Projet : Régressions pénalisées

Import des librairies utiles

```r
library(mvtnorm)
library(palmerpenguins)
library(glmnet)
```

```
## Loading required package: Matrix

## Loaded glmnet 4.1-8
```

```r
library(xtable)
library(stargazer)
```

```
##
## Please cite as:

##  Hlavac, Marek (2022). stargazer: Well-Formatted Regression and Summary Statistics Tables.

##  R package version 5.2.3. https://CRAN.R-project.org/package=stargazer
```

```r
library(ggplot2)
library(gridExtra)
library(ordinal)
library(knitr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:ordinal':
##
##     slice

## The following object is masked from 'package:gridExtra':
##
##     combine

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'

## The following object is masked from 'package:dplyr':
##
```

```
##     group_rows
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(pls)
```

```
##
## Attaching package: 'pls'

## The following object is masked from 'package:corrplot':
##
##     corrplot

## The following object is masked from 'package:stats':
##
##     loadings
```

```
library(pROC)
```

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

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

Lecture des données

```
data=read.table("transportmod3.txt",h=T)
head(data)
```

```
##    CO2 Incid CA CA3 Port.Port.1 Port.Port.2 Port.Port.3 Port.Port.4 Port.Port.5
## 1 6.26    11  0   0       24.61        0.65        9.51        0.25        9.03
## 2 7.51     8  0   1       24.43        1.84       12.94        0.56        8.04
## 3 4.97     8  0   0       25.40        1.57       12.32        4.38        8.88
## 4 6.27     9  0   1       27.41        2.54       12.20        1.23        9.57
## 5 7.20    34  1   2       23.85        2.45       11.46        7.94       13.02
## 6 6.81    22  0   1       24.61        2.11       12.16        6.85       12.32
##   Port.Port.6 Port.Port.7 Port.Port.8 Port.Port.9 Port.Port.10 Port.Port.11
## 1        0.71       15.22       18.33       19.99        21.24         7.47
## 2        1.31       12.62       19.36       17.67        19.93         2.94
## 3        5.90       14.58       17.83       17.52        18.59         5.71
## 4        2.49       15.45       18.50       19.31        21.66         9.41
## 5        4.96       15.04       17.77       12.99        20.52         2.81
## 6        4.50       16.29       17.39       17.21        23.69         9.59
##   Port.Port.12 Port.Port.13 Port.Port.14 Port.Port.15 Port.Port.16 Port.Port.17
## 1         5.00         1.73        17.80        29.01         6.67        15.17
## 2         5.24         3.55        16.19        30.09         8.86        14.78
## 3         7.42         1.69        16.79        30.23         9.18        15.16
## 4         6.59         2.05        19.74        29.41         8.52        14.20
## 5         7.48         2.16        12.47        30.44         3.43        18.64
## 6         9.78         3.15        17.65        30.89         8.01        17.16
##   Port.Port.18 Port.Port.19 Port.Port.20 Port.Port.21 Port.Port.22 Port.Port.23
```

```
## 1         0.61        28.62         4.27        20.73        12.86         4.56
## 2         3.70        26.90         7.34        25.08         8.39         1.61
## 3         0.28        28.31        10.12        17.94        12.73         0.67
## 4         2.75        29.99         5.48        23.85         9.06         5.65
## 5         0.25        24.90         9.59        20.20        13.80         1.49
## 6         3.52        27.16         9.44        21.28         9.26         5.50
##   Port.Port.24 Port.Port.25 Port.Port.26 Port.Port.27 Port.Port.28 Port.Port.29
## 1         1.09        10.99        22.48         4.34        21.84         0.85
## 2         9.96         9.07        23.33         5.95        22.71         5.79
## 3         2.94         8.51        23.62         4.30        24.88         4.11
## 4         4.19        11.43        22.56         6.03        19.47         2.98
## 5         2.08         5.56        23.32         6.55        25.57         7.83
## 6         3.92        10.38        23.53         7.13        23.42         5.00
##   Port.Port.30 Port.Port.31 Port.Port.32 Port.Port.33 Port.Port.34 Port.Port.35
## 1        13.83         2.09        10.58        22.06        15.57        11.34
## 2         7.18         2.67         8.43        17.61        12.13         9.34
## 3        14.20         0.06        12.98        29.33        16.37        13.41
## 4        10.73         1.67         5.53        17.74        14.02        13.18
## 5        11.58         2.59        11.65        24.67        18.42        16.19
## 6        17.87         4.00        12.50        25.82        16.04        17.55
##   Port.Port.36 Port.Port.37 Port.Port.38 Port.Port.39 Port.Port.40 Port.Port.41
## 1         6.81        10.44         6.36         4.86         8.65         4.16
## 2         0.07        11.57         9.58         4.78         2.49         4.92
## 3         3.28        12.30         8.64         7.86         4.89         2.73
## 4         4.64        15.74         9.96         5.25         6.00         2.94
## 5         4.11        12.31         5.52         8.78         6.50         5.05
## 6         4.22        17.43        10.77         6.83         6.15         6.62
##   Port.Port.42 Port.Port.43 Port.Port.44 Port.Port.45 Port.Port.46 Port.Port.47
## 1         8.23         4.18         0.86        10.33         1.69        10.05
## 2        11.30         6.46         2.71        13.28         5.60         9.83
## 3         9.71         6.00         2.59         4.05         1.02        11.28
## 4        10.52         9.50         1.72         9.60         8.42        14.22
## 5         8.73         2.90         3.59        10.54         1.61         9.63
## 6         7.26         2.88         4.95        16.28         0.24         5.82
##   Port.Port.48 Port.Port.49 Port.Port.50 Port.Port.51 Port.Port.52 Port.Port.53
## 1         0.25         2.02         7.09        15.17         6.83         6.16
## 2         4.73         4.78         7.43        10.85         5.95         6.82
## 3         4.51         5.42         3.25        10.10         2.38         8.97
## 4         1.91         5.10        11.50         9.38         1.71         8.04
## 5         6.86         3.67         4.06        15.79         5.93         6.70
## 6         0.10         0.18        10.82        10.63         6.84         5.33
##   Port.Port.54 Port.Port.55 Port.Port.56 Port.Port.57 Port.Port.58 Port.Port.59
## 1        18.91        10.60        14.74        18.42        16.27        13.34
## 2        19.10        10.29        12.02        16.95        16.63        14.57
## 3        15.14         8.57        15.61        14.38         7.92        13.26
## 4        15.97        10.47        12.99        15.60        12.39        12.41
## 5        18.09        11.76        14.07        23.97        18.75        14.56
## 6        18.05        12.00         8.52        18.96        17.94        10.34
##   Port.Port.60
## 1        17.17
## 2        19.50
## 3        22.00
## 4        24.49
## 5        20.99
```

```
## 6        22.67
```

Statistiques descriptives

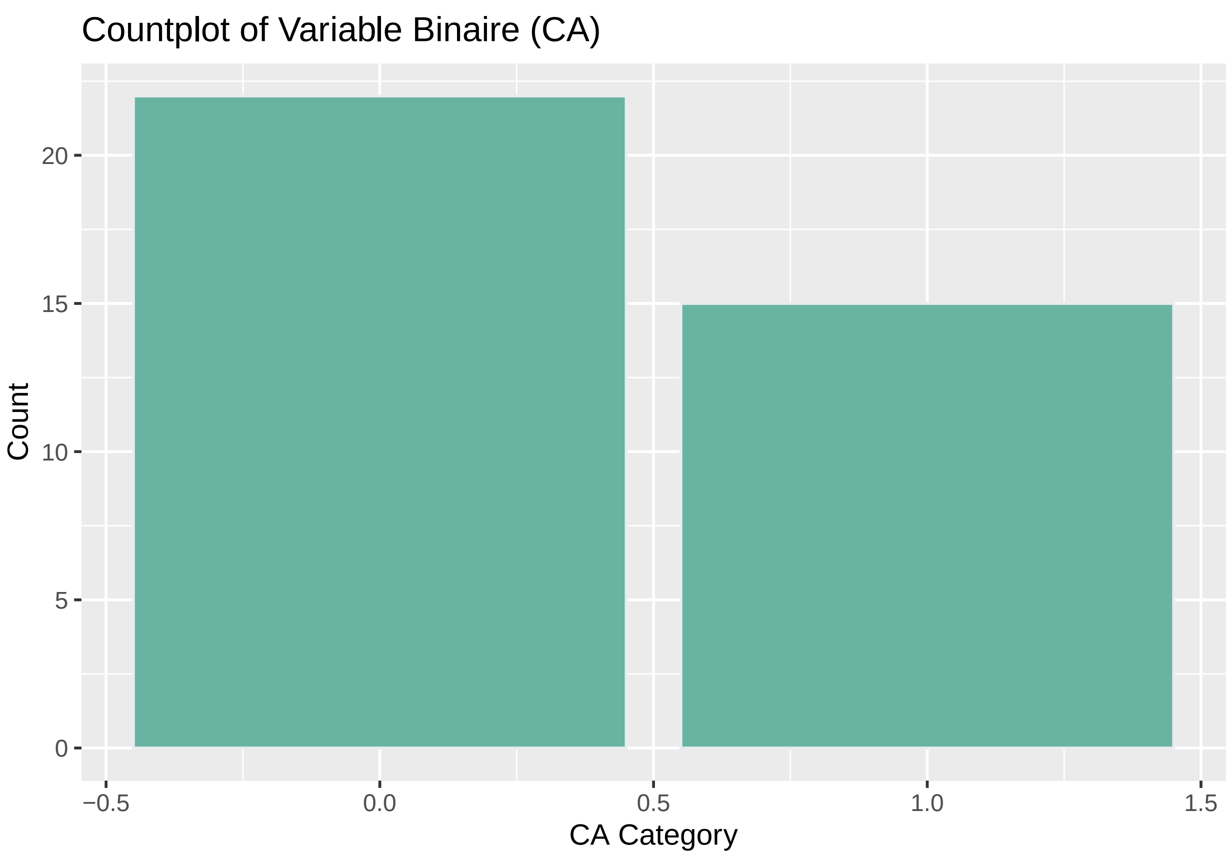
Valeurs descriptives

```r
# Sélection des dolonnes à décrire
colonnes_a_analyser <- c("CO2", "Incid", "CA")
donnees_selectionnees <- data[colonnes_a_analyser]

# Affichage des statistiques descriptives
res_summary <- summary(donnees_selectionnees)
table_latex <- xtable(res_summary)

# Génération du code LaTeX
print(table_latex)
```

```
## % latex table generated in R 4.3.2 by xtable 1.8-4 package
## % Thu Dec 14 18:43:40 2023
## \begin{table}[ht]
## \centering
## \begin{tabular}{rlll}
##   \hline
##  &      CO2 &     Incid &       CA \\
##   \hline
## X & Min.   : 4.970   & Min.   :  3.0   & Min.   :0.0000   \\
##   X.1 & 1st Qu.: 6.710   & 1st Qu.: 13.0   & 1st Qu.:0.0000   \\
##   X.2 & Median : 7.410   & Median : 27.0   & Median :0.0000   \\
##   X.3 & Mean   : 7.465   & Mean   : 34.7   & Mean   :0.4054   \\
##   X.4 & 3rd Qu.: 8.290   & 3rd Qu.: 42.0   & 3rd Qu.:1.0000   \\
##   X.5 & Max.   :10.550   & Max.   :152.0   & Max.   :1.0000   \\
##    \hline
## \end{tabular}
## \end{table}
```

Graphe des corrélations

```r
# Définition un seuil pour les fortes corrélations
threshold <- 0.7
correlation_matrix <- cor(data)

# Récupérer les noms de lignes et de colonnes de corrélations fortes
strong_correlations <- as.data.frame(which(abs(correlation_matrix) > threshold, arr.ind = TRUE))

# Créer un graphique en violon
ggplot(data, aes(x = CA)) +
  geom_bar(fill = "#69b3a2", color = "#e9ecef") +
  labs(title = "Countplot of Variable Binaire (CA)",
       x = "CA Category",
       y = "Count")
```



Conditionnenment de XX

```
X=as.matrix(data[,5:64])

# Calculer la matrice X'X
XTX <- t(X) %*% X

# Calculer les valeurs propres de X'X
eigen_XTX <- eigen(XTX)
cat("Valeurs propres de X'X:", eigen_XTX$values, "\n")
```

```
## Valeurs propres de X'X: 382245.6 1422.419 1215.233 1045.545 892.915 830.2399 743.8491 643.2138 570.9
```

```
# Calculer l'indice de conditionnement
condition_index <- max(eigen_XTX$values) / min(eigen_XTX$values)

# Afficher l'indice de conditionnement
cat("Indice de conditionnement:", condition_index, "\n")
```

```
## Indice de conditionnement: -2.591569e+16
```

