	2.94	113.25
423	True	171.53	103.53	103.63	4.040000	2.96	112.76
424	True	171.70	104.32	104.07	4.210000	2.67	112.73
425	True	171.70	103.74	103.89	4.430000	2.80	112.59
426	True	171.91	103.97	103.98	4.230000	3.01	112.98
427	True	171.91	103.99	103.50	3.410000	2.92	113.02
428	True	171.82	103.64	104.21	4.080000	3.00	112.97
429	True	171.86	103.77	103.93	4.830000	3.17	113.35
430	True	172.09	103.78	104.22	4.160000	3.18	113.30
431	True	172.12	104.08	103.93	4.720000	3.18	113.38
432	True	172.14	103.80	103.56	3.940000	3.12	113.38
433	True	171.47	104.46	104.30	4.120000	3.06	113.59
434	True	171.71	103.71	103.36	4.100000	3.17	113.07
435	True	172.04	103.82	103.99	3.880000	3.14	113.23
436	True	171.55	103.92	103.77	4.120000	3.25	113.19
437	True	171.88	104.58	104.31	4.880000	2.82	113.71
438	True	171.65	103.88	103.59	3.940000	3.05	113.28
439	True	172.07	104.07	103.67	3.820000	3.17	113.91
440	True	171.99	103.74	103.46	4.000000	2.76	113.33
441	True	172.45	103.87	104.40	3.750000	3.05	113.62
442	True	171.93	103.88	104.35	4.260000	3.29	112.70
443	True	172.03	103.73	104.19	3.750000	2.87	112.49
444	True	171.61	103.93	103.41	4.450000	2.95	113.19
445	True	172.23	103.94	103.46	3.950000	3.15	113.77
446	True	172.34	104.42	103.22	4.177420	3.01	112.97
447	True	172.51	104.53	103.50	4.510000	3.07	113.21
448	True	171.59	103.86	104.33	4.120000	2.84	114.00
449	True	172.01	103.93	103.69	3.890000	3.34	113.16

450	True	171.92	103.91	103.93	4.340000	2.91	112.40
451	True	172.01	104.11	103.13	4.530000	2.96	113.45
452	True	172.17	103.79	103.54	4.070000	2.78	113.03
453	True	171.68	103.48	103.47	3.320000	3.11	112.46
454	True	172.54	104.05	103.78	4.340000	2.93	114.14
455	True	172.30	104.21	104.46	3.690000	3.12	113.28
456	True	171.70	103.57	104.20	4.220000	2.81	112.71
457	True	172.25	104.22	103.62	3.960000	3.15	112.74
458	True	171.75	103.48	103.94	3.760000	2.97	113.28
459	True	172.00	104.07	103.55	3.860000	3.30	114.03
460	True	171.64	104.45	103.97	4.160000	3.23	113.67
461	True	172.16	103.69	103.71	3.680000	3.12	113.04
462	True	172.01	103.95	103.77	4.160000	3.10	113.19
463	True	171.48	103.95	103.93	4.520000	2.91	113.21
464	True	172.19	104.14	103.59	4.110000	3.08	113.81
465	True	172.16	103.87	103.34	4.320000	2.91	113.47
466	True	171.55	103.50	104.51	4.610000	2.95	113.31
467	True	171.98	103.77	103.91	4.750000	3.05	113.38
468	True	171.80	104.06	103.57	4.060000	3.14	112.98
469	True	171.83	103.97	104.00	3.720000	3.13	113.83
470	True	172.36	104.34	103.70	4.220000	3.08	112.96
471	True	171.73	104.02	103.92	4.100000	3.19	112.87
472	True	172.58	104.17	103.13	4.640000	3.17	113.05
473	True	171.97	104.04	104.24	4.230000	2.84	113.54
474	True	172.26	104.08	103.86	3.930000	3.01	113.44
475	True	172.29	103.78	103.76	4.700000	3.05	113.20
476	True	172.02	104.36	103.64	4.120000	2.82	113.38
477	True	171.41	104.47	104.08	3.870000	3.18	113.14
478	True	171.95	104.25	103.64	3.240000	2.81	113.42
479	True	171.71	104.49	103.54	3.770000	3.20	113.43
480	True	171.87	103.86	103.27	4.140000	3.14	113.32
481	True	171.97	103.62	103.64	4.470000	2.65	113.70
482	True	171.81	103.53	103.96	3.768554	2.71	113.99
483	True	172.35	104.20	103.75	3.700000	2.96	113.05
484	True	172.14	103.91	103.65	4.190000	3.23	113.54
485	True	172.10	104.02	103.88	3.950000	3.10	113.39
486	True	171.83	104.29	103.58	4.000000	3.32	113.21
487	True	171.95	104.39	103.59	4.690000	2.81	113.65
488	True	171.75	103.89	103.81	3.640000	2.92	113.48
489	True	172.19	104.07	104.02	4.190000	3.37	113.27
490	True	171.98	104.36	103.44	4.190000	3.20	112.92
491	True	172.13	103.71	103.60	3.590000	3.12	113.80
492	True	172.60	104.83	103.56	4.100000	2.94	113.38

493	True	172.33	104.46	103.79	4.080000	2.94	113.39
494	True	172.55	103.97	104.04	4.250000	3.34	113.36
495	True	172.08	104.19	104.03	4.110000	2.99	113.50
496	True	171.81	103.93	103.43	4.610000	2.95	113.46
497	True	172.21	103.67	104.00	4.120000	3.20	113.54
498	True	172.45	103.95	103.50	4.100000	3.08	113.23
499	True	171.76	104.01	103.68	4.070000	2.75	112.81
500	True	172.04	104.34	103.55	3.520000	3.02	113.64
501	True	171.95	104.11	103.87	4.200000	2.81	113.52
502	True	171.87	103.73	103.69	4.060000	3.10	112.96
503	True	171.70	103.79	103.98	4.440000	3.03	113.94
504	True	172.46	103.51	103.85	4.230000	3.14	113.38
505	True	171.68	103.95	103.60	3.980000	2.93	113.22
506	True	172.01	103.97	104.05	4.058764	2.98	113.65
507	True	171.82	104.27	103.55	4.210000	3.20	113.28
508	True	171.97	104.05	104.11	4.150000	2.99	113.66
509	True	171.82	104.12	103.95	4.180000	3.01	113.47
510	True	171.40	103.96	103.85	4.150000	2.81	113.45
511	True	171.87	103.87	104.27	4.440000	2.94	112.95
512	True	172.44	103.69	104.16	3.730000	3.14	112.98
513	True	171.95	104.09	103.73	4.340000	3.07	113.49
514	True	171.77	104.19	103.69	4.170000	2.85	112.89
515	True	171.43	104.42	103.41	4.060000	3.13	113.26
516	True	172.21	103.25	103.78	4.480000	3.00	113.25
517	True	171.91	104.34	103.73	3.710000	2.95	113.69
518	True	171.53	103.69	103.51	3.970000	3.08	112.71
519	True	171.36	104.15	103.76	3.440000	3.31	112.76
520	True	172.13	103.76	103.33	4.370000	3.48	113.34
521	True	172.12	103.66	104.22	3.810000	3.04	112.97
522	True	171.89	104.30	103.57	3.980000	2.75	112.92
524	True	172.13	103.57	103.80	4.400000	3.16	113.75
525	True	172.09	103.61	103.90	4.580000	3.26	112.88
526	True	171.97	103.61	103.60	4.890000	3.17	112.96
527	True	171.88	103.61	103.90	4.400000	3.00	112.97
528	True	171.92	103.79	103.34	3.840000	2.80	113.00
529	True	172.32	103.69	103.60	4.240000	3.29	112.74
530	True	172.16	104.17	103.27	4.100000	3.11	113.08
531	True	171.90	103.94	103.38	3.670000	2.87	113.28
532	True	172.22	104.00	103.98	3.990000	3.41	112.94
533	True	171.73	103.77	103.95	4.250000	3.08	112.75
534	True	172.33	103.73	103.80	3.770000	2.86	113.58
535	True	171.76	104.04	103.88	4.180000	3.21	113.15
536	True	171.68	104.18	103.89	4.400000	3.33	113.21

537	True	172.38	103.64	103.81	4.460000	3.14	114.07
538	True	171.61	103.86	104.06	4.310000	3.10	112.64
539	True	171.90	104.50	103.49	4.080000	2.82	113.50
540	True	171.87	103.75	103.55	3.860000	2.82	113.38
541	True	171.96	103.68	103.84	4.460000	3.07	112.74
542	True	171.49	103.63	103.83	3.850000	3.26	113.10
543	True	171.64	103.95	103.45	3.980000	3.11	113.21
544	True	172.30	103.98	103.72	4.280000	3.01	112.97
545	True	172.25	104.08	103.59	4.520000	3.01	113.46
546	True	171.99	103.75	104.14	4.190000	2.82	112.83
547	True	171.73	103.76	103.91	4.280000	3.17	113.09
548	True	172.37	104.41	103.75	3.710000	2.80	113.41
549	True	172.05	103.75	103.47	4.190000	2.98	112.89
550	True	171.89	104.22	104.18	4.110000	2.89	113.53
551	True	171.79	104.37	103.37	4.310000	3.18	113.61
552	True	171.56	103.44	104.09	4.410000	2.97	113.59
553	True	171.86	103.93	103.80	4.270000	2.86	113.97
554	True	171.51	104.49	103.40	3.850000	3.12	112.76
555	True	171.61	103.76	103.89	4.020000	3.17	112.81
556	True	172.07	104.18	104.10	4.470000	2.70	113.99
557	True	171.83	104.54	103.91	4.110000	3.05	113.55
558	True	171.24	103.81	103.74	4.690000	3.29	113.53
559	True	171.81	103.79	103.94	4.140000	3.08	112.62
560	True	172.12	103.59	103.77	3.740000	3.21	113.32
561	True	172.09	104.02	103.79	3.800000	3.28	112.98
562	True	172.25	103.71	103.97	3.930000	3.00	113.38
563	True	172.45	103.78	103.79	3.900000	2.87	112.34
564	True	171.87	103.99	104.11	3.660000	3.15	113.35
565	True	172.00	104.10	104.36	3.830000	2.92	113.00
566	True	172.23	104.00	103.35	4.430000	2.95	113.12
567	True	172.15	103.51	103.76	4.260000	3.13	112.94
568	True	172.13	104.08	103.64	3.930000	2.82	112.85
569	True	172.05	104.11	104.07	3.600000	2.78	112.93
570	True	171.93	104.08	103.78	4.030000	2.85	113.61
571	True	171.57	104.07	103.85	3.820000	3.44	113.40
572	True	171.95	103.76	103.99	3.250000	2.78	113.20
573	True	171.83	103.80	104.22	4.660000	2.76	113.25
574	True	172.28	104.09	104.20	3.660000	2.83	113.48
575	True	172.35	104.13	103.58	4.750000	3.19	113.42
576	True	171.99	104.10	103.57	3.640000	3.42	113.64
577	True	172.03	103.63	104.00	3.700000	3.37	113.77
578	True	172.01	103.62	103.26	4.910000	2.86	113.16
579	True	172.53	103.80	103.82	4.640000	3.28	113.54

580	True	172.03	104.19	104.01	3.990000	3.04	113.15
581	True	171.76	104.48	104.01	4.350000	3.37	113.09
582	True	172.51	103.57	104.27	3.720000	3.33	113.47
583	True	172.13	103.99	104.22	3.610000	2.91	113.39
584	True	171.68	103.84	103.98	3.520000	3.01	113.45
585	True	171.98	104.18	103.96	4.410000	2.84	113.01
586	True	171.97	104.04	103.82	4.040000	3.38	113.05
587	True	171.89	104.33	103.97	3.950000	3.17	113.12
588	True	171.74	103.72	103.52	3.940000	2.77	113.09
589	True	171.92	103.67	104.24	4.110000	3.09	113.23
590	True	172.04	104.16	104.05	4.000000	3.28	112.78
591	True	171.85	103.58	104.17	4.320000	3.07	113.59
592	True	171.67	103.81	103.76	4.590000	3.30	112.18
593	True	171.99	103.54	103.44	3.800000	3.08	113.04
594	True	172.29	104.46	104.42	3.970000	2.90	113.38
595	True	171.65	103.43	103.83	4.360000	3.10	113.53
596	True	171.96	103.83	103.87	3.520000	2.71	113.52
597	True	172.24	104.51	103.90	4.060000	2.88	112.58
598	True	171.95	103.85	104.28	4.130000	3.22	113.24
599	True	171.67	103.88	103.94	4.200000	2.93	112.84
600	True	172.31	103.93	103.32	4.730000	2.85	113.23
601	True	172.20	103.82	103.74	4.220000	3.03	113.69
602	True	171.74	103.25	104.07	3.800000	2.95	113.07
603	True	171.68	103.88	104.14	4.000000	2.96	113.71
604	True	172.08	104.03	103.42	3.790000	2.77	113.27
605	True	172.45	103.85	103.89	4.030000	3.03	113.24
606	True	172.37	104.26	103.87	4.470000	3.12	112.82
607	True	171.52	104.52	103.96	3.430000	3.01	113.44
608	True	171.91	104.17	103.62	4.010000	2.89	112.73
609	True	172.34	104.50	103.66	4.190000	3.08	112.68
610	True	172.53	103.47	103.99	3.840000	2.71	112.74
611	True	171.80	103.71	103.67	4.480000	3.18	112.76
612	True	171.80	103.68	103.49	4.298047	3.30	112.84
613	True	171.95	103.35	103.76	3.850000	3.29	113.07
614	True	172.45	104.52	103.72	4.120000	3.34	113.06
615	True	172.40	103.86	103.88	3.870000	2.97	113.49
616	True	171.95	103.77	103.81	4.130000	3.09	113.24
617	True	172.51	104.23	103.85	4.740000	2.91	112.81
618	True	172.36	103.89	103.67	4.340000	3.07	113.75
619	True	172.22	103.75	103.55	3.930000	3.05	113.48
620	True	172.26	104.06	103.99	4.030000	2.84	113.14
621	True	171.87	103.40	103.82	4.060000	2.94	113.02
622	True	171.99	103.69	103.51	3.900000	3.02	113.04

623	True	171.56	104.32	104.33	3.920000	2.97	113.04
624	True	171.85	104.33	104.03	4.230000	3.19	112.46
625	True	172.09	103.98	103.39	4.100000	2.95	112.79
626	True	172.05	103.42	104.10	4.450000	2.99	112.67
627	True	171.98	104.44	104.08	4.670000	3.21	112.73
628	True	172.00	104.23	103.74	4.410000	2.76	112.96
629	True	171.44	103.52	103.49	4.090000	3.12	113.23
630	True	172.01	104.03	103.67	3.900000	3.18	112.61
631	True	172.49	104.33	104.03	4.280000	3.07	112.71
632	True	172.75	104.33	103.97	4.340000	3.14	113.12
633	True	171.66	104.17	104.16	4.750000	2.94	113.52
634	True	172.40	104.19	103.98	4.080000	2.93	113.44
635	True	172.20	103.93	103.49	3.800000	2.99	113.63
636	True	172.21	104.27	104.01	4.230000	2.79	113.78
637	True	171.13	104.28	103.14	4.160000	2.92	113.00
638	True	171.51	103.85	103.36	4.490000	2.80	113.87
639	True	171.81	104.10	103.69	4.290000	2.95	112.72
640	True	171.88	103.66	103.52	4.660000	2.75	113.25
641	True	171.91	104.34	103.77	4.450000	2.95	112.66
642	True	171.79	103.51	103.25	4.050000	3.08	112.71
643	True	171.85	103.90	103.74	4.130000	3.07	113.15
644	True	172.07	103.83	103.71	4.440000	2.99	113.11
645	True	171.81	103.91	103.78	3.660000	3.28	113.59
646	True	171.73	103.82	103.85	3.970000	3.12	112.85
647	True	171.59	103.23	103.64	4.010000	2.94	113.59
648	True	171.71	103.83	103.51	3.800000	3.02	113.01
649	True	172.22	104.48	104.06	4.590000	2.91	112.82
650	True	171.59	104.06	103.99	3.930000	3.24	112.80
651	True	172.32	104.16	104.14	3.780000	3.25	113.38
652	True	171.62	103.49	103.58	3.950000	3.00	113.10
653	True	172.14	103.74	103.52	4.560000	2.83	113.43
654	True	172.53	103.99	103.55	4.500000	3.10	113.03
655	True	171.97	103.69	103.54	4.160607	2.70	112.79
656	True	172.09	104.06	103.90	3.970000	3.32	113.09
657	True	172.07	103.97	103.84	4.050000	3.12	113.18
658	True	171.82	103.77	103.79	4.360000	2.77	113.79
659	True	172.19	104.05	103.81	3.900000	3.22	113.52
660	True	171.84	103.75	103.38	4.080000	2.70	113.72
661	True	172.14	104.01	104.00	3.640000	3.16	113.37
662	True	171.69	104.22	104.62	4.110000	3.25	113.40
663	True	171.83	103.76	103.76	4.400000	2.88	113.84
664	True	171.80	103.78	103.65	3.730000	3.12	113.63
666	True	172.57	104.65	104.44	4.540000	2.99	113.16

667	True	172.38	103.55	103.80	3.970000	2.90	113.30
668	True	171.58	103.65	103.37	3.540000	3.19	113.38
669	True	171.96	103.51	103.75	4.060000	3.33	113.53
670	True	172.14	104.34	104.20	4.630000	3.02	112.47
671	True	172.27	104.29	104.22	3.890000	3.53	113.50
672	True	172.07	103.64	103.67	3.860000	3.20	113.83
673	True	172.19	104.61	103.69	4.000000	3.26	112.91
674	True	171.82	103.78	103.76	3.810000	3.25	113.36
675	True	172.04	103.94	103.76	3.810000	3.24	113.41
676	True	171.60	103.85	103.91	4.094065	2.56	113.27
677	True	171.69	103.90	104.13	4.070000	2.92	113.52
678	True	172.05	103.90	103.76	4.520000	2.71	113.42
679	True	172.15	103.65	103.66	3.600000	3.50	113.24
680	True	171.75	104.16	104.00	4.190000	3.03	113.55
681	True	172.03	103.87	103.40	4.290000	3.01	113.09
682	True	172.49	104.44	103.98	4.080000	3.07	113.16
683	True	172.24	104.51	103.97	4.180000	3.22	113.21
684	True	172.59	104.22	104.01	4.470000	2.95	113.19
685	True	172.13	103.76	103.85	3.650000	3.24	112.92
686	True	172.21	104.28	104.37	4.060000	3.30	113.92
687	True	172.41	104.14	104.06	4.450000	2.94	113.98
688	True	172.02	104.23	104.26	4.920000	2.89	113.49
689	True	172.11	103.67	103.43	4.190000	2.98	113.09
690	True	172.22	103.75	103.49	3.690000	3.17	113.14
691	True	172.33	103.83	103.54	3.980000	3.18	113.31
692	True	171.64	103.58	103.46	3.720000	3.20	113.30
693	True	172.49	103.92	103.91	4.420000	2.84	113.38
694	True	172.00	104.32	104.26	4.530000	3.04	112.93
695	True	171.49	103.77	103.60	4.010000	3.09	112.95
696	True	172.10	103.98	103.86	4.470000	3.06	113.00
697	True	171.81	103.96	103.47	4.000000	3.00	113.10
698	True	171.90	104.00	104.07	4.520000	3.09	113.60
699	True	172.24	103.97	104.11	4.020000	2.84	113.15
700	True	171.71	103.52	103.34	4.120000	3.31	113.15
701	True	172.22	103.96	103.97	4.940000	2.99	113.01
702	True	172.03	103.64	103.59	3.630000	3.04	112.96
703	True	172.11	104.06	104.21	4.420000	3.06	113.18
704	True	172.10	103.85	103.68	4.250000	2.91	113.40
705	True	171.75	103.60	103.40	4.020000	2.84	112.90
706	True	172.46	103.75	104.02	4.260000	3.22	113.63
707	True	172.17	103.31	103.89	4.380000	3.13	113.48
708	True	172.48	104.56	104.03	3.790000	2.97	113.65
709	True	171.99	104.03	103.43	4.080000	2.98	113.28

