Rapport de projet Analyse de données & Eléments de modélisation statistique

Xiaoya Wang, Mickael Song, Yessine Jmal, Florian Grivet

2022-11-22

Contents

1 Analyse du jeu de données

1.1 Statistiques descriptives et préparation du jeu de données

```
# Chargement des données
data = read.delim("Data_Etudiants_2023.txt",header=TRUE, sep=";")
```

```
datapy = pd.read_csv("Data_Etudiants_2023.txt", sep=";")
```

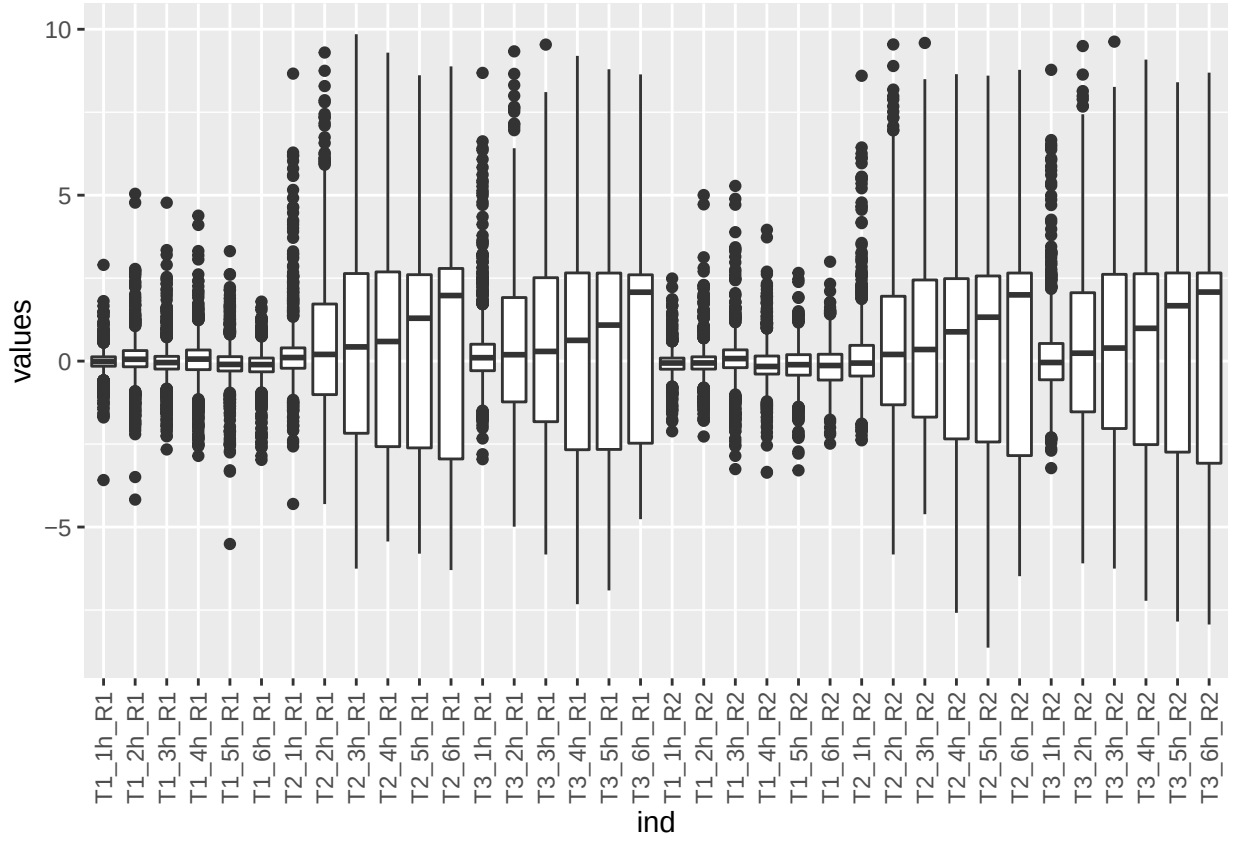
Table 1: Les premières lignes du jeu de données.

	T1_1h_R1	T1_2h_R1	T1_3h_R1	T1_4h_R1	T1_5h_R1	T1_6h_R1	T2_1h_R1	T2_2h_R1
G1	0.17	0.68	-0.18	0.08	0.00	0.50	-0.59	0.19
G2	0.19	0.82	-0.05	0.18	0.47	-0.76	0.38	2.51
G3	-0.05	-0.03	0.26	-0.32	-0.39	0.42	0.21	-1.00
G4	-0.23	-0.75	-0.24	-0.70	-0.12	-0.38	0.41	0.23
G5	-0.21	-0.69	-0.18	-0.07	0.52	0.45	-0.45	-1.51
G6	-0.62	-0.86	-0.02	-0.14	0.49	0.45	-0.57	-1.48

Le jeu de données contient 1615 individus et 36 variables, toutes quantitatives.
Les attributs du jeu de données sont : T1_1h_R1, T1_2h_R1, T1_3h_R1, T1_4h_R1, T1_5h_R1, T1_6h_R1, T2_1h_R1, T2_2h_R1, T2_3h_R1, T2_4h_R1, T2_5h_R1, T2_6h_R1, T3_1h_R1, T3_2h_R1, T3_3h_R1, T3_4h_R1, T3_5h_R1, T3_6h_R1, T1_1h_R2, T1_2h_R2, T1_3h_R2, T1_4h_R2, T1_5h_R2, T1_6h_R2, T2_1h_R2, T2_2h_R2, T2_3h_R2, T2_4h_R2, T2_5h_R2, T2_6h_R2, T3_1h_R2, T3_2h_R2, T3_3h_R2, T3_4h_R2, T3_5h_R2, T3_6h_R2