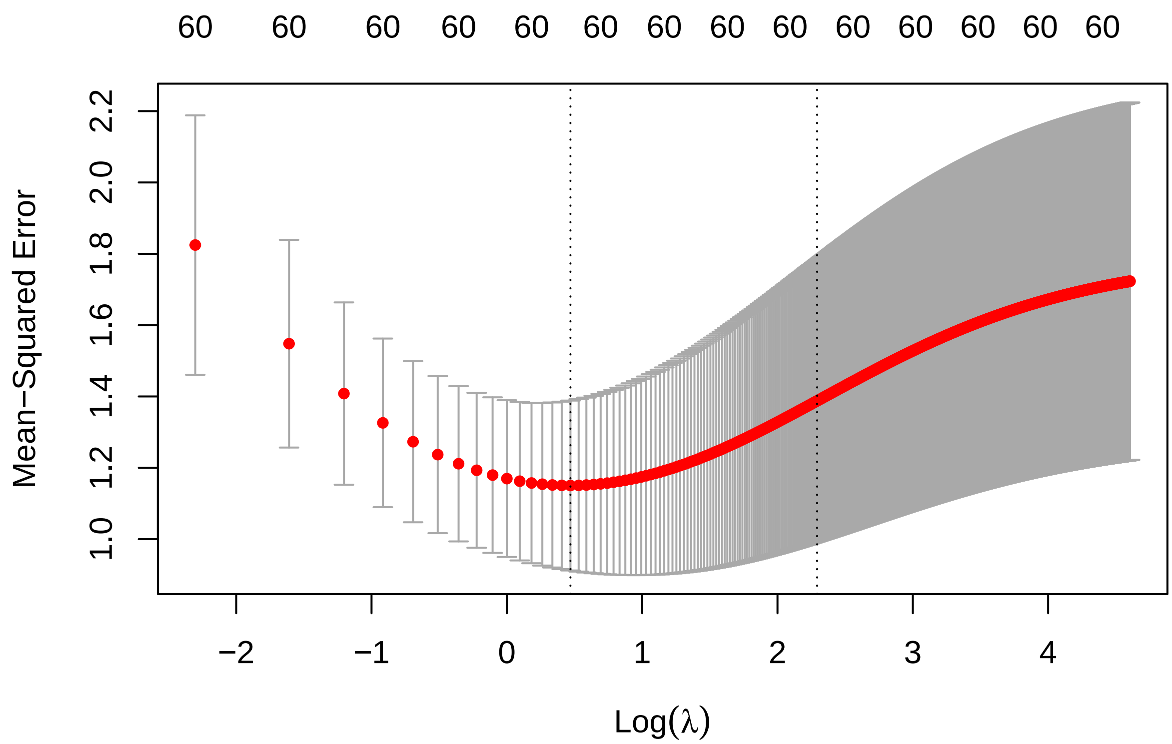
Modélisation du CO2

Standardisation de la donnée

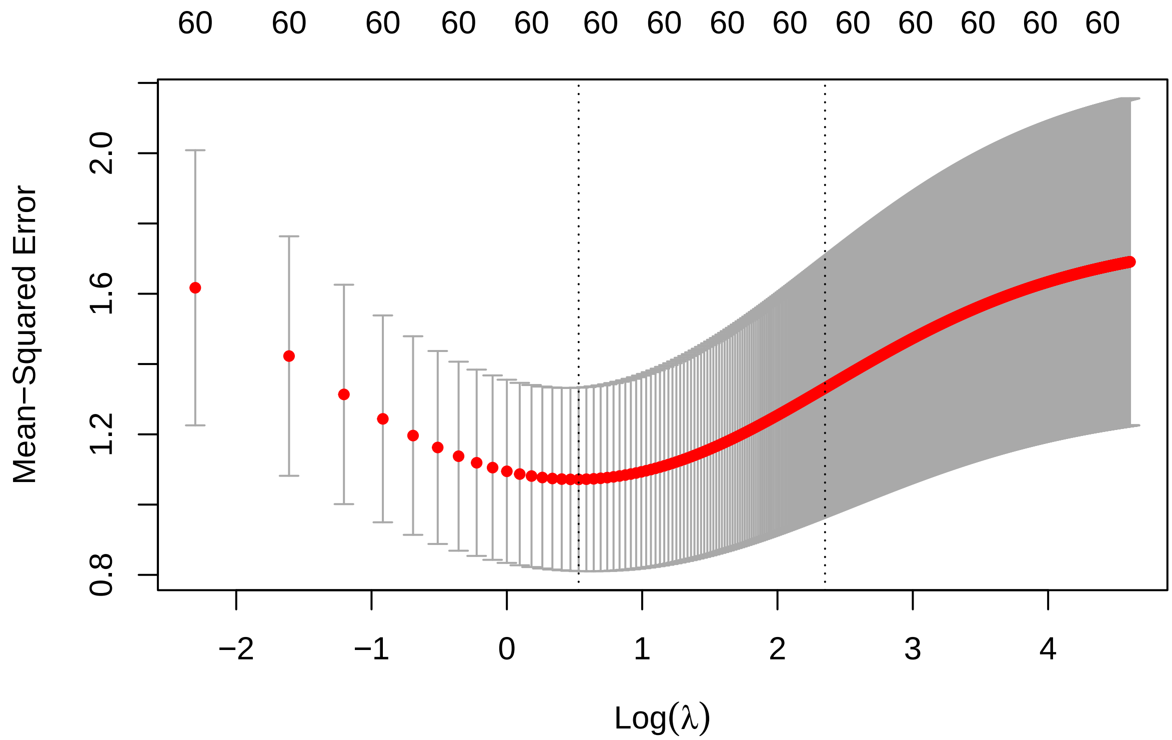
```r
#Centrer et réduire les variables explicatives
Y=data$CO2
X=as.matrix(data[,5:64])
XS=scale(X)
```

Modèle RIDGE

```r
# On a un modèle linéaire. On va prendre family = 'gaussian'
rescv=cv.glmnet(XS,Y,family='gaussian',alpha=0, lambda=seq(0.1, 100, 0.1))
plot(rescv)
```




```r
# Stabilisation du lambda avec une boucle
rescv=cv.glmnet(XS,Y,family='gaussian',alpha=0, lambda=seq(0.1, 100, 0.1))
plot(rescv)
```




```
lambda=rescv$lambda.min
lambda
```

```
## [1] 1.7
```

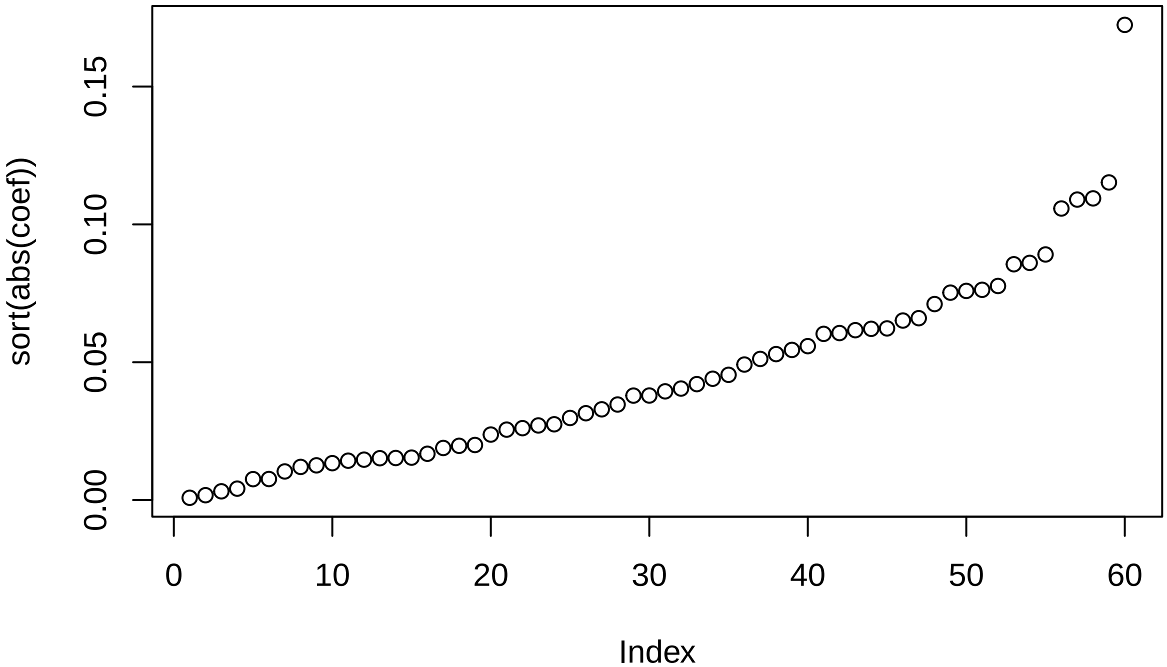
```
# Autre méthode
lambda=0
for (j in 1:10)
{
  rescv=cv.glmnet(XS,Y,family='gaussian',alpha=0, lambda=seq(0.1, 100, 0.1))
  print(rescv$lambda.min)
  lambda=lambda+rescv$lambda.min
}
```

```
## [1] 2.3
## [1] 1.9
## [1] 2.2
## [1] 1.9
## [1] 1.8
## [1] 2
## [1] 2
## [1] 2.2
## [1] 1.9
## [1] 2.3
```

Redéfinition du seuil

```
seuil=lambda/10
resridge=glmnet(XS,Y,family='gaussian',alpha=0, lambda=seuil)
#On retire le coefficient 1
coef=coefficients(resridge)[-1]
```

```r
#On affiche
plot(sort(abs(coef)))
```



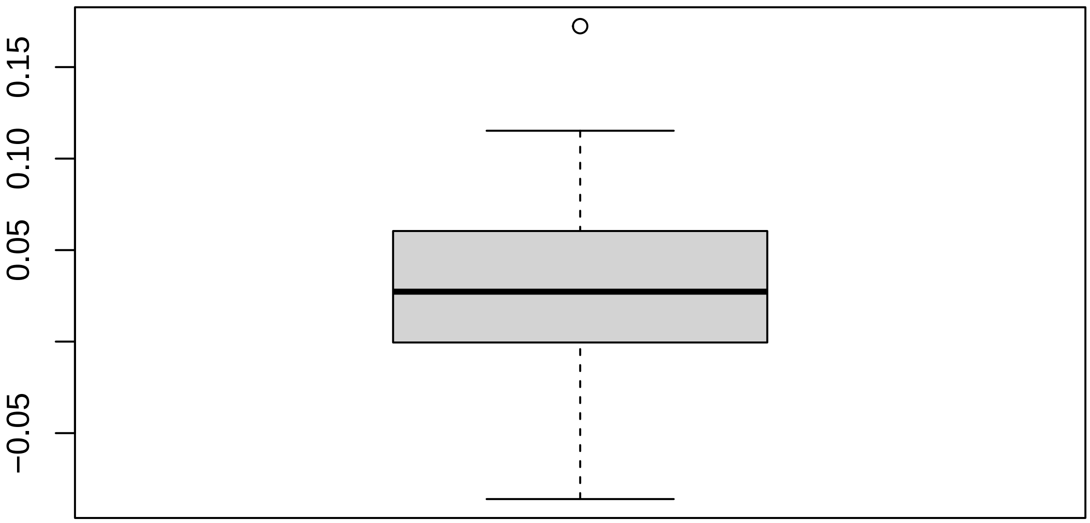
On prend les 8 premiers qui se détachent du groupe.

Choix des ports

```r
#En fonction du plot
which(abs(coef)>0.6)
```

```
## integer(0)
```

```r
# A l'aide d'un boxplot
resbox=boxplot(coef)
```




```r
which(coef>resbox$stats[5,])
```

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

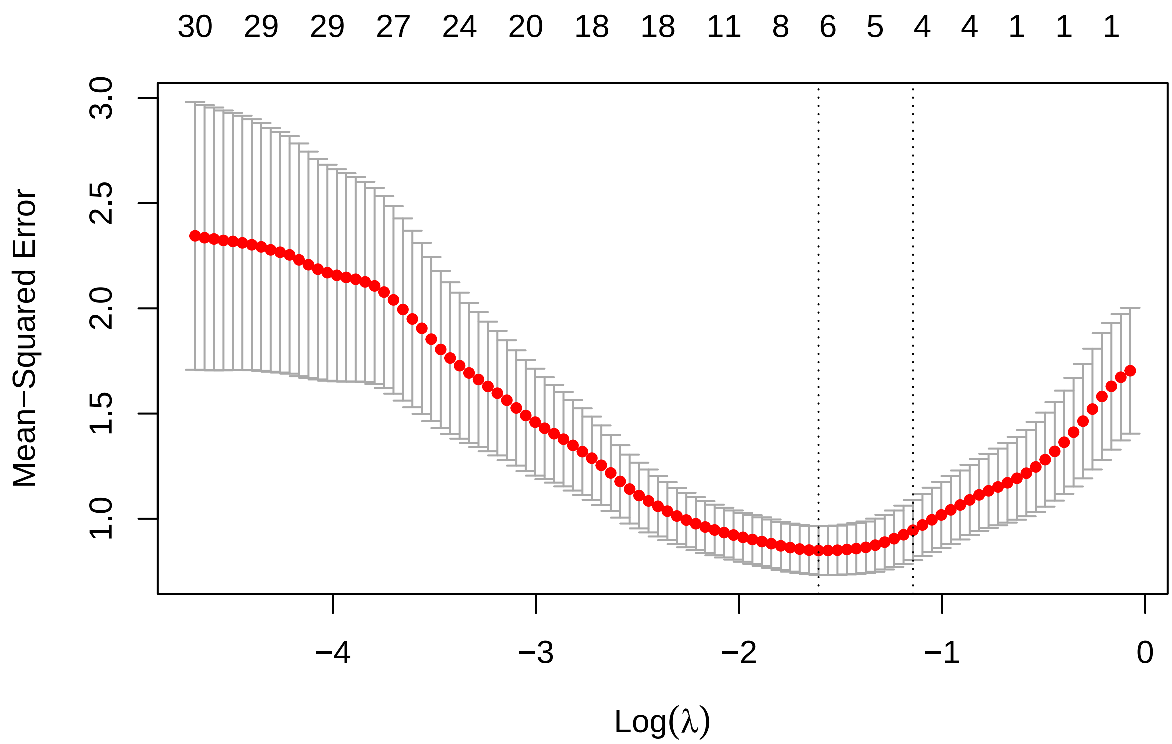
```
selecridge=order(coef,decreasing=TRUE)[1:5]
colnames(X[,selecridge])
```

```
## [1] "Port.Port.5"  "Port.Port.3"  "Port.Port.58" "Port.Port.34" "Port.Port.8"
```

Les ports retenus sont: 5 3 58 34 8.

