TP Analyse des données

Zhang shurong Meng fangzhou

5/3/2021

Partie 4: 1) On transforme les variables qualitatives aux variables quantitatives.

```
donnees<-read.table("desbois.csv",header=T,sep=";",dec=",",na.strings="NA")
row.names(donnees) < as.character(donnees$Num)
```

```
##    [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   [97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [121] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [133] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [145] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [157] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [169] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [181] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [193] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [205] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [217] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [229] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [241] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [253] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [265] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [277] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [289] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [301] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [313] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [325] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [337] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [349] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [361] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [373] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [385] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [397] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [409] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [421] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
##  [433] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [445] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [457] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [469] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [481] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [493] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [505] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [517] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [529] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [541] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [553] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [565] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [577] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [589] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [601] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [613] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [625] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [637] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [649] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [661] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [673] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [685] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [697] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [709] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [721] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [733] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [745] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [757] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [769] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [781] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [793] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [805] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [817] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [829] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [841] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [853] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [865] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [877] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [889] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [901] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [913] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [925] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [937] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [949] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [961] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [973] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [985] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [997] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1009] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1021] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1033] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1045] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1057] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1069] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
## [1081] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1093] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1105] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1117] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1129] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1141] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1153] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1165] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1177] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1189] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1201] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1213] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1225] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1237] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [1249] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
donnees$Num<-NULL
donnees$DIFF<- factor(donnees$DIFF,label=c('healthy','failing'))
donnees$CNTY<- factor(donnees$CNTY)
#donnees$CNTY<- Null
donnees$OWNLAND<- factor(donnees$OWNLAND,label=c('Yes','No'))
donnees$ToF<- factor(donnees$ToF)
donnees$STATUS<- factor(donnees$STATUS)
```

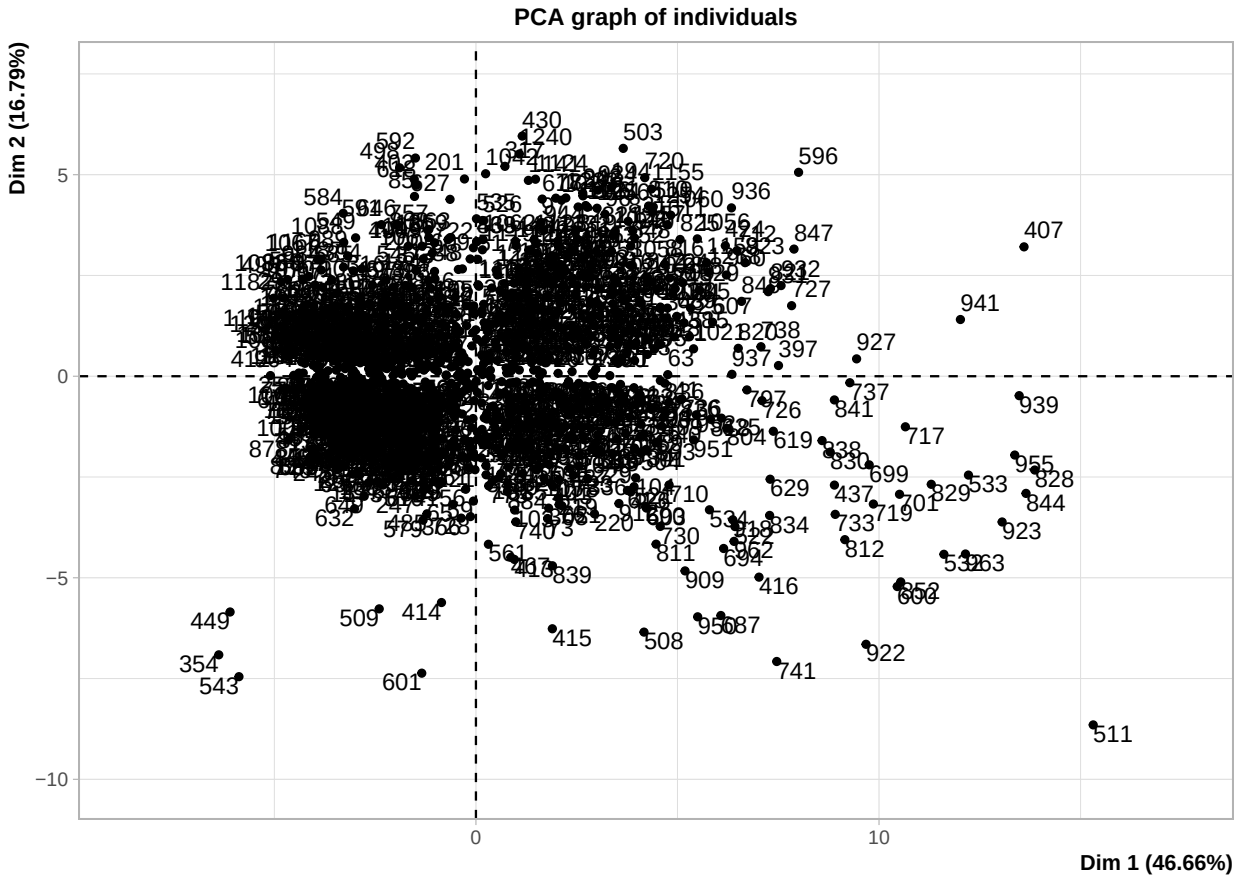
2)

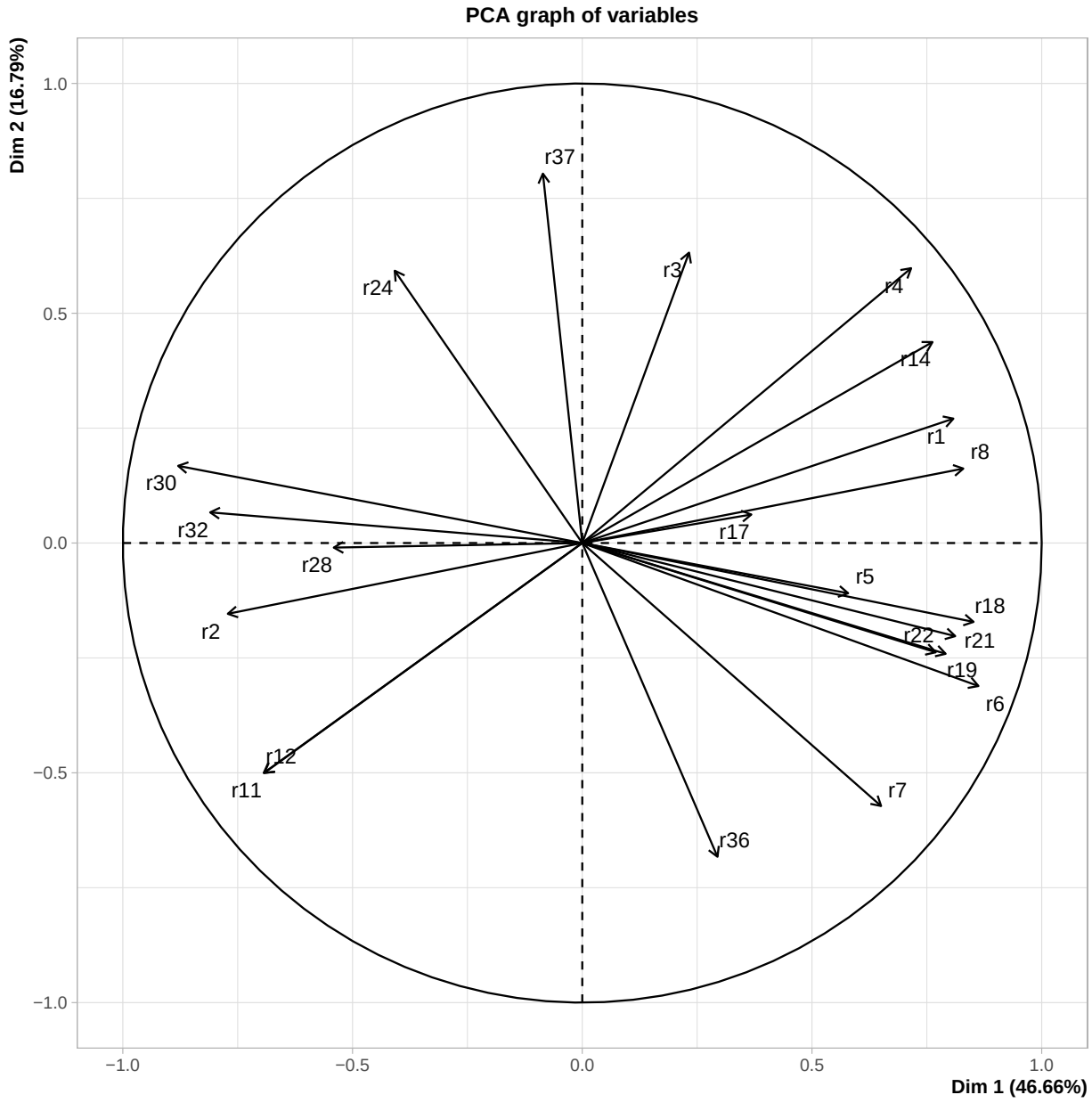
```
varACP<-donnees[,9:30] #Chercher toutes les variables du colonne 9 au colonne 30
```

Et on va étudier l'ACP sur ce tableau extrait.

3)

```
library(FactoMineR) # faire appel a library factorminer
res.pca=PCA(varACP,scale.unit = TRUE,graph=T, ncp=22)
```

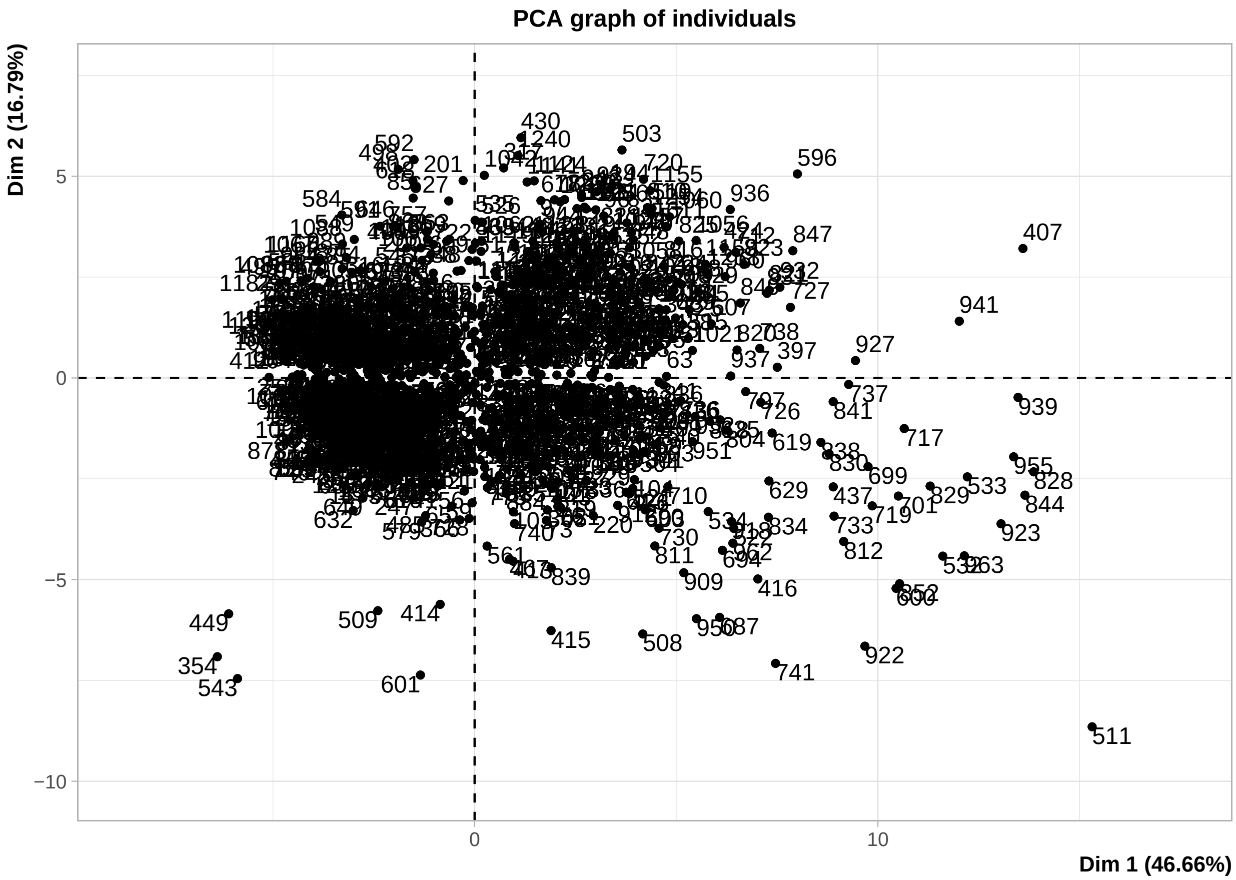

PCA graph of individuals
Dim 2 (16.79%)
Dim 1 (46.66%)
5
0
−5
−10
0
10
430
503
592
201
596
936
847
407
727
941
927
397
737
939
717
955
828
844
923
449
509
414
415
508
922
741
354
543
601
511