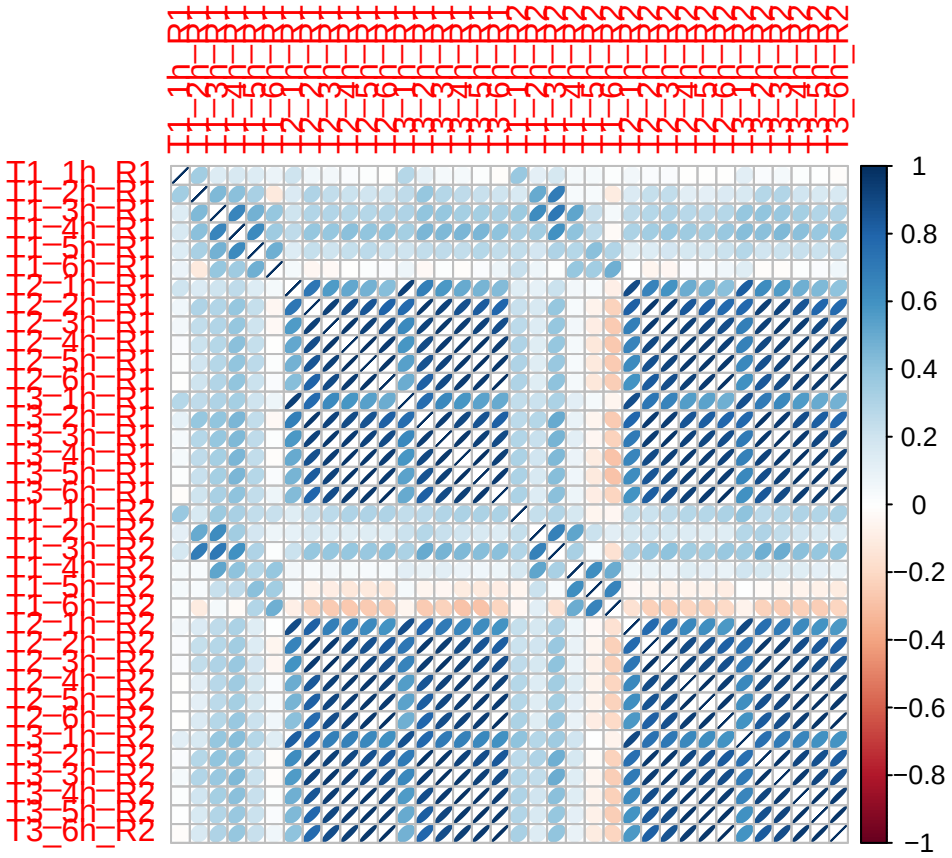
Avec le résultat de la commande `datapy.isnull().sum()`, on voit bien que notre jeu de données est complet.

```
ggplot(stack(data), aes(x = ind, y = values))+
  geom_boxplot()+
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
```



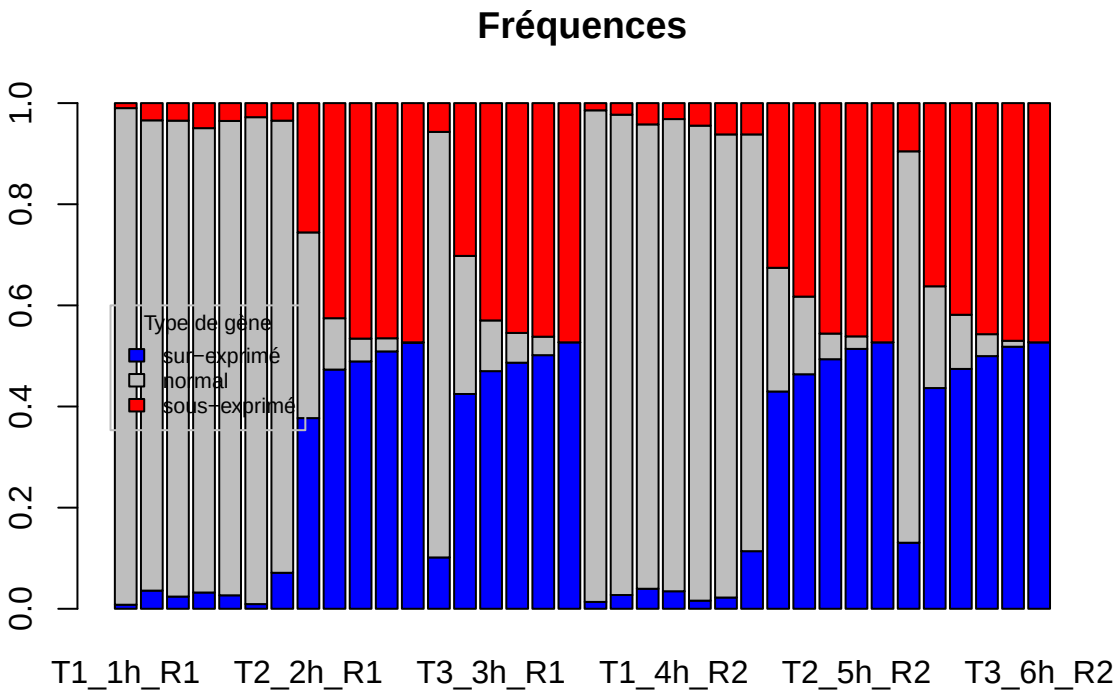
On remarque que les traitements 2 et 3 ont des boxplots similaires que ça soit pour le réplicat 1 ou pour le réplicat 2. On peut faire l'hypothèse que ces deux traitements ont des résultats similaires. On remarque que ces deux traitements ne sont pas centrés donc qu'ils ont un effet non nul sur les gènes. On note également une dissymétrie pour ces traitements ainsi que de nombreux outliers et une forte variabilité entre les individus. Le traitement 1 est quant à lui beaucoup plus réduit et centré en 0. Le traitement semble donc ne pas avoir d'effet sur les gènes. On remarque que le traitement 1 est symétrique mais possède également beaucoup d'outliers.

```
corrplot(cor(data), method="ellipse")
```