Modèle LASSO

```
#Modèle LASSO
rescv=cv.glmnet(XS,Y,family='gaussian',alpha=1)
plot(rescv)
```




```
# Prenons le lambda_1se qui sÃ©lectionne un peu moins de variables
seuil=rescv$lambda.1se
reslasso=glmnet(XS,Y,family='gaussian',alpha=1,lambda=seuil)

#Boucle sur le lambda
lambda=0
for (j in 1:10)
{
  reslasso=cv.glmnet(XS,Y,family='gaussian',alpha=1, lambda=seq(0.1, 100, by = 0.1))
  print(reslasso$lambda.1se)
  lambda=lambda+reslasso$lambda.1se
}
```

```
## [1] 0.3
## [1] 0.3
## [1] 0.3
## [1] 0.3
## [1] 0.4
```

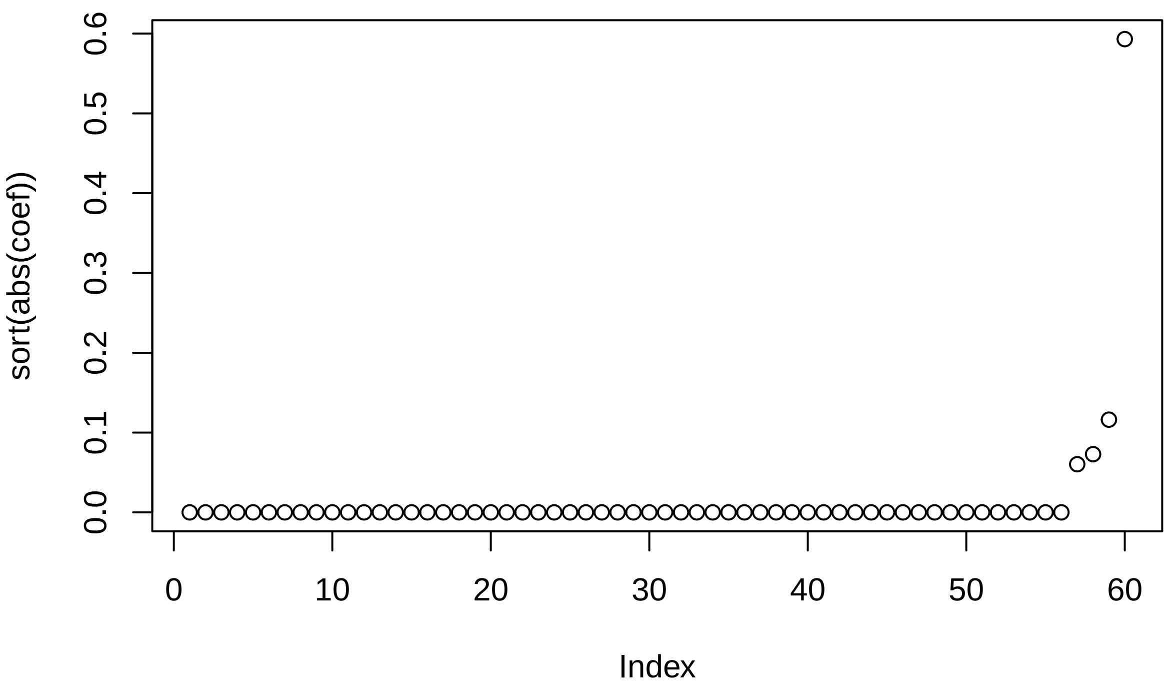
```
## [1] 0.3
## [1] 0.4
## [1] 0.3
## [1] 0.3
## [1] 0.3
```

#Définition du seuil

```
#Par calcul de la moyenne
seuil=lambda/10
seuil
```

```
## [1] 0.32
```

```
# On va alors redefinir un meilleur seuil et relancer notre régression
reslasso=glmnet(XS,Y,family='gaussian',alpha=1, lambda=seuil)
coef=coefficients(reslasso)[-1]
plot(sort(abs(coef)))
```



On sélectionne 4 variables

```
selected_va <- which(coef!=0)
selected_va
```

```
## [1]  3  5  7 58
```

On obtient les ports: 3 5 7 58.

Modèle Elasticnet

Modèle

```
"On procède de la meme manière en stabilisant notre modèle sur un intervalle de test de valeur"
```

```
## [1] "On procède de la meme manière en stabilisant notre modèle sur un intervalle de test de valeur"
```

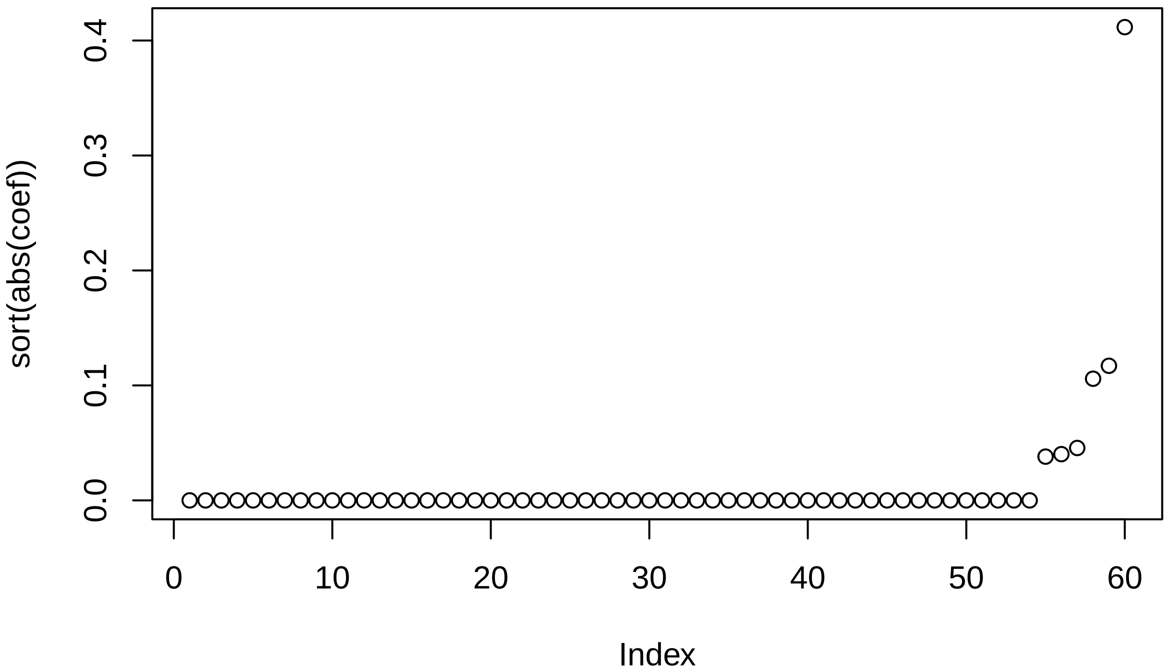
```r
lambda=0
for (j in 1:10)
{
  resenet=cv.glmnet(XS,Y,family='gaussian',alpha=0.5,lambda=seq(0.1, 100, 0.1))
  print(resenet$lambda.1se)
  lambda=lambda+resenet$lambda.1se
}
```

```
## [1] 0.5
## [1] 0.6
## [1] 0.7
## [1] 0.6
## [1] 0.6
## [1] 0.6
## [1] 0.5
## [1] 0.7
## [1] 0.7
## [1] 0.8
```

```r
lambda.1se=lambda/10
lambda.1se
```

```
## [1] 0.63
```

```r
resenet=glmnet(XS,Y,family='gaussian',alpha=0.5,lambda=lambda.1se)
coef=coefficients(resenet)[-1]
plot(sort(abs(coef)))
```




```r
which(coef!=0)
```

```
## [1]  3  5  7  8 34 58
```

```r
coef
```

```
##  [1]  0.00000000  0.00000000  0.10582428  0.00000000  0.41164710  0.00000000
```

```
##  [7] -0.04021975  0.03806654  0.00000000  0.00000000  0.00000000  0.00000000
## [13]  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000
## [19]  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000
## [25]  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000
## [31]  0.00000000  0.00000000  0.00000000  0.04558936  0.00000000  0.00000000
## [37]  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000
## [43]  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000
## [49]  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000
## [55]  0.00000000  0.00000000  0.00000000  0.11707652  0.00000000  0.00000000
```

Les ports: 3 5 7 8 34 58.

Modèle Linéaire

```r
linear_model <- lm(Y ~ X[, selected_va])
stargazer(linear_model, title = "Linear Regression Results", out = "table.tex")
```

```
##
## % Table created by stargazer v.5.2.3 by Marek Hlavac, Social Policy Institute. E-mail: marek.hlavac a
## % Date and time: Thu, Dec 14, 2023 - 18:43:49
## \begin{table}[!htbp] \centering
##   \caption{Linear Regression Results}
##   \label{}
## \begin{tabular}{@{\extracolsep{5pt}}lc}
## \\[-1.8ex]\hline
## \hline \\[-1.8ex]
##  & \multicolumn{1}{c}{\textit{Dependent variable:}} \\
## \cline{2-2}
## \\[-1.8ex] & Y \\
## \hline \\[-1.8ex]
##  X[, selected\_va]Port.Port.3 & 0.094 \\
##   & (0.078) \\
##   & \\
##  X[, selected\_va]Port.Port.5 & 0.430$^{***}$ \\
##   & (0.074) \\
##   & \\
##  X[, selected\_va]Port.Port.7 & $-$0.313$^{***}$ \\
##   & (0.093) \\
##   & \\
##  X[, selected\_va]Port.Port.58 & 0.060 \\
##   & (0.040) \\
##   & \\
##  Constant & 5.065$^{***}$ \\
##   & (1.792) \\
##   & \\
## \hline \\[-1.8ex]
## Observations & 37 \\
## R$^{2}$ & 0.729 \\
## Adjusted R$^{2}$ & 0.695 \\
## Residual Std. Error & 0.723 (df = 32) \\
## F Statistic & 21.555$^{***}$ (df = 4; 32) \\
## \hline
## \hline \\[-1.8ex]
## \textit{Note:}  & \multicolumn{1}{r}{$^{*}$p$<$0.1; $^{**}$p$<$0.05; $^{***}$p$<$0.01} \\
```

```
## \end{tabular}
## \end{table}
```

```r
print(selected_va)
```

```
## [1]  3  5  7 58
```

```r
# Afficher le modèle linéaire
summary(linear_model)
```

```
##
## Call:
## lm(formula = Y ~ X[, selected_va])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.09525 -0.63607  0.09453  0.57374  1.44511
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  5.06467    1.79191   2.826  0.00805 **
## X[, selected_va]Port.Port.3  0.09383    0.07793   1.204  0.23742
## X[, selected_va]Port.Port.5  0.42989    0.07447   5.773  2.1e-06 ***
## X[, selected_va]Port.Port.7 -0.31274    0.09281  -3.370  0.00198 **
## X[, selected_va]Port.Port.58 0.06026    0.04019   1.499  0.14355
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7227 on 32 degrees of freedom
## Multiple R-squared:  0.7293, Adjusted R-squared:  0.6955
## F-statistic: 21.55 on 4 and 32 DF,  p-value: 1.052e-08
```