710	True	172.07	104.21	103.82	4.290000	3.35	113.30
711	True	172.03	103.97	103.86	4.439846	3.07	112.65
712	True	172.43	103.60	104.12	4.570000	3.08	112.95
713	True	172.09	103.68	103.99	3.740000	3.28	113.27
714	True	171.56	103.92	103.50	4.410000	2.66	113.55
715	True	172.03	104.69	103.87	4.170000	2.79	112.85
716	True	172.36	104.11	103.31	4.040000	2.98	113.37
717	True	171.77	103.92	103.90	4.610000	2.95	112.95
718	True	172.43	103.97	103.93	3.990000	3.08	112.66
719	True	172.43	103.58	103.77	3.950000	2.62	112.81
720	True	171.79	103.65	103.61	4.190000	3.06	113.60
721	True	171.86	103.47	103.59	4.040000	2.97	113.22
722	True	171.65	103.95	103.61	4.030000	3.25	113.06
723	True	172.30	104.04	103.93	4.330000	2.92	113.19
724	True	171.99	103.97	103.89	4.220000	3.17	113.12
725	True	172.16	104.43	104.06	4.510000	3.19	112.69
726	True	171.73	103.60	103.34	3.820000	3.15	112.89
727	True	171.79	103.74	103.48	4.600000	2.80	113.35
728	True	172.05	103.72	103.81	4.210000	2.97	113.61
729	True	171.94	104.11	104.16	4.080000	3.35	111.76
731	True	172.17	103.93	103.62	4.060000	3.08	113.10
732	True	171.97	103.69	104.17	4.320000	3.00	112.82
733	True	171.52	103.92	103.66	3.810000	3.15	113.54
734	True	172.10	103.94	103.75	3.660000	3.20	113.78
735	True	171.35	103.70	103.43	3.710000	3.22	113.28
736	True	171.92	103.93	104.06	4.380000	2.97	113.01
737	True	171.69	103.85	103.53	3.860000	3.19	112.68
738	True	172.16	104.39	103.85	3.770000	3.32	112.55
739	True	171.70	103.88	103.56	3.890000	3.03	113.60
740	True	172.07	103.74	103.76	4.470650	3.09	112.41
741	True	171.95	103.84	103.68	3.790000	3.09	112.68
742	True	172.17	103.75	103.29	4.430000	2.88	113.38
743	True	172.14	104.06	103.96	4.341643	3.24	113.07
744	True	172.30	104.58	104.17	4.360000	3.33	112.98
745	True	172.10	103.95	103.72	4.490000	3.07	113.15
746	True	172.16	103.92	103.76	4.350000	2.84	113.21
747	True	172.02	103.73	103.31	4.350000	3.07	113.62
748	True	171.91	104.28	103.92	3.640000	3.36	113.15
749	True	171.99	103.91	103.79	4.050000	3.11	113.67
750	True	171.77	103.73	103.48	4.210000	2.92	113.24
751	True	171.69	104.05	103.73	3.870000	3.21	113.56
752	True	172.67	104.69	104.18	4.310000	2.94	113.59
753	True	171.45	104.21	104.19	3.910000	3.01	113.11

754	True	171.96	103.40	104.29	3.770000	3.24	112.87
755	True	172.34	104.20	103.69	3.320000	3.09	113.44
756	True	171.62	104.12	103.81	4.380000	3.13	112.64
757	True	171.72	103.95	104.11	3.750000	3.13	113.51
758	True	172.44	104.06	103.79	4.280000	3.16	113.31
759	True	171.66	104.12	103.94	4.440000	3.19	113.25
760	True	172.24	104.06	103.53	4.130000	2.89	112.38
761	True	171.54	103.70	103.89	3.880000	3.10	113.06
762	True	172.16	103.93	103.04	4.140000	2.99	113.26
763	True	172.02	104.01	103.57	4.500000	2.77	112.72
764	True	172.41	104.43	103.65	3.970000	3.35	113.71
765	True	171.74	103.59	103.94	3.960000	3.12	113.71
766	True	171.91	103.88	103.79	4.170000	3.29	113.38
767	True	172.46	103.89	103.74	3.570000	3.02	112.70
768	True	172.62	104.00	104.03	4.310000	3.29	113.10
769	True	171.69	103.91	103.26	4.040000	3.22	113.56
770	True	171.99	103.27	103.31	3.780000	3.00	113.51
771	True	171.80	104.17	103.80	4.670000	2.88	113.03
772	True	172.10	104.30	103.74	3.850000	2.76	113.10
773	True	172.12	103.95	104.08	4.500000	3.06	113.18
774	True	172.03	103.93	103.53	4.430000	3.10	113.37
775	True	172.00	103.69	103.91	4.120000	3.33	112.84
776	True	172.53	104.29	103.87	4.170000	2.98	112.64
777	True	171.26	103.87	104.41	4.430000	2.88	113.90
778	True	172.26	104.03	103.98	3.970000	2.97	112.89
779	True	171.92	103.60	103.92	4.330000	3.02	113.16
780	True	172.13	103.34	103.62	3.930000	2.86	113.66
781	True	172.41	103.95	103.79	4.080414	3.13	113.41
782	True	171.93	104.34	103.72	3.780000	2.93	113.52
783	True	172.25	104.02	104.16	3.870000	3.16	113.77
784	True	172.28	104.03	104.14	4.480000	3.07	113.32
785	True	172.51	103.78	103.94	4.240000	3.39	113.13
786	True	172.19	103.42	103.61	4.180000	3.12	112.74
787	True	171.42	103.33	103.54	3.870000	3.39	112.87
788	True	172.25	103.96	103.49	4.500000	3.29	113.29
789	True	171.99	103.67	103.76	3.120000	2.99	113.22
790	True	172.00	103.84	103.72	4.770000	2.87	113.84
791	True	171.55	104.05	104.05	4.290000	2.99	113.37
792	True	171.82	103.67	103.80	4.600000	3.49	113.32
793	True	172.28	103.98	103.98	4.510000	2.92	113.07
794	True	172.47	104.46	103.41	4.400000	2.99	113.36
795	True	171.88	103.84	103.87	4.180000	3.54	113.31
796	True	171.90	103.81	103.80	4.150000	3.01	112.82

797	True	171.64	103.75	104.17	4.170000	2.98	113.14
798	True	171.96	104.18	104.04	4.570000	2.86	113.58
799	True	171.96	103.84	103.62	3.614306	3.01	114.44
800	True	172.50	104.03	103.79	4.270000	2.90	112.72
801	True	171.96	104.09	103.67	3.750000	2.58	113.28
802	True	172.19	103.84	104.62	4.140000	3.21	113.45
803	True	171.35	104.62	103.58	4.320000	2.88	113.42
804	True	171.87	103.27	104.50	3.850000	3.03	113.19
805	True	171.34	103.95	103.29	4.890000	2.76	112.58
806	True	171.38	103.75	103.16	4.320000	3.02	113.25
807	True	171.91	103.99	103.76	3.620000	3.04	113.00
808	True	171.36	103.61	103.96	3.970000	3.22	113.45
809	True	172.16	104.36	104.02	4.110000	2.97	113.55
810	True	172.01	103.51	103.48	4.420000	3.13	113.50
811	True	172.17	104.44	103.31	4.150000	3.31	112.96
812	True	171.56	103.81	103.80	4.240000	3.35	113.80
813	True	171.58	103.34	103.82	3.530000	2.92	113.63
814	True	172.29	103.95	103.99	3.860000	2.83	113.35
815	True	171.87	103.94	103.22	3.710000	3.09	113.19
816	True	171.90	104.22	103.56	3.520000	3.20	113.23
817	True	171.84	103.62	103.93	4.290000	2.95	113.22
818	True	172.21	103.96	103.55	3.910000	2.73	113.36
819	True	172.15	104.13	103.52	4.000000	3.09	112.93
820	True	172.20	104.01	103.74	4.060000	3.17	112.94
821	True	171.88	104.44	103.50	4.530000	3.06	113.52
822	True	172.29	104.04	103.94	3.810000	3.00	113.60
823	True	171.97	103.79	103.56	4.170000	3.11	113.58
824	True	171.86	103.78	103.50	3.880000	3.08	113.62
825	True	172.10	104.09	103.83	4.490000	3.25	113.37
826	True	171.94	104.03	103.49	4.160000	2.87	113.05
827	True	172.21	103.95	103.74	3.790000	2.99	113.51
828	True	172.45	103.94	103.90	4.280000	2.85	113.37
830	True	172.26	104.14	103.73	4.410000	2.96	112.91
831	True	171.49	103.94	103.85	4.320000	3.32	113.67
832	True	172.10	103.57	103.80	4.200000	3.02	113.27
833	True	171.80	104.30	103.58	4.750000	3.00	112.94
834	True	172.22	103.71	103.43	4.130000	3.06	113.42
835	True	172.07	103.61	104.34	4.340000	2.85	113.03
836	True	172.15	104.08	104.10	4.340000	2.78	113.33
837	True	172.52	103.98	103.74	4.660000	3.19	113.44
838	True	172.18	103.60	103.76	3.590000	2.75	112.24
839	True	171.72	103.96	103.19	3.810000	2.83	113.27
840	True	172.09	104.20	103.85	4.870000	3.04	113.24

841	True	172.48	103.71	103.79	3.910000	2.96	112.49
842	True	172.36	103.92	103.08	3.690000	3.10	112.88
844	True	172.61	103.98	103.73	3.870000	3.16	113.11
845	True	171.62	104.14	104.49	4.371811	2.99	113.35
846	True	172.02	104.21	104.05	4.093621	2.90	113.62
847	True	171.83	104.01	103.88	4.180000	3.04	112.69
848	True	172.14	104.07	103.83	3.790000	3.14	113.85
849	True	172.44	104.26	103.56	3.480000	3.12	112.70
850	True	172.02	104.07	104.13	3.720000	2.96	112.56
851	True	172.19	103.77	103.39	4.170000	3.17	113.14
852	True	172.16	103.76	103.72	4.180000	3.02	113.43
853	True	171.92	103.72	103.66	4.410000	2.65	113.40
854	True	171.74	104.00	103.83	3.950000	3.03	114.15
855	True	171.87	103.95	103.52	3.940000	3.02	113.33
856	True	172.26	104.66	103.58	4.360000	2.80	112.44
857	True	171.94	104.52	103.77	4.100000	3.08	112.75
858	True	172.18	104.04	104.12	4.260000	3.34	113.67
859	True	172.30	104.18	104.10	4.520000	3.02	113.02
860	True	171.85	103.85	103.45	3.850000	3.09	113.10
861	True	171.38	103.83	103.99	4.440000	3.12	113.48
862	True	171.86	104.21	103.74	4.430000	2.90	113.65
863	True	172.08	103.78	103.13	3.440000	2.88	113.22
864	True	171.97	104.00	103.37	3.860000	3.21	112.87
865	True	171.68	103.54	104.31	3.900000	3.10	112.91
866	True	172.34	103.94	103.91	4.130000	3.23	112.85
867	True	171.67	103.76	103.59	3.820000	3.17	112.77
868	True	172.12	104.32	103.19	4.040000	3.28	113.11
869	True	172.23	103.87	104.05	3.900000	3.09	113.38
870	True	172.25	104.21	103.79	3.840000	3.25	112.71
871	True	172.14	104.34	103.98	3.840000	2.98	113.05
872	True	171.37	104.07	103.75	4.249629	3.07	113.27
873	True	171.66	104.42	103.50	3.760000	3.20	112.94
874	True	172.29	103.93	103.82	4.250000	3.11	113.54
875	True	172.22	103.75	103.91	4.500000	3.19	113.06
876	True	171.94	104.09	103.28	3.980000	3.13	113.53
877	True	172.18	103.85	103.67	3.830000	2.96	113.09
878	True	171.95	103.62	103.91	4.910000	2.90	113.17
879	True	172.27	104.21	103.59	3.900000	3.07	113.31
880	True	172.34	104.14	104.06	4.110000	3.01	113.78
881	True	171.71	104.31	104.10	4.540000	3.12	113.11
882	True	171.54	104.36	103.83	3.860000	3.06	112.92
883	True	172.16	103.77	103.63	4.060000	3.12	113.02
884	True	171.73	104.38	103.26	4.000000	3.04	112.72

885	True	172.57	103.78	103.65	4.270000	3.04	113.00
886	True	171.71	104.14	103.99	4.490000	3.15	113.21
887	True	172.00	104.35	103.29	3.730000	3.11	113.18
888	True	172.08	104.04	104.08	3.660000	3.30	113.41
889	True	172.15	104.28	103.67	4.280000	3.24	113.35
890	True	172.31	103.94	103.89	4.000000	3.20	113.80
891	True	172.35	104.04	104.18	4.470000	3.03	113.15
892	True	172.16	103.63	103.59	4.250000	3.38	113.58
893	True	172.63	104.13	104.17	3.770000	3.45	113.34
894	True	172.20	103.52	103.78	3.660000	3.40	113.35
895	True	172.02	103.67	104.22	4.400000	3.04	113.33
896	True	171.81	103.68	103.80	3.893748	2.98	113.82
897	True	171.66	104.00	103.47	3.900000	2.99	112.65
898	True	172.00	103.65	103.83	2.980000	2.76	113.17
899	True	171.95	103.65	103.90	3.980000	3.20	112.68
900	True	172.39	103.98	104.05	4.260000	2.98	113.41
901	True	171.65	103.84	104.25	3.560000	3.22	113.63
902	True	171.83	104.08	103.70	3.280000	2.88	113.56
903	True	171.50	104.27	104.04	3.970000	3.37	112.98
904	True	171.41	104.03	104.21	4.260000	3.11	113.49
905	True	172.16	104.10	103.78	3.840000	3.03	113.13
906	True	171.99	104.76	104.55	4.390000	3.02	113.70
907	True	172.32	103.97	104.33	4.060000	3.08	113.73
908	True	171.99	104.44	103.88	4.040000	2.89	113.01
909	True	172.22	104.05	103.86	4.680000	3.05	113.04
910	True	171.62	103.80	103.48	4.160000	3.35	112.83
911	True	171.84	104.03	103.88	4.310000	3.18	113.07
912	True	172.05	103.75	104.15	4.390000	2.87	112.83
913	True	172.25	103.93	103.49	4.170000	2.80	113.56
914	True	171.98	103.75	103.83	4.390000	3.32	112.97
915	True	171.65	103.82	103.37	4.370000	2.98	113.04
916	True	172.37	104.11	104.09	4.780000	3.09	113.42
917	True	171.64	103.90	104.07	3.600000	3.39	113.18
918	True	171.99	103.62	103.42	3.480000	3.37	112.94
919	True	172.48	103.59	104.03	4.100000	2.74	112.78
920	True	171.92	103.68	103.45	3.746333	2.58	113.68
921	True	171.85	103.27	103.96	4.000000	2.65	113.62
922	True	171.36	103.72	104.76	4.170000	2.88	113.14
923	True	172.24	103.95	103.52	4.330000	2.88	113.48
924	True	171.49	104.24	104.32	3.580000	3.07	113.80
925	True	172.45	103.99	104.39	3.430000	2.66	112.93
926	True	171.82	103.61	103.84	3.990000	2.73	113.40
927	True	172.15	103.98	103.89	4.210000	3.08	113.61

928	True	172.18	104.65	103.81	3.950000	3.07	113.03
929	True	172.39	104.08	103.76	3.850000	2.95	113.23
930	True	172.13	104.10	103.69	4.390000	2.86	113.06
931	True	172.36	103.52	103.70	4.170000	3.29	113.33
932	True	171.83	104.01	103.92	4.350000	3.08	112.47
933	True	171.77	104.27	103.94	4.340000	2.87	113.21
934	True	172.05	103.78	103.81	4.170000	3.08	113.35
935	True	172.17	104.27	104.32	4.520000	3.32	113.48
936	True	171.52	103.85	103.75	3.950000	3.09	112.97
937	True	172.17	104.06	103.37	4.320000	3.31	113.66
938	True	171.64	103.60	103.76	4.420000	2.98	112.94
939	True	172.19	104.08	104.17	4.170000	3.26	113.07
940	True	171.94	104.64	103.70	4.360000	2.98	112.83
941	True	171.90	104.11	103.69	3.680000	3.08	112.74
942	True	172.13	103.35	104.17	4.230000	3.27	113.06
943	True	172.10	104.17	103.78	4.610000	2.84	113.35
944	True	171.79	103.56	103.77	4.150000	2.80	112.67
945	True	171.66	103.68	103.41	4.350000	3.33	113.74
946	True	172.09	103.74	103.52	4.237415	3.02	112.78
947	True	171.63	103.87	104.66	4.710533	3.27	112.68
948	True	171.90	104.08	103.94	4.170000	2.97	113.05
949	True	171.84	104.39	103.83	3.770000	2.97	113.05
950	True	172.02	103.89	103.93	4.140000	3.18	113.75
951	True	172.40	104.09	103.74	3.740000	3.13	113.46
952	True	171.95	103.86	103.80	4.510000	2.87	112.17
953	True	172.39	104.20	103.88	3.950000	3.36	113.07
954	True	171.66	103.54	104.16	4.000000	3.33	113.53
955	True	171.77	103.63	103.59	3.520000	3.32	112.77
956	True	172.37	104.06	103.69	4.090000	2.87	113.68
957	True	172.22	103.72	103.80	4.360000	3.22	112.84
958	True	172.24	103.80	104.36	4.180000	2.99	113.81
959	True	172.15	103.84	103.32	4.210000	3.03	112.89
960	True	172.31	103.98	103.53	4.040000	2.97	113.47
961	True	171.87	103.88	104.07	3.800000	2.90	113.36
962	True	171.79	103.77	104.00	4.100000	3.05	112.67
963	True	172.12	103.64	103.65	4.230000	3.18	112.67
964	True	172.09	103.65	103.25	3.960000	2.85	113.20
965	True	171.84	103.72	103.57	4.330000	2.83	113.31
966	True	171.42	104.79	104.16	3.450000	3.16	112.64
967	True	171.92	104.57	104.17	4.090000	2.88	113.12
968	True	172.36	104.09	103.75	3.980000	2.93	112.70
969	True	172.06	103.56	103.76	3.830000	3.05	113.42
970	True	171.75	103.60	103.06	4.030000	2.98	112.96