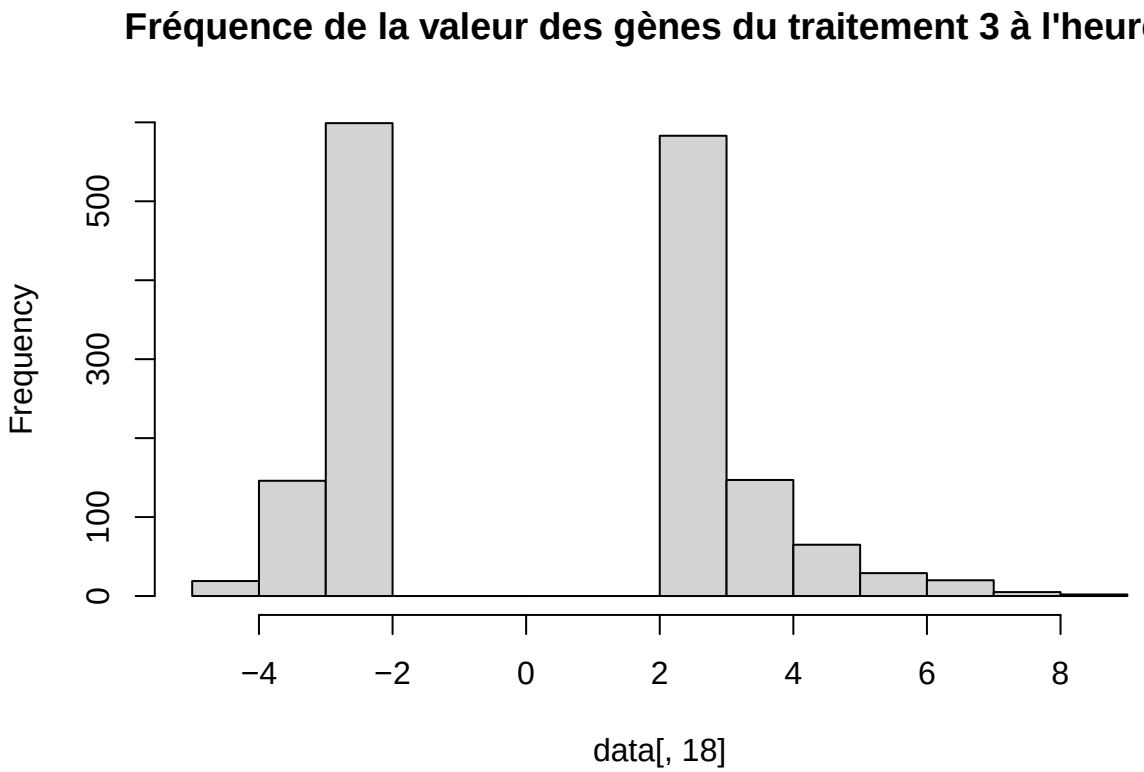
1.1.1 ANALYSE CORRPLOT A FAIRE

```
BP = array(rep(rep(0,3),36), dim=c(3,36))
BP[1,] = apply(data>1, 2, sum)
BP[2,] = apply(data<=1&data>=(-1), 2, sum)
BP[3,] = apply(data<(-1), 2, sum)
BP = BP/nrow(data)
barplot(BP, main="Fréquences", col=c("blue", "grey", "red"), names.arg=c(attributes(data)$names))
legend(0,0.6, legend=c("sur-exprimé", "normal", "sous-exprimé"), title="Type de gène", box.col="grey", i
```



Ce graphique représente la fréquence des gènes “sous-exprimés”, “normaux” et “sur-exprimés” pour chaque traitement à toute heure sur les deux réplicats. Ce graphique appuie notre hypothèse que les traitements 2 et 3 sont similaires et que le traitement 1 n’a pas beaucoup d’effet.

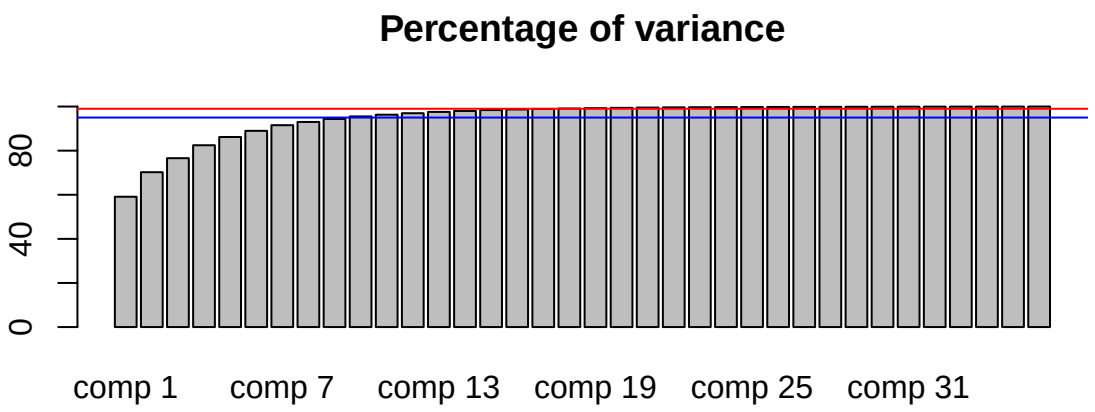
```
hist(data[,18], main="Fréquence de la valeur des gènes du traitement 3 à l'heure 6")
```



On remarque qu’à l’heure 6 (la dernière) du traitement 3, tous les gènes sont soit très sur-exprimé (valeurs ≥ 2), soit très sous-exprimé (valeurs ≤ 2). C’est comme ça que les gènes du jeu de données ont été choisi.

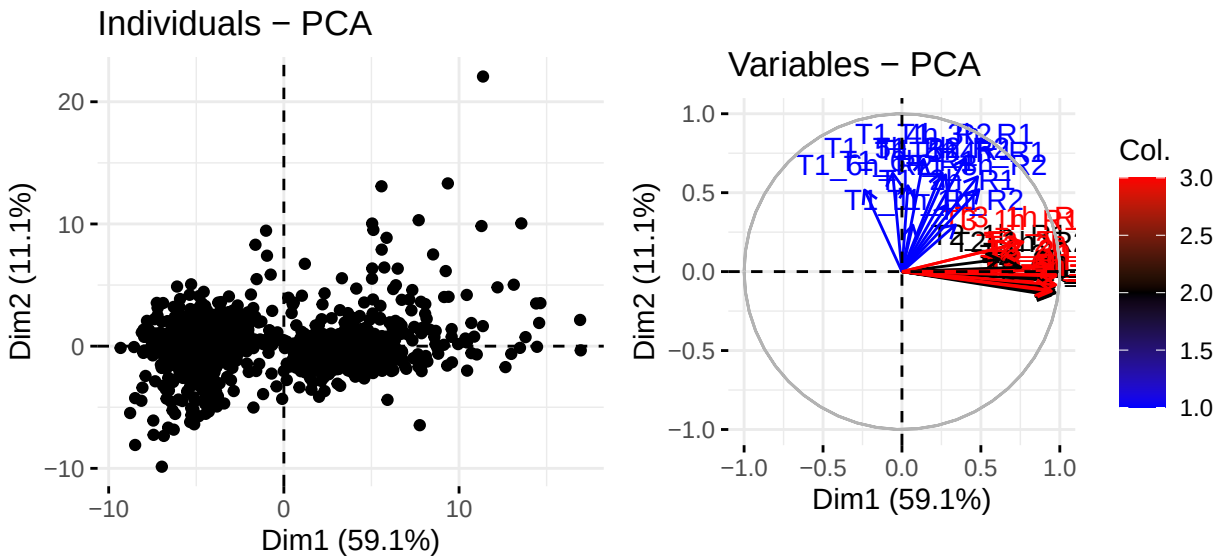
1.2 Analyse en composante principale

```
barplot(cumsum(res.pca$eig[, "percentage of variance"]), main="Percentage of variance")
abline(h=95, col="blue")
abline(h=99, col="red")
```