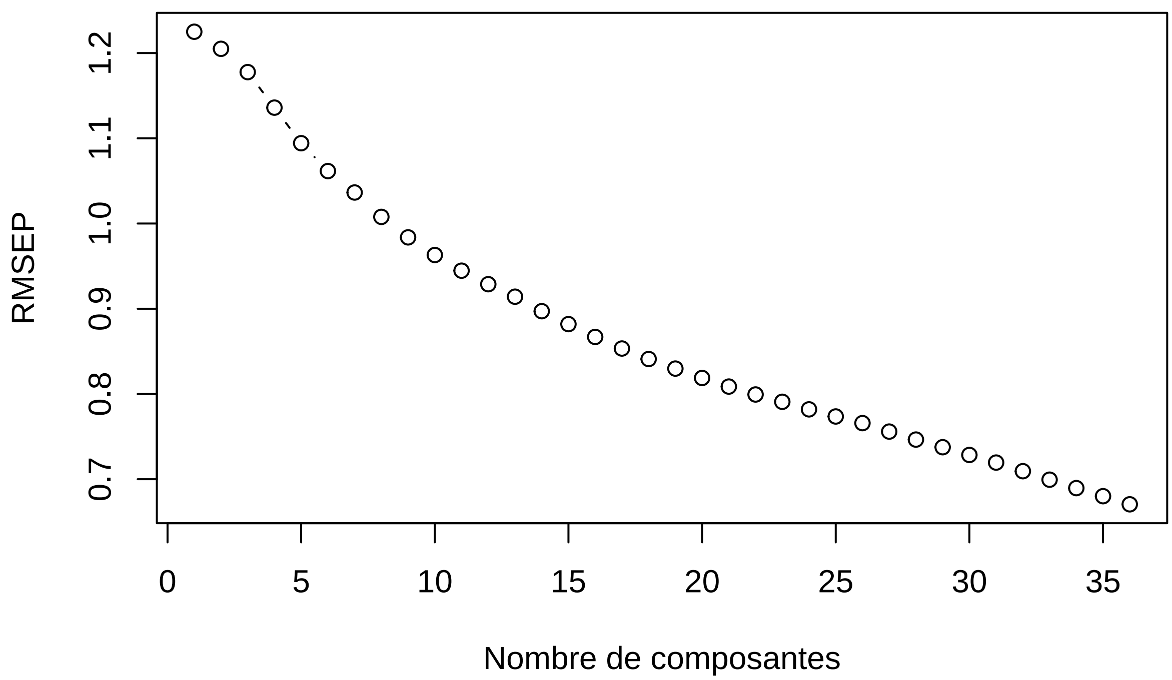
Modèle PCA

```r
# Effectuez le PCR
pcr_model <- pcr(Y ~ XS, scale = TRUE)

# Résumé du modèle PCR
summary(pcr_model)
```

```
## Data:    X dimension: 37 60
##  Y dimension: 37 1
## Fit method: svdpc
## Number of components considered: 36
## TRAINING: % variance explained
##    1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X    11.91    21.74    30.67    38.54    45.42    51.98    57.54    62.55
## Y    10.07    15.90    24.69    40.01    50.56    53.59    54.61    63.65
##    9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X    67.17     71.21     75.09     78.12     80.87     83.17     85.12
## Y    64.96     66.15     67.54     67.90     69.36     75.77     76.02
##    16 comps  17 comps  18 comps  19 comps  20 comps  21 comps  22 comps
## X     87.02     88.73     90.30     91.63     92.82     93.88     94.88
## Y     78.59     78.80     78.87     78.99     80.46     80.49     80.50
##    23 comps  24 comps  25 comps  26 comps  27 comps  28 comps  29 comps
```

```
## X      95.77     96.45     97.10     97.64     98.16     98.55     98.89
## Y      80.51     82.61     82.77     82.82     89.36     89.38     89.65
##    30 comps  31 comps  32 comps  33 comps  34 comps  35 comps  36 comps
## X      99.20     99.43     99.58     99.71     99.83     99.92       100
## Y      91.38     92.27     96.67     97.66     98.64     98.72       100
```

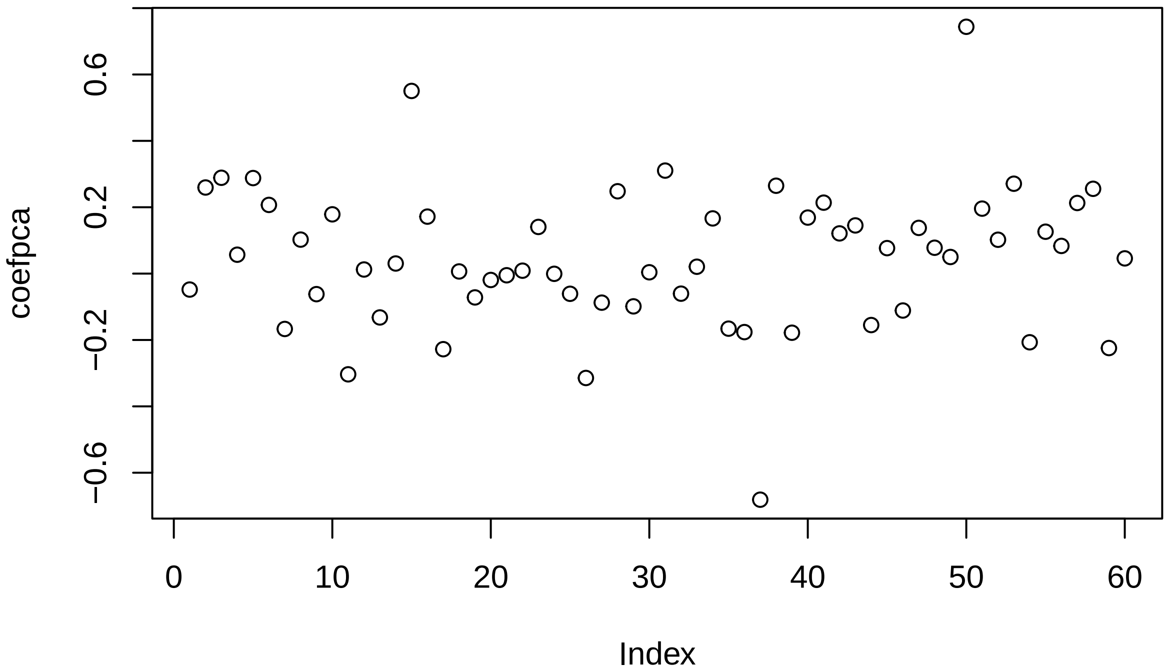
```r
# Initialisation du vecteur pour stocker les RMSEP
rmsep_values <- numeric(36)

# Boucle sur le nombre de composantes
for (i in 1:36) {
  pcr_model <- pcr(Y ~ XS, ncomp = i, scale = TRUE)

  # Prédiction sur l'ensemble de données
predicted_values <- predict(pcr_model, as.data.frame(X))

  # Calcul du RMSEP
rmsep_values[i] <- sqrt(mean((Y - predicted_values)^2))
}

# Tracer le graphique du RMSEP
plot(1:36, rmsep_values, type = "b", xlab = "Nombre de composantes", ylab = "RMSEP")
```



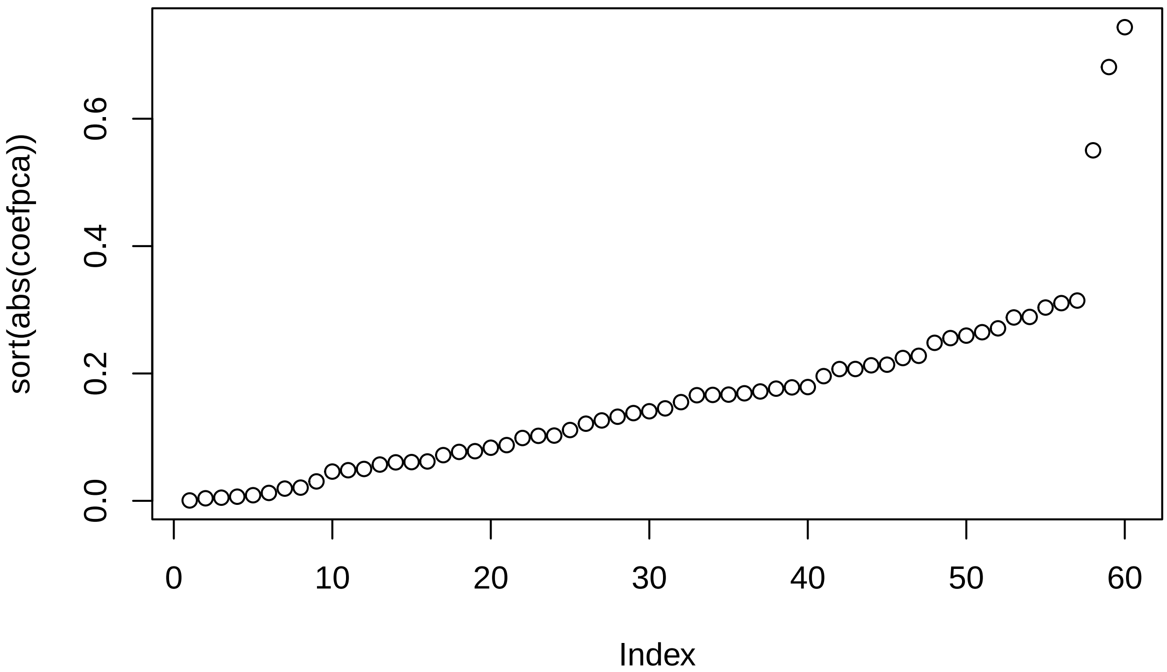

```r
# Trouver le nombre optimal de composantes
optimal_components <- which.min(rmsep_values)

# Seuillage pour les coefficients
threshold <- 0.3 # Choisissez votre seuil
significant_coef <- abs (coef(pcr_model)) > threshold

respca2=plsr(Y ~XS, ncomp=16)
summary(respca2)
```

```
## Data:    X dimension: 37 60
##  Y dimension: 37 1
## Fit method: kernelpls
## Number of components considered: 16
## TRAINING: % variance explained
##    1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X    9.075    17.28    23.17    28.79    34.25    38.51    44.43    49.21
## Y   69.775    80.21    86.23    89.98    92.86    95.56    96.56    97.35
##    9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X    54.01     57.50     60.37     64.54     67.84     71.85     74.51
## Y    98.11     98.85     99.39     99.62     99.76     99.82     99.89
##    16 comps
## X     77.17
## Y     99.93
```

```r
coefpca=coef(respca2)
par(mfrow=c(1,1))
plot(coefpca) # on pourrait faire du seuillage ici
```




```r
plot(sort(abs(coefpca)))
```




```
selecpca=order(abs(coefpca),decreasing=TRUE)[1:9]

colnames(X[,selecpca])
```

```
## [1] "Port.Port.50" "Port.Port.37" "Port.Port.15" "Port.Port.26" "Port.Port.31"
## [6] "Port.Port.11" "Port.Port.3"  "Port.Port.5"  "Port.Port.53"
```

##Modèle PLS

```
# Réaliser la régression PLS
pls_model <- plsr(Y ~ XS, ncomp = 16)  # Vous pouvez ajuster ncomp selon votre analyse

# Afficher un résumé du modèle PLS
summary(pls_model)
```

```
## Data:    X dimension: 37 60
##  Y dimension: 37 1
## Fit method: kernelpls
## Number of components considered: 16
## TRAINING: % variance explained
##    1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X    9.075    17.28    23.17    28.79    34.25    38.51    44.43    49.21
## Y   69.775    80.21    86.23    89.98    92.86    95.56    96.56    97.35
##    9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X    54.01     57.50     60.37     64.54     67.84     71.85     74.51
## Y    98.11     98.85     99.39     99.62     99.76     99.82     99.89
##    16 comps
## X     77.17
## Y     99.93
```

```
# Visualiser les coefficients PLS
coef_pls <- coef(pls_model, ncomp = 16)  # Vous pouvez ajuster ncomp ici aussi
print(coef_pls)
```

```
## , , 16 comps
```