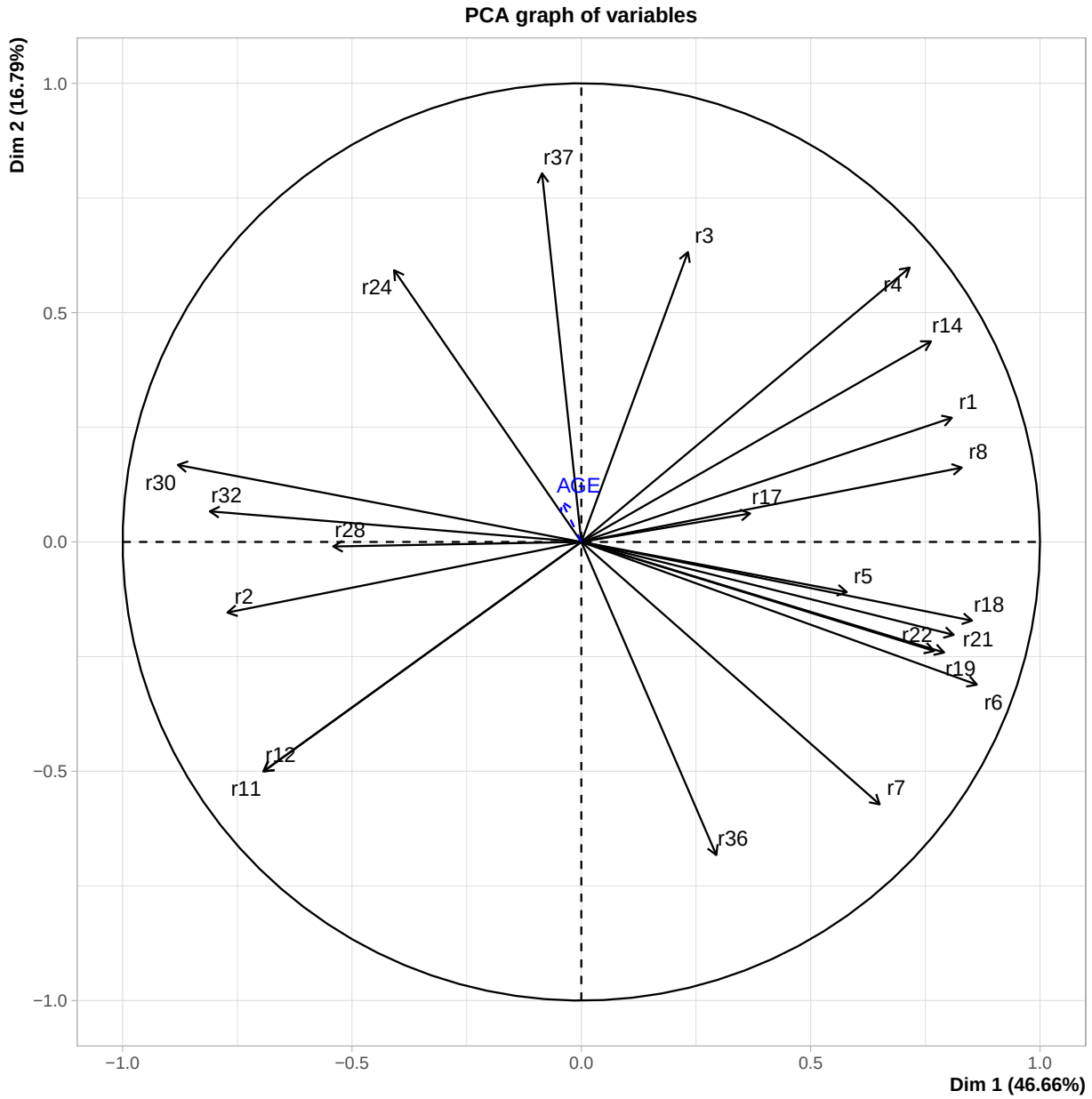
On remarque qu'il y a 46.66% informations sur le premier axe, 16.79% informations sur le second axe. Ensuite, on voit que r37 et r36 sont bien représentés car ils ont les flèches long(leur qualtiés sont très proches que 1). Ainsi, r37 est corrélé positivement à l'axe2 et r36 est corrélé négativement à l'axe2. Cependant, r36 et r37 sont très peu corrélés à l'axe1. De plus, on a r5 et r28 sont mal représentrés(les flèches courtes). Car on perd des informations quand on fait la projection.

4) On peut ajouter les variables illustratives, ici on ajoute la variable "age", puis on refait l'ACP avec "age"

```
varACP<-donnees[,c(7,9:30)]
res.pca=PCA(varACP,scale.unit=TRUE,quanti.sup=1,graph=T,ncp=22)
```



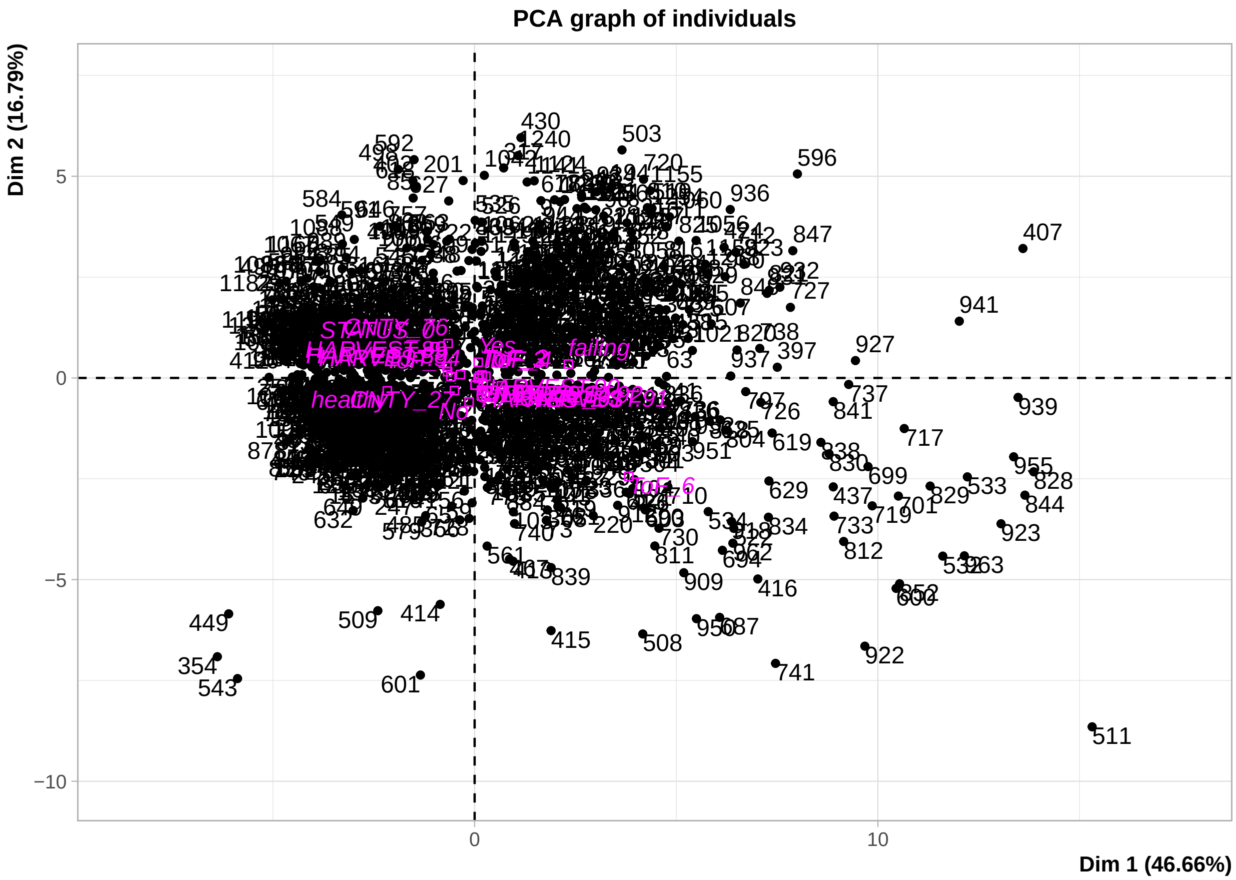
PCA graph of individuals
Dim 2 (16.79%)
Dim 1 (46.66%)