Pour avoir 95% de l'information, on peut réduire nos données à 10 dimensions.
Pour avoir 99% de l'information il suffit de se placer en dimension 28.

```
col_trait = rep(rep(c(1,2,3), each=6),2)
g1 = fviz_pca_ind(res.pca, label="none")
g2 = fviz_pca_var(res.pca, col.var=col_trait) + scale_color_gradient2(low="blue", mid="black", high="red
grid.arrange(g1,g2,ncol=2)
```



Les traitements 2 et 3 semblent avoir le même comportement et son selon l'axe 1.
Le traitement 1 suit l'axe 2.

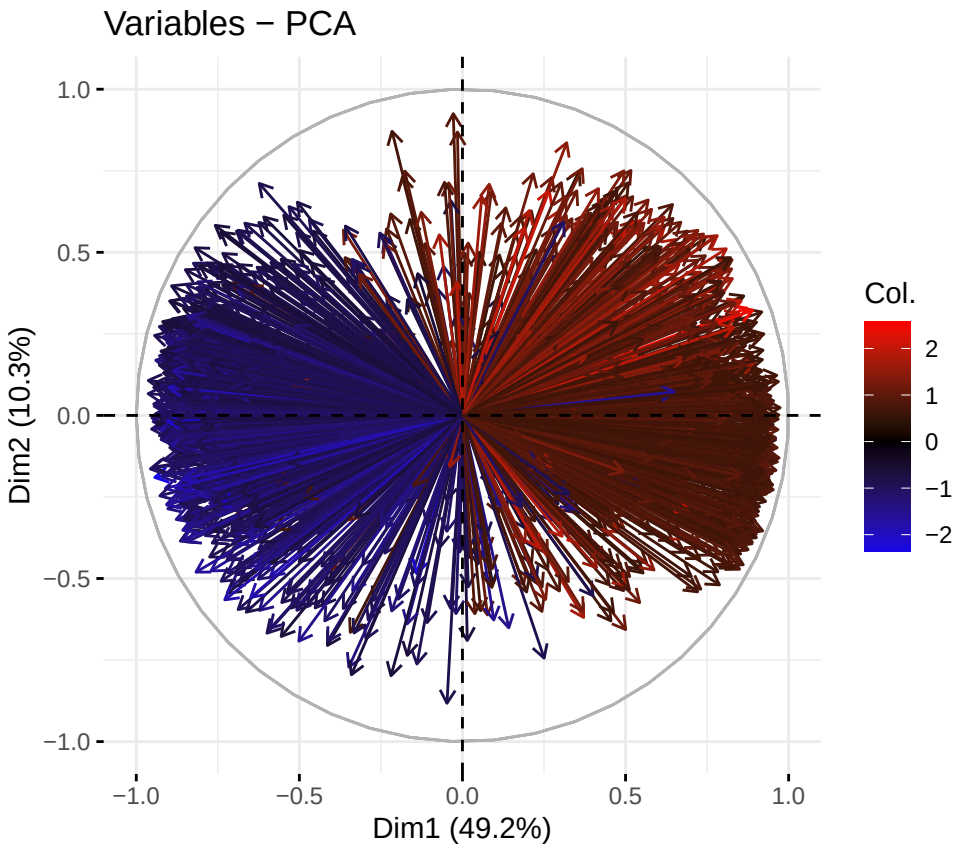
Pourcentage des variables dans la construction des dimensions 1 et 2 :

L'axe 1 nous dit si un gène réagit au traitement 2 ou 3. Si le gène réagit fortement à l'un de ces traitements, il se retrouve sur un côté du graphique (si le gène devient très sous-exprimé ou très sur-exprimé) et s'il ne réagit pas beaucoup à l'un de ces traitement il se trouve au milieu.