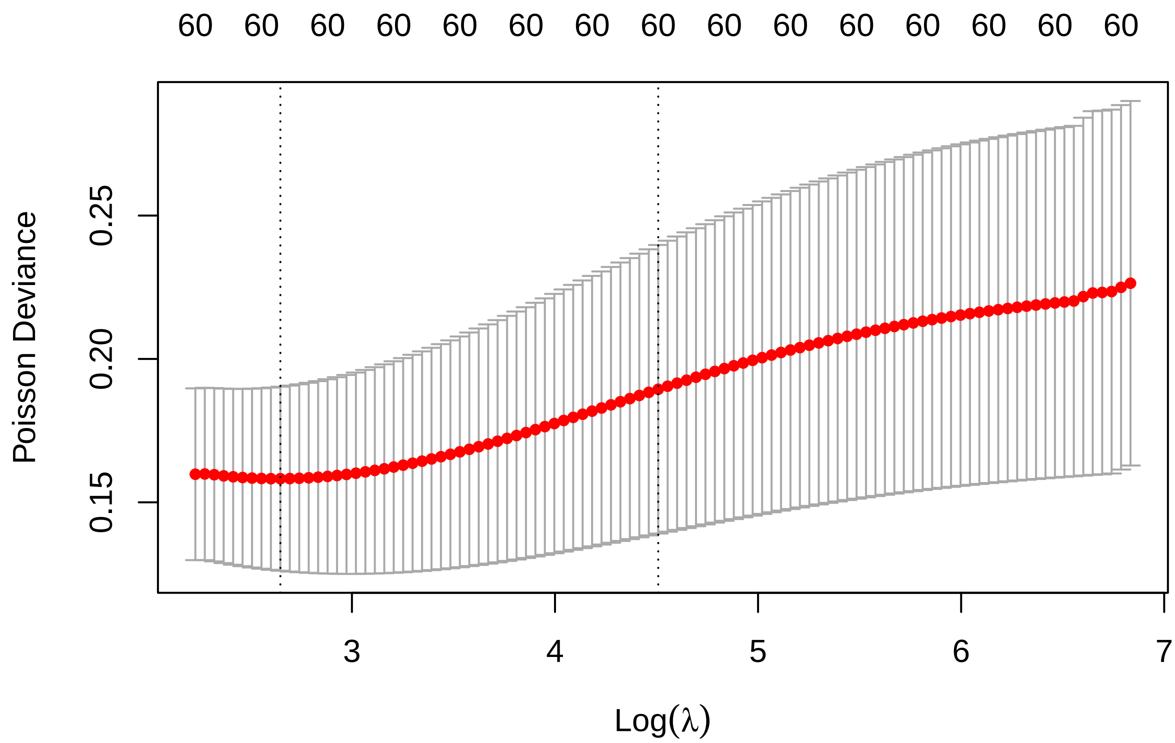
```
##
##                        Y
## Port.Port.1  -0.0480640114
## Port.Port.2   0.2594567631
## Port.Port.3   0.2887707800
## Port.Port.4   0.0569410378
## Port.Port.5   0.2879171414
## Port.Port.6   0.2070196206
## Port.Port.7  -0.1667723657
## Port.Port.8   0.1024610846
## Port.Port.9  -0.0618313286
## Port.Port.10  0.1786375903
## Port.Port.11 -0.3035112929
## Port.Port.12  0.0124025913
## Port.Port.13 -0.1320879834
## Port.Port.14  0.0304897491
## Port.Port.15  0.5504000094
## Port.Port.16  0.1717246771
## Port.Port.17 -0.2277282950
## Port.Port.18  0.0065216484
## Port.Port.19 -0.0717609931
## Port.Port.20 -0.0192261395
## Port.Port.21 -0.0049550482
## Port.Port.22  0.0088235516
## Port.Port.23  0.1406508179
## Port.Port.24 -0.0006142111
## Port.Port.25 -0.0608353876
## Port.Port.26 -0.3144918732
## Port.Port.27 -0.0874973000
## Port.Port.28  0.2481399271
## Port.Port.29 -0.0986739100
## Port.Port.30  0.0040104813
## Port.Port.31  0.3104189204
## Port.Port.32 -0.0604015128
## Port.Port.33  0.0207618401
## Port.Port.34  0.1663228307
## Port.Port.35 -0.1658395022
## Port.Port.36 -0.1761967634
## Port.Port.37 -0.6811737881
## Port.Port.38  0.2647416575
## Port.Port.39 -0.1781382938
## Port.Port.40  0.1688296535
## Port.Port.41  0.2137988165
## Port.Port.42  0.1211669439
## Port.Port.43  0.1451863711
## Port.Port.44 -0.1549744168
## Port.Port.45  0.0768054369
## Port.Port.46 -0.1111946686
## Port.Port.47  0.1377664477
## Port.Port.48  0.0779499755
## Port.Port.49  0.0500654422
## Port.Port.50  0.7437104125
## Port.Port.51  0.1957948988
## Port.Port.52  0.1020543766
```

```
## Port.Port.53  0.2708461860
## Port.Port.54 -0.2068762343
## Port.Port.55  0.1262169362
## Port.Port.56  0.0834096003
## Port.Port.57  0.2127966093
## Port.Port.58  0.2555194380
## Port.Port.59 -0.2242603330
## Port.Port.60  0.0459284650
```

#Modélisation du nombre d'incidents

Modélisation Ridge de Poisson

```r
#Modèle de Ridge de poisson
rescv_p <- cv.glmnet(XS, Y, family = "poisson", alpha = 0)
plot(rescv_p)
```




```r
# Sélectionner le meilleur modèle en # utilisant la validation croisée
lambda=0
for (j in 1:10)
{
  rescv_poisson <- cv.glmnet(XS, Y, family = "poisson", alpha=0, lambda=seq(0.1, 100, 0.1))
  # Trouver le meilleur lambda (paramètre de régularisation)
  print(rescv_poisson$lambda.min)
  lambda=lambda+rescv_poisson$lambda.min
}
```

```
## [1] 13.8
## [1] 15
## [1] 10.4
## [1] 17.1
```

```
## [1] 11.4
## [1] 13.6
## [1] 15.1
## [1] 13.5
## [1] 15.8
## [1] 13.6
```

```r
seuil=lambda/10

#On refait notre modèle avec le nouveau seuil trouvé
rescv_poisson <- glmnet(XS, Y, family = "poisson", alpha=0, lambda=seuil)

# Obtenir le vecteur des coefficients du meilleur modèle
best_coeffs <- coef(rescv_poisson)

# Sélectionner les variables non nulles du meilleur modèle
selected_vars <- which(best_coeffs != 0)

# Obtenez le nom des variables sélectionnées (en supposant que vous avez des noms de variables)
selected_var_names <- colnames(XS)[selected_vars]

# Deuxieme méthode
coef=coefficients(rescv_poisson)[-1] # on retire le coefficient 1
'la plupart de nos coeff sont différents de zéro donc une autre technique serait optimal
notamment un choix basé sur le seuillage ??'
```

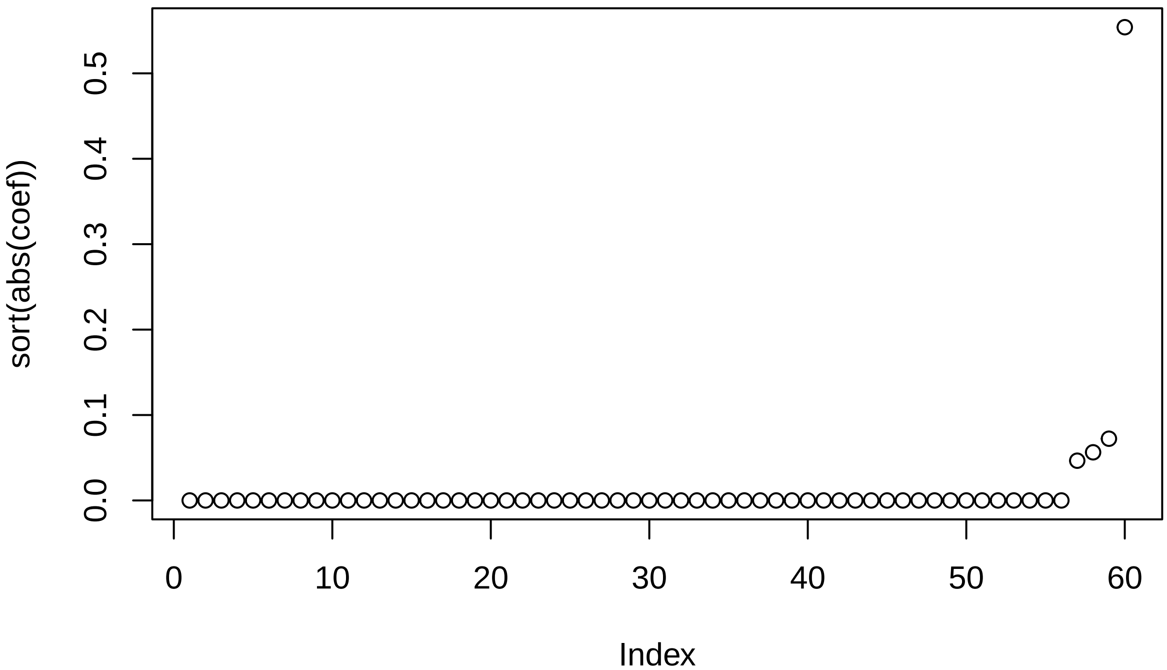
```
## [1] "la plupart de nos coeff sont différents de zéro donc une autre technique serait optimal\nnotamme
```

#Lasso de Poisson

```r
lambda=0
for (j in 1:10)
{
Lasso_poisson <- cv.glmnet(XS, Y, family = "poisson", alpha=1, lambda=seq(0.1, 100, 0.1))
  # Trouver le meilleur lambda (paramètre de régularisation)
  print(Lasso_poisson$lambda.1se)
  lambda=lambda+Lasso_poisson$lambda.1se
}
```

```
## [1] 0.6
## [1] 0.3
## [1] 0.3
## [1] 0.3
## [1] 0.3
## [1] 0.3
## [1] 0.3
## [1] 0.3
## [1] 0.6
## [1] 0.3
```

```r
seuil=lambda/10

Lasso_poisson=glmnet(XS,Y,family='gaussian',alpha=1, lambda=seuil)

coef=coefficients(Lasso_poisson)[-1]
plot(sort(abs(coef)))
```




```
which(coef!=0)
```

```
## [1]  3  5  7 58
```

```
# Extraire les coefficients du modèle Lasso (s = 0)
lasso_coefficients <- coef(Lasso_poisson, s = 0)
# Obtenir les indices des variables retenues
selected_indices <- which(lasso_coefficients != 0)
# Obtenir les noms des variables retenues
selected_variables <- colnames(lasso_coefficients)[selected_indices]
# Afficher les noms des variables retenues
print(selected_variables)
```

```
## [1] "s1" NA   NA   NA   NA
```

Elasticnet de Poisson

```
lambda=0
for (j in 1:10)
{
  Elastinet_poisson <- cv.glmnet(XS, Y, family = "poisson", alpha=0.5, lambda=seq(0.1, 100, 0.1))
  # Trouver le meilleur lambda (paramètre de régularisation)
  print(Elastinet_poisson$lambda.1se)
  lambda=lambda+Elastinet_poisson$lambda.1se
}
```

```
## [1] 0.9
## [1] 0.7
## [1] 0.6
## [1] 0.8
## [1] 0.7
## [1] 0.7
## [1] 0.8
## [1] 0.7
```

```
## [1] 0.7
## [1] 0.6
```

```
seuil=lambda/10

Lasso_poisson=glmnet(XS,Y,family='gaussian',alpha=0.5, lambda=seuil)
```

Régression de Poisson

```
# Modèle de Regression de poisson
# Exemple de régression de Poisson
# Supposons que votre dataframe s'appelle "df" et que vous voulez sélectionner les colonnes "col1", "co
ports_selectionnees <- X[, c("Port.Port.3","Port.Port.5", "Port.Port.7", "Port.Port.58")]
model_poisson <- glm(Y ~ ports_selectionnees, family = poisson)
```

```
## Warning in dpois(y, mu, log = TRUE): non-integer x = 6.260000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.510000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 4.970000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 6.270000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.200000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 6.810000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.410000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.410000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.650000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 6.330000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 5.750000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 10.550000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.020000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.320000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.500000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 5.010000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.510000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 6.590000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.780000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 5.040000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 5.960000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.060000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 6.710000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.670000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.400000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 9.740000
```

```
## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.230000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 9.520000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.580000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.500000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 9.470000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.120000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.290000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.250000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.970000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.830000
```

```r
summary(model_poisson)
```

```
##
## Call:
## glm(formula = Y ~ ports_selectionnees, family = poisson)
##
## Coefficients:
##                                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)                      1.638705   0.921643   1.778   0.0754 .
## ports_selectionneesPort.Port.3   0.014696   0.039737   0.370   0.7115
## ports_selectionneesPort.Port.5   0.055896   0.036998   1.511   0.1308
## ports_selectionneesPort.Port.7  -0.039960   0.046935  -0.851   0.3946
## ports_selectionneesPort.Port.58  0.008395   0.020704   0.405   0.6851
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 8.3391  on 36  degrees of freedom
## Residual deviance: 2.3366  on 32  degrees of freedom
## AIC: Inf
##
## Number of Fisher Scoring iterations: 4
```

Modélisation de CA

Modèle Ridge

```r
# Modèle de Regression de poisson
# Exemple de régression de Poisson
# Supposons que votre dataframe s'appelle "df" et que vous voulez sélectionner les colonnes "col1", "co
ports_selectionnees <- X[, c("Port.Port.3","Port.Port.5", "Port.Port.7", "Port.Port.58")]
model_poisson <- glm(Y ~ ports_selectionnees, family = poisson)
```