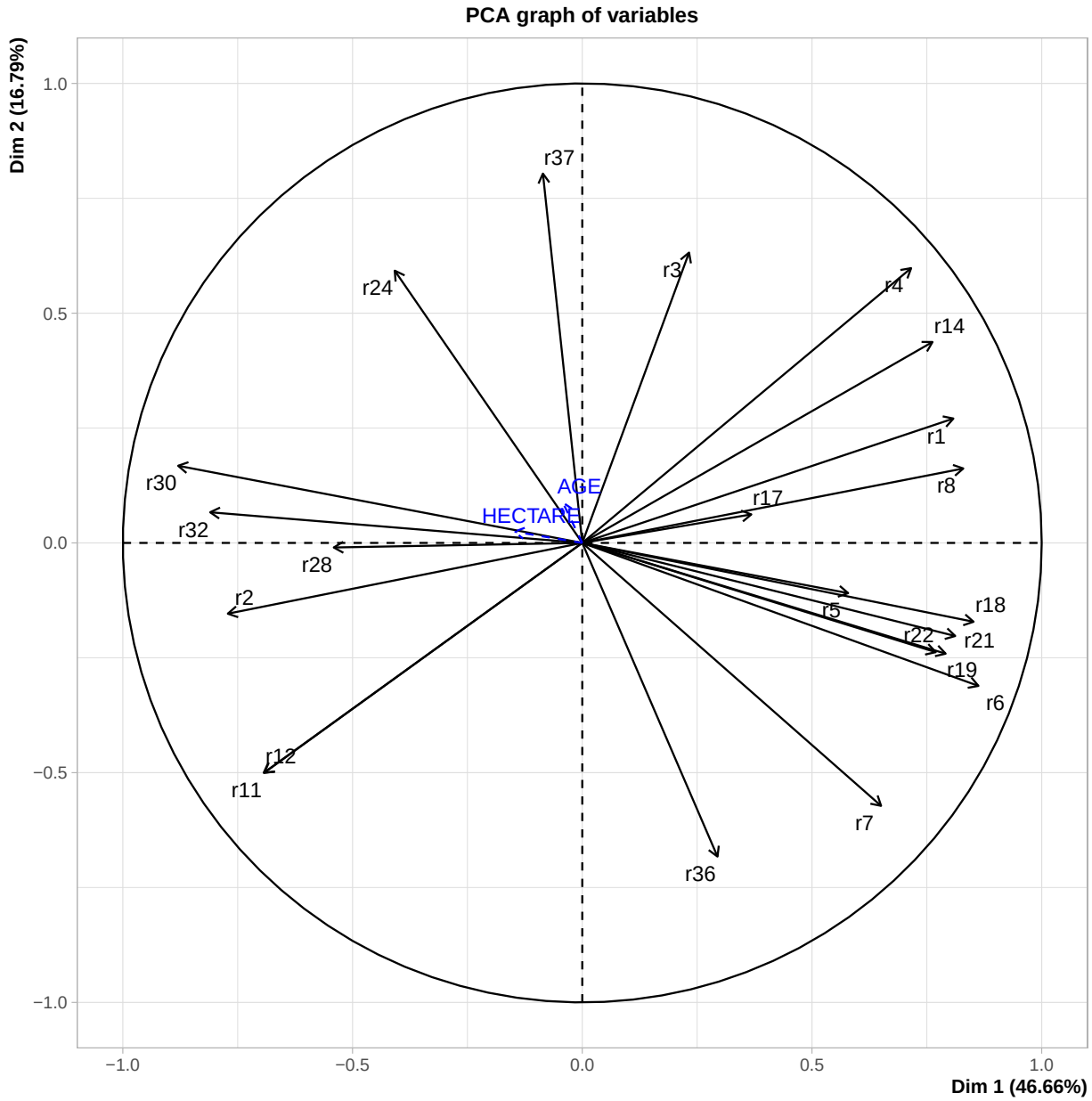


On voit qu'il fait la même chose pour le calcul des axes mais le variable "AGE" est très mal projétée sur l'axe1 ou l'axe2. Donc il n'est pas lié à la première dimension. Et il est déssiné en bleu, cela est aussi un moyen de voir qu'on ne s'est pas trompé des extractions des données, et on a le même pourcentrage d'inertie partout.

5) L'option quali.sup permet aussi de rajouter une ou plusieurs variable-s qualitative-s en illustratif en précisant leur place dans le fichier.

```
qual<-c(1,2,3,5,6,8)#les variables qualitatives
quant<-c(4,7)#les variables quantitatives
res.pca=PCA(donnees,scale.unit=TRUE,quali.sup=qual,quanti.sup = quant,graph=T,ncp=22)
```


PCA graph of individuals
Dim 2 (16.79%)
Dim 1 (46.66%)
5
0
−5
−10
0
10



On remarque que les variables “AGE” et “HECTARE” sont les variables quantitatives et les six autres premiers variables sont les variables qualitatives.Et on obtient le même résultat comme précédemment sauf qu’on a ajouté la variable “HECTARE”. Cette variable est très mal représentée car sa flèche est très courte. Donc c’est difficile de l’interpréter.

Partie 5: 1)2)3)

```
res.pca$eig
```

```
##          eigenvalue percentage of variance cumulative percentage of variance
## comp 1  10.264920946            46.65873157                           46.65873
## comp 2   3.692948498            16.78612954                           63.44486
## comp 3   2.469798802            11.22635819                           74.67122
## comp 4   1.581187667             7.18721667                           81.85844
```

```
## comp 5   1.191288024            5.41494556                    87.27338
## comp 6   0.826548474            3.75703852                    91.03042
## comp 7   0.598006136            2.71820971                    93.74863
## comp 8   0.386044115            1.75474598                    95.50338
## comp 9   0.250361757            1.13800799                    96.64138
## comp 10  0.167046145            0.75930066                    97.40068
## comp 11  0.137042488            0.62292040                    98.02360
## comp 12  0.127531273            0.57968760                    98.60329
## comp 13  0.093677608            0.42580731                    99.02910
## comp 14  0.058506740            0.26593973                    99.29504
## comp 15  0.044486406            0.20221094                    99.49725
## comp 16  0.040319183            0.18326901                    99.68052
## comp 17  0.022568574            0.10258443                    99.78310
## comp 18  0.014627243            0.06648747                    99.84959
## comp 19  0.012014521            0.05461146                    99.90420
## comp 20  0.010473116            0.04760507                    99.95181
## comp 21  0.006360138            0.02890972                    99.98072
## comp 22  0.004242146            0.01928248                   100.00000
```

La première colonne est la valeur de la valeur propre(par exemle la première vaut 10.26). La deuxième colonne est le pourcentage de variance(par exemple 46.65% pour la première valeur propre) et on rappelle qu'on peut le calculer par (valeur propre/22 *100). La troisième colonne est le pourcentrage du premier axe, du premier plan factoriel, du premier espace de dimension 3, cette pourcentage est d'inertie conservée par la projection sur d'espace qu'on rajoute chaque fois une dimension plus(par exemple, 46.66% d'inerties sont expliquées par le premier composant principale,63.44% d'inerties sont expliquées par les deux premiers composants pricipales. . .)

On peut récupérer les valeurs propres en prennant la première colonne :

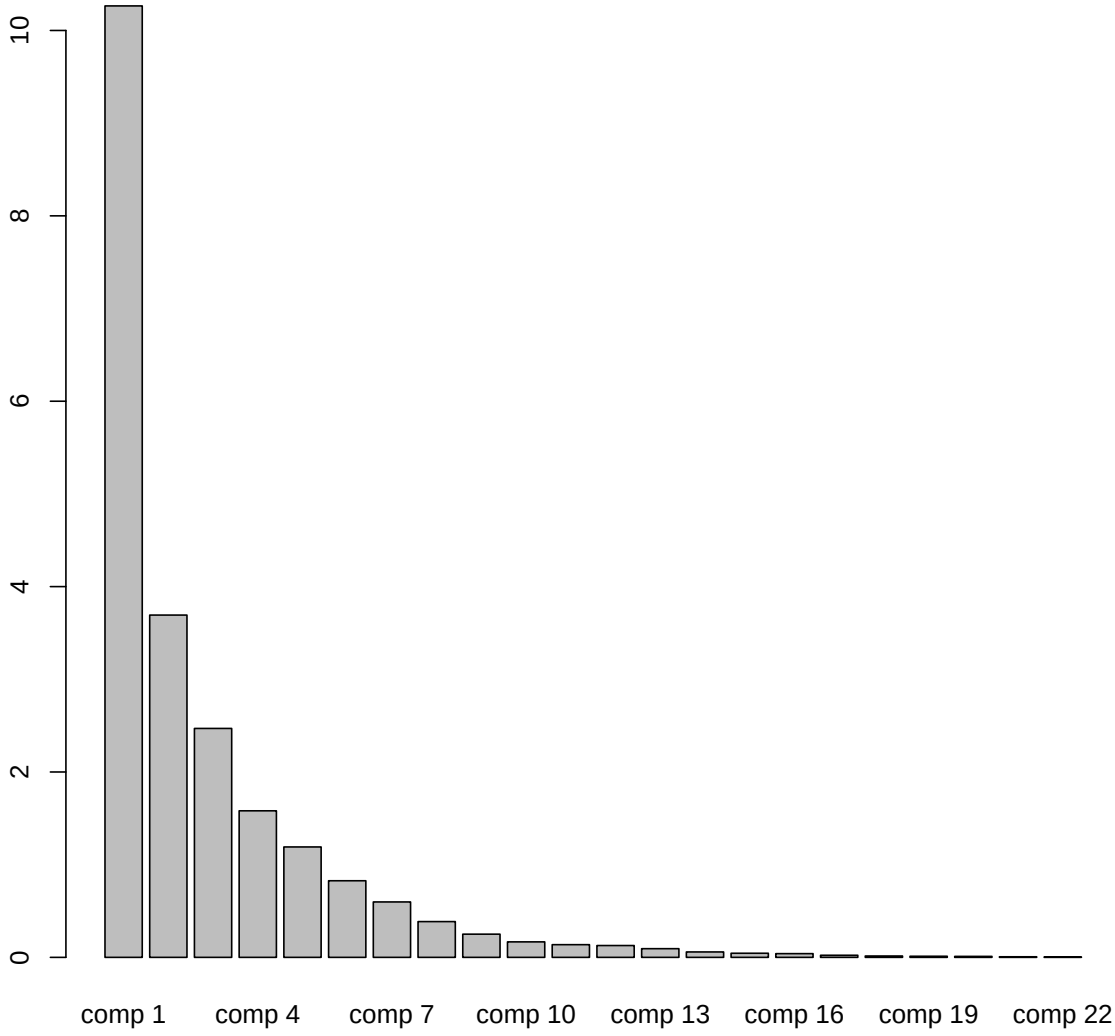
```
vp<-res.pca$eig[,1]
vp
```

```
##       comp 1       comp 2       comp 3       comp 4       comp 5       comp 6
## 10.264920946  3.692948498  2.469798802  1.581187667  1.191288024  0.826548474
##       comp 7       comp 8       comp 9      comp 10      comp 11      comp 12
##  0.598006136  0.386044115  0.250361757  0.167046145  0.137042488  0.127531273
##      comp 13      comp 14      comp 15      comp 16      comp 17      comp 18
##  0.093677608  0.058506740  0.044486406  0.040319183  0.022568574  0.014627243
##      comp 19      comp 20      comp 21      comp 22
##  0.012014521  0.010473116  0.006360138  0.004242146
```

```
sum(vp)
```

```
## [1] 22
```

```
barplot(vp)
```



On vérifie que la somme de toutes les valeurs propres est égale à 22. On remarque les valeurs propres de la composante 1 à la composante 5 qui est supérieure à 1(Règle de Kaiser), et on peut faire le "break" après 5 composantes(Règle d'Elbow), puis on garde q=5 les composantes principales pour décrire les entreprises qui était décrit par 22 variables. Et on perd juste 12.8% de l'information (Inertie des données). Mais on fait attention qu'on ne coupe pas les valeurs propres qui sont proches entre elles.

4)

```
ncp<-10
psi<-res.pca$ind$coord
cos2<-res.pca$ind$cos2
#pour un individu
v1<-psi[1,]
v12<-v1^2
```

```
qlt11<-v12[1]/sum(v12)
cos2[1,]
```

```
##        Dim.1        Dim.2        Dim.3        Dim.4        Dim.5        Dim.6
## 4.995183e-01 1.167633e-02 2.141767e-01 7.685096e-02 9.274840e-02 6.003811e-02
##        Dim.7        Dim.8        Dim.9       Dim.10       Dim.11       Dim.12
## 7.223461e-03 1.715586e-02 6.113384e-03 2.537858e-03 1.270790e-04 2.432885e-05
##       Dim.13       Dim.14       Dim.15       Dim.16       Dim.17       Dim.18
## 3.936356e-04 4.303750e-03 3.520076e-04 2.018902e-04 7.087604e-07 1.705804e-03
##       Dim.19       Dim.20       Dim.21       Dim.22
## 3.864900e-03 2.456350e-04 1.286868e-04 6.122249e-04
```

```
qlt12<-sum(v12[1:2])/sum(v12)
qlt16<-sum(v12[1:ncp])/sum(v12)
qlt11
```

```
##     Dim.1
## 0.4995183
```

```
qlt12
```

```
## [1] 0.5111947
```

```
qlt16
```

```
## [1] 0.9880393
```

On a trouvé exactement la même chose pour le premier élément de cos2[1,] et qlt11. Car la commande de cos2 dans Factominer nous donne la qualité de projection l'individu1 sur les différents axes.