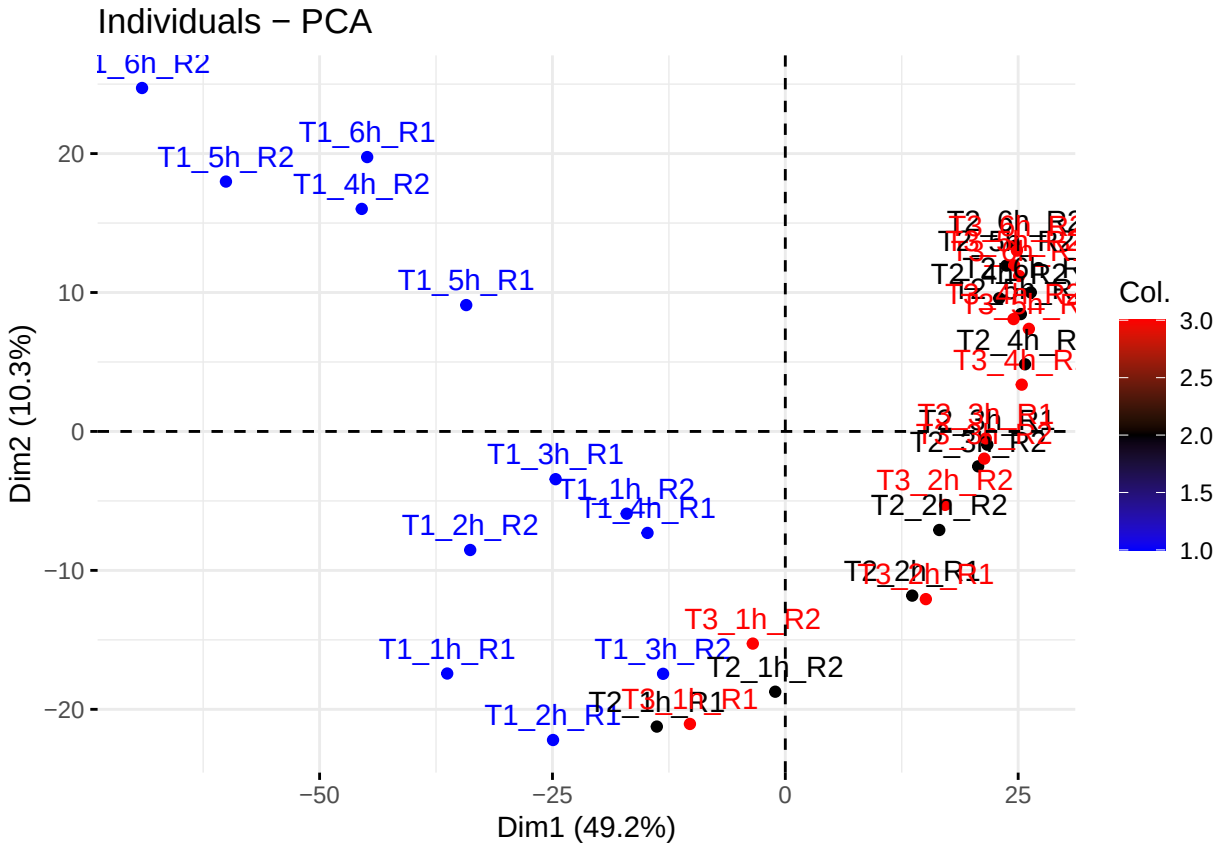
L'axe 2 nous dit si un gène réagit au traitement 1. Si le gène réagit fortement à ce traitement, il se retrouve en haut s'il devient sur-exprimé, en bas s'il devient sous-exprimé et s'il ne réagit pas beaucoup, il se trouve

au milieu.

```
fviz_pca_var(res.pca.transpose, col.var=data$T3_6h_R2, label="none") + scale_color_gradient2(low="blue"
```




```
fviz_pca_ind(res.pca.transpose, axes=c(1, 2), autoLab="yes", col.ind=col_trait) + scale_color_gradient2
```



1.2.1 Commentaires à faire

2 Modèle linéaire

Pour cela on a besoin des données centrées réduites qu'on obtient avec la commande suivante :

```
data_new = as.data.frame(scale(data))
```

2.1 Etude de l'expression des gènes pour le traitement T3 à 6h

On récupère les valeurs du traitement T3 à 6h grâce aux commandes suivantes :

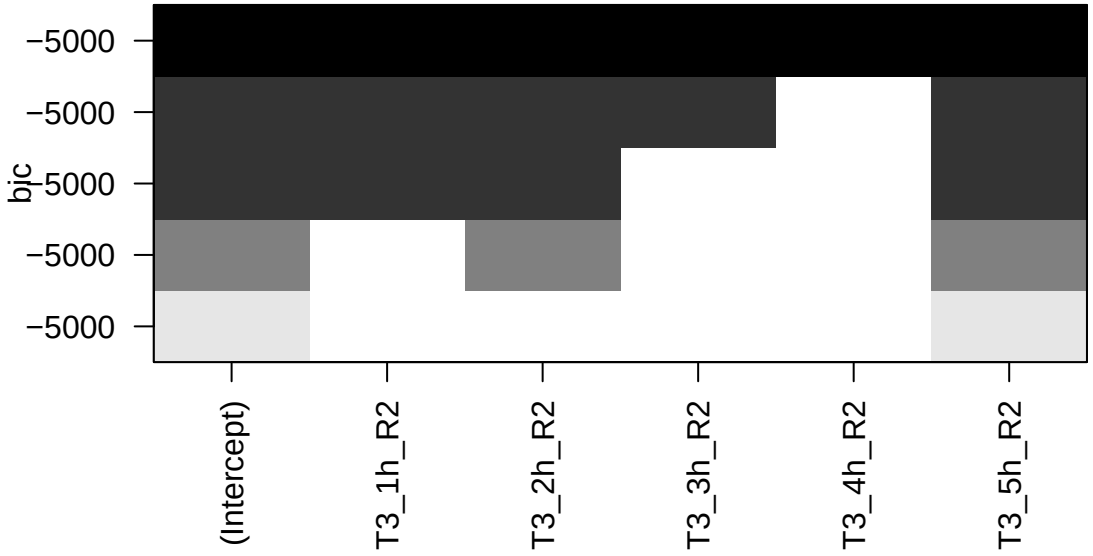
```
T3 = data[grep("T3", names(data), value=TRUE)]
T3R2 = T3[grep("R2", names(T3), value=TRUE)]
```

```
reg.mul <- lm(T3_6h_R2 ~ ., data = T3R2)
summary(reg.mul)
```

```
##
## Call:
## lm(formula = T3_6h_R2 ~ ., data = T3R2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.77335 -0.11316  0.00137  0.10221  1.12628
##
```

```
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.294e-17  5.212e-03   0.000        1
## T3_1h_R2     4.397e-02  8.067e-03   5.451 5.79e-08 ***
## T3_2h_R2    -1.697e-01  2.635e-02  -6.441 1.56e-10 ***
## T3_3h_R2     1.545e-01  3.293e-02   4.691 2.95e-06 ***
## T3_4h_R2    -1.642e-01  3.832e-02  -4.285 1.93e-05 ***
## T3_5h_R2     1.117e+00  2.815e-02  39.686  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2095 on 1609 degrees of freedom
## Multiple R-squared:  0.9563, Adjusted R-squared:  0.9561
## F-statistic:  7035 on 5 and 1609 DF,  p-value: < 2.2e-16
```

```
choix=regsubsets(T3_6h_R2~., data = T3R2, nbest = 1, nvmax = 5, method = "backward")
plot(choix,scale = "bic")
```



On a réalisé notre sélection de variables avec tous les critères (BIC, adjr2, Cp) et avec les méthodes forward et backward. Nous avons eu les mêmes résultats :

On garde toutes les variables mais on observe quand même une gradation. Le temps précédent (5h) est le plus influent suivi du temps de démarrage (1h, 2h). On peut faire l'hypothèse d'une périodicité de temps sur l'influence des traitements sur les gènes. Il faudrait avoir tester sur plus d'heures pour valider ou non cette hypothèse.