971	True	172.06	104.45	103.73	4.180000	3.50	113.36
972	True	171.57	103.58	104.06	3.860000	2.94	113.03
973	True	171.40	103.88	103.54	3.750000	2.80	113.01
974	True	172.41	104.10	103.90	4.060000	3.14	112.61
975	True	172.06	104.08	103.47	4.470000	2.97	114.32
976	True	172.43	104.04	104.26	4.560000	3.24	112.82
977	True	172.31	104.29	103.72	3.980000	2.87	112.40
978	True	171.62	103.94	104.14	4.590000	3.30	113.50
979	True	172.43	103.65	104.30	4.090000	2.94	113.06
980	True	172.00	103.76	104.07	4.360000	2.59	113.64
981	True	172.12	104.08	104.49	4.090000	3.56	113.50
982	True	172.02	104.23	103.72	4.137780	2.99	113.37
983	True	171.43	104.52	103.42	4.050000	2.82	113.02
984	True	172.33	103.69	103.66	3.760000	3.09	113.66
985	True	171.63	103.70	103.63	4.210000	3.25	112.76
986	True	171.88	103.87	104.02	4.120000	3.33	112.47
987	True	171.92	103.61	103.83	4.080000	2.72	112.40
988	True	171.83	103.77	103.84	4.100000	3.10	113.10
989	True	172.25	103.99	103.84	3.810000	2.99	113.36
990	True	171.68	103.90	103.68	3.780000	2.74	113.60
991	True	171.74	103.90	103.60	3.870000	2.89	113.36
992	True	172.21	104.42	103.53	4.480000	3.05	113.11
993	True	171.79	103.69	103.68	3.650000	2.77	112.78
994	True	171.91	103.73	103.51	4.600000	3.10	113.53
995	True	172.11	104.12	103.83	3.900000	2.72	113.28
996	True	171.66	103.92	103.47	4.260000	2.83	113.20
997	True	172.16	103.72	103.61	4.300000	2.72	113.51
998	True	171.78	103.38	104.22	4.230000	3.07	113.77
999	True	171.44	103.96	103.92	3.680000	2.89	113.21
1000	True	171.56	104.07	103.58	3.550000	3.02	112.96
1001	False	172.28	103.95	103.91	4.780000	3.31	111.40
1002	False	171.92	103.86	104.30	4.960000	3.13	111.29
1003	False	171.59	104.14	104.38	4.970000	3.47	111.22
1004	False	172.02	104.33	104.33	5.190000	3.21	111.99
1005	False	172.55	104.25	104.23	5.600000	3.13	111.72
1006	False	171.88	104.30	104.18	5.340000	3.33	112.69
1007	False	171.63	104.05	104.25	4.610000	3.10	110.91
1008	False	171.83	104.13	104.52	4.940000	3.27	111.72
1009	False	172.30	104.28	103.90	5.100000	3.57	110.66
1010	False	172.09	104.26	103.96	4.450000	3.31	111.25
1011	False	171.43	103.85	103.94	4.370000	3.23	111.71
1012	False	172.09	104.44	104.24	4.900000	3.13	111.93
1013	False	171.37	103.73	104.18	4.970000	3.34	110.98

1014	False	171.98	104.18	103.59	5.440000	3.39	112.07
1015	False	171.73	103.99	103.86	5.550000	3.33	111.28
1016	False	171.86	104.51	104.29	5.970000	3.54	110.76
1017	False	171.99	104.23	104.33	5.130000	3.21	110.99
1018	False	171.81	104.37	104.18	5.900000	3.52	110.95
1019	False	171.79	104.18	103.87	5.550000	3.25	111.88
1020	False	172.16	104.03	104.44	5.330000	3.33	111.21
1021	False	171.14	104.31	104.30	5.460000	3.29	111.57
1022	False	171.91	104.20	103.57	4.530000	3.52	111.14
1024	False	172.02	104.26	104.20	6.200000	3.58	111.25
1025	False	171.52	104.05	104.31	4.240000	3.70	112.60
1026	False	172.17	104.20	104.13	3.860000	3.38	112.44
1027	False	171.88	103.85	103.96	5.100000	3.31	112.64
1029	False	171.74	104.05	103.93	5.380000	3.49	111.34
1031	False	171.96	104.45	104.48	5.310000	3.42	111.75
1032	False	172.40	104.00	103.82	6.330000	3.10	112.11
1033	False	172.08	104.22	104.41	4.510000	3.30	111.40
1034	False	171.72	104.06	103.80	5.690000	3.64	111.77
1035	False	172.35	103.79	104.35	5.180000	3.35	112.02
1036	False	172.05	104.37	104.14	5.380000	3.27	111.14
1037	False	171.58	104.13	104.67	5.860000	3.52	112.47
1038	False	172.09	104.24	103.83	5.100000	3.37	111.69
1039	False	172.08	104.10	103.96	5.640000	3.13	111.51
1040	False	171.65	104.12	104.33	5.560000	3.19	111.48
1041	False	171.91	104.08	104.16	5.910000	3.41	111.32
1043	False	171.75	103.99	104.21	4.740000	3.40	110.50
1044	False	172.05	104.29	103.80	4.940000	3.27	111.33
1045	False	171.55	104.18	103.83	4.460000	3.42	111.60
1046	False	172.40	104.05	103.94	5.010000	3.43	111.53
1047	False	172.07	104.36	104.04	5.600000	3.29	111.73
1048	False	171.78	103.95	104.27	4.640000	3.29	110.73
1049	False	172.16	104.24	104.47	6.080000	3.49	111.59
1050	False	171.69	104.09	104.49	5.280000	3.51	112.68
1051	False	171.96	104.24	103.75	5.500000	3.29	111.08
1052	False	171.81	103.91	104.77	5.280000	3.58	111.33
1053	False	172.13	104.09	104.16	5.450000	3.38	111.79
1054	False	171.85	104.52	104.05	6.210000	3.43	111.96
1055	False	172.39	104.16	103.92	5.660000	3.39	111.83
1056	False	171.74	104.07	104.08	5.060000	3.52	111.76
1057	False	171.50	104.32	104.66	4.920000	3.33	111.95
1058	False	172.02	104.43	104.37	5.590000	3.06	110.35
1059	False	171.98	104.33	104.28	4.860000	3.45	111.55
1060	False	172.15	104.41	104.40	5.640000	3.25	111.05

1061	False	171.75	104.34	104.58	4.260000	3.29	110.98
1062	False	172.02	104.06	104.16	4.740000	3.35	110.94
1063	False	171.65	104.19	103.96	5.370000	2.99	112.86
1064	False	171.44	103.99	104.04	4.810000	3.49	111.74
1065	False	171.87	104.16	104.44	6.040000	3.32	111.99
1066	False	171.87	104.21	103.82	4.810000	3.12	112.06
1067	False	171.92	104.39	103.93	5.180000	3.02	111.44
1068	False	171.37	104.08	104.16	5.550000	3.55	111.56
1069	False	171.89	104.49	104.38	5.200000	3.63	110.48
1070	False	171.21	104.29	104.41	5.140000	3.26	111.16
1071	False	172.08	104.38	104.22	5.510000	3.41	111.42
1072	False	171.50	103.90	103.98	4.820000	3.40	112.07
1073	False	171.41	104.06	104.42	5.040000	3.40	111.02
1074	False	172.13	103.67	103.82	4.270000	3.22	112.15
1075	False	172.14	104.49	103.76	4.360000	3.24	110.36
1076	False	172.02	104.51	103.69	6.230000	3.39	112.35
1077	False	171.57	104.27	104.44	5.050277	3.21	111.87
1078	False	171.80	104.12	104.06	5.660000	3.29	111.09
1079	False	172.21	104.40	104.28	5.280000	3.42	112.99
1080	False	172.17	104.33	104.24	4.930000	3.43	112.06
1081	False	172.23	103.83	104.29	5.610000	3.47	111.72
1082	False	171.82	103.90	103.67	4.790000	3.36	112.43
1084	False	171.85	103.60	103.82	4.600000	3.21	112.50
1085	False	171.94	104.27	103.94	4.460000	3.14	112.13
1086	False	171.39	103.51	103.50	5.520000	3.45	111.54
1087	False	171.54	104.53	104.15	5.690000	3.43	111.70
1088	False	171.91	104.28	103.86	5.480000	3.37	113.13
1089	False	171.91	104.26	104.62	5.860000	3.44	110.68
1090	False	171.13	104.13	104.26	3.860000	3.69	111.52
1091	False	172.11	104.88	104.10	4.800000	3.73	110.78
1092	False	172.20	104.31	104.40	4.600000	3.42	111.92
1093	False	171.87	104.76	104.02	6.300000	3.61	111.29
1094	False	172.08	104.33	103.94	4.770000	3.64	110.68
1095	False	172.13	104.40	103.43	4.440000	3.65	111.17
1096	False	171.83	104.11	103.92	5.630000	3.21	112.18
1097	False	172.34	104.36	103.82	4.920000	3.47	111.33
1098	False	172.51	104.43	104.17	5.110000	3.08	111.70
1099	False	171.32	104.28	104.36	5.080000	3.69	112.09
1100	False	171.19	104.18	104.01	4.820000	3.18	111.73
1101	False	172.29	104.09	103.92	4.510000	3.54	110.94
1102	False	171.86	104.49	103.87	5.210000	3.60	111.11
1103	False	172.31	104.11	103.74	4.960000	3.60	111.78
1104	False	171.88	104.05	103.75	4.410000	3.21	112.52

1105	False	171.95	103.89	103.69	4.450000	3.42	112.02
1106	False	172.37	104.03	104.47	4.950000	3.25	111.99
1107	False	172.08	103.95	103.90	5.450000	3.29	112.03
1108	False	172.12	104.18	104.31	5.360000	3.31	111.43
1109	False	171.78	104.24	103.94	4.640000	3.11	111.31
1110	False	172.03	104.16	104.31	5.460000	3.24	112.82
1112	False	172.03	104.52	103.96	4.300000	3.48	112.16
1113	False	172.06	104.28	104.31	5.610000	3.27	111.61
1114	False	171.84	104.52	104.19	4.600000	3.47	110.64
1115	False	172.10	104.10	104.20	5.160000	3.16	111.85
1116	False	172.18	104.14	104.37	6.040000	3.62	110.97
1117	False	172.26	103.90	104.12	4.990000	3.42	111.27
1118	False	171.94	104.14	104.14	4.690000	3.06	112.04
1119	False	171.38	104.52	104.22	4.580000	3.41	111.84
1120	False	172.00	104.20	104.35	5.180000	3.31	111.95
1121	False	171.48	104.46	104.24	4.940000	3.61	110.80
1122	False	171.40	104.38	104.19	4.802145	3.17	112.39
1123	False	172.09	104.15	104.17	4.150000	3.40	113.85
1124	False	172.07	104.03	103.58	5.330000	3.64	111.27
1126	False	172.04	104.26	103.87	5.210000	3.49	112.09
1127	False	171.84	104.06	104.10	4.870000	3.25	111.04
1128	False	172.04	103.76	104.03	5.070000	3.42	111.87
1129	False	172.37	104.25	103.67	5.120000	3.30	112.03
1130	False	171.68	104.04	104.68	5.320000	3.56	111.04
1131	False	171.56	104.29	104.19	5.230000	3.63	112.94
1132	False	171.77	104.34	103.44	5.120000	3.36	111.49
1133	False	172.04	104.32	103.81	5.570000	3.57	112.37
1134	False	171.79	103.99	103.67	6.160000	3.52	110.93
1135	False	171.91	103.70	104.41	6.340000	3.50	113.05
1136	False	171.81	104.17	103.93	4.770000	3.34	110.80
1137	False	172.41	104.32	103.93	5.820000	3.36	112.36
1138	False	171.77	103.98	103.76	5.160000	3.38	111.07
1139	False	172.01	104.06	104.12	5.710000	3.35	111.01
1140	False	171.32	104.20	104.09	5.670000	3.19	112.02
1141	False	172.15	104.26	104.20	5.500000	3.49	112.33
1142	False	171.98	103.81	104.35	4.570000	3.52	111.73
1144	False	171.90	104.25	104.64	4.460000	3.07	110.86
1145	False	171.88	103.84	104.18	5.020000	3.05	112.07
1146	False	171.79	103.81	103.97	5.860000	3.20	111.04
1147	False	171.94	104.53	103.81	5.070000	3.14	112.46
1148	False	172.43	104.06	104.02	4.930000	3.33	111.59
1149	False	171.84	104.63	104.43	5.240000	3.21	111.48
1150	False	172.02	104.08	104.01	4.630000	3.28	111.03

1151	False	171.79	104.86	104.34	5.390000	3.14	113.02
1152	False	171.97	104.19	104.05	4.790000	3.18	111.46
1153	False	171.83	103.83	103.92	5.340000	3.22	112.03
1154	False	172.42	104.33	104.40	5.340000	3.13	112.11
1155	False	172.32	103.95	103.94	4.230000	3.31	111.41
1156	False	172.36	104.04	104.31	4.760000	3.56	111.57
1157	False	171.78	104.33	103.58	6.010000	3.26	111.30
1158	False	172.09	103.70	104.27	5.530000	3.51	112.20
1159	False	172.11	104.10	104.18	4.520000	3.21	111.45
1160	False	171.47	104.33	104.03	5.250000	3.01	111.68
1161	False	172.39	104.05	104.32	4.130000	3.41	112.66
1162	False	171.92	104.29	104.27	5.760000	3.13	111.37
1163	False	172.46	103.87	104.36	5.560000	3.33	110.69
1164	False	172.34	104.36	103.83	6.030000	3.44	111.44
1165	False	171.50	104.52	104.19	4.840000	3.12	110.37
1166	False	171.50	104.11	104.14	5.380000	3.53	111.24
1167	False	171.89	104.23	104.52	4.600000	3.15	110.94
1168	False	171.84	104.21	104.07	5.850000	3.50	111.16
1169	False	171.83	103.56	103.76	5.560000	3.49	110.70
1171	False	171.63	104.30	103.85	5.450000	3.26	111.88
1172	False	171.86	103.94	103.81	5.120000	3.23	111.55
1173	False	171.87	104.07	103.54	4.590000	3.50	112.06
1174	False	172.12	104.38	104.31	4.820000	3.18	111.82
1175	False	171.84	104.17	104.12	4.420000	3.57	112.01
1176	False	172.13	104.06	103.92	5.540000	3.13	111.01
1177	False	171.59	104.05	103.94	5.067584	3.02	111.29
1178	False	172.07	103.91	104.21	4.120000	3.42	111.91
1179	False	172.21	104.50	104.46	4.920000	3.34	112.12
1180	False	171.47	104.28	103.89	5.780000	3.21	111.80
1181	False	171.68	104.00	104.29	4.900000	3.10	111.77
1182	False	171.26	104.02	103.93	5.600000	3.12	112.48
1183	False	172.05	104.15	104.56	5.350000	3.56	111.20
1184	False	172.25	103.96	103.92	4.670000	3.24	111.06
1185	False	171.97	103.95	103.73	5.150000	3.24	110.87
1186	False	171.38	104.39	104.56	5.960000	3.25	111.59
1187	False	171.56	104.34	104.43	5.920000	3.66	111.87
1188	False	171.52	104.18	103.99	5.930000	3.26	110.87
1189	False	171.74	103.77	104.31	4.990000	3.40	111.52
1190	False	172.18	104.53	104.14	5.970000	3.18	111.11
1191	False	171.45	104.21	104.18	4.550000	3.52	113.21
1192	False	171.42	103.87	104.51	5.070000	3.30	110.22
1193	False	171.95	104.08	104.08	5.660000	3.32	110.93
1194	False	171.59	103.93	103.95	5.250000	3.38	111.71

1195	False	171.67	104.06	104.15	4.870000	3.21	111.49
1196	False	172.61	104.31	104.13	5.850000	3.27	111.63
1197	False	171.94	104.04	104.27	4.750000	3.16	111.61
1198	False	171.94	104.20	103.65	5.000000	3.43	112.08
1199	False	172.05	104.20	104.11	4.090000	3.37	111.18
1200	False	172.03	104.32	104.87	4.490000	3.77	111.04
1201	False	171.99	104.04	104.25	4.770000	3.43	111.83
1202	False	171.85	104.12	103.69	4.860000	3.50	111.55
1203	False	172.26	104.23	103.98	6.030000	3.29	111.83
1204	False	172.02	104.22	104.19	5.140000	3.73	110.49
1205	False	171.96	104.01	104.54	5.210000	3.31	111.35
1206	False	171.92	103.90	104.58	5.560000	3.50	110.59
1207	False	171.73	104.09	104.47	4.810000	3.46	112.30
1208	False	171.71	104.12	104.14	4.840000	3.38	111.68
1209	False	171.96	104.38	103.82	5.190000	3.32	111.68
1210	False	172.25	104.18	104.23	5.110000	3.37	111.17
1211	False	172.29	104.43	104.80	4.860000	3.30	111.53
1212	False	172.31	103.98	104.09	5.280000	3.47	111.71
1213	False	171.90	104.33	104.40	4.700000	3.31	112.12
1214	False	171.76	104.13	104.19	5.220000	3.55	111.90
1215	False	171.87	104.26	103.99	4.870000	3.60	110.48
1216	False	171.88	104.14	104.42	4.340000	3.18	111.38
1217	False	171.93	104.09	104.51	4.870000	3.58	111.63
1218	False	171.63	104.61	104.29	5.310000	3.14	111.51
1219	False	171.68	104.67	103.71	5.260000	3.22	111.32
1220	False	172.23	103.81	104.13	5.240000	3.17	110.91
1221	False	171.91	104.13	104.24	5.280000	3.19	112.31
1222	False	172.27	104.06	104.24	5.430000	3.35	110.29
1223	False	171.69	104.51	103.81	5.020000	3.42	112.08
1224	False	172.24	104.31	104.66	4.200000	3.61	112.37
1225	False	171.74	104.52	104.23	5.590000	3.61	112.14
1226	False	172.03	104.15	104.24	5.120000	3.30	112.36
1227	False	171.60	103.90	104.29	6.110000	3.55	110.77
1228	False	172.26	103.94	103.95	5.700000	3.26	112.61
1229	False	172.27	104.36	103.68	5.740000	3.51	112.35
1230	False	171.94	103.88	104.10	6.050000	3.39	111.52
1231	False	171.92	104.43	103.77	5.790000	3.11	111.03
1232	False	171.75	103.96	104.15	4.840000	3.32	111.76
1233	False	171.90	103.94	104.59	5.040000	3.23	111.57
1234	False	172.13	104.35	104.00	5.690000	3.30	111.92
1235	False	171.95	104.00	103.88	5.800000	2.96	111.79
1236	False	172.27	103.87	104.35	5.580000	3.52	112.32
1237	False	172.57	104.13	104.19	5.100000	3.45	112.18