On peut aussi faire pour tous les individus:

```
#pour tous les individus
psi2<-psi*psi
head(psi2)
```

```
##       Dim.1        Dim.2      Dim.3     Dim.4      Dim.5     Dim.6        Dim.7
## 1  6.928094 0.1619454579 2.97053359 1.0658881 1.28637847 0.8327015 1.001862e-01
## 2 10.560493 0.7937683618 3.87748418 0.1610651 0.91205243 0.9577653 1.086038e-01
## 3  4.221470 0.0001875542 2.25224759 0.3778563 0.45247888 0.2491030 2.553573e-05
## 4  8.185969 0.1474111352 0.28246600 2.6791712 0.01089057 0.4170123 1.510294e-02
## 5  9.335273 0.2893395057 1.38636527 0.4473248 0.10887503 2.0495623 2.495947e-02
## 6 14.103383 1.7811896672 0.03565741 0.2237325 1.57682176 0.8659415 4.814630e-03
##         Dim.8      Dim.9       Dim.10       Dim.11       Dim.12       Dim.13
## 1 0.237943990 0.08478988 0.0351989393 0.0017625281 0.0003374302 0.0054595485
## 2 0.046247000 0.01867690 0.0004900831 0.1148545476 0.0588778530 0.0215182941
## 3 0.003096761 0.04255478 0.0057230271 0.0888054062 0.0187370814 0.0011559551
## 4 0.298157439 0.05727785 0.0505782466 0.0010388053 0.0238730651 0.0004304480
## 5 0.215064458 0.23251457 0.0137766752 0.0001110777 0.0161999711 0.0005249215
## 6 0.229035218 0.30974846 0.0200872923 0.0111468550 0.0108816851 0.0029506642
```

```
##         Dim.14      Dim.15       Dim.16       Dim.17       Dim.18       Dim.19
## 1 5.969107e-02 0.004882187 0.0028001259 9.830187e-06 0.0236587292 5.360442e-02
## 2 4.056324e-02 0.008722809 0.0126699166 2.192452e-04 0.0056571008 6.534319e-04
## 3 1.154977e-03 0.002141684 0.0038567474 1.394257e-09 0.0001961925 1.631121e-05
## 4 2.342496e-03 0.002725369 0.0004706312 3.053916e-04 0.0555307788 1.034611e-02
## 5 1.006739e-05 0.006723871 0.0084618657 2.563653e-04 0.0093429417 4.784119e-03
## 6 1.060263e-01 0.008954722 0.0054180612 2.866102e-04 0.0046290968 1.328167e-03
##         Dim.20       Dim.21       Dim.22
## 1 3.406846e-03 1.784829e-03 0.0084912833
## 2 2.155366e-04 2.269974e-04 0.0008080505
## 3 7.634348e-05 3.880512e-05 0.0001468376
## 4 1.913895e-05 9.749397e-04 0.0081014470
## 5 1.852241e-11 1.030089e-03 0.0162605478
## 6 4.699948e-05 2.731193e-07 0.0009800920
```

```
s<-apply(psi2[,], MARGIN=1,FUN=sum)
s1<-psi2[,1] #sur le premier axe factoriel
qlt1<-s1/s
s2<-apply(psi2[,1:2], MARGIN=1,FUN=sum) #le premier plan factoriel
qlt2<-s2/s
#le sous-espace engendré par le nombre de com- posantes principales
sncp<-apply(psi2[,1:ncp], MARGIN=1,FUN=sum)
qlt<-sncp/s
max(qlt1)
```

```
## [1] 0.9584851
```

```
min(qlt1)
```

```
## [1] 2.619245e-06
```

```
which(qlt1==min(qlt1))
```

```
## 1151
## 1151
```

```
which(qlt1==max(qlt1))
```

```
## 770
## 770
```

```
max(qlt2)
```

```
## [1] 0.9602937
```

```
min(qlt2)
```

```
## [1] 0.002148773
```

```
which(qlt2==min(qlt2))
```

```
## 1128
## 1128
```

```
which(qlt2==max(qlt2))
```

```
## 907
## 907
```

```
max(qlt)
```

```
## [1] 0.9987141
```

```
min(qlt)
```

```
## [1] 0.8579129
```

On remarque que

• l'individu qui est mieux représenté sur le premier axe factoriel,il est représenté à 0.9584851 qui est très proche de 1, et on a ainsi l'individu qui est très mal représenté sur le premier axe factoriel, il est représenté à 2.619245e^-06.

• l'individu qui est mieux représenté sur le premier plan factoriel,il est représenté à 0.9602937 qui est très proche de 1, et on a ainsi l'individu qui est très mal représenté sur le premier plan factoriel, il est représenté à 0.002148773.

• l'individu qui est mieux représenté sur le sous-espace engendré par le nombre de com posantes principales, il est représenté à 0.9987141 qui est très proche de 1, et on a ainsi l'individu qui est très mal représenté sur le sous-espace engendré par le nombre de composantes principales, il est représenté à 0.8579129.

5)

```
ncp<-2
#qualité du plan
sncp<-apply(psi2[,1:ncp], MARGIN=1,FUN=sum)
qlt<-sncp/s

#qlt
max(qlt)
```

```
## [1] 0.9602937
```

```
min(qlt)
```

```
## [1] 0.002148773
```

```r
which(qlt==min(qlt))
```

```
## 1128
## 1128
```

```r
which(qlt==max(qlt))
```

```
## 907
## 907
```

```r
sum(qlt>0.9) #le nombre d'entreprise qui a une qualité >0.9
```

```
## [1] 43
```

```r
top<-which(qlt>0.9) # trouver tous les entreprises qui a une qualité >0.9
top
```

```
##  122  129  289  381  398  450  477  553  638  685  709  712  770  778  820  823
##  122  129  289  381  398  450  477  553  638  685  709  712  770  778  820  823
##  826  828  830  844  847  860  881  889  907  918  923  939  951  955  991 1021
##  826  828  830  844  847  860  881  889  907  918  923  939  951  955  991 1021
## 1034 1054 1063 1064 1124 1155 1167 1189 1217 1229 1249
## 1034 1054 1063 1064 1124 1155 1167 1189 1217 1229 1249
```

```r
sum(qlt>0.75)
```

```
## [1] 376
```

```r
top2<-which(qlt>0.75)
top2
```

```
##    6   10   11   15   25   29   31   32   33   37   41   44   45   48   49   52
##    6   10   11   15   25   29   31   32   33   37   41   44   45   48   49   52
##   53   55   57   58   63   64   68   69   70   74   78   87   93  103  104  112
##   53   55   57   58   63   64   68   69   70   74   78   87   93  103  104  112
##  120  122  128  129  131  134  141  145  148  152  153  155  161  165  167  168
##  120  122  128  129  131  134  141  145  148  152  153  155  161  165  167  168
##  169  171  173  189  192  194  201  203  218  220  225  231  235  238  242  245
##  169  171  173  189  192  194  201  203  218  220  225  231  235  238  242  245
##  251  255  257  260  261  265  269  281  283  284  287  289  304  306  310  314
##  251  255  257  260  261  265  269  281  283  284  287  289  304  306  310  314
##  317  319  327  329  330  333  336  337  338  341  343  349  355  359  371  374
##  317  319  327  329  330  333  336  337  338  341  343  349  355  359  371  374
##  380  381  383  384  397  398  400  404  409  412  416  426  435  439  443  444
##  380  381  383  384  397  398  400  404  409  412  416  426  435  439  443  444
##  447  450  454  456  460  467  470  476  477  478  485  496  507  521  522  526
##  447  450  454  456  460  467  470  476  477  478  485  496  507  521  522  526
##  534  538  544  550  553  554  555  556  560  561  570  571  572  573  577  579
##  534  538  544  550  553  554  555  556  560  561  570  571  572  573  577  579
```

```
##  588  590  607  613  618  635  636  638  639  640  641  646  648  649  650  657
##  588  590  607  613  618  635  636  638  639  640  641  646  648  649  650  657
##  659  668  675  678  679  685  687  689  691  699  701  709  711  712  715  716
##  659  668  675  678  679  685  687  689  691  699  701  709  711  712  715  716
##  719  720  725  728  734  736  737  738  741  743  746  750  751  759  764  766
##  719  720  725  728  734  736  737  738  741  743  746  750  751  759  764  766
##  767  768  770  775  776  778  782  786  788  790  796  802  804  812  816  820
##  767  768  770  775  776  778  782  786  788  790  796  802  804  812  816  820
##  821  823  826  828  829  830  831  836  837  839  840  841  844  847  848  849
##  821  823  826  828  829  830  831  836  837  839  840  841  844  847  848  849
##  857  858  860  862  863  869  870  871  878  879  880  881  886  887  889  896
##  857  858  860  862  863  869  870  871  878  879  880  881  886  887  889  896
##  897  900  902  905  907  908  909  911  913  915  918  922  923  927  931  934
##  897  900  902  905  907  908  909  911  913  915  918  922  923  927  931  934
##  936  937  939  940  941  942  944  946  950  951  955  958  959  960  963  964
##  936  937  939  940  941  942  944  946  950  951  955  958  959  960  963  964
##  965  967  968  976  977  978  982  985  986  988  989  991  994  997 1001 1003
##  965  967  968  976  977  978  982  985  986  988  989  991  994  997 1001 1003
## 1006 1009 1018 1019 1021 1024 1025 1028 1034 1035 1042 1054 1056 1058 1060 1063
## 1006 1009 1018 1019 1021 1024 1025 1028 1034 1035 1042 1054 1056 1058 1060 1063
## 1064 1068 1070 1071 1073 1075 1077 1078 1079 1081 1083 1085 1087 1088 1090 1093
## 1064 1068 1070 1071 1073 1075 1077 1078 1079 1081 1083 1085 1087 1088 1090 1093
## 1097 1100 1101 1108 1109 1113 1118 1121 1123 1124 1132 1134 1139 1141 1143 1152
## 1097 1100 1101 1108 1109 1113 1118 1121 1123 1124 1132 1134 1139 1141 1143 1152
## 1153 1155 1157 1159 1160 1167 1169 1172 1174 1176 1181 1182 1186 1187 1189 1192
## 1153 1155 1157 1159 1160 1167 1169 1172 1174 1176 1181 1182 1186 1187 1189 1192
## 1193 1197 1199 1207 1208 1210 1211 1212 1214 1217 1219 1223 1229 1232 1233 1239
## 1193 1197 1199 1207 1208 1210 1211 1212 1214 1217 1219 1223 1229 1232 1233 1239
## 1240 1246 1249 1251 1254 1256 1258 1259
## 1240 1246 1249 1251 1254 1256 1258 1259
```

```
sum(qlt<0.05)
```

```
## [1] 30
```

```
top3<-which(qlt<0.05)
top3
```

```
##   83   89   94  136  193  199  294  309  408  423  465  488  513  518  546  604
##   83   89   94  136  193  199  294  309  408  423  465  488  513  518  546  604
##  608  615  628  631  673  742  763  853  865  930 1037 1128 1135 1227
##  608  615  628  631  673  742  763  853  865  930 1037 1128 1135 1227
```

```
sum(qlt<0.01)
```

```
## [1] 6
```

```
top4<-which(qlt<0.01)
top4
```

```
##  513  518  628  673  853 1128
##  513  518  628  673  853 1128
```