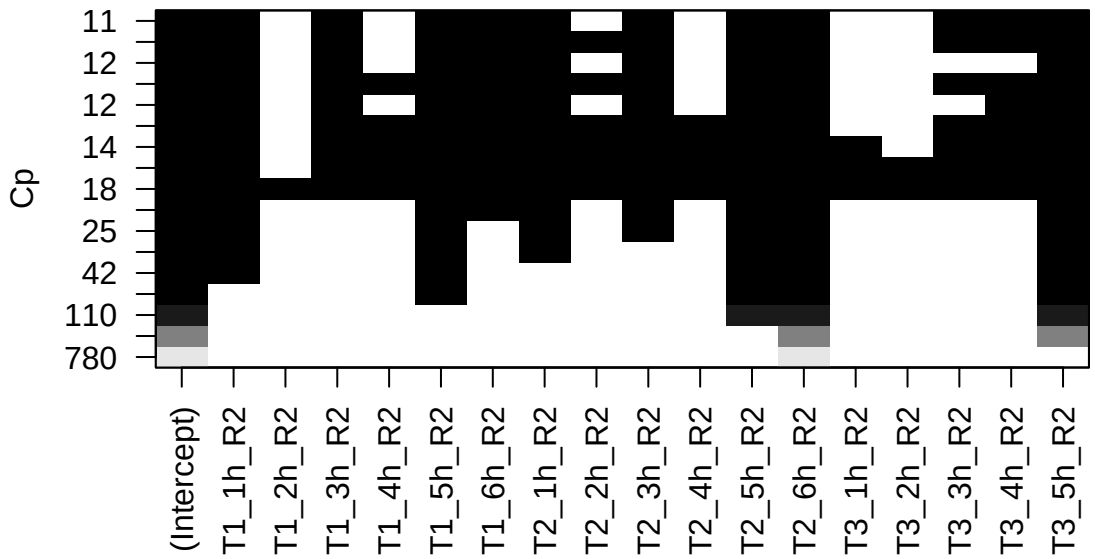
On étudie sur tous les traitements et tous les temps :

```
R2 = data_new[grep("R2", names(data_new), value = TRUE)]
reg.mul_complet <- lm(T3_6h_R2 ~ ., data = R2)
summary(reg.mul_complet)
```

```
##
## Call:
## lm(formula = T3_6h_R2 ~ ., data = R2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41156 -0.05433 -0.00149  0.05279  0.40686
```

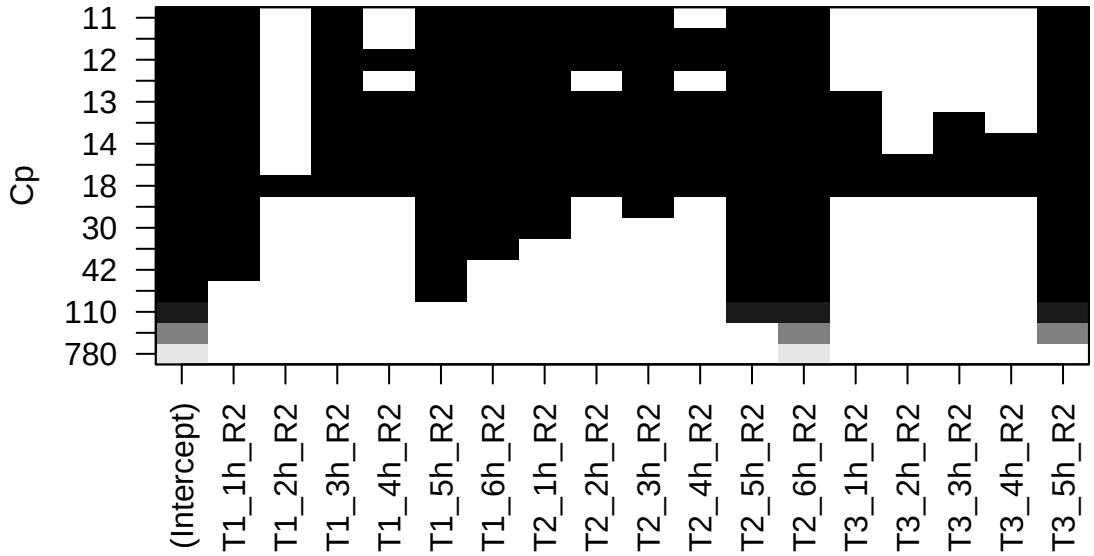
```
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.527e-17  2.399e-03   0.000 1.000000
## T1_1h_R2     1.113e-02  2.927e-03   3.801 0.000149 ***
## T1_2h_R2     4.820e-04  3.999e-03   0.121 0.904069
## T1_3h_R2     5.276e-03  4.305e-03   1.225 0.220578
## T1_4h_R2     5.396e-03  3.973e-03   1.358 0.174517
## T1_5h_R2    -2.862e-02  3.816e-03  -7.498 1.07e-13 ***
## T1_6h_R2     1.197e-02  3.821e-03   3.132 0.001766 **
## T2_1h_R2     1.657e-02  6.874e-03   2.410 0.016057 *
## T2_2h_R2     1.390e-02  1.636e-02   0.849 0.395798
## T2_3h_R2    -4.287e-02  1.757e-02  -2.440 0.014777 *
## T2_4h_R2    -2.099e-02  1.932e-02  -1.086 0.277425
## T2_5h_R2    -2.106e-01  2.347e-02  -8.972  < 2e-16 ***
## T2_6h_R2     8.069e-01  1.194e-02  67.559  < 2e-16 ***
## T3_1h_R2    -6.494e-03  6.962e-03  -0.933 0.351023
## T3_2h_R2     3.340e-03  1.846e-02   0.181 0.856433
## T3_3h_R2     1.977e-02  2.040e-02   0.969 0.332687
## T3_4h_R2    -2.701e-02  2.006e-02  -1.346 0.178355
## T3_5h_R2     4.410e-01  2.307e-02  19.119  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09642 on 1597 degrees of freedom
## Multiple R-squared:  0.9908, Adjusted R-squared:  0.9907
## F-statistic: 1.012e+04 on 17 and 1597 DF,  p-value: < 2.2e-16
```

```
choix=regsubsets(T3_6h_R2~., data = R2, nbest = 1, nvmax = 18, method = "backward")
plot(choix,scale = "Cp")
```