1238	False	171.72	104.05	104.36	5.400000	3.75	111.59
1239	False	172.27	103.88	104.16	5.910000	3.31	110.91
1240	False	171.99	104.02	104.06	5.340000	3.61	112.12
1241	False	172.16	104.12	104.21	5.010000	3.43	110.99
1242	False	171.66	104.17	104.02	4.710000	3.22	110.91
1243	False	171.58	104.23	104.50	5.140000	3.70	111.75
1244	False	171.64	104.20	104.10	5.430000	3.60	112.72
1245	False	171.91	103.80	103.99	6.140000	3.32	110.78
1246	False	172.26	104.04	104.43	3.820000	3.47	111.95
1247	False	171.76	104.14	104.54	5.140000	3.41	111.50
1248	False	171.57	104.04	104.14	5.270000	3.42	112.68
1249	False	172.31	103.97	104.03	4.600000	3.14	111.17
1250	False	171.45	104.03	104.26	4.880000	3.44	111.92
1251	False	172.27	104.18	103.78	5.580000	3.25	112.38
1252	False	172.25	104.21	103.98	5.710000	3.50	111.88
1253	False	172.03	104.60	104.00	4.360000	3.26	112.08
1254	False	171.80	104.51	104.57	5.500000	3.30	112.94
1256	False	171.90	104.22	104.09	5.280000	3.18	112.16
1257	False	172.00	104.23	104.52	5.760000	3.37	110.94
1258	False	172.39	104.34	104.10	5.890000	3.14	111.56
1259	False	172.42	103.69	104.29	5.250000	3.33	110.70
1260	False	172.20	104.45	104.02	4.810000	3.58	111.20
1261	False	171.88	103.89	104.41	4.750000	3.59	112.10
1262	False	171.56	104.03	103.93	5.470000	3.50	111.67
1263	False	171.93	104.43	104.15	5.800000	3.40	111.82
1264	False	171.59	104.05	104.40	5.050000	3.45	111.14
1265	False	171.92	104.33	104.06	4.450000	3.41	111.46
1266	False	172.37	104.54	104.03	5.820000	3.41	111.45
1267	False	171.67	104.12	103.98	5.680000	3.18	111.55
1268	False	172.13	103.99	103.78	5.470000	2.94	112.30
1269	False	171.94	104.38	103.81	4.870000	3.39	111.22
1270	False	172.04	104.37	104.09	4.810000	3.50	111.25
1271	False	171.26	104.22	104.07	4.780000	3.81	112.88
1272	False	172.40	104.42	104.18	4.620000	3.31	111.27
1273	False	171.73	103.74	104.38	5.140000	3.16	111.73
1274	False	172.33	104.36	104.31	4.400000	3.33	112.03
1275	False	171.96	104.00	103.95	5.630000	3.26	110.96
1276	False	171.82	103.97	103.88	4.730000	3.55	111.87
1277	False	172.47	104.27	104.10	4.880000	3.33	110.68
1279	False	172.11	104.23	104.45	5.240000	3.58	111.78
1280	False	171.43	104.26	103.97	5.730000	3.14	111.82
1281	False	171.74	104.02	104.22	5.330000	2.94	112.26
1282	False	171.62	104.53	103.64	4.890000	3.16	112.54

1283	False	171.80	104.14	104.26	5.670000	3.35	111.46
1284	False	171.60	104.23	104.45	5.570000	3.45	110.34
1285	False	172.10	104.16	104.34	6.000000	3.09	111.62
1286	False	172.12	104.38	103.93	4.710000	3.44	111.52
1287	False	171.93	104.24	103.71	5.480000	3.16	111.84
1288	False	171.31	103.95	104.07	5.140000	3.41	112.44
1289	False	172.06	104.00	104.02	5.090000	3.53	112.07
1290	False	171.57	104.08	104.12	5.460000	3.21	111.49
1292	False	171.84	104.46	104.15	4.510000	3.39	110.73
1293	False	171.86	104.04	104.43	4.630000	3.64	111.63
1294	False	171.57	104.29	104.28	5.120000	3.68	110.98
1295	False	172.40	104.27	104.18	4.920000	3.17	111.79
1296	False	171.90	103.97	104.36	5.590000	3.61	112.05
1297	False	171.55	104.46	104.36	4.690000	3.31	111.57
1298	False	171.96	104.47	104.06	4.840000	3.76	111.04
1299	False	172.00	104.36	104.16	4.940000	3.25	111.82
1300	False	172.11	104.24	104.06	5.610000	3.60	111.76
1301	False	171.38	104.13	104.22	5.670000	3.46	112.48
1302	False	171.33	104.28	104.19	5.720000	3.25	111.99
1303	False	172.08	103.98	104.23	4.990000	3.31	112.19
1304	False	172.17	104.49	103.76	5.047570	2.93	111.21
1305	False	171.91	104.00	104.49	4.680000	3.42	111.82
1306	False	172.10	104.36	104.08	4.960000	3.46	111.93
1307	False	172.13	103.99	104.06	5.800000	3.28	111.30
1308	False	171.68	104.24	104.00	4.490000	3.24	112.18
1309	False	172.36	104.02	104.32	5.620000	3.40	112.13
1310	False	172.00	104.45	104.19	5.500000	2.99	111.99
1311	False	171.92	104.84	104.60	5.960000	2.92	110.69
1312	False	171.78	104.07	104.16	5.770000	3.30	111.27
1313	False	171.92	104.37	104.05	4.950000	3.04	110.61
1314	False	171.83	104.12	104.01	5.590000	3.20	112.53
1315	False	172.43	104.32	103.95	4.130000	3.39	111.50
1316	False	172.08	104.15	104.17	4.778967	3.40	112.29
1317	False	172.40	104.55	104.22	5.180000	3.51	111.94
1318	False	171.99	104.28	104.32	4.710000	3.45	112.18
1319	False	171.91	103.99	104.23	5.010000	3.42	111.77
1320	False	171.74	104.40	104.39	4.870000	3.06	112.00
1321	False	172.05	104.60	104.32	5.120000	3.35	111.78
1322	False	172.29	104.72	104.86	5.710000	3.16	112.15
1323	False	172.07	104.50	104.23	6.190000	3.07	111.21
1324	False	171.84	104.23	104.31	5.100000	3.68	111.72
1325	False	171.51	104.13	103.90	4.990000	3.60	111.23
1326	False	171.83	104.39	104.17	5.510000	3.33	113.64

1327	False	171.75	104.36	104.02	6.000000	3.13	111.79
1328	False	171.78	104.51	104.06	5.900000	3.18	111.91
1329	False	172.10	104.22	103.99	5.260000	3.24	111.94
1330	False	172.50	104.07	103.71	3.820000	3.63	110.74
1331	False	171.89	104.32	103.94	5.640000	3.30	111.56
1332	False	172.32	104.60	104.83	4.840000	3.51	112.55
1333	False	172.09	104.40	104.21	5.280000	3.41	112.11
1334	False	172.10	104.30	104.21	4.070000	3.41	111.27
1335	False	172.22	104.41	104.64	5.200000	3.37	112.20
1336	False	172.04	104.17	103.90	5.050000	3.62	111.56
1337	False	171.79	104.05	104.30	5.020000	3.44	112.01
1338	False	172.22	104.17	104.07	4.520000	3.67	112.13
1339	False	171.99	104.18	104.20	5.260000	3.23	111.83
1340	False	171.75	104.16	104.23	5.750000	3.25	111.68
1341	False	172.04	104.34	104.48	4.880000	3.28	112.15
1342	False	171.94	104.21	104.10	4.280000	3.47	112.23
1343	False	171.97	104.38	104.18	5.590000	3.47	110.98
1344	False	171.62	104.14	104.45	4.940000	3.66	111.93
1345	False	171.43	104.14	103.95	5.340000	3.14	111.76
1346	False	171.56	104.17	103.87	6.160000	3.38	111.55
1347	False	171.94	104.37	104.14	5.370000	3.46	111.94
1348	False	171.72	104.46	104.12	5.726993	3.61	110.31
1349	False	171.84	104.32	104.50	6.280000	3.00	111.06
1350	False	171.53	104.03	104.05	5.770000	3.22	111.93
1351	False	171.38	103.78	103.70	5.220000	3.43	111.60
1352	False	172.10	103.98	104.28	5.780000	3.16	111.09
1353	False	171.62	104.21	103.99	5.500000	3.45	111.35
1354	False	171.61	104.04	104.06	6.190000	3.08	110.73
1355	False	171.67	103.79	103.44	5.130000	3.32	111.47
1357	False	171.95	104.26	103.97	5.880000	3.16	112.44
1358	False	171.91	103.91	103.98	4.780000	3.65	111.41
1359	False	171.67	104.16	104.08	5.420000	3.30	111.63
1360	False	172.25	104.52	104.22	4.650000	3.43	110.48
1361	False	171.98	104.44	104.26	5.750000	3.20	110.93
1362	False	171.95	104.47	104.34	5.920000	3.10	113.17
1363	False	171.56	103.80	103.87	5.660000	2.98	112.95
1364	False	172.00	104.46	104.30	5.270000	3.37	111.85
1365	False	171.69	104.18	104.28	5.620000	3.23	110.53
1366	False	171.74	103.96	103.47	5.140000	3.30	111.40
1367	False	171.83	104.18	104.26	5.000000	3.60	111.55
1368	False	171.60	104.37	104.20	5.820000	3.08	112.84
1369	False	171.65	104.32	104.38	5.650000	3.24	112.30
1370	False	171.94	104.56	104.25	4.600000	3.37	110.64

1371	False	171.69	103.87	104.16	5.460000	3.31	111.42
1372	False	171.86	104.12	104.10	6.010000	3.34	111.91
1373	False	171.38	104.04	104.20	5.540000	3.38	112.80
1374	False	171.69	104.17	104.37	5.310000	3.54	111.89
1375	False	172.02	103.90	104.10	5.920000	3.19	111.46
1376	False	171.74	104.41	104.08	4.660000	3.22	111.04
1377	False	171.80	104.31	104.49	5.970000	3.51	111.20
1378	False	172.30	104.18	104.54	5.040000	3.35	111.11
1379	False	171.41	104.20	103.73	4.310000	3.32	111.74
1380	False	172.15	104.23	104.41	5.470000	3.37	112.53
1381	False	171.58	103.87	104.07	4.670000	3.51	111.53
1382	False	172.08	104.04	103.44	5.390000	3.19	111.04
1383	False	171.97	104.85	104.52	5.870000	3.56	110.98
1384	False	171.75	104.16	104.16	4.230000	2.99	111.83
1385	False	171.26	104.19	104.26	5.170000	3.35	111.04
1386	False	171.46	104.40	104.30	5.200000	3.30	111.58
1387	False	172.05	104.22	104.05	5.870000	3.18	111.13
1388	False	171.80	103.97	103.93	5.390000	3.07	111.22
1389	False	171.05	104.09	104.50	4.720000	3.10	112.44
1390	False	172.31	104.31	104.72	4.860000	3.41	112.06
1391	False	171.43	104.26	103.91	5.140000	3.53	111.90
1392	False	172.36	104.01	104.41	5.230000	3.07	110.71
1393	False	171.59	104.17	104.09	5.250000	3.08	112.14
1394	False	171.67	104.08	104.73	4.930000	3.31	111.81
1395	False	172.00	104.17	104.44	4.720000	3.40	111.17
1396	False	171.50	104.28	104.22	5.200000	3.24	111.56
1397	False	171.91	104.53	104.23	4.750000	3.34	112.28
1398	False	172.24	104.32	103.83	5.600000	3.44	111.37
1399	False	171.73	104.12	104.36	4.710000	3.50	112.15
1400	False	171.81	104.44	104.23	5.290000	3.39	111.64
1401	False	172.03	104.46	103.80	5.590000	3.56	111.86
1402	False	171.62	104.32	104.76	4.920000	3.41	112.40
1403	False	171.89	103.79	104.17	5.110000	3.40	111.76
1404	False	171.53	104.25	104.17	4.300000	3.50	112.23
1405	False	171.76	103.79	104.05	5.120000	3.22	112.08
1406	False	171.79	104.30	104.31	5.650000	3.11	111.80
1407	False	171.54	103.98	103.96	5.570000	3.09	111.88
1408	False	172.02	103.99	104.28	4.220000	2.98	112.42
1409	False	171.99	103.95	104.27	5.460000	3.48	111.06
1410	False	171.90	104.09	104.73	4.590000	3.27	112.47
1411	False	171.96	103.86	103.86	5.060000	3.55	110.44
1412	False	171.95	104.39	104.31	5.980000	3.23	111.33
1413	False	172.25	103.94	103.68	4.140000	3.25	112.48

1414	False	171.83	104.42	104.03	6.000000	2.95	112.98
1415	False	171.92	104.17	104.11	5.590000	3.14	112.20
1416	False	171.93	104.58	104.35	6.040000	3.46	111.46
1417	False	171.76	104.25	104.65	4.250000	3.22	111.45
1418	False	171.21	103.92	104.29	5.180000	3.12	111.70
1419	False	171.79	104.64	103.74	5.970000	3.26	111.86
1420	False	172.13	104.33	103.69	4.750000	3.08	110.75
1421	False	171.56	104.47	104.04	6.380000	3.43	112.12
1422	False	172.24	104.11	103.55	5.020000	3.63	111.33
1423	False	171.49	104.12	104.33	5.930000	3.09	111.56
1424	False	171.74	104.17	104.56	5.540000	3.05	111.67
1425	False	171.68	103.90	104.41	4.740000	3.29	111.85
1426	False	171.94	104.36	104.01	5.140000	3.35	111.83
1427	False	172.22	103.92	104.03	6.250000	3.14	110.89
1428	False	172.27	104.39	104.50	5.090000	3.11	112.51
1429	False	172.39	104.04	104.32	5.090000	3.55	111.82
1430	False	171.49	104.64	104.25	4.310000	3.49	112.00
1431	False	171.76	104.55	103.80	5.280000	3.26	111.40
1432	False	172.52	104.11	104.35	5.300000	3.31	111.55
1433	False	171.96	103.92	104.23	5.000000	3.35	111.16
1434	False	172.43	104.18	104.43	4.570000	3.65	111.92
1435	False	171.66	103.80	104.09	4.730000	3.12	111.56
1436	False	172.66	104.33	104.41	5.185862	3.56	111.47
1437	False	171.76	104.03	103.85	5.790000	3.37	111.30
1438	False	171.92	104.18	103.90	5.670000	3.58	112.37
1439	False	171.90	104.28	104.29	5.140043	3.24	111.49
1440	False	171.65	103.94	104.19	4.670000	3.32	110.95
1441	False	171.53	104.41	103.85	4.640000	3.39	111.94
1443	False	171.40	104.21	104.44	5.590000	3.30	111.09
1444	False	172.19	104.26	104.12	5.230000	3.49	110.52
1445	False	171.84	104.17	103.98	4.910000	3.39	111.26
1446	False	172.14	104.39	104.10	4.440000	3.34	111.86
1447	False	171.65	104.41	104.06	5.370000	3.56	111.36
1448	False	172.07	104.25	103.62	5.100000	3.31	111.28
1449	False	171.74	104.32	103.64	5.470000	3.28	111.56
1450	False	171.66	104.10	104.15	4.530000	3.40	111.83
1451	False	172.72	104.05	104.17	4.210000	3.37	111.53
1452	False	171.94	104.20	103.81	5.100000	3.43	111.69
1453	False	171.93	104.15	103.98	4.570000	3.57	112.71
1455	False	172.20	104.56	103.58	5.190000	3.62	111.17
1456	False	171.56	103.65	103.75	4.850000	3.56	111.31
1457	False	172.48	104.41	104.47	4.810000	3.62	111.31
1458	False	171.34	104.43	104.28	4.700000	3.39	111.01

1459	False	171.65	104.00	104.53	5.690000	3.41	111.09
1460	False	171.78	104.31	103.82	6.190000	3.25	111.14
1461	False	171.99	104.28	104.48	5.270000	3.46	110.94
1462	False	171.92	104.66	104.31	5.150000	3.33	112.75
1463	False	171.90	104.02	103.91	5.380000	3.48	111.98
1464	False	172.23	104.27	104.28	4.350000	3.68	111.64
1466	False	172.17	103.89	103.87	5.220000	3.68	110.68
1467	False	171.58	104.55	103.83	5.500000	3.32	110.97
1468	False	171.16	103.80	103.93	4.950000	3.36	112.29
1469	False	172.21	103.88	104.65	4.820000	3.58	111.11
1470	False	172.37	104.52	104.37	5.360000	3.26	111.27
1471	False	171.49	104.42	104.40	5.080000	3.36	111.45
1472	False	171.93	104.58	103.73	5.280000	3.36	112.41
1473	False	171.85	104.27	103.83	4.150000	3.37	112.00
1474	False	171.76	104.04	104.12	6.290000	3.20	112.87
1475	False	171.70	104.42	104.04	4.750000	3.32	111.63
1476	False	172.03	104.48	104.33	5.660000	3.01	112.15
1477	False	171.43	104.46	103.95	5.130000	3.05	111.20
1478	False	172.16	104.23	104.19	5.090000	3.61	112.43
1479	False	172.36	103.93	104.00	6.080000	3.76	112.89
1480	False	171.55	103.76	104.21	5.110000	3.01	111.48
1481	False	171.64	103.92	104.66	5.710000	3.38	112.33
1482	False	172.17	104.33	104.53	5.060000	3.71	111.75
1483	False	172.30	104.04	103.85	4.120000	3.27	111.69
1484	False	172.08	104.16	104.58	4.790000	3.72	111.04
1486	False	172.52	104.48	104.17	5.160000	3.39	110.71
1487	False	172.47	104.16	103.85	3.990000	3.32	111.25
1488	False	171.30	104.49	103.89	4.880000	3.17	111.51
1489	False	171.78	103.93	103.93	5.880000	3.19	111.07
1490	False	172.11	104.14	104.15	4.840000	3.28	110.98
1491	False	171.82	104.32	104.05	6.060000	3.03	111.68
1492	False	171.79	104.18	104.54	5.130000	3.51	112.40
1493	False	172.01	103.97	104.40	5.520000	3.31	111.18
1494	False	171.63	104.33	104.61	4.880000	3.35	112.16
1495	False	171.57	104.14	104.14	5.410000	3.23	111.76
1496	False	171.75	104.38	104.17	4.420000	3.09	111.28
1497	False	172.19	104.63	104.44	5.270000	3.37	110.97
1498	False	171.80	104.01	104.12	5.510000	3.36	111.95
1499	False	172.06	104.28	104.06	5.170000	3.46	112.25
1500	False	171.47	104.15	103.82	4.630000	3.37	112.07

3- ACP

```
if (!require(FactoMineR)) install.packages("FactoMineR")
```

```
Le chargement a nécessité le package : FactoMineR
```

```
if (!require(factoextra)) install.packages("factoextra")
```

```
Le chargement a nécessité le package : factoextra
```

```
Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(FactoMineR)
library(factoextra)
```