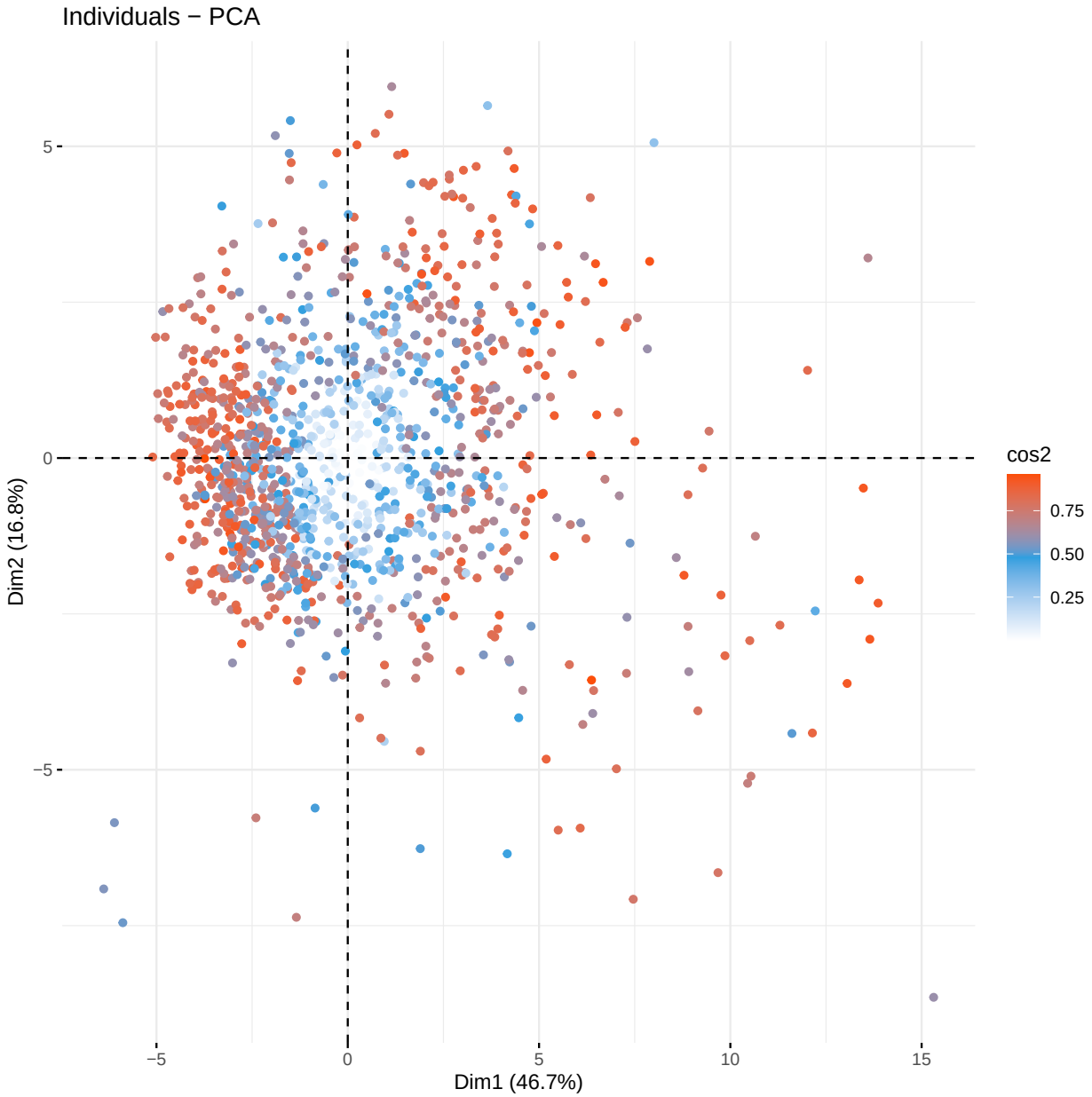
On remarque dans le premier plan factoriel, il y a 43 entreprises qui ont une qualité >0.9, c'est à dire qu'ils sont très bien représentés. Ensuite, il y a 376 entreprises qui ont une qualité >0.75, ils ne sont pas mal représentés. Après ça, il y a 30 entreprise qui ont une qualité <0.05 , ils sont mal représentés et enfin qu'il y a 6 entreprises qui ont une qualité <0.01, ils sont très mal représentés.

6)

```
library(ggplot2)
library(factoextra)
```

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

```
fviz_pca_ind(res.pca, col.ind="cos2", geom = "point",gradient.cols = c("white", "#2E9FDF", "#FC4E07" ))
```



On remarque que: • Les individus en rouge sont les individus qui sont bien représentés, et on observe bien

qu'ils sont plutôt éloignés de l'origine. • Les individus en bleu sont les individus qui sont mal représentés, et on observe bien qu'ils sont plutôt proches de l'origine.

7)

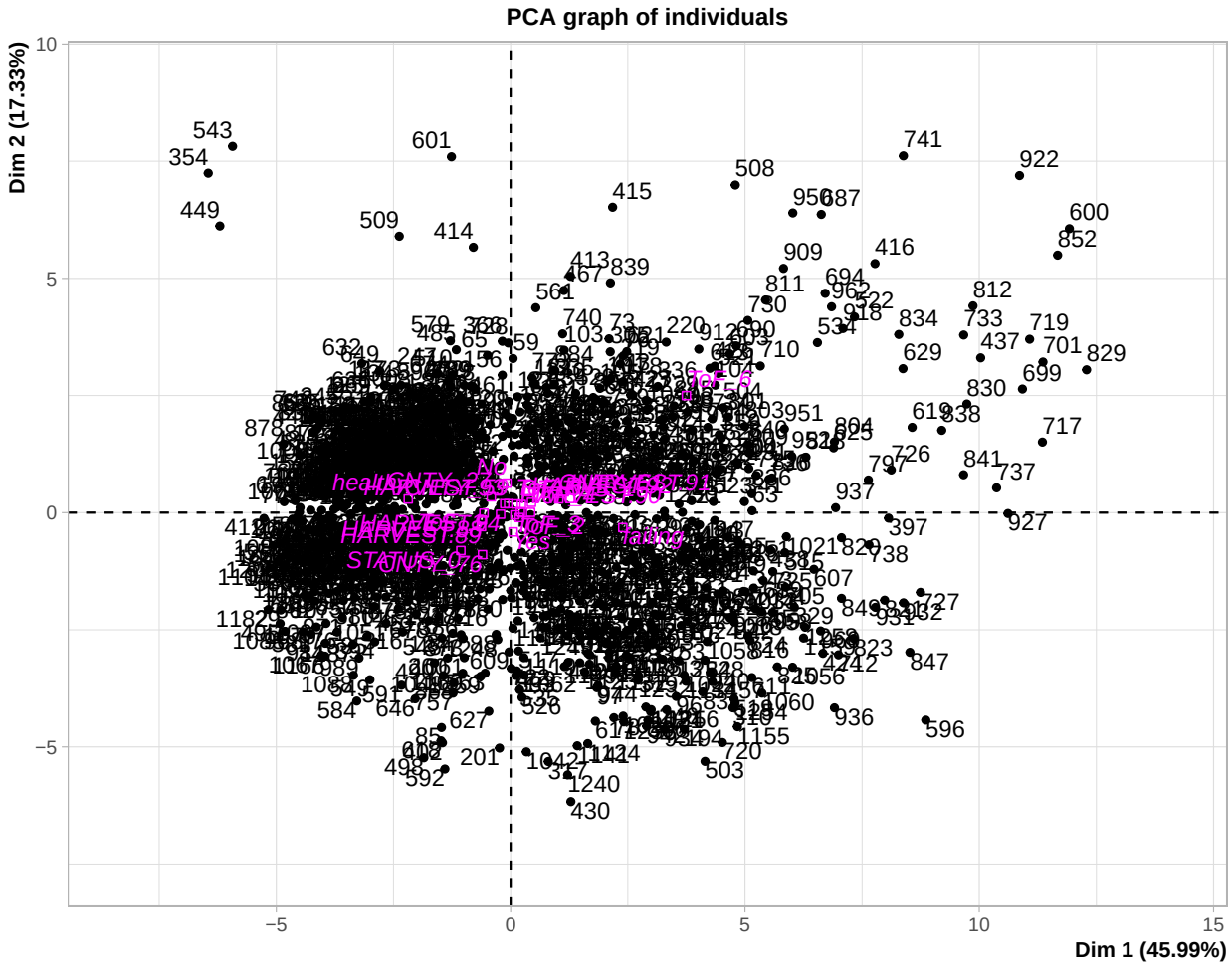
```
cind<-res.pca$ind$contrib[,1]
l<-500/1260
trop<-which(cind>l)
trop<-which(cind>1)
trop
```

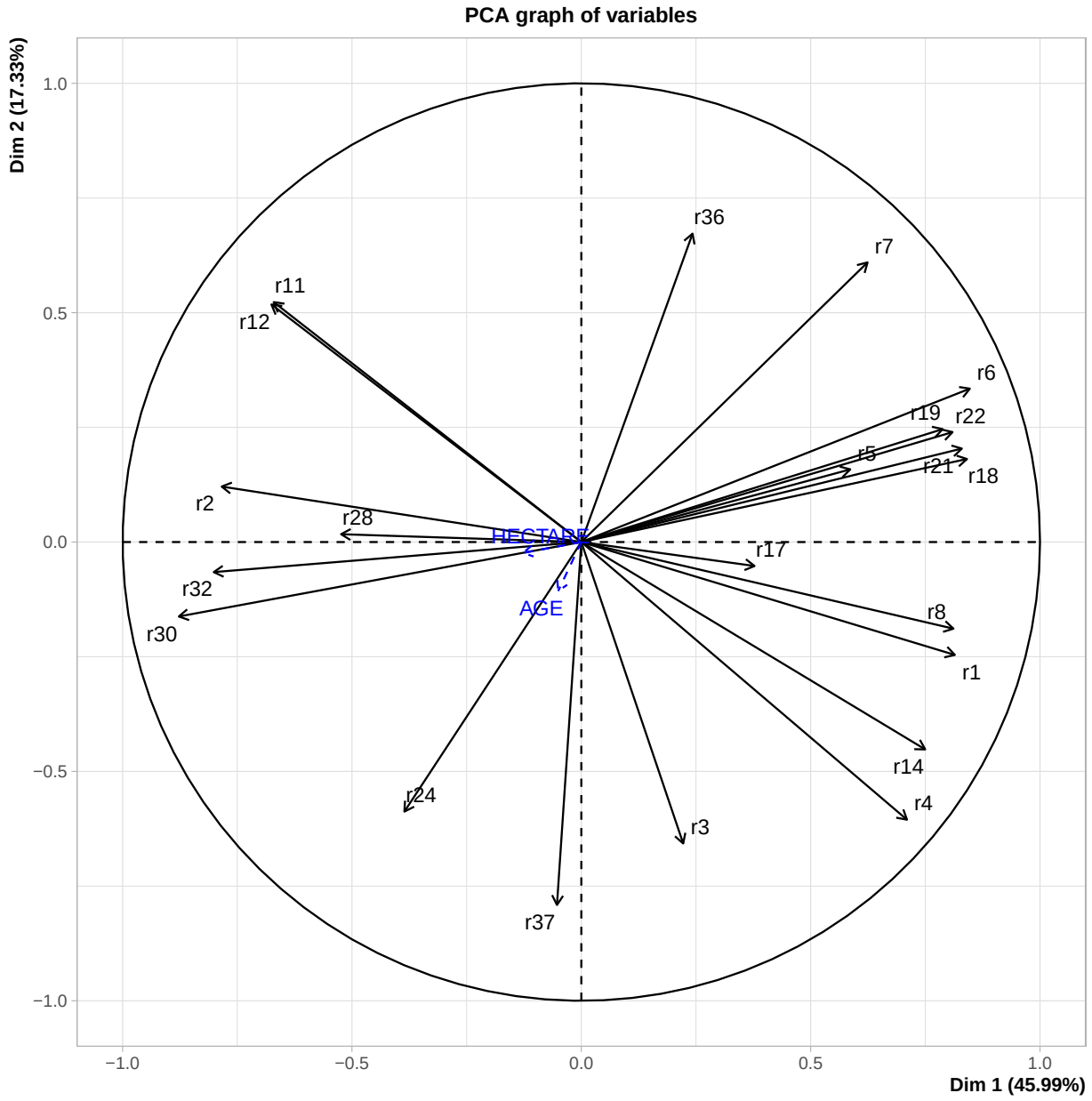
```
## 407 511 532 533 828 844 923 939 941 955 963
## 407 511 532 533 828 844 923 939 941 955 963
```

```
cind[trop]
```

```
##      407      511      532      533      828      844      923      939
## 1.429552 1.813210 1.042110 1.154417 1.485990 1.439893 1.317542 1.404372
##      941      955      963
## 1.117019 1.381594 1.140371
```

```
#les individus qui ont une trop forte contribution à la formation du premier axe
#principal.
res.pca = PCA(donnees[-trop,], scale.unit = TRUE, quali.sup = qual,
              quanti.sup = quant, graph = T, ncp = 12)
```