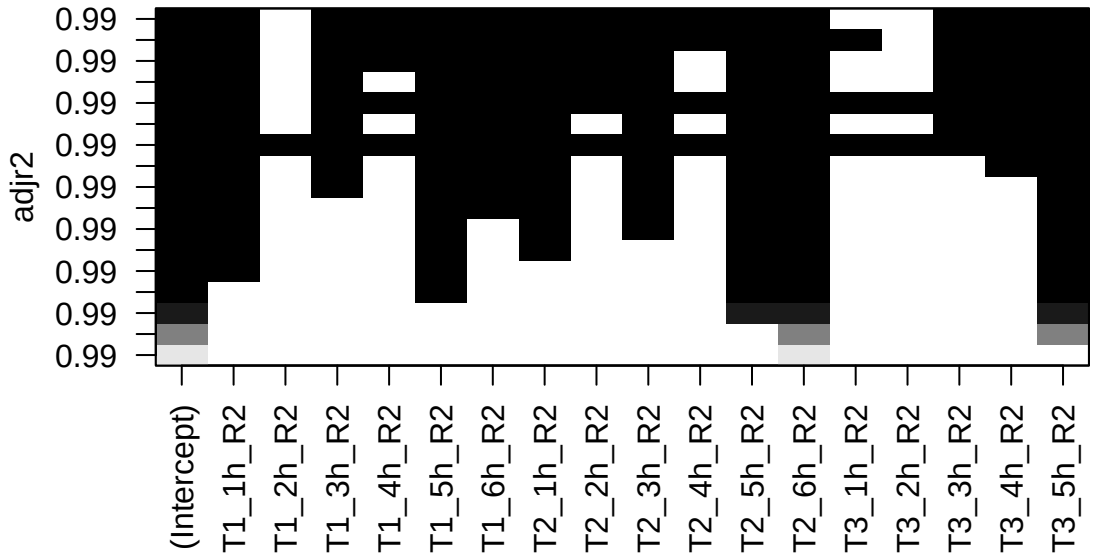
Critère Cp, méthode backward: T1: 1h, 3h, 5h, 6h T2: 1h, 3h, 5h, 6h T3: 3h, 4h, 5h

```
choix=regsubsets(T3_6h_R2~., data = R2, nbest = 1, nvmax = 18, method = "forward")
plot(choix,scale = "Cp")
```



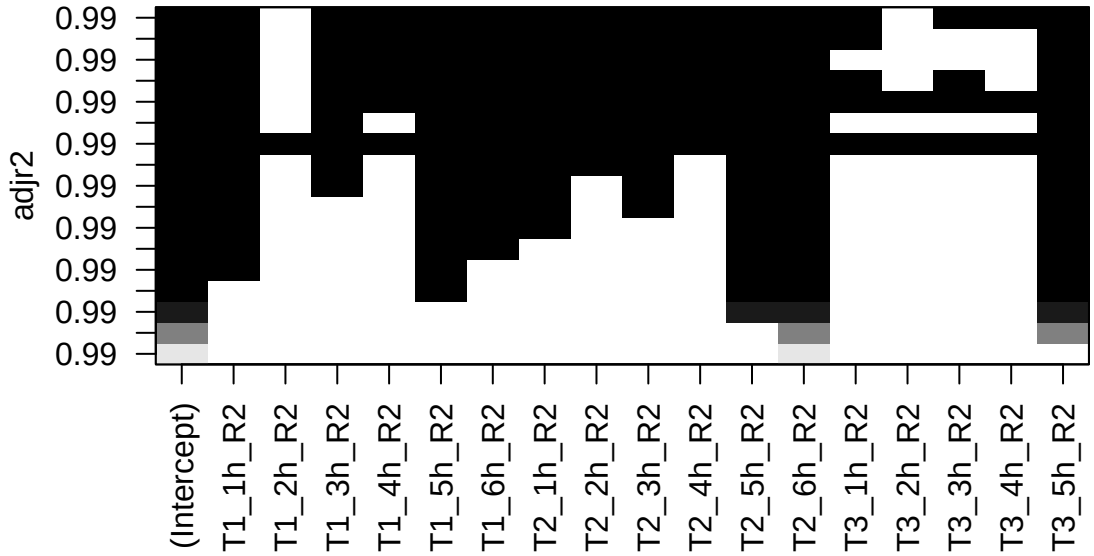
Critère Cp, méthode forward: T1: 1h, 3h, 5h, 6h T2: 1h, 2h, 3h, 5h, 6h T3: 5h

```
choix=regsubsets(T3_6h_R2~., data = R2, nbest = 1, nvmax = 18, method = "backward")
plot(choix,scale = "adjr2")
```