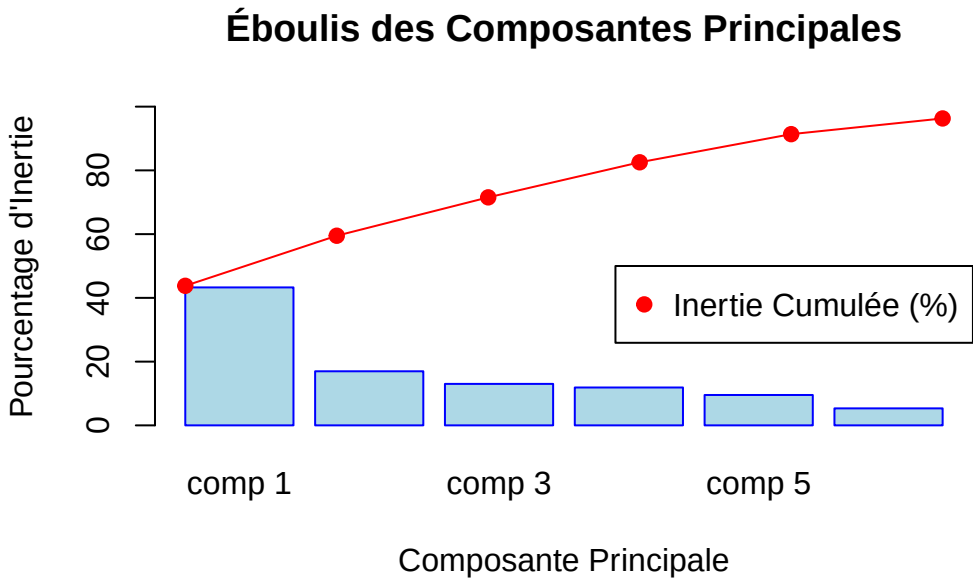
```
# Extraire les colonnes numériques
datanum_without_ouliers_z <- Filter(is.numeric, data_without_ouliers_z)
```

```
# Réaliser l'ACP sans spécifier ncp
resultat_acp <- PCA(datanum_without_ouliers_z, scale.unit = TRUE, graph = FALSE)

# Extraire les valeurs propres
valeurs_propres <- resultat_acp$eig[,1]
```

```
# Calculer les pourcentages d'inertie expliquée par chaque composante
pourcentage_inertie <- 100 * valeurs_propres / sum(valeurs_propres)
# Calculer l'inertie cumulée
cumulative_inertia <- cumsum(pourcentage_inertie)
```

```
# Créer un graphique avec un seul appel
par(mfrow = c(1, 1)) # Assurez-vous d'avoir une seule fenêtre graphique

# Créer un plot avec des barres pour les pourcentages d'inertie
barplot(pourcentage_inertie, main = "Éboulis des Composantes Principales",
        xlab = "Composante Principale", ylab = "Pourcentage d'Inertie",
        col = "lightblue", border = "blue", ylim = c(0, 100))

# Ajouter la courbe cumulée en superposition sur le même graphique
```

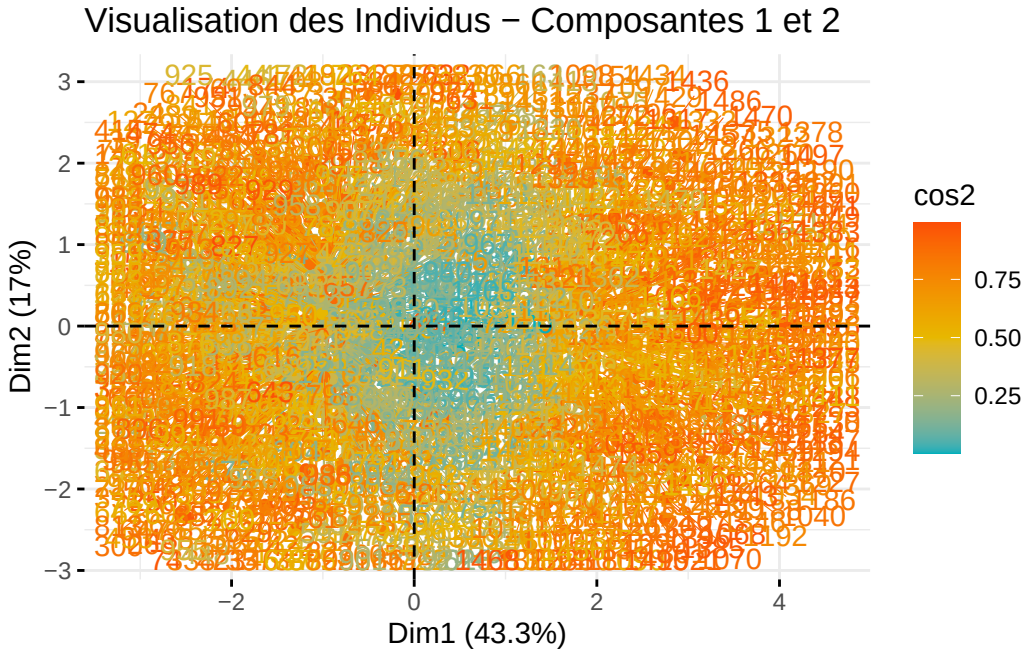
```r
# Recréer le graphique avec la courbe cumulée en utilisant `plot` pour ne pas effacer les ba
par(new = TRUE) # Permet de superposer le nouveau graphique sur le précédent
plot(cumulative_inertia, type = "o", col = "red", pch = 19,
     ylim = c(0, 100), xlab = "", ylab = "", axes = FALSE)

# Ajouter une légende
legend("topright", legend = c("Inertie Cumulée (%)"), col = "red", pch = 19, inset = c(0, 0.5
```

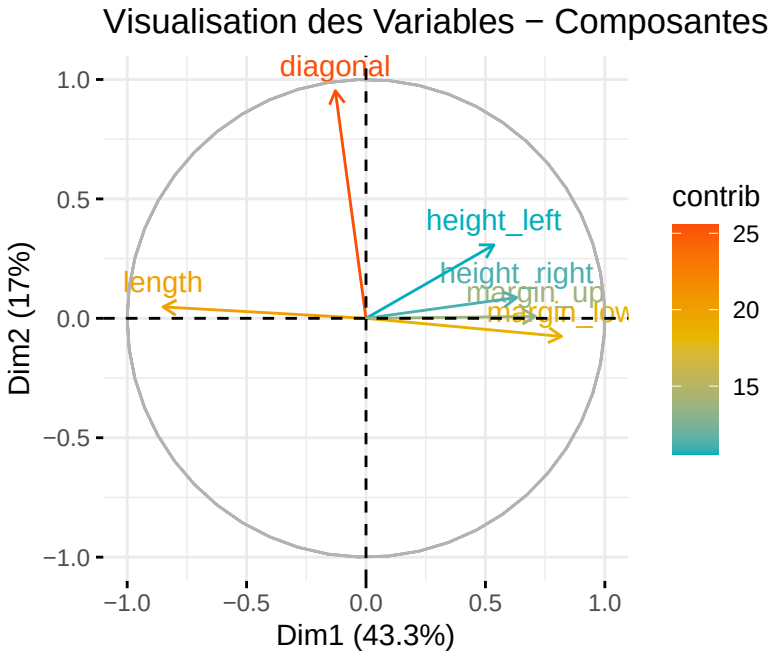


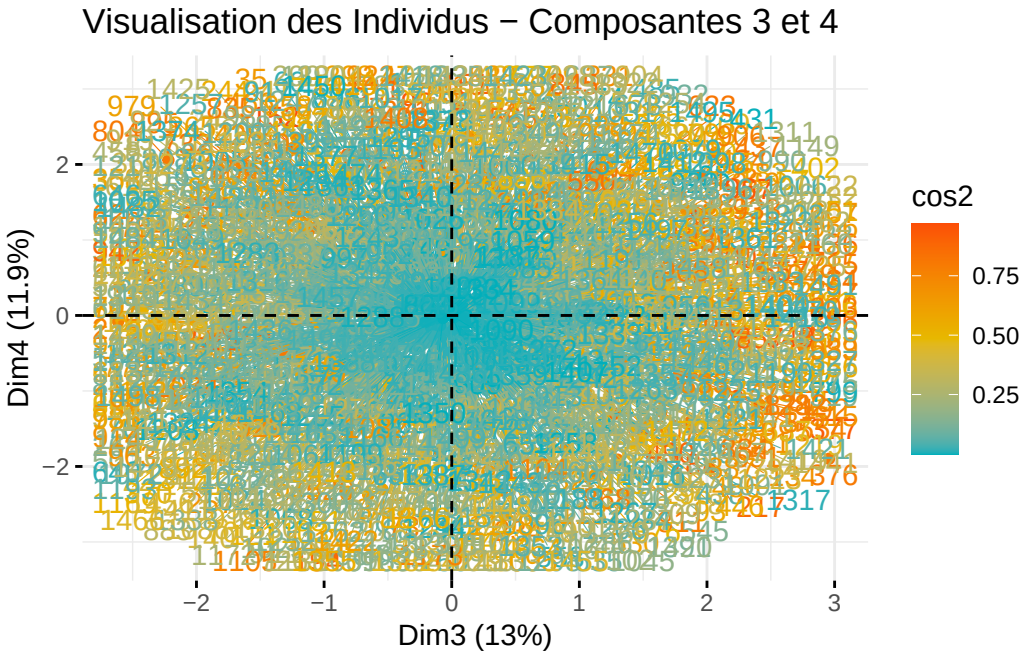

```r
# on choisit 4 composantes principales qui représentent 80% de la variance
resultat_acp_optimal <- PCA(datanum_without_ouliers_z, scale.unit = TRUE, ncp = 4, graph = FA
```

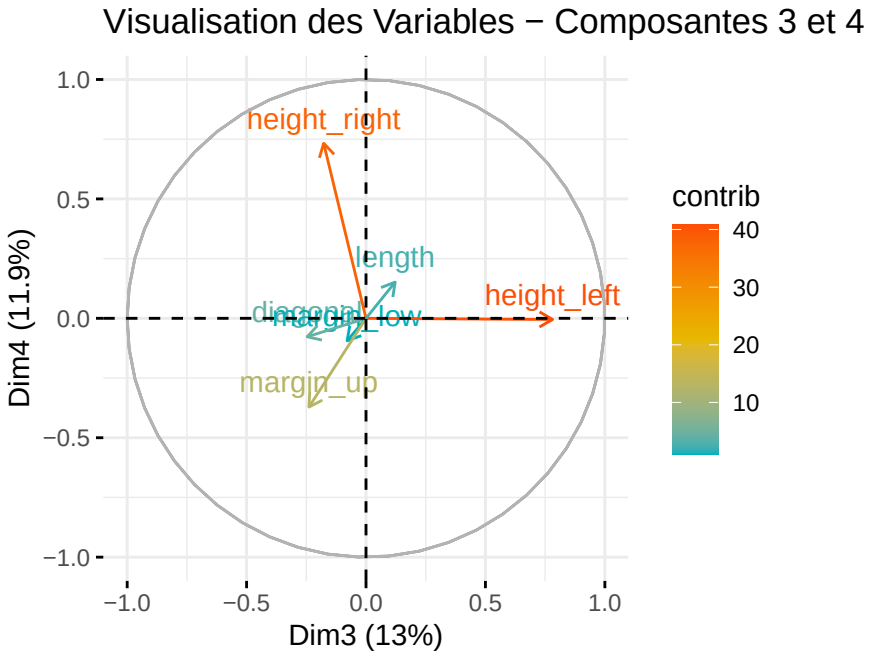
```r
# Visualiser les individus et les variables pour les premières et dernières combinaisons de c
pairs_to_plot <- list(c(1, 2), c(3, 4))  # Choisir les paires de composantes à afficher

# Visualiser les individus et les variables pour les différentes combinaisons de composantes
for (pair in pairs_to_plot) {
# Visualiser les individus
  print(fviz_pca_ind(resultat_acp_optimal,
                     axes = pair,
                     col.ind = "cos2",
                     gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
                     repel = TRUE) +
```

```
        ggtitle(paste("Visualisation des Individus - Composantes", pair[1], "et", pair[2])))

  # Visualiser les variables
  print(fviz_pca_var(resultat_acp_optimal,
                     axes = pair,
                     col.var = "contrib",
                     gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07")) +
        ggtitle(paste("Visualisation des Variables - Composantes", pair[1], "et", pair[2])))
}
```




Visualisation des Variables − Composantes 1 et 2
diagonal
height_left
height_right
margin_up
length
contrib
25
20
15
1.0
0.5
0.0
−0.5
−1.0
Dim2 (17%)
Dim1 (43.3%)


Visualisation des Individus − Composantes 3 et 4
cos2
0.75
0.50
0.25
2
0
−2
−2
−1
0
1
2
3
Dim4 (11.9%)
Dim3 (13%)



Axes principaux :
Dim1 (43.3%) : Cet axe explique 43.3% de la variance totale des données.
Dim2 (17%) : Cet axe explique 17% de la variance totale des données.

Dim1 On observe :
- une forte contribution positibe de la marge basse (margin_low)
- une forte contribution négative de la longueur (length)
une forte valeur de Dim1 indiquera un longueur de billet faible avec une marge basse élevée

Dim2 On observe :
- une forte contribution positibe de la diagonale Une forte valeur de Dim2 indiquera une valeur de diagonale élevée

```
# Extraire les scores des individus
scores <- resultat_acp_optimal$ind$coord
```

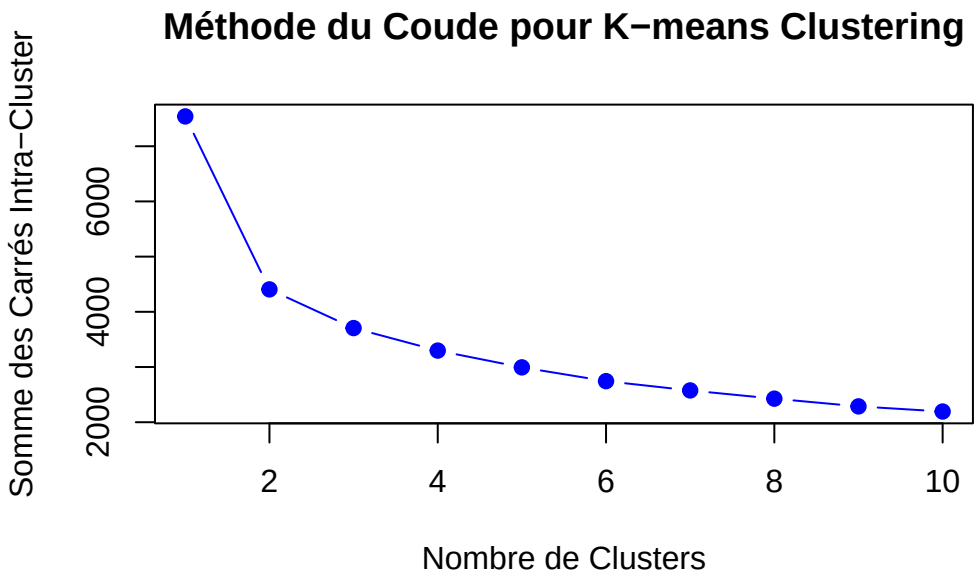
```
head(scores)
```

```
       Dim.1      Dim.2      Dim.3       Dim.4
2 -2.0305446 -2.2234316 -1.0048435 -0.01956135
3 -0.9427170  2.6760577  1.2489821 -0.84876466
4 -1.3898268 -1.8603367  0.6064674  0.83245323
5  0.1694694 -0.5337031  0.8893891 -1.76130391
6 -0.6711784  0.4183698 -0.8915549  0.79572264
```

```
7   0.2146804   1.3325731   0.0091752 -0.50813505
```

4- Clustering

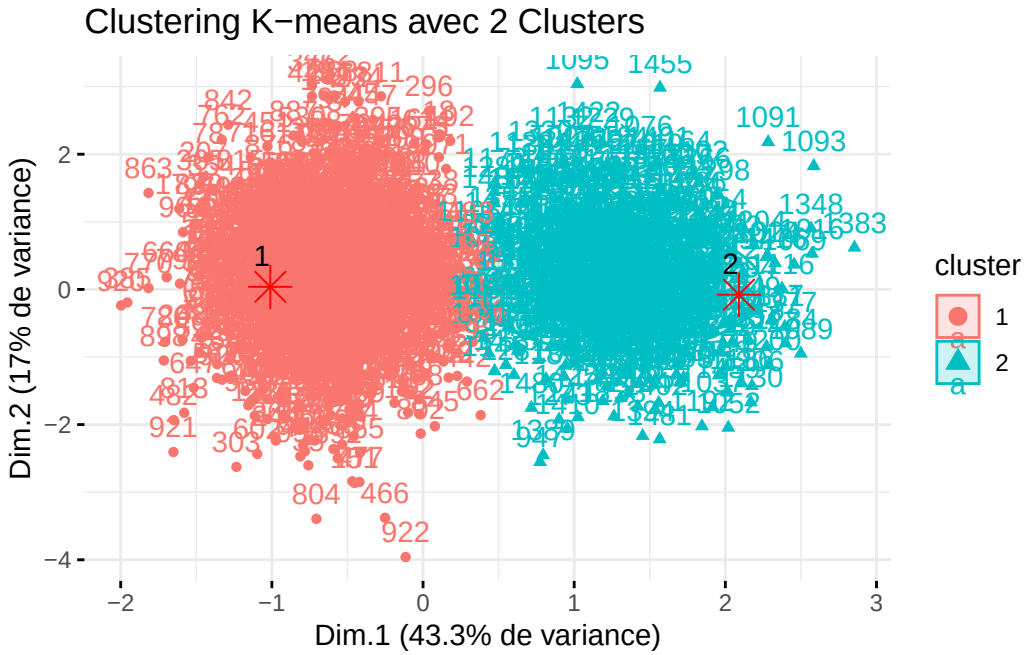
```r
# Définir une plage de nombres de clusters à tester
k_values <- 1:10  # Tester de 1 à 10 clusters
wss <- numeric(length(k_values))  # Pour stocker la somme des carrés intra-cluster

# Calculer l'inertie pour chaque nombre de clusters
for (k in k_values) {
  kmeans_result <- kmeans(scores, centers = k, nstart = 25)  # nstart pour la reproductibilit
  wss[k] <- kmeans_result$tot.withinss
}

# Tracer la méthode du coude
plot(k_values, wss, type = "b", pch = 19, col = "blue",
     xlab = "Nombre de Clusters", ylab = "Somme des Carrés Intra-Cluster",
     main = "Méthode du Coude pour K-means Clustering")
```



TEST AVEC 2 CLUSTERS

```r
# Appliquer K-means clustering avec 2 clusters
kmeans_result <- kmeans(scores, centers = 2, nstart = 23)  # nstart pour la reproductibilité
```

```r
# Installer et charger le package factoextra si ce n'est pas déjà fait
if (!require(factoextra)) install.packages("factoextra")
library(factoextra)

# Installer et charger le package ggplot2 si ce n'est pas déjà fait
if (!require(ggplot2)) install.packages("ggplot2")
library(ggplot2)

# Visualiser les clusters dans les deux premières composantes principales
plot_clusters <- fviz_cluster(kmeans_result, data = scores,
                              ellipse.type = "euclid",  # Ajouter des ellipses autour des clu
                              ggtheme = theme_minimal())

# Modifier les étiquettes des axes pour refléter la variance expliquée
plot_clusters <- plot_clusters +
  labs(x = "Dim.1 (43.3% de variance)", y = "Dim.2 (17% de variance)") +
  ggtitle("Clustering K-means avec 2 Clusters")

# Ajouter les centres des clusters au graphique
centroids <- kmeans_result$centers

plot_clusters <- plot_clusters +
  geom_point(data = as.data.frame(centroids), aes(x = Dim.1, y = Dim.2),
             color = "red", size = 5, shape = 8) +  # Points rouges pour les centres
  geom_text(data = as.data.frame(centroids), aes(x = Dim.1, y = Dim.2, label = rownames(centi
            color = "black", vjust = -1, hjust = 1)  # Étiquettes pour les centres

# Afficher le graphique final
print(plot_clusters)
```




```
# Centres des clusters
print(kmeans_result$centers)
```

```
      Dim.1       Dim.2       Dim.3       Dim.4
1 -1.010287  0.03782197  0.04525871  0.04633596
2  2.089887 -0.07823879 -0.09362248 -0.09585090
```

```
# Taille de chaque cluster
print(kmeans_result$size)
```

```
[1] 995 481
```

```
# Ajouter les résultats du clustering à un DataFrame
data_with_clusters <- data.frame(data_without_ouliers_z, Cluster = kmeans_result$cluster)

# Afficher les premières lignes du DataFrame avec les clusters
head(data_with_clusters)
```

```
  is_genuine diagonal height_left height_right margin_low margin_up length
2       True   171.46      103.36       103.66       3.77      2.99 113.09
3       True   172.69      104.48       103.50       4.40      2.94 113.16
```

```
4       True   171.36      103.91       103.94       3.62      3.01 113.51
5       True   171.73      104.28       103.46       4.04      3.48 112.54
6       True   172.17      103.74       104.08       4.42      2.95 112.81
7       True   172.34      104.18       103.85       4.58      3.26 112.81
  Cluster
2       1
3       1
4       1
5       1
6       1
7       1
```

```r
count_values  <- data_with_clusters %>%
  group_by(Cluster, is_genuine) %>%
  count()

print(count_values)
```