PCA graph of individuals
Dim 2 (17.33%)
Dim 1 (45.99%)
10
5
0
−5
−5
0
5
10
15




```
cind<-res.pca$ind$contrib[,1]
```

On obtient donc les entreprises qui ont une trop forte contribution à la formation du premier axe principal sont “407 511 532 533 828 829 844 923 939 941 955 963”, et on observe bien qu’ils se situent très loin que l’origine sur le graph. De plus, on remarque qu’il y a 46.62% informations sur le premier axe, 16.77% informations sur le second axe. Ensuite, on voit que r37 et r36 sont bien représentés car ils ont les flèches plus long que les autres. Ainsi, r37 est corrélé positivement à l’axe2 et r36 est corrélé négativement à l’axe2. Cependant, r36 et r37 sont très peu corrélés à l’axe1. De plus, on a r5 et r28 sont mal représentrés(les flèches courtes). Car on perd des informations quand on fait la projection.

8)

```r
phi<-res.pca$var$coord
phi #la matrice phi
```

```
##           Dim.1       Dim.2       Dim.3       Dim.4       Dim.5        Dim.6
## r1   0.81433924 -0.24599616  0.42892899 -0.14015318 -0.16657489  0.030649923
## r2  -0.78396342  0.12095783 -0.47804437  0.13285571  0.14031869  0.033407847
## r3   0.22212079 -0.65712702 -0.52996639  0.15999781 -0.04508950  0.360085580
## r4   0.70997545 -0.60518788 -0.06010836  0.06851546 -0.10017739  0.218093425
## r5   0.58632940  0.15839175  0.67513877 -0.27478591 -0.15397754 -0.154508097
## r6   0.84708547  0.33439775  0.15426828  0.15150432 -0.24181816  0.178707222
## r7   0.62402785  0.60958220  0.38824560  0.01057216 -0.20774458  0.032857970
## r8   0.81137364 -0.18885001 -0.22611707  0.26919676 -0.19022573  0.306208801
## r11 -0.67069002  0.52300379  0.07299323 -0.29636465  0.08261994  0.360674916
## r12 -0.67557053  0.51783921  0.12514616 -0.22824174  0.09047798  0.397940721
## r14  0.74971553 -0.45199607 -0.06576447  0.26040580 -0.13549718 -0.153437040
## r17  0.37768986 -0.05221726  0.11889112  0.26476402  0.81434482 -0.155316256
## r18  0.84085867  0.18101366  0.15634759  0.24959828  0.29737739  0.109093894
## r19  0.78768230  0.24612425  0.19843026  0.17412406  0.24593921  0.157097433
## r21  0.82965101  0.20375336 -0.14288981 -0.15095625  0.19576834  0.105786179
## r22  0.80935354  0.23962568 -0.11588631 -0.25437681  0.15774034  0.111232795
## r24 -0.38571587 -0.58761520  0.60448366  0.09600958  0.12652751  0.109415868
## r28 -0.52373394  0.01710903  0.55680225  0.59703931  0.04790143  0.176420901
## r30 -0.87740963 -0.16226425  0.19657408  0.24240148 -0.14843554 -0.007561319
## r32 -0.80135351 -0.06550810  0.39213781  0.37468371 -0.09274669  0.102717195
## r36  0.24254051  0.67229126 -0.24323204  0.48972272 -0.17227047 -0.266780441
## r37 -0.05309405 -0.79109179  0.30231157 -0.40772840  0.11868178 -0.053045036
##            Dim.7         Dim.8        Dim.9       Dim.10        Dim.11
## r1  -0.118510020 -0.0225258296 -0.053830921  0.051626175  0.0559028088
## r2   0.171262475  0.0070873530  0.093007547  0.102323475  0.0749632857
## r3   0.010523514 -0.0579937316  0.113614178  0.057996025  0.2122475615
## r4  -0.137741694 -0.0249608243 -0.091669738  0.024410813 -0.0910014406
## r5  -0.043847478 -0.0075897898 -0.014905293  0.085526839  0.1564276287
## r6  -0.025224191  0.0344417974  0.131753963 -0.033241550 -0.0451531471
## r7   0.005989645  0.0111177371  0.149086003  0.002034435  0.0269362107
## r8  -0.064425870  0.0562162927  0.016571644 -0.056435009 -0.1258678415
## r11 -0.130875666 -0.0539870391  0.005000456  0.126442152 -0.0395325164
## r12 -0.112104639 -0.0074119850 -0.022734455  0.128783578 -0.0560062909
## r14 -0.020580013 -0.0109727702 -0.124718430  0.249990107 -0.0313115228
## r17 -0.250317816  0.0107468976 -0.003064513  0.040072393 -0.0204441851
## r18 -0.118636412  0.0516032230  0.141623807 -0.029252442  0.0654100350
## r19  0.238248889 -0.3148551189 -0.044504462 -0.030832799  0.0009822131
## r21  0.119539066  0.3925870971 -0.031984884  0.009649505  0.0186434999
## r22  0.343587339  0.1093208837 -0.169667993  0.029315033  0.0093878149
## r24  0.235029359  0.0711174976  0.121931788  0.023689423 -0.0669862003
## r28  0.080288538  0.0395991889 -0.079553438 -0.016684343  0.0436087014
## r30 -0.130145733  0.2428602505  0.003604524  0.015299690  0.0350718661
## r32  0.121531983  0.0104757202 -0.124959946  0.001375250  0.0092207817
## r36  0.111000880 -0.0003106087  0.120451698  0.146972400 -0.0796932821
## r37  0.148559892 -0.0062773339  0.208839622  0.072040402 -0.0951960667
##           Dim.12
## r1   0.106256522
## r2   0.005743355
## r3   0.056665172
```

```
## r4   0.089970396
## r5   0.072498981
## r6  -0.011383133
## r7  -0.050945104
## r8   0.041434422
## r11  0.002363036
## r12 -0.012639118
## r14 -0.196729817
## r17  0.092082536
## r18 -0.114515854
## r19 -0.025723042
## r21 -0.015703448
## r22  0.060745124
## r24 -0.032448928
## r28  0.003059090
## r30  0.013320525
## r32  0.052153483
## r36  0.155753732
## r37  0.041741445
```

```
phi2<-phi*phi
s<-apply(phi2[,], MARGIN=1,FUN=sum)
s1<-phi2[,1]
qltv1<-s1/s #la qualité de la représentation sur le premier axe
s2<-apply(phi2[,1:2], MARGIN=1,FUN=sum)
qltv2<-s2/s #la qualité de la représentation sur le premier plan
sncp<-apply(phi2[,1:ncp], MARGIN=1,FUN=sum)
qltv<-sncp/s #la qualité de la représentation sur le sous-espace vectoriel choisi
qltv1
```

```
##          r1          r2          r3          r4          r5          r6
## 0.669506873 0.646696817 0.049971402 0.513725167 0.348305074 0.725538204
##          r7          r8         r11         r12         r14         r17
## 0.396630905 0.674607478 0.453885286 0.460135142 0.562811172 0.144064511
##         r18         r19         r21         r22         r24         r28
## 0.720054136 0.623329276 0.693395053 0.660806410 0.152133895 0.276781061
##         r30         r32         r36         r37
## 0.775086441 0.646646683 0.059439217 0.002862207
```

```
qltv2
```

```
##        r1        r2        r3        r4        r5        r6        r7        r8
## 0.7306012 0.6620917 0.4873341 0.8869965 0.3737231 0.8386043 0.7751111 0.7111538
##       r11       r12       r14       r17       r18       r19       r21       r22
## 0.7298872 0.7304898 0.7673798 0.1468182 0.7534230 0.6841882 0.7352165 0.7187312
##       r24       r28       r30       r32       r36       r37
## 0.5052170 0.2770764 0.8015953 0.6509679 0.5161271 0.6382852
```

```
qltv
```

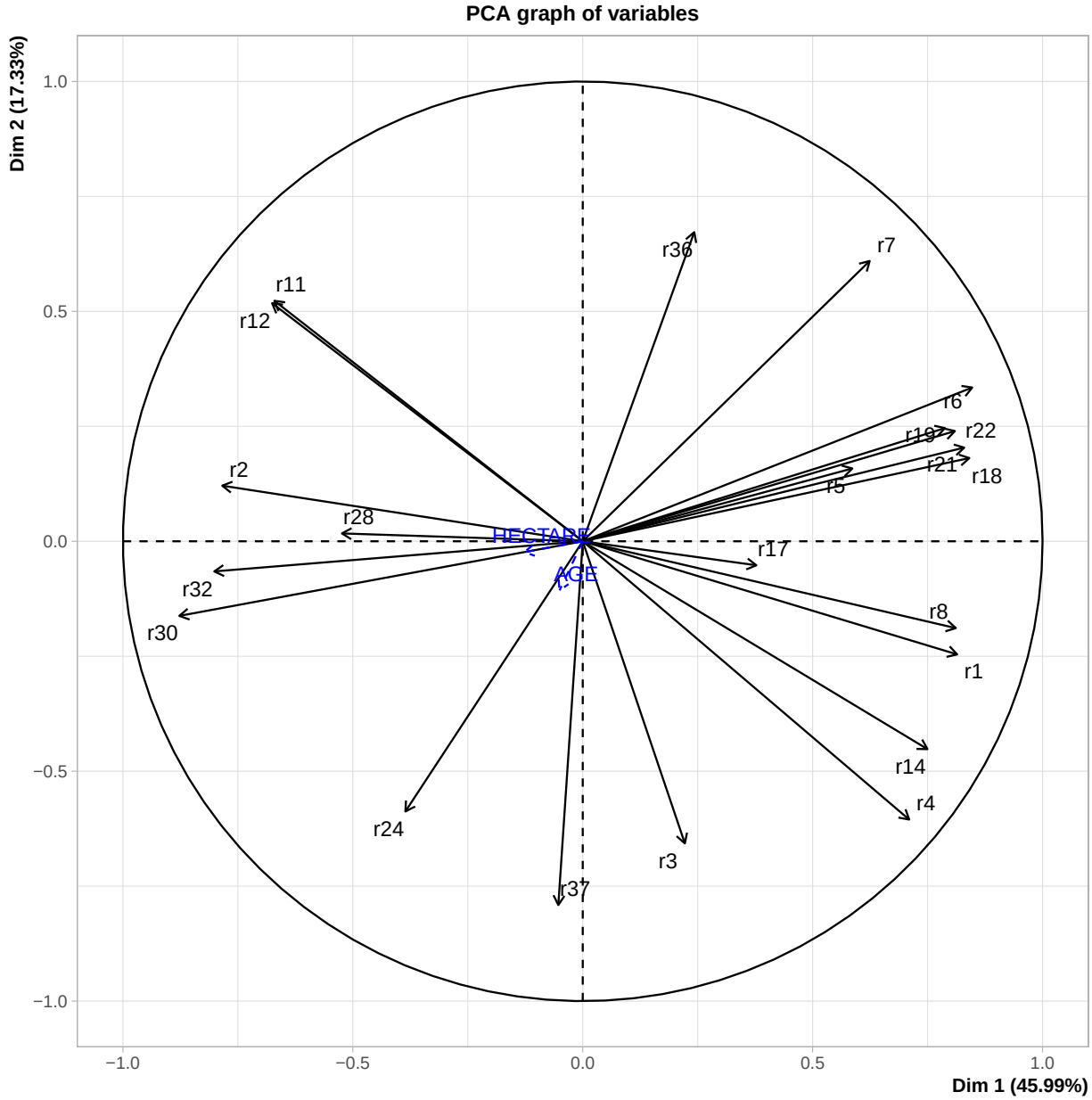
```
##        r1        r2        r3        r4        r5        r6        r7        r8
## 0.7306012 0.6620917 0.4873341 0.8869965 0.3737231 0.8386043 0.7751111 0.7111538
```

```
##       r11       r12       r14       r17       r18       r19       r21       r22
## 0.7298872 0.7304898 0.7673798 0.1468182 0.7534230 0.6841882 0.7352165 0.7187312
##       r24       r28       r30       r32       r36       r37
## 0.5052170 0.2770764 0.8015953 0.6509679 0.5161271 0.6382852
```