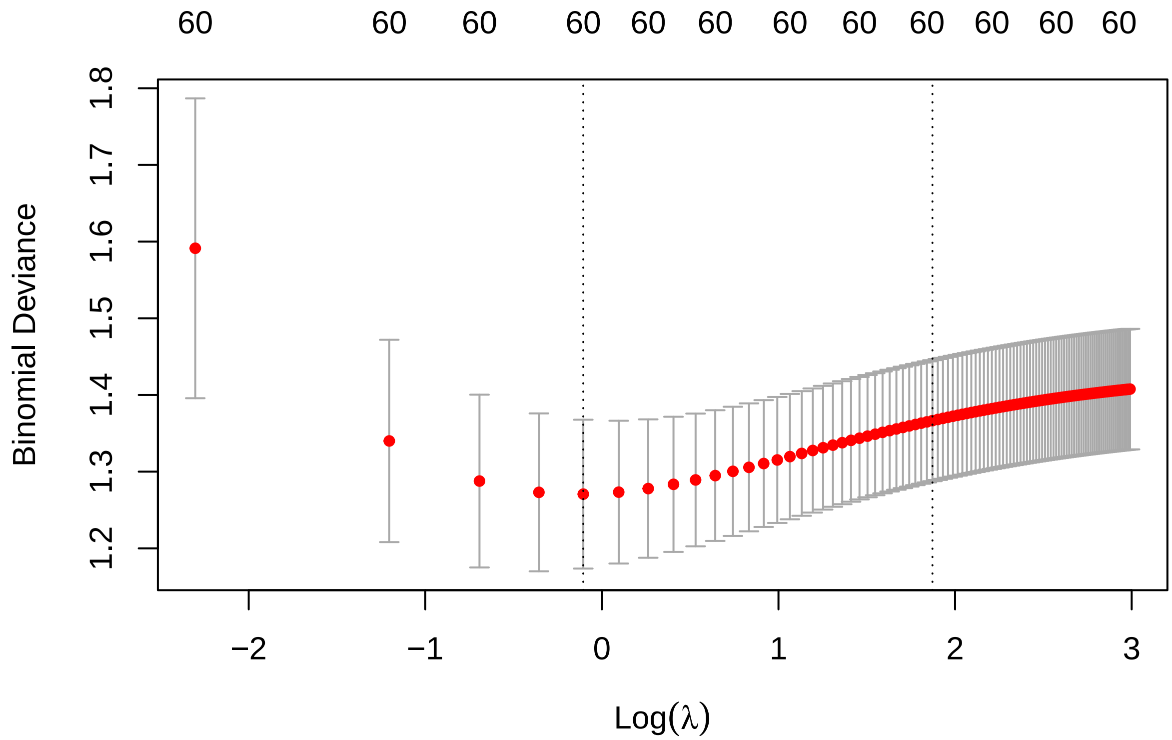
```
## Warning in dpois(y, mu, log = TRUE): non-integer x = 6.260000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.510000

## Warning in dpois(y, mu, log = TRUE): non-integer x = 4.970000
```

```
## Warning in dpois(y, mu, log = TRUE): non-integer x = 6.270000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.200000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 6.810000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.410000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.410000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.650000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 6.330000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 5.750000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 10.550000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.020000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.320000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.500000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 5.010000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.510000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 6.590000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.780000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 5.040000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 5.960000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.060000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 6.710000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.670000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.400000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 9.740000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.230000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 9.520000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.580000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.500000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 9.470000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.120000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.290000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 7.250000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.970000
## Warning in dpois(y, mu, log = TRUE): non-integer x = 8.830000
```

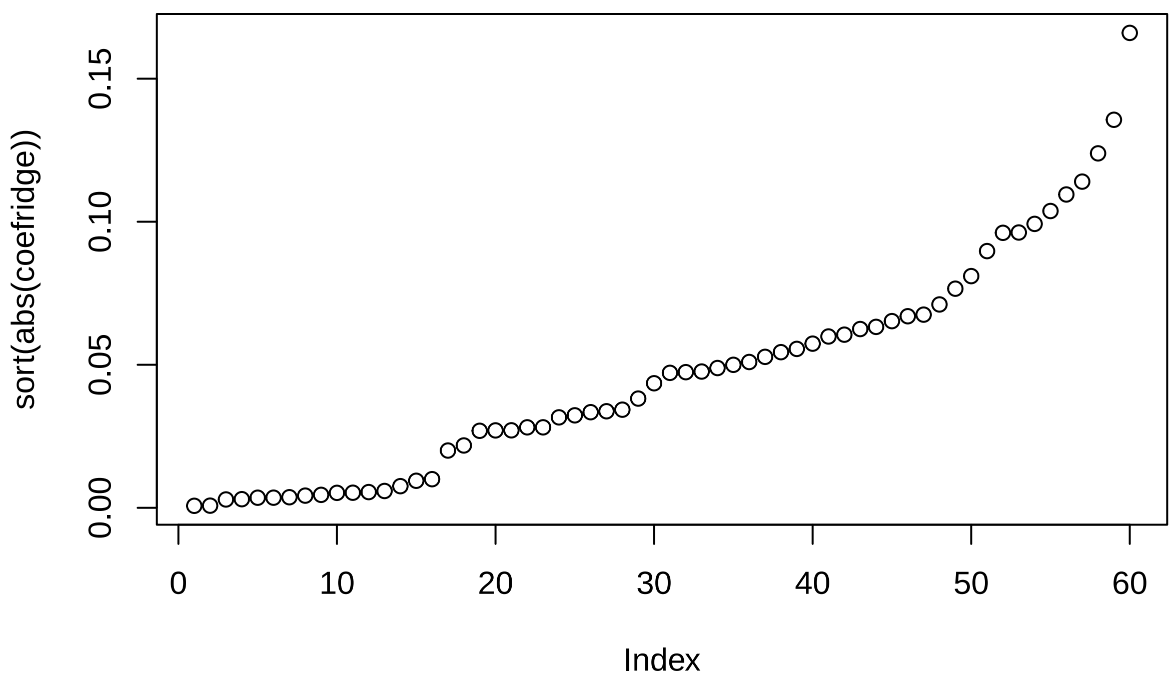
```r
summary(model_poisson)
```

```
##
## Call:
## glm(formula = Y ~ ports_selectionnees, family = poisson)
##
## Coefficients:
##                                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)                       1.638705   0.921643   1.778   0.0754 .
## ports_selectionneesPort.Port.3    0.014696   0.039737   0.370   0.7115
## ports_selectionneesPort.Port.5    0.055896   0.036998   1.511   0.1308
## ports_selectionneesPort.Port.7   -0.039960   0.046935  -0.851   0.3946
## ports_selectionneesPort.Port.58   0.008395   0.020704   0.405   0.6851
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 8.3391  on 36  degrees of freedom
## Residual deviance: 2.3366  on 32  degrees of freedom
## AIC: Inf
##
## Number of Fisher Scoring iterations: 4
```

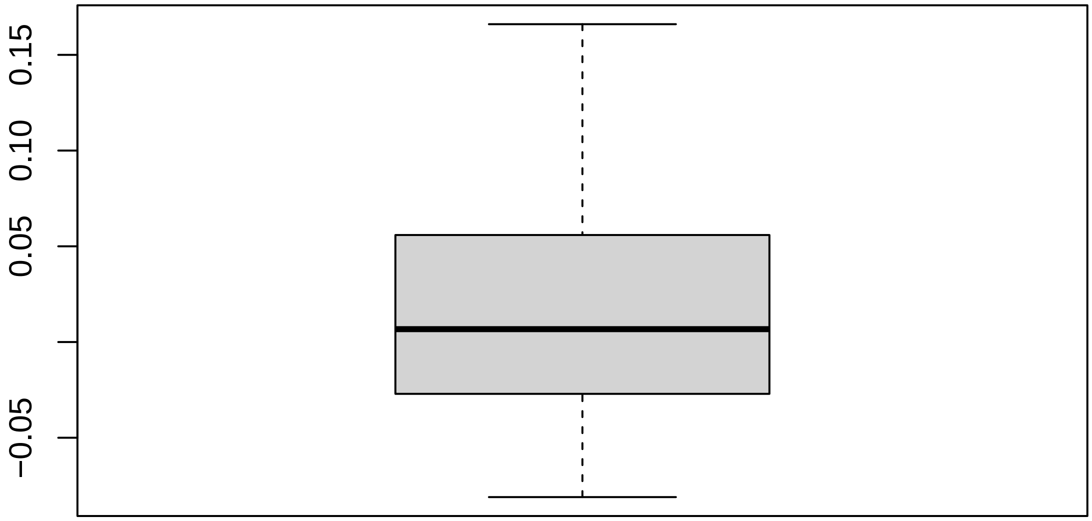
```r
Y=data$CA
X=as.matrix(data[,5:64])
XS=scale(X)# Standardisation de la donées

# Supposons que y est initialement une variable binaire (0 ou 1)
Y <- as.factor(Y)

# Définir "1" comme le niveau par défaut
levels(Y) <- c("0", "1")

# Réalisez une régression logistique Ridge avec validation croisée
cv.ridge <- cv.glmnet(XS, Y, alpha = 0, family = "binomial",lambda=seq(0.1, 20, 0.2))
best_lambda <- cv.ridge$lambda.min  # ou cv.ridge$lambda.1se selon votre choix
plot(cv.ridge)
```




```r
# Entraînez le modèle final avec le meilleur lambda
ridge_model <- glmnet(XS, Y, alpha = 0, lambda = best_lambda, family = "binomial")

# Sélection des variables
coefridge=coefficients(ridge_model)[-1]
plot(sort(abs(coefridge)))
```



```r
boxplot(coefridge)
```




```r
selec3=order(abs(coefridge),decreasing=TRUE)[1:3]
selec3
```

```
## [1]  5 29 58
```

```r
selec12=order(abs(coefridge),decreasing=TRUE)[1:6]
selec12
```

```
## [1]  5 29 58 55 24 40
```

```r
colnames(X[,selec3])
```

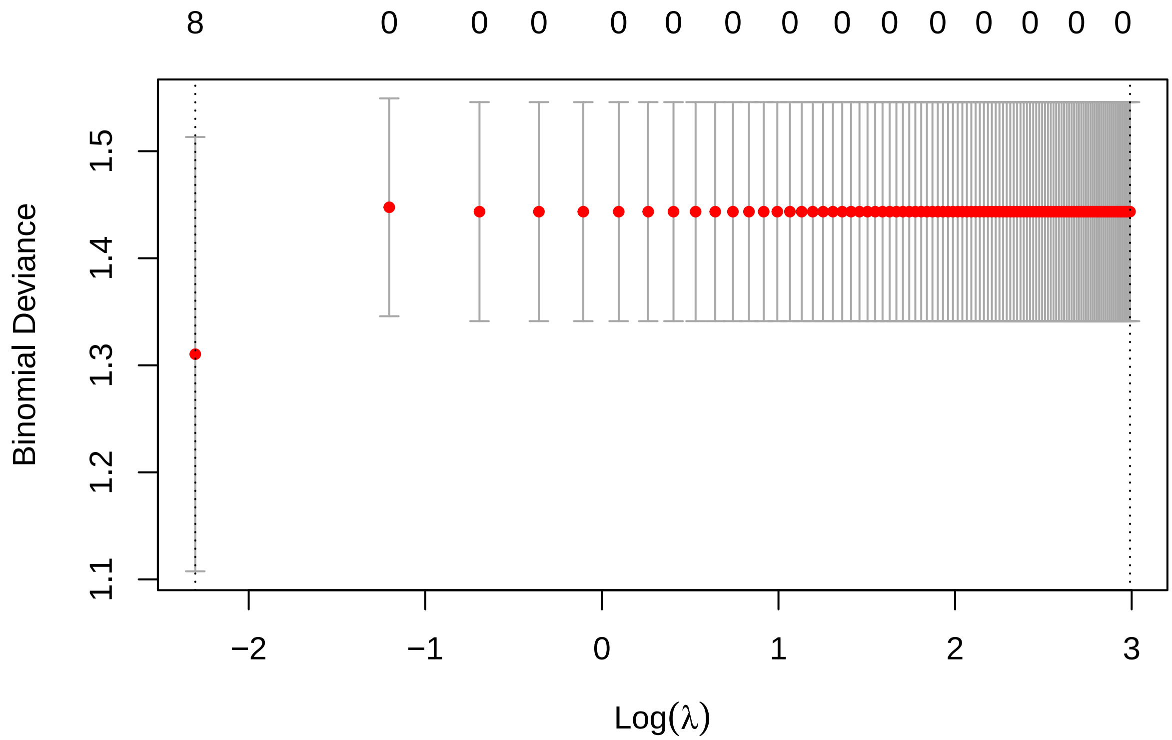
```
## [1] "Port.Port.5"  "Port.Port.29" "Port.Port.58"
```

```r
colnames(X[,selec12])
```

```
## [1] "Port.Port.5"  "Port.Port.29" "Port.Port.58" "Port.Port.55" "Port.Port.24"
## [6] "Port.Port.40"
```

