# Devoir 2

Raphael Lacoste

## Question No 1 a)

```r
rm(list=ls())
set.seed(1835188)
library(ISLR)
data = ISLR::College
echant = sample(1:nrow(data), 400)
  data_test = data[-echant, ]
  data_train = data[echant, ]
```

# Question No 1 b) 1.

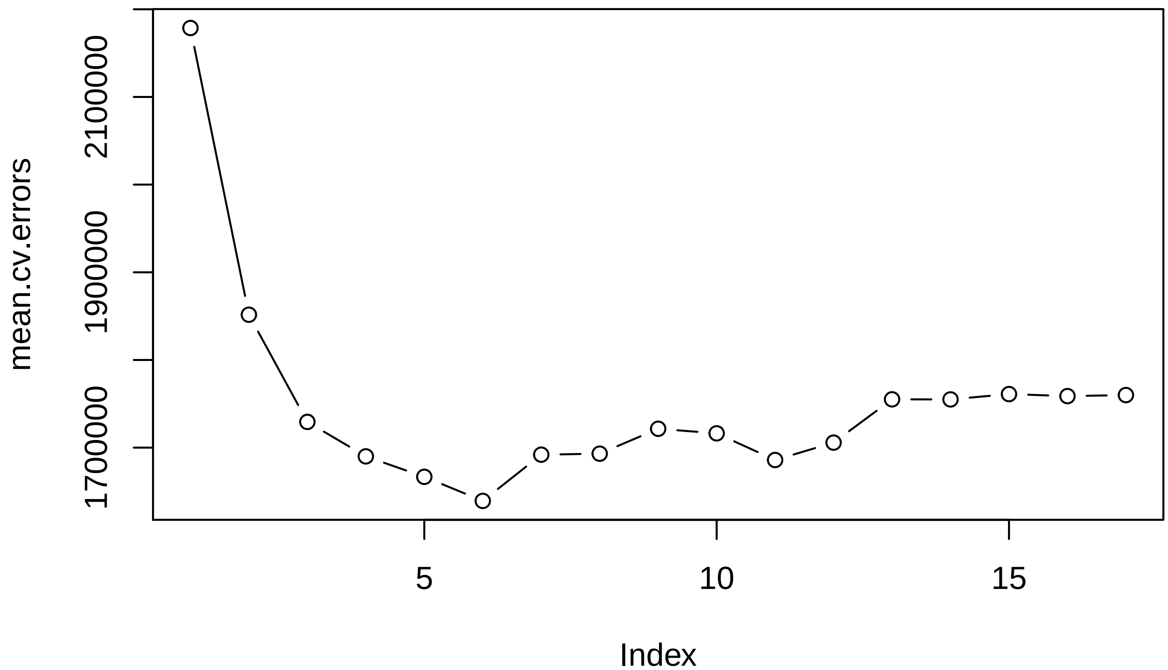
```
library(leaps)
regfit.best = regsubsets(Apps~., data = data_train, nvmax = 18)
regfit.summary = summary(regfit.best)
regfit.summary$adjr2
```

```
##  [1] 0.8716190 0.9050408 0.9104749 0.9152205 0.9196477 0.9216612 0.9225823
##  [8] 0.9231221 0.9237263 0.9243611 0.9250622 0.9253453 0.9253497 0.9252813
## [15] 0.9251673 0.9250132 0.9248278
```

```
predict.regsubsets = function(object, newdata, id,...){
  form <- as.formula(object$call[[2]])
  mat <- model.matrix(form, newdata)
  coefi <- coef(object, id=id)
  xvars <- names(coefi)
  mat[,xvars]%*%coefi
}

k = 10
folds = sample(1:k, nrow(data_train), replace = TRUE)
cv.errors = matrix(NA,k,17, dimnames = list(NULL, paste(1:17)))
  for(j in 1:k){
    best.fit <- regsubsets(Apps~., data_train[folds!=j,], nvmax=17)
    for(i in 1:17) {
      pred <- predict(best.fit, data_train[folds == j,], id = i)
      cv.errors[j,i] <- mean((data_train$Apps[folds == j ]-pred)^2)
    }
  }
```

## Question No 1 b) 2.

```
mean.cv.errors <- apply(cv.errors, 2, mean)
plot(mean.cv.errors, type = "b")
```




```
which.min(mean.cv.errors)
```

```
## 6
## 6
```

```
reg.best = regsubsets(Apps~., data = data_train, nvmax = 4)
coef(reg.best, 4)
```

```
##   (Intercept)        Accept     Top10perc      Outstate        Expend
## -964.77209860    1.47320317   28.99629282   -0.09905163    0.12777535
```

On constate sur le graphique un MSE intéressant lorsque le modèle possède 4 variables. Après celà, le MSE diminue encore mais la contribution est beaucoup moins importante. 4 Variables semble un bon choix pour avoir un modèle relativement simple et précis.