9)

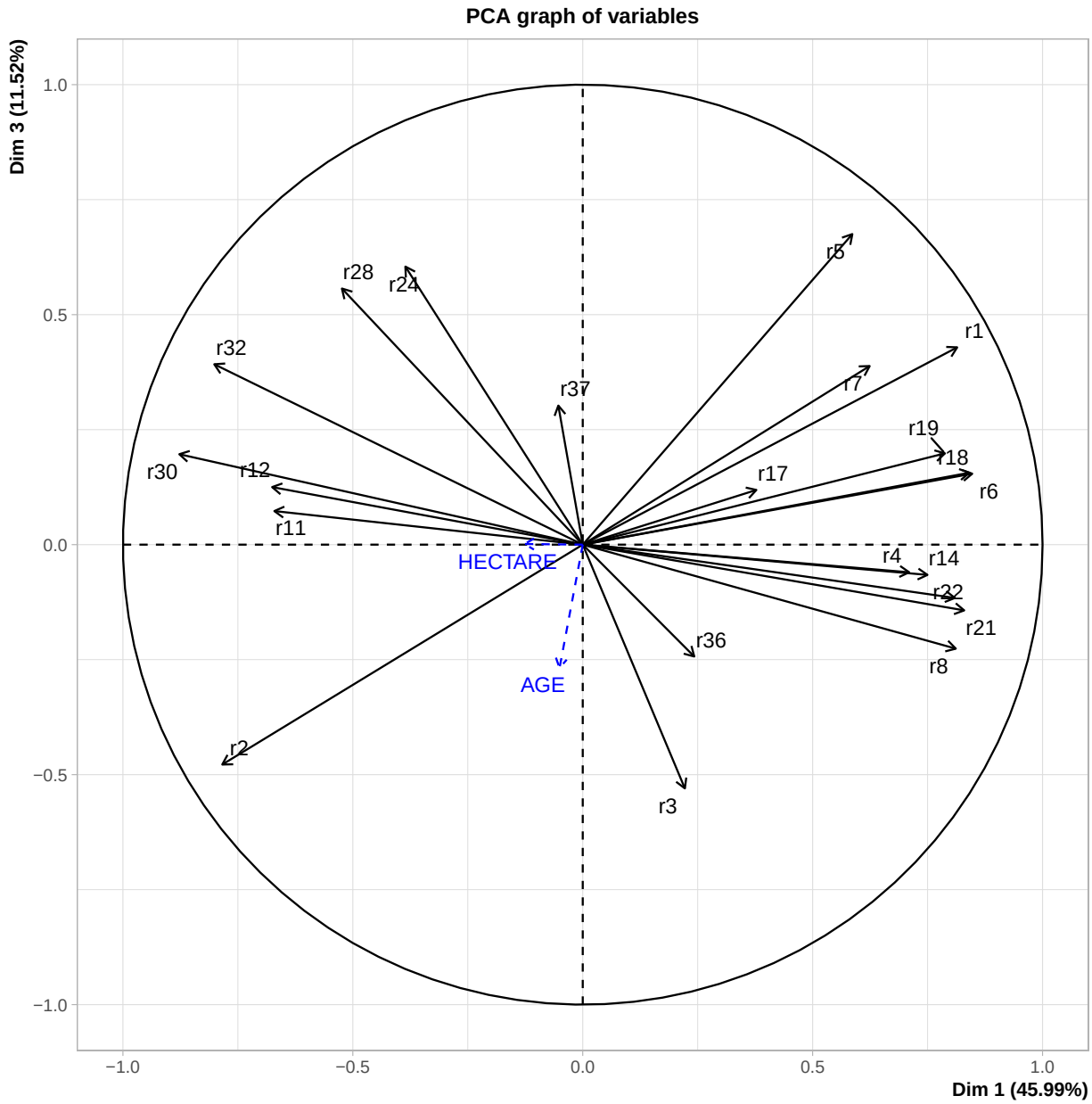
```
plot.PCA(res.pca, axes=c(1,2), choix="var", habillage=1)
```

```
## Warning in plot.PCA(res.pca, axes = c(1, 2), choix = "var", habillage = 1):
## Habillage must be in c('contrib','cos2','none')
```



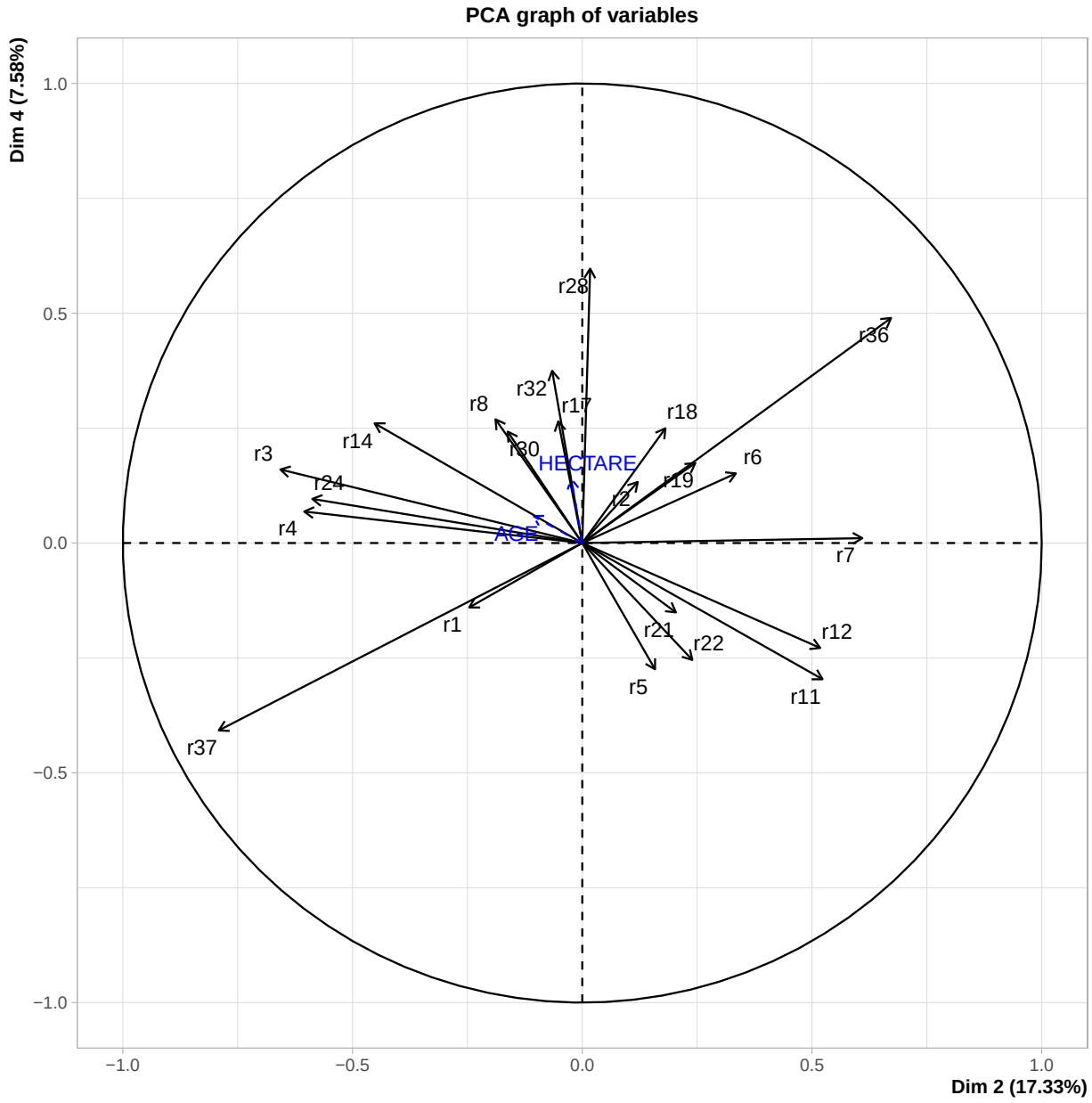
```
plot.PCA(res.pca, axes=c(1,3), choix="var", habillage=1)
```

```
## Warning in plot.PCA(res.pca, axes = c(1, 3), choix = "var", habillage = 1):
## Habillage must be in c('contrib','cos2','none')
```




```
plot.PCA(res.pca, axes=c(2,4), choix="var", habillage=1)
```

```
## Warning in plot.PCA(res.pca, axes = c(2, 4), choix = "var", habillage = 1):
## Habillage must be in c('contrib','cos2','none')
```



On remarque que :

• Il y a 45.99% informations sur le premier axe, 17.33% informations sur le second axe. Ensuite, on voit que r37 et r36 sont bien représentés car ils ont les flèches longs(leurs qualité sont proches de 1). Ainsi, r37 est corrélé négativement à l'axe2 et r36 est corrélé positivement à l'axe2. Cependant, r36 et r37 sont très peu corrélés à l'axe1. De plus, on a r17 et r28 sont mal représentrés(les flèches courtes). Car on perd des informations quand on fait la projection. r30,r32 sont bien présentés et corrélés négativement à l'axe1. Et r8,r21 sont bien présentés et corrélés positivement à l'axe1.

• Il y a 45.99% informations sur le premier axe, 11.52% informations sur le troisième axe. Ensuite, on voit que r37 et r36 sont très mauvaises représentés car ils ont les flèches très courtes(leur qualités sont très petites). Ainsi, r30,r32 sont bien représentés et corrélés négativement à l'axe1 et r6,r18,r19,r14,r21,r8,r22 sont bien représentés et corrélés positivement à l'axe1.