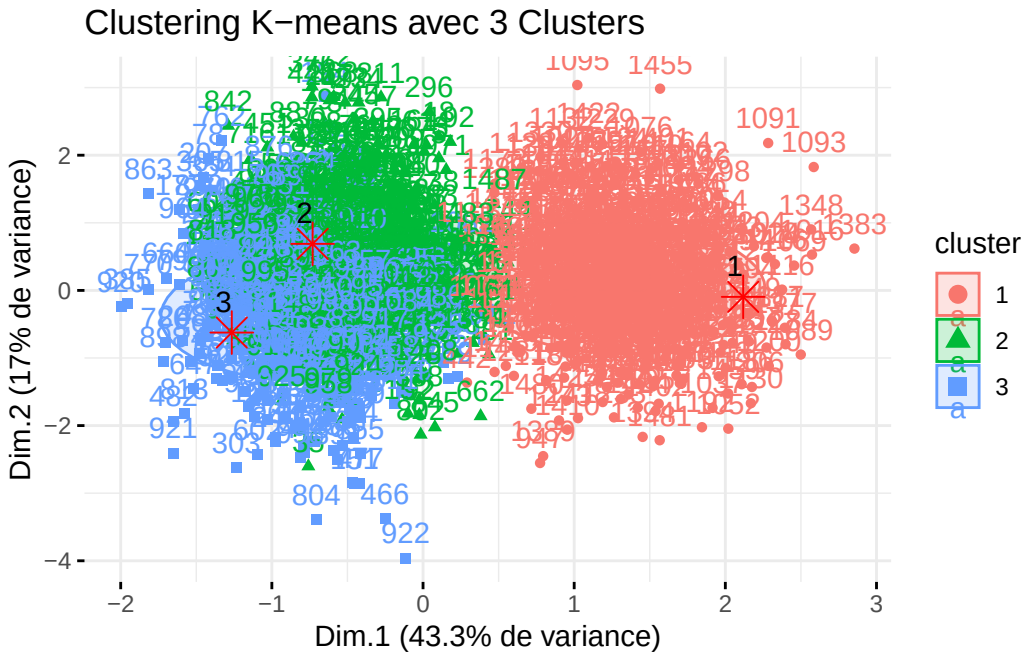
```
# A tibble: 4 x 3
# Groups:   Cluster, is_genuine [4]
  Cluster is_genuine     n
    <int> <chr>      <int>
1       1 False         12
2       1 True         983
3       2 False        471
4       2 True          10
```

TEST AVEC 3 CLUSTERS

```r
# Appliquer K-means clustering avec 3 clusters
kmeans_result_2 <- kmeans(scores, centers = 3, nstart = 33)  # nstart pour la reproductibili
```

```r
# Installer et charger le package factoextra si ce n'est pas déjà fait
if (!require(factoextra)) install.packages("factoextra")
library(factoextra)

# Installer et charger le package ggplot2 si ce n'est pas déjà fait
if (!require(ggplot2)) install.packages("ggplot2")
library(ggplot2)

# Visualiser les clusters dans les deux premières composantes principales
```

```r
plot_clusters_2 <- fviz_cluster(kmeans_result_2, data = scores,
                                ellipse.type = "euclid",  # Ajouter des ellipses autour des clu
                                ggtheme = theme_minimal())

# Modifier les étiquettes des axes pour refléter la variance expliquée
plot_clusters_2 <- plot_clusters_2 +
  labs(x = "Dim.1 (43.3% de variance)", y = "Dim.2 (17% de variance)") +
  ggtitle("Clustering K-means avec 3 Clusters")

# Ajouter les centres des clusters au graphique
centroids_2 <- kmeans_result_2$centers

plot_clusters_2 <- plot_clusters_2 +
  geom_point(data = as.data.frame(centroids_2), aes(x = Dim.1, y = Dim.2),
             color = "red", size = 5, shape = 8) +  # Points rouges pour les centres
  geom_text(data = as.data.frame(centroids_2), aes(x = Dim.1, y = Dim.2, label = rownames(cen
            color = "black", vjust = -1, hjust = 1)  # Étiquettes pour les centres

# Afficher le graphique final
print(plot_clusters_2)
```



```r
# Centres des clusters
print(kmeans_result_2$centers)
```

```
       Dim.1       Dim.2      Dim.3       Dim.4
1  2.1180703 -0.09563134 -0.1050411 -0.10193927
2 -0.7302239  0.68851932  0.4853524  0.13042833
3 -1.2651856 -0.62369073 -0.4037040 -0.03806472
```

```r
# Taille de chaque cluster
print(kmeans_result_2$size)
```

```
[1] 471 512 493
```

```r
# Ajouter les résultats du clustering à un DataFrame
data_with_clusters_2 <- data.frame(data_without_ouliers_z, Cluster = kmeans_result_2$cluster)

# Afficher les premières lignes du DataFrame avec les clusters
head(data_with_clusters_2)
```

```
  is_genuine diagonal height_left height_right margin_low margin_up length
2       True   171.46      103.36       103.66       3.77      2.99 113.09
3       True   172.69      104.48       103.50       4.40      2.94 113.16
4       True   171.36      103.91       103.94       3.62      3.01 113.51
5       True   171.73      104.28       103.46       4.04      3.48 112.54
6       True   172.17      103.74       104.08       4.42      2.95 112.81
7       True   172.34      104.18       103.85       4.58      3.26 112.81
  Cluster
2       3
3       2
4       3
5       2
6       3
7       2
```

```r
count_values  <- data_with_clusters_2 %>%
  group_by(Cluster, is_genuine) %>%
  count()

print(count_values)
```

```
# A tibble: 6 x 3
# Groups:   Cluster, is_genuine [6]
  Cluster is_genuine     n
    <int> <chr>      <int>
1       1 False        467
2       1 True           4
3       2 False         11
4       2 True         501
5       3 False          5
6       3 True         488
```

Le but était de vérifier comment se comportait le modèle avec 3 clusters. L'idéal aurait été un clustering avec 100 % de vrai dans un cluster 100 % de faux dans un autre cluster et les douteux dans un 3ème cluster

VII- Régression Logistique

1- Remplacement des valeurs de "is_genuine" par "0" et "1"

Nous allons commencer par remplacer "False" par "0" et "True" par "1" afin que "is_genuine" contienne des valeurs numériques binaires.

```
# Convertir 'is_genuine' en binaire : True -> 1 et False -> 0
data$is_genuine <- ifelse(data$is_genuine == "True", 1, 0)

# Vérifiez les premiers enregistrements pour confirmer la transformation
head(data$is_genuine)
```

```
[1] 1 1 1 1 1 1
```

2- Régréssion logistique complète (toutes les variables)

```
reg_log <- glm(is_genuine~diagonal+height_left+height_right+margin_low+margin_up+length,
family="binomial",data=data)
summary(reg_log)
```

```
Call:
glm(formula = is_genuine ~ diagonal + height_left + height_right +
```

```
    margin_low + margin_up + length, family = "binomial", data = data)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -204.5582   241.7681  -0.846   0.3975
diagonal        0.0680     1.0913   0.062   0.9503
height_left    -1.7162     1.1038  -1.555   0.1200
height_right   -2.2584     1.0720  -2.107   0.0351 *
margin_low     -5.7756     0.9370  -6.164 7.11e-10 ***
margin_up     -10.1531     2.1078  -4.817 1.46e-06 ***
length          5.9129     0.8458   6.991 2.73e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1909.543  on 1499  degrees of freedom
Residual deviance:   84.685  on 1493  degrees of freedom
AIC: 98.685

Number of Fisher Scoring iterations: 10
```

Certaines des variables obtenues ont des p-valeurs qui sont inférieures au niveau de test de 5 %, ce qui nous indique qu'elles sont bien significatives. Certaines autres ne sont pas en dessous de ce seuil.

On peut donc passer sur une procédure de sélection en retirant les variables non significatives au fur et à mesure, mais nous pouvons aussi sélectionner automatiquement un modèle avec une commande telle que stepAIC , qui sélectionne de manière automatique un modèle en se basant sur le critère AIC.

3- Sélection du modèle (méthode AIC)

```
library(MASS)
```

```
Attachement du package : 'MASS'

L'objet suivant est masqué depuis 'package:dplyr':

    select
```

```r
stepAIC(reg_log, direction = "both")
```

```
Start:  AIC=98.68
is_genuine ~ diagonal + height_left + height_right + margin_low +
    margin_up + length

               Df Deviance    AIC
- diagonal      1   84.689  96.69
<none>              84.685  98.68
- height_left   1   87.168  99.17
- height_right  1   89.799 101.80
- margin_up     1  127.341 139.34
- margin_low    1  205.785 217.79
- length        1  308.723 320.72

Step:  AIC=96.69
is_genuine ~ height_left + height_right + margin_low + margin_up +
    length

               Df Deviance    AIC
<none>              84.689  96.69
- height_left   1   87.173  97.17
+ diagonal      1   84.685  98.68
- height_right  1   89.925  99.92
- margin_up     1  128.287 138.29
- margin_low    1  211.175 221.17
- length        1  309.492 319.49



Call:  glm(formula = is_genuine ~ height_left + height_right + margin_low +
    margin_up + length, family = "binomial", data = data)

Coefficients:
 (Intercept)   height_left  height_right    margin_low     margin_up
    -193.147        -1.716        -2.263        -5.794       -10.167
      length
       5.920

Degrees of Freedom: 1499 Total (i.e. Null);  1494 Residual
Null Deviance:      1910
Residual Deviance: 84.69    AIC: 96.69
```

La procédure de sélection de modèle nous a permis de déterminer que les variables “diagonal” et “height_left” n’étaient pas nécessaires pour le modèle final. Les variables restantes (“height_right”, “margin_low”, “margin_up”, et “length”) sont significatives et ont un impact important sur le modèle.
Nous avons réussi à réduire l’AIC, indiquant que le modèle final est plus parcimonieux tout en conservant une bonne qualité d’ajustement.

Analyse des coefficients : 3 variables ont une valeur absolue élevée ce qui indique qu’elles ont plus d’importance et d’influence dans la prédiction. - “margin_up” a un coeff négatif de -10.167 ce qui indique que plus la marge haute haute augmente, plus il y a de chance qu’il s’agisse d’un faux billet. - “margin_low” avec un coeff négatif également de -5.794 a également une influence importante et indique que l’augmentation de la marge basse augmente les probabilités qu’il s’agisse d’un faux billet. - “length” a un coeff positif de 5.920 qui indique à l’inverse que l’augmentation de la longueur du billet augmente la probabilité qu’il s’agisse d’un vrai billet.

Calculons les Odds Ratios pour indiquer comment les changements dans les variables indépendantes affectent la probabilité de vrais ou faux billets :

```r
reg_log_model <- glm(is_genuine ~ height_left + height_right + margin_low +
    margin_up + length,
family="binomial",data=data)
summary(reg_log_model)
```

```
Call:
glm(formula = is_genuine ~ height_left + height_right + margin_low +
    margin_up + length, family = "binomial", data = data)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  -193.1473   157.6814  -1.225    0.221
height_left    -1.7156     1.1031  -1.555    0.120
height_right   -2.2632     1.0675  -2.120    0.034 *
margin_low     -5.7942     0.8908  -6.504 7.80e-11 ***
margin_up     -10.1666     2.0988  -4.844 1.27e-06 ***
length          5.9204     0.8386   7.060 1.67e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1909.543  on 1499  degrees of freedom
Residual deviance:   84.689  on 1494  degrees of freedom
```

```
AIC: 96.689

Number of Fisher Scoring iterations: 10
```

```r
# Récupérer les coefficients du modèle final
coefficients <- coef(reg_log_model)

# Calculer les odds ratios en exponentiant les coefficients
odds_ratios <- exp(coefficients)

# Afficher les odds ratios sans exposant
formatted_odds_ratios <- formatC(odds_ratios, format = "f", digits = 5)


# Imprimer les odds ratios formatés
formatted_odds_ratios
```

```
 (Intercept)  height_left height_right   margin_low    margin_up       length
   "0.00000"    "0.17986"    "0.10402"    "0.00305"    "0.00004"  "372.54651"
```

height_left: 0.17986 Une augmentation d'une unité de height_left diminue les chances que le billet soit authentique d'environ 82% (puisque 0.17986<1)

height_right: 0.10402 Une augmentation d'une unité de height_right diminue les chances que le billet soit authentique d'environ 90%.

margin_low: 0.00305 Une augmentation d'une unité de margin_low diminue les chances que le billet soit authentique d'environ 99.7%.

margin_up: 0.00004 Une augmentation d'une unité de margin_up diminue fortement les chances que le billet soit authentique.

length: 372.5465 Une augmentation d'une unité de length augmente de manière significative (par 372 fois) les chances que le billet soit authentique.

4- Evaluation des performances du modèle

a- Métrique de performance

Matrice de confusion

La matrice de confusion permet de comparer les prédictions du modèle avec les valeurs réelles.

```
# Prédictions du modèle
predic_reg_log <- predict(reg_log_model, type = "response")

# Convertir les probabilités en classes (0 ou 1) en utilisant un seuil de 0.5
predic_reg_log_classes <- ifelse(predic_reg_log > 0.5, "True", "False")

# Créer la matrice de confusion
confusion_reg_log <- table(Predicted = predic_reg_log_classes, Actual = data$is_genuine)
print(confusion_reg_log)
```

```
         Actual
Predicted   0   1
    False 491   4
    True    9 996
```

Vrais Négatifs (VN) : 491
Faux Négatifs (FN) : 4
Faux Positifs (FP) : 9
Vrais Positifs (VP) : 996

La matrice de confusion montre que votre modèle a une très bonne performance en classant les cas True et False correctement avec peu d'erreurs.

Mesures de performance des positifs

Précision : Proportion des prédictions correctes parmi toutes les prédictions. VP/(VP+FP)
Sensibilité (Recall) : Proportion des vrais positifs parmi tous les cas positifs réels. VP/(VP+FN)
Spécificité : Proportion des vrais négatifs parmi tous les cas négatifs réels. VN/(VN+FP)
F1-Score : Moyenne harmonique de la précision et de la sensibilité. 2x((precision x recall)/(precision + recall))

```
# Calculer les mesures de performance
precision <- confusion_reg_log[2, 2] / (confusion_reg_log[2, 2] + confusion_reg_log[2, 1])
recall <- confusion_reg_log[2, 2] / (confusion_reg_log[2, 2] + confusion_reg_log[1, 2])
specificity <- confusion_reg_log[1, 1] / (confusion_reg_log[1, 1] + confusion_reg_log[2, 1])
f1_score <- 2 * (precision * recall) / (precision + recall)
```

```r
# Afficher les résultats
cat("Précision: ", precision, "\n")
```

```
Précision:  0.9910448
```

```r
cat("Sensibilité (Recall): ", recall, "\n")
```

```
Sensibilité (Recall):  0.996
```

```r
cat("Spécificité: ", specificity, "\n")
```

```
Spécificité:  0.982
```

```r
cat("F1-Score: ", f1_score, "\n")
```

```
F1-Score:  0.9935162
```

Précision (Precision) : 0.991 Cela signifie que parmi toutes les instances que le modèle a classées comme True, 99.1% sont effectivement True. C'est un excellent résultat, indiquant que le modèle est très précis dans ses prédictions positives.

Sensibilité (Recall) : 0.996 La sensibilité mesure la proportion des véritables True correctement identifiés par le modèle. Avec une sensibilité de 99.6%, votre modèle détecte presque tous les cas positifs.

Spécificité : 0.982 La spécificité mesure la proportion des véritables False correctement identifiés par le modèle. Avec une spécificité de 98.2%, le modèle est aussi très efficace pour identifier les cas négatifs.

F1-Score : 0.993 L'F1-Score est la moyenne harmonique de la précision et du rappel. Un F1-Score de 99.3% indique un excellent équilibre entre précision et rappel.

Mon objectif est d'obtenir un modèle avec une spécificité proche de 100% pour minimiser au maximum le risque de faux positif, il est donc important que tous les vrais négatifs soient identifier même si cela déteriore un peu la précision.