Régression LASSO

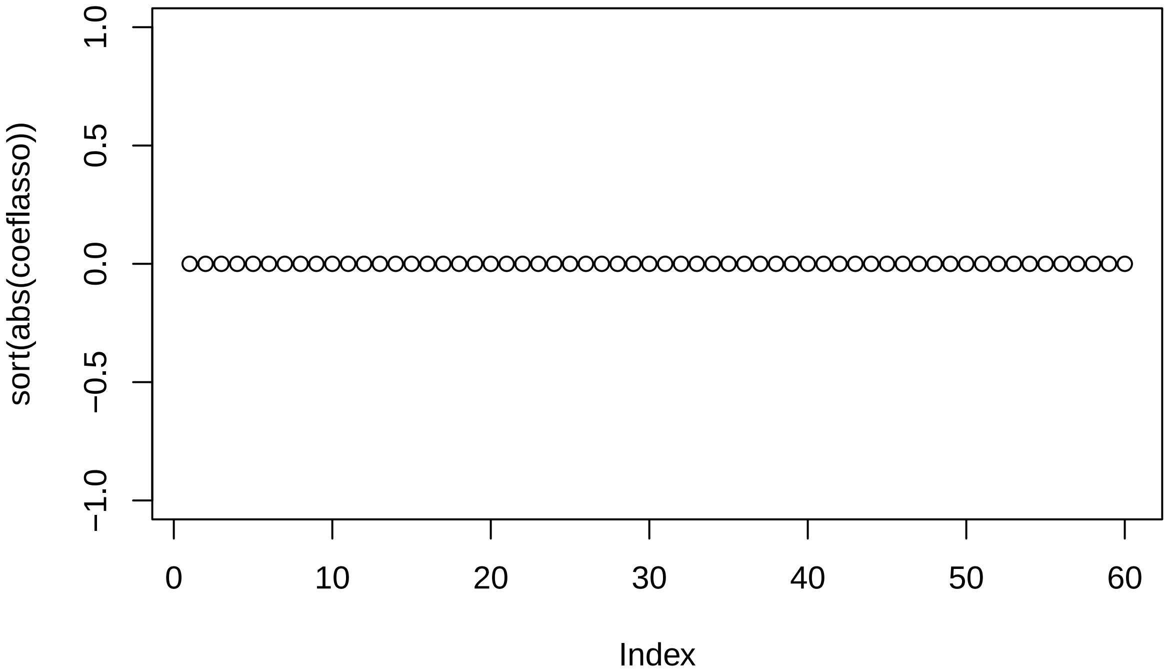
```r
rescv=cv.glmnet(XS,Y,family='binomial',alpha=1,lambda=seq(0.1, 20, 0.2))
plot(rescv)
```




```r
seuil=rescv$lambda.1se # sélection => lambda.1SE
# le lambda.1se varie car il d?pend de la partition utilisée pour la CV
# l'idéal
reslasso=glmnet(X,Y,family='binomial',alpha=1,lambda=seuil)
reslasso
```

```
##
## Call:  glmnet(x = X, y = Y, family = "binomial", alpha = 1, lambda = seuil)
##
##   Df %Dev Lambda
## 1  0    0   19.9
```

```r
coeflasso=coefficients(reslasso)[-1]
plot(sort(abs(coeflasso)))
```




```r
selec6=order(abs(coeflasso),decreasing=TRUE)[1:6]
colnames(X[,selec6])
```

```
## [1] "Port.Port.1" "Port.Port.2" "Port.Port.3" "Port.Port.4" "Port.Port.5"
## [6] "Port.Port.6"
```

#AUC & Interprétation

```r
# Obtenez les probabilités prédites
probas_lasso <- predict(reslasso, newx = as.matrix(XS), s = seuil, type = "response")

# Supposons que vous avez déjà les probabilités prédites dans une variable 'probas'
# et que 'Y' est votre variable de réponse binaire

roc_curve <- roc(Y, probas_lasso)
```

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

## Warning in roc.default(Y, probas_lasso): Deprecated use a matrix as predictor.
## Unexpected results may be produced, please pass a numeric vector.

## Setting direction: controls < cases
```

```r
roc_curve
```

```
##
## Call:
## roc.default(response = Y, predictor = probas_lasso)
##
## Data: probas_lasso in 22 controls (Y 0) < 15 cases (Y 1).
## Area under the curve: 0.5
```

```r
auc_value <- auc(roc_curve)

# Obtenez les coefficients
coefficients_lasso <- coef(reslasso, s = seuil)
```

```r
# Affichez les coefficients
print(coefficients_lasso)
```

```
## 61 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept)  -0.3829923
## Port.Port.1   0.0000000
## Port.Port.2   .
## Port.Port.3   .
## Port.Port.4   .
## Port.Port.5   .
## Port.Port.6   .
## Port.Port.7   .
## Port.Port.8   .
## Port.Port.9   .
## Port.Port.10  .
## Port.Port.11  .
## Port.Port.12  .
## Port.Port.13  .
## Port.Port.14  .
## Port.Port.15  .
## Port.Port.16  .
## Port.Port.17  .
## Port.Port.18  .
## Port.Port.19  .
## Port.Port.20  .
## Port.Port.21  .
## Port.Port.22  .
## Port.Port.23  .
## Port.Port.24  .
## Port.Port.25  .
## Port.Port.26  .
## Port.Port.27  .
## Port.Port.28  .
## Port.Port.29  .
## Port.Port.30  .
## Port.Port.31  .
## Port.Port.32  .
## Port.Port.33  .
## Port.Port.34  .
## Port.Port.35  .
## Port.Port.36  .
## Port.Port.37  .
## Port.Port.38  .
## Port.Port.39  .
## Port.Port.40  .
## Port.Port.41  .
## Port.Port.42  .
## Port.Port.43  .
## Port.Port.44  .
## Port.Port.45  .
## Port.Port.46  .
## Port.Port.47  .
## Port.Port.48  .
```

```
## Port.Port.49  .
## Port.Port.50  .
## Port.Port.51  .
## Port.Port.52  .
## Port.Port.53  .
## Port.Port.54  .
## Port.Port.55  .
## Port.Port.56  .
## Port.Port.57  .
## Port.Port.58  .
## Port.Port.59  .
## Port.Port.60  .
```

```
# Supposons que vous avez déjà les coefficients dans une variable 'coefficients'
variable_significative <- names(coeflasso)[which.max(abs(coeflasso))]
cat("La variable la plus significative est :", variable_significative)
```

```
## La variable la plus significative est :
```

Modélisation de CA3

```
# Supposons que X soit votre matrice de variables explicatives et Y votre variable réponse CA3

# Convertir la variable réponse en facteur
Y=data$CA3
X=as.matrix(data[,5:64])
XS=scale(X)# Standardisation de la donées
Y <- as.factor(Y)

# Appliquer la régression multinomiale pénalisée avec la validation croisée
cv_fit_multinom <- cv.glmnet(x = XS, y = Y, family = "multinomial", alpha = 1, lambda=seq(0.1, 20, 0.2))
```

```
## Warning in lognet(xd, is.sparse, ix, jx, y, weights, offset, alpha, nobs, : one
## multinomial or binomial class has fewer than 8 observations; dangerous ground
```

```
# Obtenir le meilleur lambda à partir de la validation croisée
best_lambda_multinom <- cv_fit_multinom$lambda.min

# Appliquer la régression multinomiale pénalisée avec le meilleur lambda
fit_multinom_best_lambda <- glmnet(x = XS, y = Y, family = "multinomial", alpha = 1, lambda = best_lamb
fit_multinom_best_lambda
```

```
##
## Call:  glmnet(x = XS, y = Y, family = "multinomial", alpha = 1, lambda = best_lambda_multinom)
##
##   Df  %Dev Lambda
## 1 13 37.72    0.1
```

```
# Sélectionner les variables
selected_vars <- coef(fit_multinom_best_lambda)

# Afficher les variables sélectionnées
print(selected_vars)
```

```
## $`0`
## 61 x 1 sparse Matrix of class "dgCMatrix"
```

```
##                        s0
##               -0.26488018
## Port.Port.1   .
## Port.Port.2  -0.26869940
## Port.Port.3   .
## Port.Port.4   .
## Port.Port.5   .
## Port.Port.6   .
## Port.Port.7   .
## Port.Port.8   .
## Port.Port.9   .
## Port.Port.10  .
## Port.Port.11  .
## Port.Port.12  .
## Port.Port.13  .
## Port.Port.14  .
## Port.Port.15  .
## Port.Port.16  .
## Port.Port.17  .
## Port.Port.18  .
## Port.Port.19  .
## Port.Port.20  .
## Port.Port.21  .
## Port.Port.22  .
## Port.Port.23  .
## Port.Port.24  .
## Port.Port.25  .
## Port.Port.26  .
## Port.Port.27 -0.35282103
## Port.Port.28  .
## Port.Port.29 -0.01203646
## Port.Port.30  .
## Port.Port.31  .
## Port.Port.32  .
## Port.Port.33  0.10941111
## Port.Port.34  .
## Port.Port.35  .
## Port.Port.36  .
## Port.Port.37  .
## Port.Port.38  .
## Port.Port.39  .
## Port.Port.40  .
## Port.Port.41  .
## Port.Port.42  .
## Port.Port.43  .
## Port.Port.44  .
## Port.Port.45  .
## Port.Port.46 -0.34437403
## Port.Port.47  .
## Port.Port.48  .
## Port.Port.49  .
## Port.Port.50  .
## Port.Port.51  .
## Port.Port.52  .
```

```
## Port.Port.53  .
## Port.Port.54  .
## Port.Port.55  .
## Port.Port.56  .
## Port.Port.57  .
## Port.Port.58 -0.03163133
## Port.Port.59  .
## Port.Port.60  .
##
## $`1`
## 61 x 1 sparse Matrix of class "dgCMatrix"
##                         s0
##               0.153760453
## Port.Port.1   .
## Port.Port.2   0.143987133
## Port.Port.3   .
## Port.Port.4   .
## Port.Port.5   .
## Port.Port.6   .
## Port.Port.7   .
## Port.Port.8   .
## Port.Port.9   .
## Port.Port.10  .
## Port.Port.11  .
## Port.Port.12  .
## Port.Port.13  .
## Port.Port.14  .
## Port.Port.15  .
## Port.Port.16  .
## Port.Port.17  .
## Port.Port.18  .
## Port.Port.19  .
## Port.Port.20  .
## Port.Port.21  .
## Port.Port.22  .
## Port.Port.23  .
## Port.Port.24  .
## Port.Port.25  .
## Port.Port.26  .
## Port.Port.27  .
## Port.Port.28  .
## Port.Port.29  .
## Port.Port.30  .
## Port.Port.31  .
## Port.Port.32 -0.056285411
## Port.Port.33  .
## Port.Port.34  .
## Port.Port.35  .
## Port.Port.36  .
## Port.Port.37  .
## Port.Port.38  .
## Port.Port.39  .
## Port.Port.40  .
## Port.Port.41  .
```

```
## Port.Port.42  0.007496989
## Port.Port.43  .
## Port.Port.44  .
## Port.Port.45  .
## Port.Port.46  .
## Port.Port.47  .
## Port.Port.48  .
## Port.Port.49  .
## Port.Port.50  .
## Port.Port.51  .
## Port.Port.52  .
## Port.Port.53  .
## Port.Port.54  .
## Port.Port.55  .
## Port.Port.56  .
## Port.Port.57  .
## Port.Port.58  .
## Port.Port.59  .
## Port.Port.60  .
##
## $`2`
## 61 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
##              0.111119730
## Port.Port.1  .
## Port.Port.2  .
## Port.Port.3  .
## Port.Port.4  .
## Port.Port.5  0.774572842
## Port.Port.6  .
## Port.Port.7  .
## Port.Port.8  .
## Port.Port.9  .
## Port.Port.10 .
## Port.Port.11 .
## Port.Port.12 .
## Port.Port.13 .
## Port.Port.14 .
## Port.Port.15 .
## Port.Port.16 .
## Port.Port.17 .
## Port.Port.18 .
## Port.Port.19 .
## Port.Port.20 .
## Port.Port.21 .
## Port.Port.22 .
## Port.Port.23 .
## Port.Port.24 0.144681561
## Port.Port.25 .
## Port.Port.26 .
## Port.Port.27 .
## Port.Port.28 .
## Port.Port.29 0.275670330
## Port.Port.30 .
```

```
## Port.Port.31 .
## Port.Port.32 .
## Port.Port.33 .
## Port.Port.34 .
## Port.Port.35 .
## Port.Port.36 .
## Port.Port.37 .
## Port.Port.38 .
## Port.Port.39 .
## Port.Port.40 0.217898160
## Port.Port.41 .
## Port.Port.42 .
## Port.Port.43 0.006916185
## Port.Port.44 .
## Port.Port.45 .
## Port.Port.46 .
## Port.Port.47 .
## Port.Port.48 .
## Port.Port.49 .
## Port.Port.50 .
## Port.Port.51 .
## Port.Port.52 .
## Port.Port.53 .
## Port.Port.54 0.031622876
## Port.Port.55 .
## Port.Port.56 .
## Port.Port.57 .
## Port.Port.58 .
## Port.Port.59 .
## Port.Port.60 .
```

Régression polytomique ordonnée

```
# Supposons que Y_ordered est votre variable réponse ordonnée et X_Select est votre matrice d'variables
# Convertir Y_ordered en une variable ordonnée si ce n'est pas déjà fait
Y_ordered <- as.ordered(Y)

# Créer une matrice X_Select (remplacez cela par votre propre matrice)
X_Select <- data[, c("Port.Port.5", "Port.Port.24", "Port.Port.29",
                     "Port.Port.40","Port.Port.43","Port.Port.54")]

# Appliquer la régression polytomique ordonnée
fit_ord <- clm(Y_ordered ~ ., data = cbind(X_Select, Y_ordered))

# Afficher les résultats
summary(fit_ord)
```

```
## formula:
## Y_ordered ~ Port.Port.5 + Port.Port.24 + Port.Port.29 + Port.Port.40 + Port.Port.43 + Port.Port.54
## data:    cbind(X_Select, Y_ordered)
##
##  link  threshold nobs logLik AIC   niter max.grad cond.H
##  logit flexible  37   -16.26 48.52 7(0)  4.86e-10 1.1e+06
```