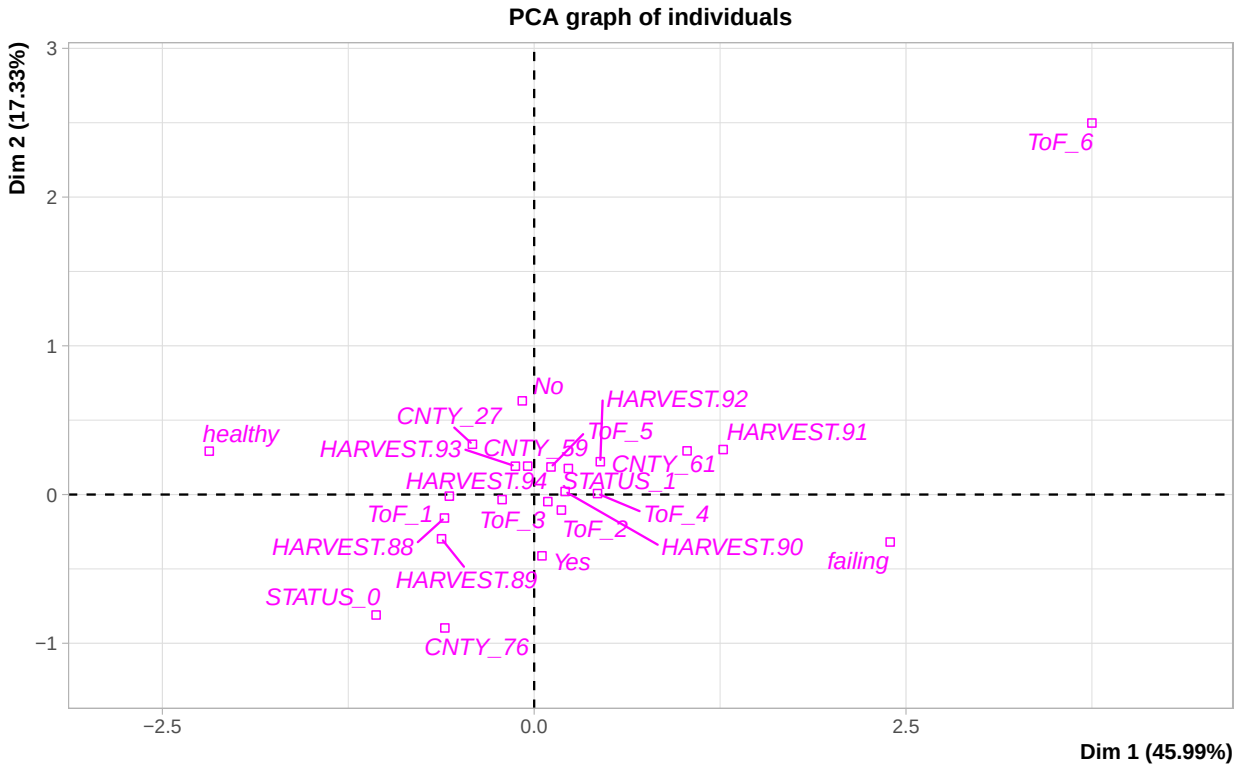
• Il y a 17.33% informations sur le second axe, 7.58% informations sur le quartième axe. Ensuite, on voit que r37 et r36 sont bien représentés car ils ont les flèches long(leurs qualités sont proches de 1). Ainsi, r28,r36

sont bien représentés et corrélés positivement à l'axe4 ,r37 est corrélé negativement à l'axe2. Et presque tous les autres variables ne sont pas bien représentés car leurs flèches sont courtes(leurs qualités sont très petites).

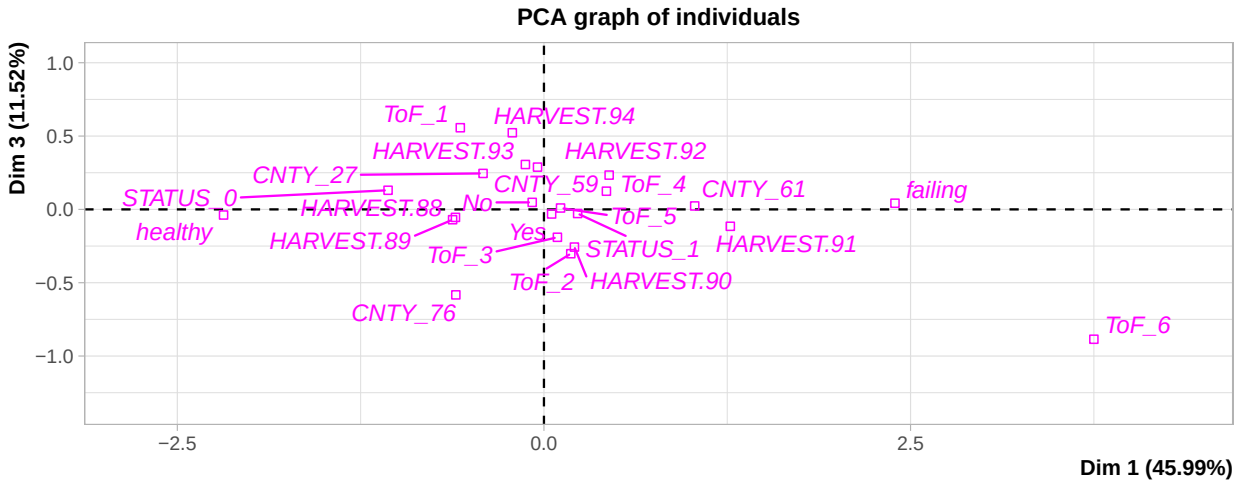
10) On peut donc conclure que l'importance de tous les 22 variables par rapport à chaque composante(dimension ou axe). Par exemple, pour l'activité productive, on a "gross"product / total assets" (r37) est très important et corrélé négativement pour la composante 2 et la composante 4. Ensuite "immobilized assets / gross product" (r36) est aussi important mais corrélé positivement pour la composante 2 et la composante 4. Cependant, r36 et r37 sont très peu corrélés à la composante 1 et à la composante 3. Un autre exemple, on a "financial expenses / total debt" (r17) et "EBITDA / gross product" (r28) ne sont pas importants pour la composante 1 ou la composante 2...

11)

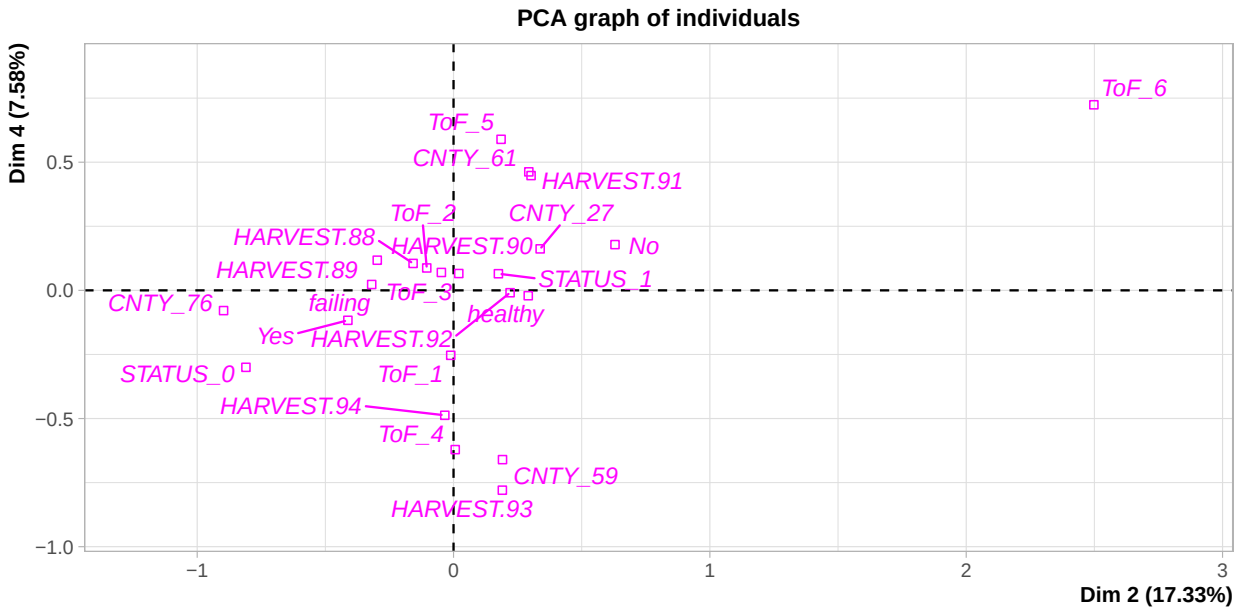
```
options(ggrepel.max.overlaps = Inf)
plot.PCA(res.pca, axes = c(1,2), invisible = c("ind"))
```




```
plot.PCA(res.pca, axes = c(1,3), invisible = c("ind"))
```




```
plot.PCA(res.pca, axes = c(2,4), invisible = c("ind"))
```



On remarque que:

• Il y a 45.99% informations sur le premier axe, 17.33% informations sur le second axe. Ensuite, “ToF_6” est bien représenté car il est très loin que l’origine. Les autres individus qui sont proches de l’origine sont plutôts mal représentés. De plus, les individus qui se situent à la droite du plan sont les individus qui ont le plus grand valeurs des variables(qui se situent à la droite du plan) en moyenne pour l’axe 1. Les individus qui se situent à la gauche du plan sont les individus qui ont les plus grands valeurs des variables(qui se situent à la gauche du plan) en moyenne pour l’axe 1. Ainsi, les individus qui se situent au-dessus du plan sont les individus qui ont les plus grand valeurs des variables(qui se situent au-dessus du plan) en moyenne pour l’axe 2. Les individus qui se situent en dessous du plan sont les individus qui ont les plus grands valeurs des variables(qui se situent en dessous du plan) en moyenne pour l’axe 2.

• Il y a 45.99% informations sur le premier axe, 11.52% informations sur le troisième axe. Ensuite, “ToF_6” est bien représenté car il est très loin que l’origine. Les autres individus qui sont proches de l’origine sont plutôts mal représentés. De plus, les individus qui se situent à la droite du plan sont les individus qui ont le

plus grand valeurs des variables(qui se situent à la droite du plan) en moyenne pour l'axe 1. Les individus qui se situent à la gauche du plan sont les individus qui ont le plus grand valeurs des variables(qui se situent à la gauche du plan) en moyenne pour l'axe 1. Ainsi, les individus qui se situent au-dessus du plan sont les individus qui ont les plus grand valeurs des variables(qui se situent au-dessus du plan en moyenne) pour l'axe 3. Les individus qui se situent en dessous du plan sont les individus qui ont les plus grand valeurs des variables(qui se situent en dessous du plan) en moyenne pour l'axe 3.

• Il y a 17.33% informations sur le deuxième axe, 7.58% informations sur le troisième axe. Ensuite, "ToF_6" est bien représenté car il est très loin que l'origine. Les autres individus qui sont proches de l'origine sont plutôts mal représentés. De plus, les individus qui se situent à la droite du plan sont les individus qui ont le plus grand valeurs des variables(qui se situent à la droite du plan en moyenne) pour l'axe 2. Les individus qui se situent à la gauche du plan sont les individus qui ont les plus grands valeurs des variables(qui se situent à la gauche du plan) en moyenne pour l'axe 2. Ainsi, les individus qui se situent au-dessus du plan sont les individus qui ont les plus grand valeurs des variables(qui se situent au-dessus du plan en moyenne) pour l'axe 4. Les individus qui se situent en dessous du plan sont les individus qui ont les plus grands valeurs des variables(qui se situent en dessous du plan) en moyenne pour l'axe 4.