Courbe ROC et AUC

La courbe ROC (Receiver Operating Characteristic) est utile pour visualiser la performance du modèle à différents seuils de classification. L'AUC (Area Under the Curve) mesure la capacité globale du modèle à discriminer entre les classes.

```
# Installer et charger le package pROC si ce n'est pas déjà fait
if (!require(pROC)) install.packages("pROC")
```

```
Le chargement a nécessité le package : pROC
```

```
Type 'citation("pROC")' for a citation.
```

```
Attachement du package : 'pROC'
```

```
Les objets suivants sont masqués depuis 'package:stats':
```

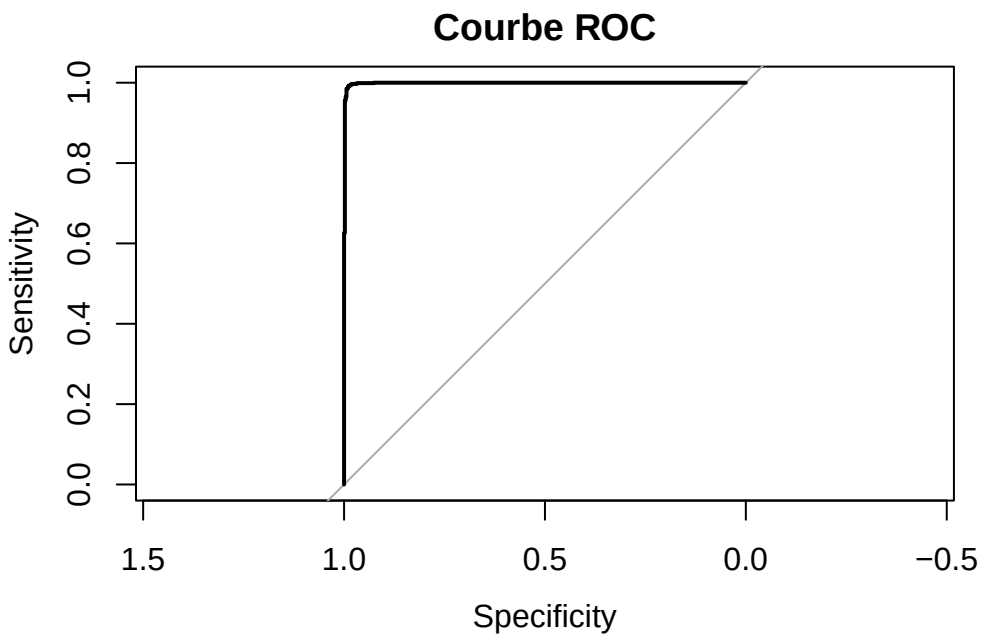
```
    cov, smooth, var
```

```
library(pROC)

# Calculer la courbe ROC
roc_curve <- roc(data$is_genuine, predic_reg_log)
```

```
Setting levels: control = 0, case = 1
```

```
Setting direction: controls < cases
```

```
# Tracer la courbe ROC
plot(roc_curve, main = "Courbe ROC")
```




```
# Calculer l'AUC
auc_value <- auc(roc_curve)
cat("AUC: ", auc_value, "\n")
```

```
AUC:  0.998894
```

AUC : 0.998894 L'AUC mesure la capacité globale du modèle à discriminer entre les classes. Une AUC proche de 1 (votre AUC est 99.9%) indique que le modèle a une très bonne performance de classification, capable de distinguer presque parfaitement entre les classes True et False.

b- Validation croisée K-Fold :

La validation croisée K-Fold permet de tester la robustesse du modèle en le validant sur différentes sous-parties du dataset.

```
# Installer et charger le package cvms si ce n'est pas déjà fait
if (!require(cvms)) install.packages("cvms")
```

```
Le chargement a nécessité le package : cvms
```

```r
if (!require(caret)) install.packages("caret")
```

```
Le chargement a nécessité le package : caret
```

```
Le chargement a nécessité le package : lattice
```

```
Attachement du package : 'caret'
```

```
L'objet suivant est masqué depuis 'package:purrr':
```

```
    lift
```

```r
library(cvms)
library(caret)

# Appliquer la validation croisée K-Fold
set.seed(123)  # Pour la reproductibilité
folds <- createFolds(data$is_genuine, k = 10)

# Calculer les performances pour chaque fold
cv_results <- lapply(folds, function(fold) {
  train_data <- data[-fold, ]
  test_data <- data[fold, ]

  # Ajuster le modèle sur les données d'entraînement
  model <- glm(is_genuine ~ height_left + height_right + margin_low + margin_up + length,
               family = "binomial", data = train_data)

  # Prédictions sur les données de test
  test_predictions <- predict(model, newdata = test_data, type = "response")
  test_classes <- ifelse(test_predictions > 0.5, 1, 0)

  # Calculer la matrice de confusion et les mesures de performance
  test_confusion_matrix <- table(Predicted = test_classes, Actual = test_data$is_genuine)
  precision <- test_confusion_matrix[2, 2] / (test_confusion_matrix[2, 2] + test_confusion_ma
  recall <- test_confusion_matrix[2, 2] / (test_confusion_matrix[2, 2] + test_confusion_matri
  specificity <- test_confusion_matrix[1, 1] / (test_confusion_matrix[1, 1] + test_confusion_
  f1_score <- 2 * (precision * recall) / (precision + recall)
```

```
  return(c(precision = precision, recall = recall, specificity = specificity, f1_score = f1_s
})

# Moyenne des résultats de validation croisée
cv_results <- do.call(rbind, cv_results)
mean_cv_results <- colMeans(cv_results)
print(mean_cv_results)
```

```
  precision      recall specificity    f1_score
  0.9949425   0.9910552   0.9900404   0.9929758
```

Le lift est une mesure souvent utilisée pour évaluer l'amélioration de la prédiction par rapport à un modèle de base (par exemple, prédire la classe majoritaire). Les valeurs élevées du lift indiquent que votre modèle offre des améliorations significatives par rapport à des méthodes naïves.

Conclusion Ces résultats montrent que votre modèle de régression logistique est très performant avec des scores de précision, de rappel, de spécificité et de F1-Score très élevés, ainsi qu'une AUC presque parfaite. Cela suggère que le modèle est bien ajusté à vos données et est capable de faire des prédictions avec une grande précision.

Prochaines Étapes Validation Externe : Si possible, testez le modèle sur un jeu de données complètement indépendant pour vérifier sa généralisation. Analyse des Résidus : Examinez les résidus pour identifier tout modèle non détecté ou problème potentiel. Exploration de Modèles Alternatifs : Bien que le modèle de régression logistique semble excellent, vous pouvez explorer d'autres modèles pour comparer les performances. Implémentation et Déploiement : Si le modèle répond à vos attentes, vous pouvez envisager de l'implémenter dans un environnement de production. Ces étapes vous aideront à assurer que votre modèle est non seulement performant mais aussi robuste et fiable dans des conditions réelles.

Nous allons ajuster le seuil pour réduire la probabilité de classer un faux billet parmis les vrais, c'est à dire minimiser le risque de faux positifs.

```
# Prédire les probabilités pour la classe positive
predicted_prob <- predict(reg_log_model, type = "response")
```

```
head(predicted_prob)
```

```
        1         2         3         4         5         6
0.8650657 0.9999951 0.9996412 0.9999990 0.7843526 0.9963137
```

Création d'une fonction pour calculer la spécificité

```r
# Fonction pour calculer la spécificité
calculate_specificity <- function(pred_probs, actual_classes, threshold) {
  predicted_classes <- ifelse(pred_probs > threshold, 1, 0)
  confusion <- table(predicted_classes, actual_classes)
  # Confusion peut avoir des noms différents selon les résultats, ajustez si nécessaire
  VN <- confusion["0", "0"]
  FP <- confusion["1", "0"]
  specificity <- VN / (VN + FP)
  return(specificity)
}
```

On teste divers seuils

```r
thresholds <- seq(0.10, 0.99, by = 0.01)
specificities <- sapply(thresholds, function(t) {
  calculate_specificity(predicted_prob, data$is_genuine, t)
})
```

On recherche le meilleur seuil

```r
best_threshold <- thresholds[which.max(specificities)]
best_specificity <- max(specificities)

cat("Meilleur seuil:", best_threshold, "\n")
```

```
Meilleur seuil: 0.97
```

```r
cat("Spécificité à ce seuil:", best_specificity, "\n")
```

```
Spécificité à ce seuil: 0.998
```

On applique ce seuil optimal à notre modèle pour obtenir les prédictions finales

```r
final_predictions <- ifelse(predicted_prob > best_threshold, "True", "False")
final_confusion_matrix <- table(final_predictions, Actual = data$is_genuine)
print(final_confusion_matrix)
```

```
                Actual
final_predictions   0   1
            False 499  50
            True    1 950
```

Avec ce modèle et ce seuil de sensibilité, on obtient un résultat intéressant pour le nombre de faux positif. Il n'y a qu'un seul faux billet détecté comme positif sur les 1500 billets de notre jeu de donnée.

```r
taux_erreur <- 1/1500 * 100
print(paste(format(taux_erreur, digits = 1, nsmall = 1), "%"))
```

```
[1] "0.07 %"
```

```r
# Calculer les mesures de performance
precision_2 <- final_confusion_matrix[2, 2] / (final_confusion_matrix[2, 2] + final_confusio
recall_2 <- final_confusion_matrix[2, 2] / (final_confusion_matrix[2, 2] + final_confusion_ma
specificity_2 <- final_confusion_matrix[1, 1] / (final_confusion_matrix[1, 1] + final_confusi
f1_score_2 <- 2 * (precision_2 * recall_2) / (precision_2 + recall_2)

# Afficher les résultats
cat("Précision: ", precision_2, "\n")
```

```
Précision:  0.9989485
```

```r
cat("Sensibilité (Recall): ", recall_2, "\n")
```

```
Sensibilité (Recall):  0.95
```

```r
cat("Spécificité: ", specificity_2, "\n")
```

```
Spécificité:  0.998
```

```
cat("F1-Score: ", f1_score_2, "\n")
```

```
F1-Score:  0.9738596
```

En regardant les performances globales, on voit que nous avons effectivement améliorer la Spécificité, c'est à dire le risque de faux positifs, passant de 0.982 à 0.998. 99.8% des faux billets sont détectés. Nous avons également augmenté la précision passant de 0.991 à 0.998, cela signifie que 99.8% des billets classés dans "True" sont réellement vrais.

Mais nous avons une baisse de sensibilité passant de 0.996 à 0.95 ce qui signifie que seulement 95% des vrais billets sont identifiés et classés comme tel, donc 5% de vrais billets sont considérés comme faux par le modèle. et une baisse du F1_score passant de 0.993 à 0.97, c'est à dire un moins bon équilibre entre détection des faux et des vrais billets dans le global.

Nous allons tester d'autres méthodes de classification pour comparer les modèles et leur performance.

Méthode RANDOM FOREST

```
# Charger les packages nécessaires
if (!require(randomForest)) install.packages("randomForest")
```

```
Le chargement a nécessité le package : randomForest
```

```
randomForest 4.7-1.1
```

```
Type rfNews() to see new features/changes/bug fixes.
```

```

Attachement du package : 'randomForest'
```

```
L'objet suivant est masqué depuis 'package:ggplot2':
```

```
    margin
```

```
L'objet suivant est masqué depuis 'package:dplyr':
```

```
    combine
```

```r
library(randomForest)
library(caret)

# Préparer les données
features <- data[ , names(data) != "is_genuine"]
target <- as.factor(data$is_genuine)

# Diviser les données en ensembles d'entraînement et de test
set.seed(123)
trainIndex <- createDataPartition(target, p = 0.7, list = FALSE)
trainData <- data[trainIndex, ]
testData <- data[-trainIndex, ]

# Préparer les caractéristiques et la cible pour l'entraînement et le test
trainFeatures <- trainData[ , names(trainData) != "is_genuine"]
trainTarget <- as.factor(trainData$is_genuine)
testFeatures <- testData[ , names(testData) != "is_genuine"]
testTarget <- as.factor(testData$is_genuine)

# Entraîner le modèle Random Forest pour classification
rf_model <- randomForest(x = trainFeatures, y = trainTarget, ntree = 100, importance = TRUE)
print(rf_model)
```

```
Call:
 randomForest(x = trainFeatures, y = trainTarget, ntree = 100,      importance = TRUE)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 2

        OOB estimate of  error rate: 0.76%
Confusion matrix:
    0   1 class.error
0 344   6 0.017142857
1   2 698 0.002857143
```

```r
# Faire des prédictions sur l'ensemble de test
predictions <- predict(rf_model, testFeatures)

# Convertir les prédictions en facteur avec les niveaux de testTarget
predictions <- factor(predictions, levels = levels(testTarget))
```

```r
# Évaluer le modèle
confusionMatrix(predictions, testTarget)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 148   2
         1   2 298

               Accuracy : 0.9911
                 95% CI : (0.9774, 0.9976)
    No Information Rate : 0.6667
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.98

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9867
            Specificity : 0.9933
         Pos Pred Value : 0.9867
         Neg Pred Value : 0.9933
             Prevalence : 0.3333
         Detection Rate : 0.3289
   Detection Prevalence : 0.3333
      Balanced Accuracy : 0.9900

       'Positive' Class : 0
```

Utilisation des variables déterminées par StepAIC pour voir si cela améliore le modèle Randomforest

```r
# Préparer les données pour Random Forest en utilisant les variables sélectionnées
selected_vars <- c("height_left", "height_right", "margin_low", "margin_up", "length")

trainData_selected <- trainData[, c(selected_vars, "is_genuine")]
testData_selected <- testData[, c(selected_vars, "is_genuine")]
```

```r
# Convertir 'is_genuine' en facteur
trainData_selected$is_genuine <- as.factor(trainData_selected$is_genuine)
testData_selected$is_genuine <- as.factor(testData_selected$is_genuine)
```

```r
library(randomForest)

# Construire le modèle Random Forest
rf_model_2 <- randomForest(is_genuine ~ ., data = trainData_selected, ntree = 100, importanc

# Afficher les résultats du modèle
print(rf_model_2)
```

```
Call:
 randomForest(formula = is_genuine ~ ., data = trainData_selected,      ntree = 100, importa
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 2

        OOB estimate of  error rate: 0.86%
Confusion matrix:
    0   1 class.error
0 343   7 0.020000000
1   2 698 0.002857143
```

```r
# Prédictions sur le jeu de test
predictions_2 <- predict(rf_model_2, newdata = testData_selected)

# Calculer la matrice de confusion et d'autres statistiques
library(caret)
confusionMatrix(predictions_2, testData_selected$is_genuine)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 148   2
         1   2 298

               Accuracy : 0.9911
                 95% CI : (0.9774, 0.9976)
```

```
    No Information Rate : 0.6667
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.98

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9867
            Specificity : 0.9933
         Pos Pred Value : 0.9867
         Neg Pred Value : 0.9933
             Prevalence : 0.3333
         Detection Rate : 0.3289
   Detection Prevalence : 0.3333
      Balanced Accuracy : 0.9900

       'Positive' Class : 0
```

Les statistiques sont les mêmes qu'avec l'intégralité des variables.

Essayons en modifiant les hyperparamètres de la méthode RandoForest

```r
# Charger les packages nécessaires
if (!require(randomForest)) install.packages("randomForest")
if (!require(caret)) install.packages("caret")
library(randomForest)
library(caret)

# Préparer les données
features <- data[ , names(data) != "is_genuine"]
target <- as.factor(data$is_genuine)

# Diviser les données en ensembles d'entraînement et de test
set.seed(123)
trainIndex <- createDataPartition(target, p = 0.7, list = FALSE)
trainData <- data[trainIndex, ]
testData <- data[-trainIndex, ]

# Préparer les caractéristiques et la cible pour l'entraînement et le test
trainFeatures <- trainData[ , names(trainData) != "is_genuine"]
trainTarget <- as.factor(trainData$is_genuine)
testFeatures <- testData[ , names(testData) != "is_genuine"]
```

```r
testTarget <- as.factor(testData$is_genuine)

# Définir la grille de recherche des hyperparamètres
tuneGrid <- expand.grid(
  mtry = c(2, 3, 4)  # Nombre de variables à essayer à chaque split
)

# Définir le contrôle de la validation croisée
control <- trainControl(method = "cv", number = 5)

# Entraîner le modèle avec recherche des hyperparamètres
rf_model_tuned <- train(
  x = trainFeatures,
  y = trainTarget,
  method = "rf",
  trControl = control,
  tuneGrid = tuneGrid,
  importance = TRUE
)

# Afficher les meilleurs paramètres trouvés
print(rf_model_tuned$bestTune)
```

```
  mtry
1    2
```

```r
# Faire des prédictions sur l'ensemble de test avec le meilleur modèle
predictions <- predict(rf_model_tuned, testFeatures)

# Évaluer le modèle
confusionMatrix(predictions, testTarget)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 148   2
         1   2 298

               Accuracy : 0.9911
                 95% CI : (0.9774, 0.9976)
```

```
    No Information Rate : 0.6667
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.98

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9867
            Specificity : 0.9933
         Pos Pred Value : 0.9867
         Neg Pred Value : 0.9933
             Prevalence : 0.3333
         Detection Rate : 0.3289
   Detection Prevalence : 0.3333
      Balanced Accuracy : 0.9900

       'Positive' Class : 0
```

Essayons avec la méthode de Réseau Neuronal

```
print(head(data))
```

```
  is_genuine diagonal height_left height_right margin_low margin_up length
1          1   171.81      104.86       104.95       4.52      2.89 112.83
2          1   171.46      103.36       103.66       3.77      2.99 113.09
3          1   172.69      104.48       103.50       4.40      2.94 113.16
4          1   171.36      103.91       103.94       3.62      3.01 113.51
5          1   171.73      104.28       103.46       4.04      3.48 112.54
6          1   172.17      103.74       104.08       4.42      2.95 112.81
```

```
print(summary(data))
```

```
   is_genuine        diagonal      height_left     height_right
 Min.   :0.0000   Min.   :171.0   Min.   :103.1   Min.   :102.8
 1st Qu.:0.0000   1st Qu.:171.8   1st Qu.:103.8   1st Qu.:103.7
 Median :1.0000   Median :172.0   Median :104.0   Median :103.9
 Mean   :0.6667   Mean   :172.0   Mean   :104.0   Mean   :103.9
 3rd Qu.:1.0000   3rd Qu.:172.2   3rd Qu.:104.2   3rd Qu.:104.2
 Max.   :1.0000   Max.   :173.0   Max.   :104.9   Max.   :105.0
   margin_low      margin_up         length
```

```
 Min.   :2.980   Min.   :2.270   Min.   :109.5
 1st Qu.:4.020   1st Qu.:2.990   1st Qu.:112.0
 Median :4.310   Median :3.140   Median :113.0
 Mean   :4.483   Mean   :3.151   Mean   :112.7
 3rd Qu.:4.870   3rd Qu.:3.310   3rd Qu.:113.3
 Max.   :6.900   Max.   :3.910   Max.   :114.4
```

```r
print(str(data))
```

```
'data.frame':   1500 obs. of  7 variables:
 $ is_genuine  : num  1 1 1 1 1 1 1 1 1 1 ...
 $ diagonal    : num  172 171 173 171 172 ...
 $ height_left : num  105 103 104 104 104 ...
 $ height_right: num  105 104 104 104 103 ...
 $ margin_low  : num  4.52 3.77 4.4 3.62 4.04 4.42 4.58 3.98 4 4.04 ...
 $ margin_up   : num  2.89 2.99 2.94 3.01 3.48 2.95 3.26 2.92 3.25 3.25 ...
 $ length      : num  113 113 113 114 113 ...
NULL
```

```r
library(reticulate)

# Utilisez le nouvel environnement Conda
use_condaenv("tf-env", required = TRUE)

# Vérifiez la configuration de Python
py_config()
```

```
python:         C:/Users/Utilisateur/Documents/Anaconda/envs/tf-env/python.exe
libpython:      C:/Users/Utilisateur/Documents/Anaconda/envs/tf-env/python39.dll
pythonhome:     C:/Users/Utilisateur/Documents/Anaconda/envs/tf-env
version:        3.9.13 | packaged by conda-forge | (main, May 27 2022, 16:51:29) [MSC v.1929
Architecture:   64bit
numpy:          C:/Users/Utilisateur/Documents/Anaconda/envs/tf-env/Lib/site-packages/numpy
numpy_version:  1.26.4

NOTE: Python version was forced by use_python() function
```

```
library(keras)
```

```
Attachement du package : 'keras'

L'objet suivant est masqué depuis 'package:cvms':

    evaluate
```

```
use_condaenv("tf-env", required=T)
```