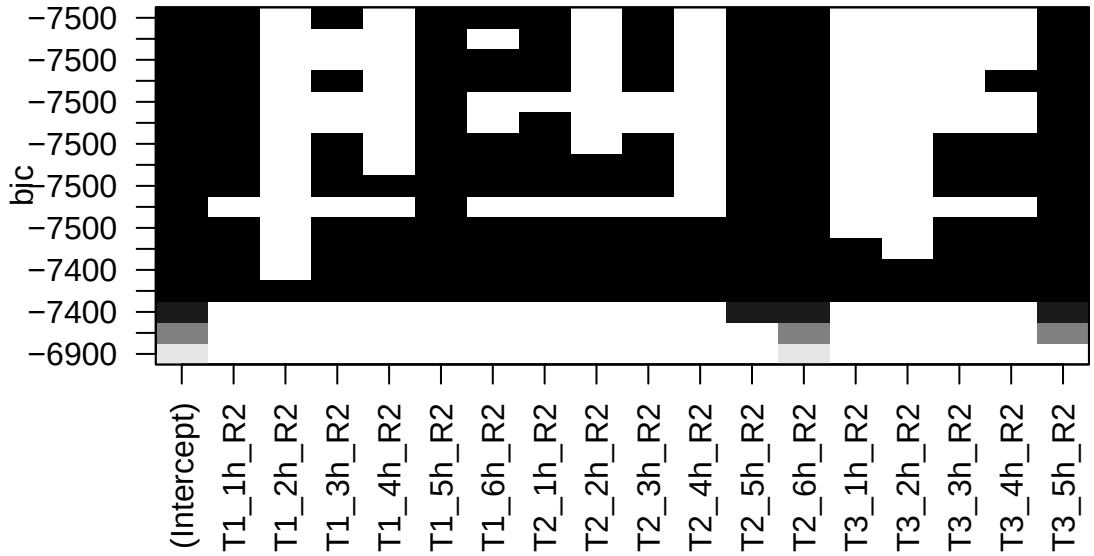
Critère bic, méthode backward: il enlève que T1 2h et T3 1h et 2h T1: 1h, 3h, 4h, 5h, 6h T2: 1h, 2h, 3h, 4h, 5h, 6h T3: 3h, 4h, 5h

```
choix=regsubsets(T3_6h_R2~., data = R2, nbest = 1, nvmax = 18, method = "forward")
plot(choix,scale = "adjr2")
```



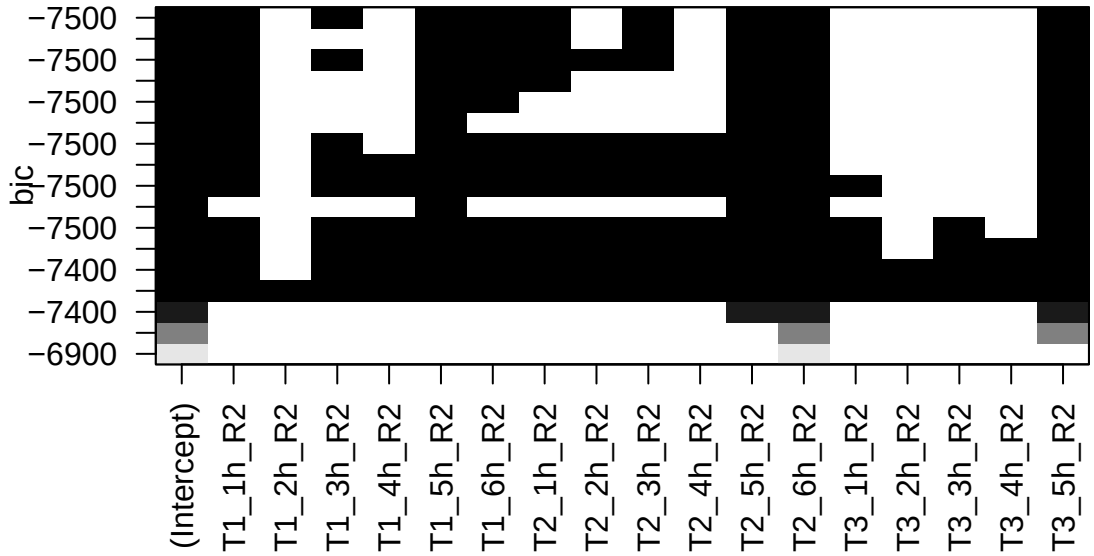
Critère bic, méthode forward: (enlève que T1 et T3 à 2h) T1: 1h, 3h, 4h, 5h, 6h T2: 1h, 2h, 3h, 4h, 5h, 6h T3: 1h, 3h, 4h, 5h

```
choix=regsubsets(T3_6h_R2~., data = R2, nbest = 1, nvmax = 18, method = "backward")
plot(choix,scale = "bic")
```



Critère bic, méthode backward: T1: 1h, 3h, 5h, 6h T2: 1h, 3h, 5h, 6h T3: 5h

```
choix=regsubsets(T3_6h_R2~., data = R2, nbest = 1, nvmax = 18, method = "forward")
plot(choix,scale = "bic")
```



Critère bic, méthode forward: T1: 1h, 3h, 5h, 6h T2: 1h, 3h, 5h, 6h T3: 5h

```
mod.debut = lm(T3_6h_R2 ~ ., data = R2)
mod.fin = lm(T3_6h_R2 ~ T1_1h_R2+T1_3h_R2+T1_5h_R2+T1_6h_R2+T2_1h_R2+T2_3h_R2+T2_5h_R2+T2_6h_R2+T3_5h_R
```

```
anova(mod.fin,mod.debut)
```

```
## Analysis of Variance Table
##
## Model 1: T3_6h_R2 ~ T1_1h_R2 + T1_3h_R2 + T1_5h_R2 + T1_6h_R2 + T2_1h_R2 +
##     T2_3h_R2 + T2_5h_R2 + T2_6h_R2 + T3_5h_R2
## Model 2: T3_6h_R2 ~ T1_1h_R2 + T1_2h_R2 + T1_3h_R2 + T1_4h_R2 + T1_5h_R2 +
##     T1_6h_R2 + T2_1h_R2 + T2_2h_R2 + T2_3h_R2 + T2_4h_R2 + T2_5h_R2 +
##     T2_6h_R2 + T3_1h_R2 + T3_2h_R2 + T3_3h_R2 + T3_4h_R2 + T3_5h_R2
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1   1605 14.937
## 2   1597 14.846  8  0.091151 1.2256 0.2798
```

p-valeur = 0.2798

ça correspond à l'analyse descriptive précedente, les gènes qui ont eu le traitement T2 ou le traitemet T3 ont des comportements similaires. Cela est cohérent avec le fait que T2 est plus influent sur T3 que T1 sur T3.

On retrouve par ailleurs les résultats de l'analyse précédente puisque les heures les plus influentes sont les heures les plus proches.

On veut chercher les variables prédictives qui permettent de discriminer les gènes sur-exprimés (Y>1) des gènes sous-exprimés (Y<-1) à 6h pour le traitement T3.

```
sur_exp = T3R2[T3R2>1]
```

```
sum(T3R2["T3_1h_R2">1])
```

```
## [1] 9.243301e-14
```