```
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## Port.Port.5    1.2583     0.4529   2.778  0.00547 **
## Port.Port.24   0.6393     0.3278   1.951  0.05110 .
## Port.Port.29   0.6591     0.2594   2.541  0.01106 *
## Port.Port.40   0.6848     0.2503   2.736  0.00623 **
## Port.Port.43   0.3929     0.2070   1.898  0.05774 .
## Port.Port.54   0.9872     0.4832   2.043  0.04103 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Threshold coefficients:
##     Estimate Std. Error z value
## 0|1    39.76      13.63   2.918
## 1|2    44.70      14.78   3.024
```

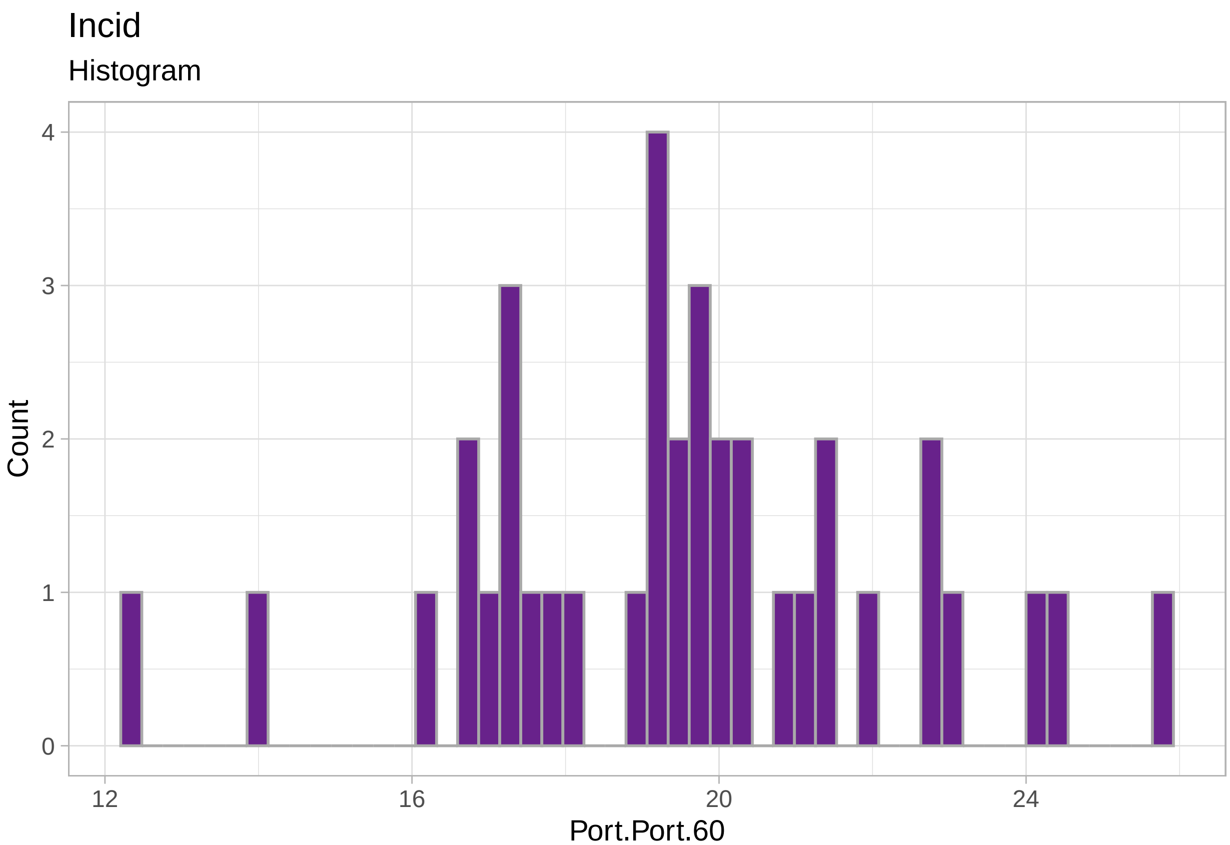
```r
stargazer(fit_ord, title = "Régression polytomique ordonnée", out = "table.tex")
```

```
##
## % Table created by stargazer v.5.2.3 by Marek Hlavac, Social Policy Institute. E-mail: marek.hlavac a
## % Date and time: Thu, Dec 14, 2023 - 18:43:59
## \begin{table}[!htbp] \centering
##   \caption{Régression polytomique ordonnée}
##   \label{}
## \begin{tabular}{@{\extracolsep{5pt}}lc}
## \\[-1.8ex]\hline
## \hline \\[-1.8ex]
##  & \multicolumn{1}{c}{\textit{Dependent variable:}} \\
## \cline{2-2}
## \\[-1.8ex] & Y\_ordered \\
## \hline \\[-1.8ex]
##  Port.Port.5 & 1.258$^{***}$ \\
##   & (0.453) \\
##   & \\
##  Port.Port.24 & 0.639$^{*}$ \\
##   & (0.328) \\
##   & \\
##  Port.Port.29 & 0.659$^{**}$ \\
##   & (0.259) \\
##   & \\
##  Port.Port.40 & 0.685$^{***}$ \\
##   & (0.250) \\
##   & \\
##  Port.Port.43 & 0.393$^{*}$ \\
##   & (0.207) \\
##   & \\
##  Port.Port.54 & 0.987$^{**}$ \\
##   & (0.483) \\
##   & \\
## \hline \\[-1.8ex]
## Observations & 37 \\
## Log Likelihood & $-$16.258 \\
## \hline
## \hline \\[-1.8ex]
```

```
## \textit{Note:}  & \multicolumn{1}{r}{$^{*}$p$<$0.1; $^{**}$p$<$0.05; $^{***}$p$<$0.01} \\
## \end{tabular}
## \end{table}
```

Histogramme

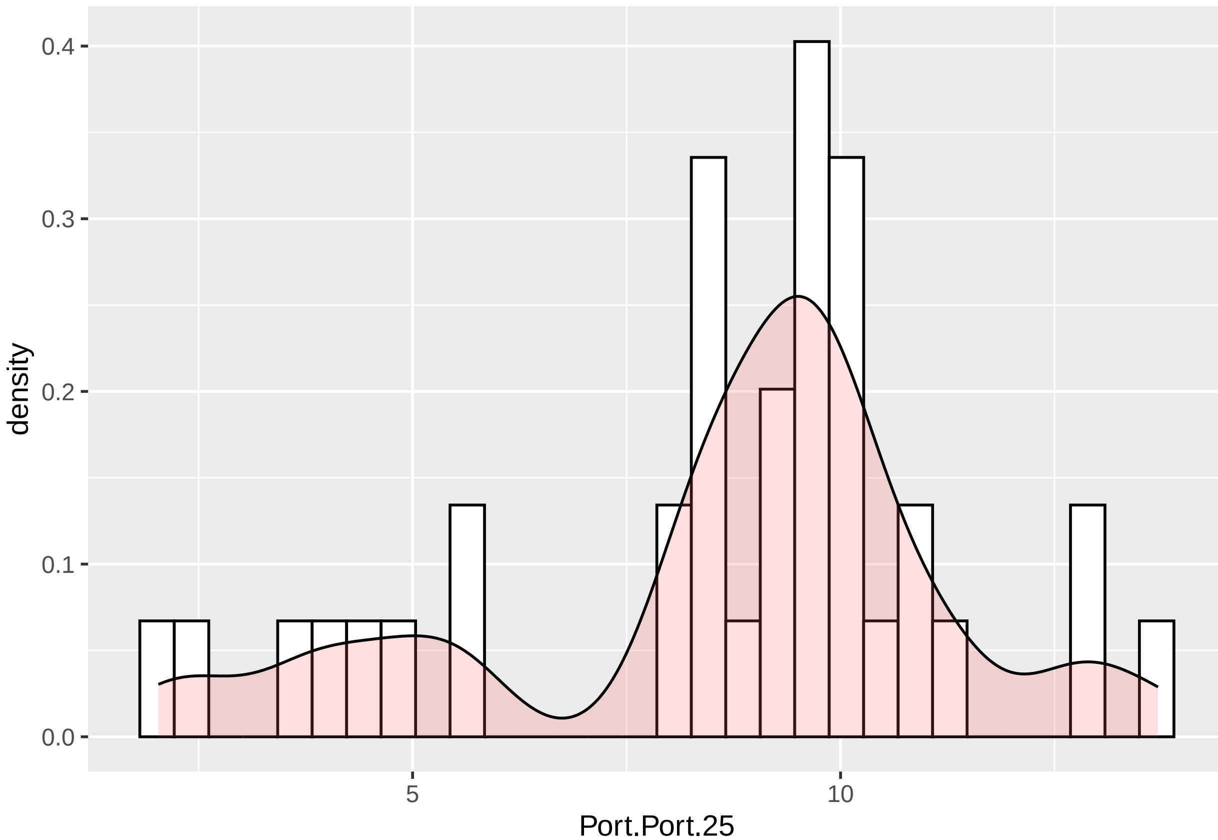
```
histo_mass <- ggplot(data)+
  geom_histogram(aes(x=Port.Port.60), fill="darkorchid4", color="darkgray", bins=50)+
  labs(title = "Incid", subtitle = "Histogram")+
  ylab("Count")+theme_light()
histo_mass
```



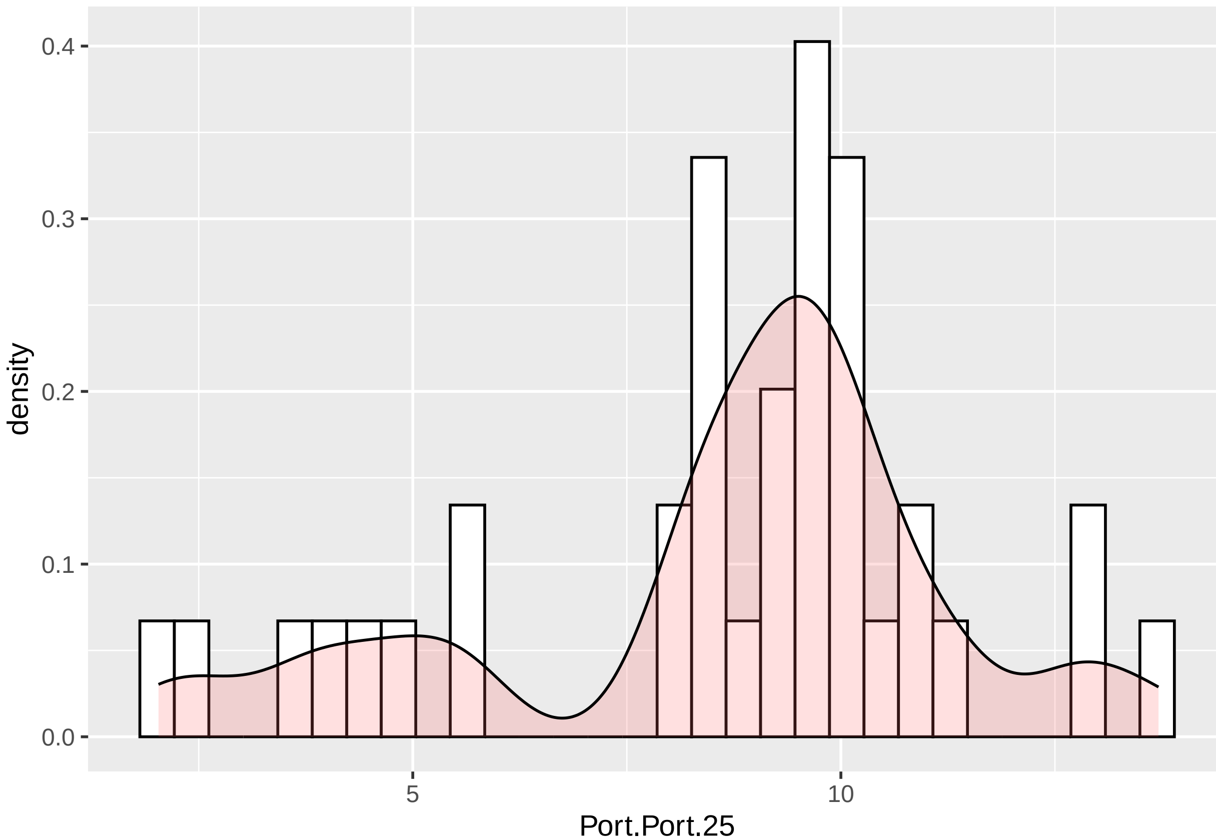

```
a <- ggplot(data, aes(x = Port.Port.25))
a
```


```r
# Histogram with density plot
a + geom_histogram(aes(y = ..density..),
                   colour="black", fill="white") +
  geom_density(alpha = 0.2, fill = "#FF6666")
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
a +geom_histogram(aes(y = after_stat(density)), bins = 30, colour = "black", fill = "white") +
  geom_density(alpha = 0.2, fill = "#FF6666")
```


0.4
0.3
0.2
0.1
0.0
density
5
10
Port.Port.25