Check if graphics card (GPU) is detected:

```
library(tensorflow)
```

```
Attachement du package : 'tensorflow'

L'objet suivant est masqué depuis 'package:caret':

    train

L'objet suivant est masqué depuis 'package:cvms':

    evaluate
```

```
tf$python$client$device_lib$list_local_devices()
```

```
[[1]]
name: "/device:CPU:0"
device_type: "CPU"
memory_limit: 268435456
locality {
}
incarnation: 11650295151070706914
xla_global_id: -1
```

```
[[2]]
name: "/device:GPU:0"
device_type: "GPU"
memory_limit: 5418840064
locality {
  bus_id: 1
  links {
  }
}
incarnation: 17566155605373544504
physical_device_desc: "device: 0, name: NVIDIA GeForce RTX 3060 Ti, pci bus id: 0000:01:00.0
xla_global_id: 416903419
```

```r
# Normalisation des variables continues
df_normalized <- data %>%
  mutate(across(c(diagonal, height_left, height_right, margin_low, margin_up, length), scale)

# Diviser les données en ensembles d'entraînement et de test
set.seed(52)  # Pour la reproductibilité
train_indices <- sample(1:nrow(df_normalized), 0.8 * nrow(df_normalized))
train_data <- df_normalized[train_indices, ]
test_data <- df_normalized[-train_indices, ]

# Préparer les entrées et sorties
x_train <- as.matrix(train_data %>% dplyr::select(everything(), -is_genuine))
y_train <- as.matrix(train_data$is_genuine)

x_test <- as.matrix(test_data %>% dplyr::select(everything(), -is_genuine))
y_test <- as.matrix(test_data$is_genuine)
```

```r
library(keras)

# Créer le modèle
model_neuronal_1 <- keras_model_sequential() %>%
  layer_dense(units = 32, activation = 'relu', input_shape = c(ncol(x_train))) %>%
  layer_dense(units = 16, activation = 'relu') %>%
  layer_dense(units = 1, activation = 'sigmoid')
```

```r
# Compiler le modèle
model_neuronal_1 %>% compile(
  loss = 'binary_crossentropy',
  optimizer = optimizer_adam(),
```

```
  metrics = c('accuracy')
)
```

```
# Évaluer le modèle
score1 <- model_neuronal_1 %>% evaluate(x_test, y_test)
```

```
10/10 - 1s - loss: 0.7722 - accuracy: 0.4733 - 948ms/epoch - 95ms/step
```

```
print(score1)
```

```
     loss  accuracy
0.7721744 0.4733333
```

```
# Créer le modèle
model_neuronal_2 <- keras_model_sequential() %>%
  layer_dense(units = 64, activation = 'relu', input_shape = c(ncol(x_train))) %>%
  layer_dropout(rate = 0.5) %>%
  layer_dense(units = 32, activation = 'relu') %>%
  layer_dropout(rate = 0.5) %>%
  layer_dense(units = 16, activation = 'relu') %>%
  layer_dense(units = 1, activation = 'sigmoid')
```

```
# Compiler le modèle
model_neuronal_2 %>% compile(
  loss = 'binary_crossentropy',
  optimizer = optimizer_adam(learning_rate = 0.001),
  metrics = c('accuracy')
)
```

```
# Évaluer le modèle
score2 <- model_neuronal_2 %>% evaluate(x_test, y_test)
```

```
10/10 - 0s - loss: 0.6681 - accuracy: 0.6667 - 178ms/epoch - 18ms/step
```

```
print(score2)
```

```
     loss  accuracy
0.6680696 0.6666667
```

```r
# Créer un modèle plus simple
model_neuronal_simplified <- keras_model_sequential() %>%
  layer_dense(units = 64, activation = 'relu', input_shape = c(ncol(x_train))) %>%
  layer_dense(units = 32, activation = 'relu') %>%
  layer_dense(units = 16, activation = 'relu') %>%
  layer_dense(units = 1, activation = 'sigmoid')

# Compiler le modèle
model_neuronal_simplified %>% compile(
  loss = 'binary_crossentropy',
  optimizer = optimizer_adam(learning_rate = 0.001),
  metrics = c('accuracy')
)

# Entraîner le modèle
history <- model_neuronal_simplified %>% fit(
  x_train, y_train,
  epochs = 20,  # Augmenter le nombre d'époques si nécessaire
  batch_size = 32,
  validation_split = 0.2
)
```

```
Epoch 1/20
30/30 - 1s - loss: 0.5194 - accuracy: 0.8292 - val_loss: 0.3124 - val_accuracy: 0.9667 - 567r
Epoch 2/20
30/30 - 0s - loss: 0.1855 - accuracy: 0.9875 - val_loss: 0.0959 - val_accuracy: 0.9875 - 120r
Epoch 3/20
30/30 - 0s - loss: 0.0607 - accuracy: 0.9906 - val_loss: 0.0528 - val_accuracy: 0.9875 - 101r
Epoch 4/20
30/30 - 0s - loss: 0.0359 - accuracy: 0.9917 - val_loss: 0.0484 - val_accuracy: 0.9875 - 112r
Epoch 5/20
30/30 - 0s - loss: 0.0301 - accuracy: 0.9927 - val_loss: 0.0480 - val_accuracy: 0.9875 - 107r
Epoch 6/20
30/30 - 0s - loss: 0.0273 - accuracy: 0.9927 - val_loss: 0.0465 - val_accuracy: 0.9875 - 112r
Epoch 7/20
30/30 - 0s - loss: 0.0255 - accuracy: 0.9927 - val_loss: 0.0471 - val_accuracy: 0.9875 - 108r
Epoch 8/20
30/30 - 0s - loss: 0.0248 - accuracy: 0.9927 - val_loss: 0.0500 - val_accuracy: 0.9875 - 104r
Epoch 9/20
30/30 - 0s - loss: 0.0239 - accuracy: 0.9927 - val_loss: 0.0459 - val_accuracy: 0.9875 - 115r
Epoch 10/20
30/30 - 0s - loss: 0.0232 - accuracy: 0.9927 - val_loss: 0.0497 - val_accuracy: 0.9875 - 107r
```

```
Epoch 11/20
30/30 - 0s - loss: 0.0225 - accuracy: 0.9927 - val_loss: 0.0464 - val_accuracy: 0.9875 - 102r
Epoch 12/20
30/30 - 0s - loss: 0.0221 - accuracy: 0.9927 - val_loss: 0.0529 - val_accuracy: 0.9875 - 98ms
Epoch 13/20
30/30 - 0s - loss: 0.0205 - accuracy: 0.9927 - val_loss: 0.0485 - val_accuracy: 0.9875 - 110r
Epoch 14/20
30/30 - 0s - loss: 0.0202 - accuracy: 0.9927 - val_loss: 0.0486 - val_accuracy: 0.9875 - 105r
Epoch 15/20
30/30 - 0s - loss: 0.0195 - accuracy: 0.9927 - val_loss: 0.0495 - val_accuracy: 0.9875 - 112r
Epoch 16/20
30/30 - 0s - loss: 0.0191 - accuracy: 0.9937 - val_loss: 0.0494 - val_accuracy: 0.9875 - 107r
Epoch 17/20
30/30 - 0s - loss: 0.0185 - accuracy: 0.9937 - val_loss: 0.0500 - val_accuracy: 0.9875 - 112r
Epoch 18/20
30/30 - 0s - loss: 0.0180 - accuracy: 0.9937 - val_loss: 0.0525 - val_accuracy: 0.9875 - 110r
Epoch 19/20
30/30 - 0s - loss: 0.0176 - accuracy: 0.9958 - val_loss: 0.0482 - val_accuracy: 0.9875 - 103r
Epoch 20/20
30/30 - 0s - loss: 0.0168 - accuracy: 0.9948 - val_loss: 0.0517 - val_accuracy: 0.9875 - 110r
```

```r
# Évaluer le modèle
score_simplified <- model_neuronal_simplified %>% evaluate(x_test, y_test)
```

```
10/10 - 0s - loss: 0.0321 - accuracy: 0.9867 - 33ms/epoch - 3ms/step
```

```r
print(score_simplified)
```

```
      loss   accuracy
0.03211671 0.98666668
```

```r
# Créer un modèle avec Dropout
model_neuronal_dropout <- keras_model_sequential() %>%
  layer_dense(units = 64, activation = 'relu', input_shape = c(ncol(x_train))) %>%
  layer_dropout(rate = 0.5) %>%
  layer_dense(units = 32, activation = 'relu') %>%
  layer_dropout(rate = 0.5) %>%
  layer_dense(units = 16, activation = 'relu') %>%
  layer_dense(units = 1, activation = 'sigmoid')
```

```r
# Compiler le modèle
model_neuronal_dropout %>% compile(
  loss = 'binary_crossentropy',
  optimizer = optimizer_adam(learning_rate = 0.001),
  metrics = c('accuracy')
)

# Entraîner le modèle
history_dropout <- model_neuronal_dropout %>% fit(
  x_train, y_train,
  epochs = 20,
  batch_size = 32,
  validation_split = 0.2
)
```

```
Epoch 1/20
30/30 - 1s - loss: 0.5976 - accuracy: 0.7094 - val_loss: 0.4654 - val_accuracy: 0.9333 - 564r
Epoch 2/20
30/30 - 0s - loss: 0.4066 - accuracy: 0.8594 - val_loss: 0.2423 - val_accuracy: 0.9792 - 101r
Epoch 3/20
30/30 - 0s - loss: 0.2269 - accuracy: 0.9510 - val_loss: 0.1009 - val_accuracy: 0.9875 - 112r
Epoch 4/20
30/30 - 0s - loss: 0.1347 - accuracy: 0.9635 - val_loss: 0.0605 - val_accuracy: 0.9875 - 110r
Epoch 5/20
30/30 - 0s - loss: 0.0841 - accuracy: 0.9771 - val_loss: 0.0543 - val_accuracy: 0.9875 - 105r
Epoch 6/20
30/30 - 0s - loss: 0.0697 - accuracy: 0.9833 - val_loss: 0.0548 - val_accuracy: 0.9875 - 103r
Epoch 7/20
30/30 - 0s - loss: 0.0630 - accuracy: 0.9812 - val_loss: 0.0563 - val_accuracy: 0.9875 - 109r
Epoch 8/20
30/30 - 0s - loss: 0.0614 - accuracy: 0.9875 - val_loss: 0.0574 - val_accuracy: 0.9875 - 105r
Epoch 9/20
30/30 - 0s - loss: 0.0510 - accuracy: 0.9833 - val_loss: 0.0581 - val_accuracy: 0.9875 - 99ms
Epoch 10/20
30/30 - 0s - loss: 0.0426 - accuracy: 0.9875 - val_loss: 0.0601 - val_accuracy: 0.9875 - 104r
Epoch 11/20
30/30 - 0s - loss: 0.0485 - accuracy: 0.9854 - val_loss: 0.0593 - val_accuracy: 0.9875 - 102r
Epoch 12/20
30/30 - 0s - loss: 0.0512 - accuracy: 0.9854 - val_loss: 0.0606 - val_accuracy: 0.9875 - 102r
Epoch 13/20
30/30 - 0s - loss: 0.0400 - accuracy: 0.9885 - val_loss: 0.0595 - val_accuracy: 0.9875 - 111r
Epoch 14/20
```

```
30/30 - 0s - loss: 0.0451 - accuracy: 0.9885 - val_loss: 0.0585 - val_accuracy: 0.9875 - 105r
Epoch 15/20
30/30 - 0s - loss: 0.0445 - accuracy: 0.9875 - val_loss: 0.0601 - val_accuracy: 0.9875 - 113r
Epoch 16/20
30/30 - 0s - loss: 0.0439 - accuracy: 0.9865 - val_loss: 0.0614 - val_accuracy: 0.9875 - 106r
Epoch 17/20
30/30 - 0s - loss: 0.0427 - accuracy: 0.9875 - val_loss: 0.0600 - val_accuracy: 0.9875 - 111r
Epoch 18/20
30/30 - 0s - loss: 0.0376 - accuracy: 0.9927 - val_loss: 0.0607 - val_accuracy: 0.9875 - 108r
Epoch 19/20
30/30 - 0s - loss: 0.0513 - accuracy: 0.9906 - val_loss: 0.0585 - val_accuracy: 0.9875 - 99ms
Epoch 20/20
30/30 - 0s - loss: 0.0288 - accuracy: 0.9927 - val_loss: 0.0579 - val_accuracy: 0.9875 - 104r
```

```r
# Évaluer le modèle
score_dropout <- model_neuronal_dropout %>% evaluate(x_test, y_test)
```

```
10/10 - 0s - loss: 0.0357 - accuracy: 0.9867 - 32ms/epoch - 3ms/step
```

```r
print(score_dropout)
```

```
      loss   accuracy
0.03571304 0.98666668
```

```r
library(keras)

# Créer un modèle avec Dropout
model_neuronal_3 <- keras_model_sequential() %>%
  layer_dense(units = 64, activation = 'relu', input_shape = c(ncol(x_train))) %>%
  layer_dropout(rate = 0.5) %>%
  layer_dense(units = 32, activation = 'relu') %>%
  layer_dropout(rate = 0.5) %>%
  layer_dense(units = 16, activation = 'relu') %>%
  layer_dense(units = 1, activation = 'sigmoid')

# Compiler le modèle
model_neuronal_3 %>% compile(
  loss = 'binary_crossentropy',
  optimizer = optimizer_adam(learning_rate = 0.001),
  metrics = c('accuracy')
```

```
)

# Ajouter un callback pour l'arrêt précoce
early_stopping <- callback_early_stopping(
  monitor = 'val_loss',
  patience = 3,
  restore_best_weights = TRUE
)

# Entraîner le modèle
history <- model_neuronal_3 %>% fit(
  x_train, y_train,
  epochs = 20,
  batch_size = 32,
  validation_split = 0.2,
  callbacks = list(early_stopping)
)
```

```
Epoch 1/20
30/30 - 1s - loss: 0.7187 - accuracy: 0.4698 - val_loss: 0.5541 - val_accuracy: 0.9292 - 571m
Epoch 2/20
30/30 - 0s - loss: 0.5050 - accuracy: 0.8583 - val_loss: 0.3462 - val_accuracy: 0.9792 - 107m
Epoch 3/20
30/30 - 0s - loss: 0.3246 - accuracy: 0.9385 - val_loss: 0.1474 - val_accuracy: 0.9875 - 106m
Epoch 4/20
30/30 - 0s - loss: 0.1634 - accuracy: 0.9708 - val_loss: 0.0739 - val_accuracy: 0.9875 - 104m
Epoch 5/20
30/30 - 0s - loss: 0.0985 - accuracy: 0.9781 - val_loss: 0.0559 - val_accuracy: 0.9875 - 108m
Epoch 6/20
30/30 - 0s - loss: 0.0807 - accuracy: 0.9781 - val_loss: 0.0523 - val_accuracy: 0.9875 - 110m
Epoch 7/20
30/30 - 0s - loss: 0.0649 - accuracy: 0.9781 - val_loss: 0.0527 - val_accuracy: 0.9875 - 103m
Epoch 8/20
30/30 - 0s - loss: 0.0596 - accuracy: 0.9833 - val_loss: 0.0529 - val_accuracy: 0.9875 - 102m
Epoch 9/20
30/30 - 0s - loss: 0.0425 - accuracy: 0.9885 - val_loss: 0.0553 - val_accuracy: 0.9875 - 109m
```

```
# Évaluer le modèle
score3 <- model_neuronal_3 %>% evaluate(x_test, y_test)
```

```
10/10 - 0s - loss: 0.0400 - accuracy: 0.9867 - 33ms/epoch - 3ms/step
```

```r
print(score3)
```

```
      loss   accuracy
0.03997809 0.98666668
```

```r
library(keras)

# Créer un modèle avec Dropout
model_neuronal_4 <- keras_model_sequential() %>%
  layer_dense(units = 64, activation = 'relu', input_shape = c(ncol(x_train))) %>%
  layer_dropout(rate = 0.5) %>%
  layer_dense(units = 32, activation = 'relu') %>%
  layer_dropout(rate = 0.5) %>%
  layer_dense(units = 16, activation = 'relu') %>%
  layer_dense(units = 1, activation = 'sigmoid')

# Compiler le modèle
model_neuronal_4 %>% compile(
  loss = 'binary_crossentropy',
  optimizer = optimizer_adam(learning_rate = 0.001),
  metrics = c('accuracy')
)

# Ajouter un callback pour l'arrêt précoce
early_stopping <- callback_early_stopping(
  monitor = 'val_loss',
  patience = 3,
  restore_best_weights = TRUE
)

# Entraîner le modèle
history <- model_neuronal_4 %>% fit(
  x_train, y_train,
  epochs = 20,
  batch_size = 32,
  validation_split = 0.2,
  callbacks = list(early_stopping)
)
```

```
Epoch 1/20
30/30 - 1s - loss: 0.5827 - accuracy: 0.7167 - val_loss: 0.4639 - val_accuracy: 0.9458 - 573r
```

```
Epoch 2/20
30/30 - 0s - loss: 0.4009 - accuracy: 0.8406 - val_loss: 0.2494 - val_accuracy: 0.9792 - 105r
Epoch 3/20
30/30 - 0s - loss: 0.2413 - accuracy: 0.9250 - val_loss: 0.1311 - val_accuracy: 0.9792 - 110r
Epoch 4/20
30/30 - 0s - loss: 0.1457 - accuracy: 0.9667 - val_loss: 0.0723 - val_accuracy: 0.9833 - 104r
Epoch 5/20
30/30 - 0s - loss: 0.0922 - accuracy: 0.9812 - val_loss: 0.0553 - val_accuracy: 0.9833 - 107r
Epoch 6/20
30/30 - 0s - loss: 0.0776 - accuracy: 0.9781 - val_loss: 0.0517 - val_accuracy: 0.9833 - 103r
Epoch 7/20
30/30 - 0s - loss: 0.0658 - accuracy: 0.9802 - val_loss: 0.0529 - val_accuracy: 0.9875 - 105r
Epoch 8/20
30/30 - 0s - loss: 0.0641 - accuracy: 0.9823 - val_loss: 0.0522 - val_accuracy: 0.9875 - 110r
Epoch 9/20
30/30 - 0s - loss: 0.0476 - accuracy: 0.9865 - val_loss: 0.0518 - val_accuracy: 0.9875 - 115r
```

```r
# Évaluer le modèle
score4 <- model_neuronal_4 %>% evaluate(x_test, y_test)
```

```
10/10 - 0s - loss: 0.0451 - accuracy: 0.9833 - 36ms/epoch - 4ms/step
```

```r
print(score4)
```

```
      loss   accuracy
0.04514809 0.98333335
```

```r
# 1. Faire des prédictions sur les données de test
y_pred <- model_neuronal_4 %>% predict(x_test)
```

```
10/10 - 0s - 93ms/epoch - 9ms/step
```

```r
# 2. Convertir les probabilités en classes (0 ou 1)
y_pred_class <- ifelse(y_pred > 0.5, 1, 0)

# 3. Créer la matrice de confusion
conf_matrix <- table(Predicted = y_pred_class, Actual = y_test)

# 4. Afficher la matrice de confusion
print(conf_matrix)
```

```
         Actual
Predicted   0   1
        0  93   4
        1   1 202
```

```r
# Installer et charger le package caret si ce n'est pas déjà fait
# install.packages("caret")
library(caret)

# Utiliser caret pour obtenir une matrice de confusion plus détaillée
confusionMatrix(as.factor(y_pred_class), as.factor(y_test))
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0  93   4
         1   1 202

               Accuracy : 0.9833
                 95% CI : (0.9615, 0.9946)
    No Information Rate : 0.6867
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9616

 Mcnemar's Test P-Value : 0.3711

            Sensitivity : 0.9894
            Specificity : 0.9806
         Pos Pred Value : 0.9588
         Neg Pred Value : 0.9951
             Prevalence : 0.3133
         Detection Rate : 0.3100
   Detection Prevalence : 0.3233
      Balanced Accuracy : 0.9850

       'Positive' Class : 0
```

Sauvegarde du modèle à utiliser

```r
# Sauvegarder le modèle neuronales (librairie keras)
model_neuronal_4 %>% save_model_hdf5("model_faux_billets.h5")
```

```r
# Sauvegarder le modèle et le seuil
saveRDS(reg_log_model, "modele_regression_logistique.rds")
saveRDS(best_threshold, "seuil_optimal.rds")
```