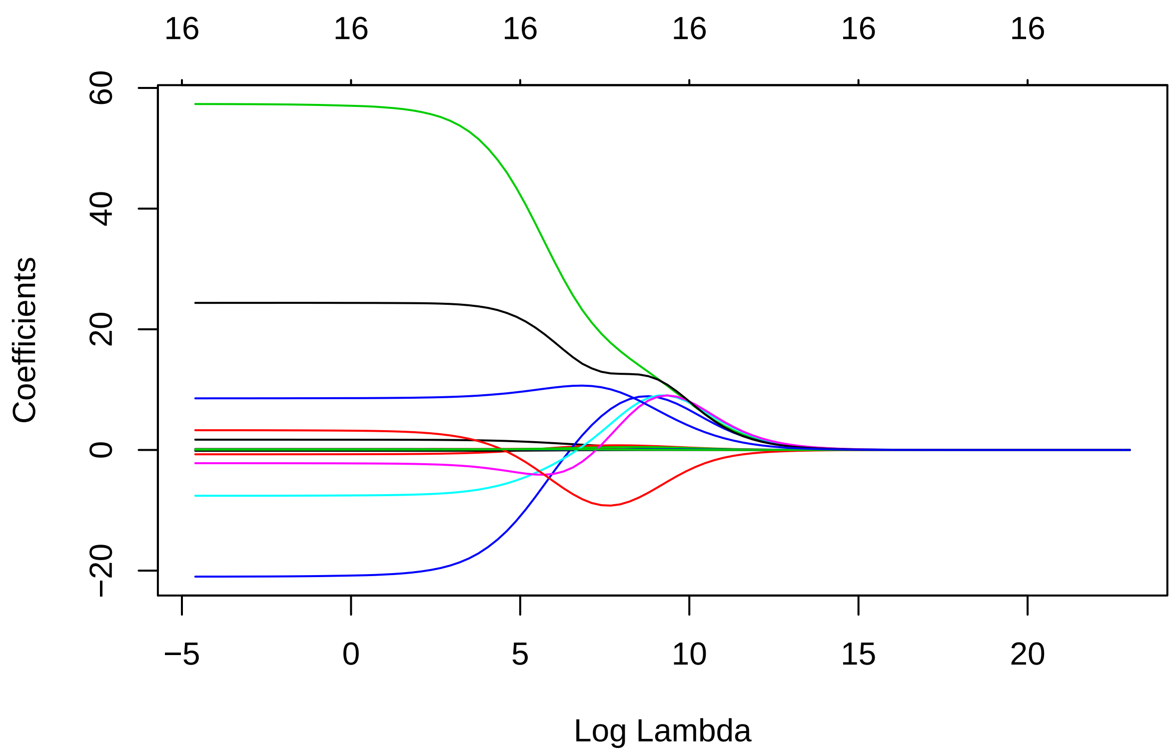
## Question No 1 c) 1.

```
library(glmnet)
```

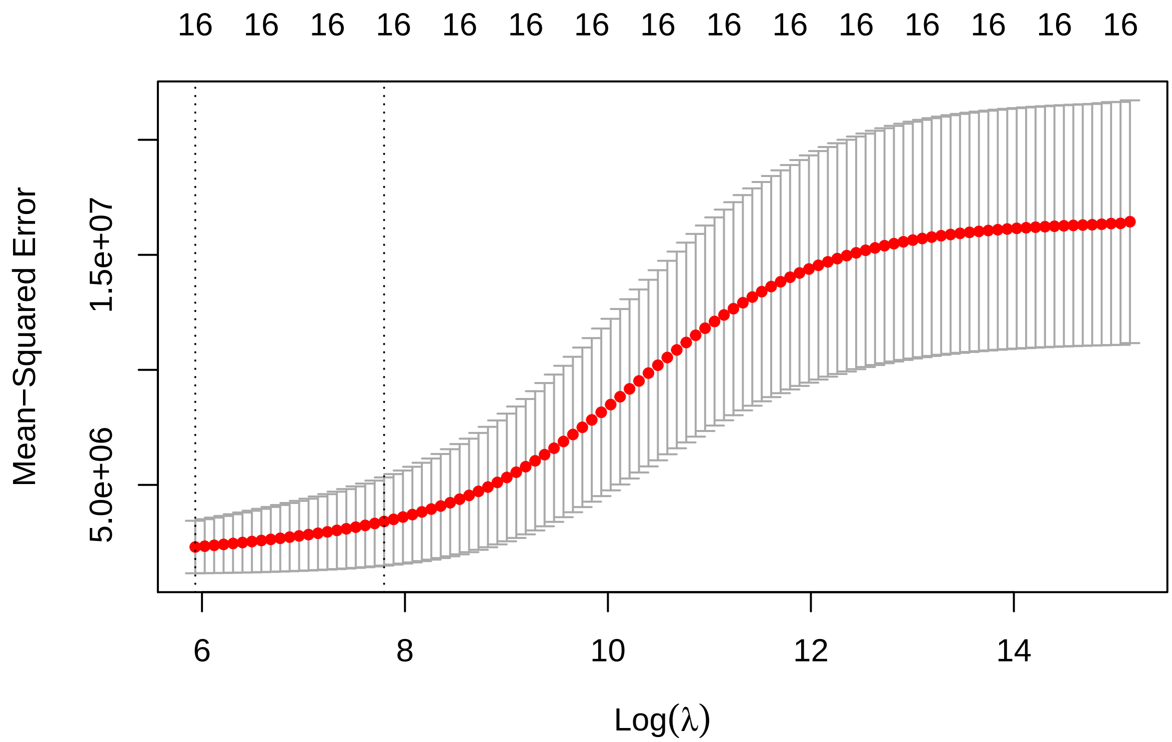
```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.0-2
```

```
Q1c1_x <- model.matrix(Apps~.,data = data_train)[,-2]
Q1c1_y <- data_train$Apps
grid <-10^seq(10,-2,length=100)
ridge.mod = glmnet(Q1c1_x, Q1c1_y, alpha = 0, lambda = grid)

plot(ridge.mod, xvar = "lambda")
```




```
cv.out <- cv.glmnet(Q1c1_x, Q1c1_y, alpha = 0)
plot(cv.out)
```




```
bestlam1 <- cv.out$lambda.min
bestlam1
```

```
## [1] 377.5471
```

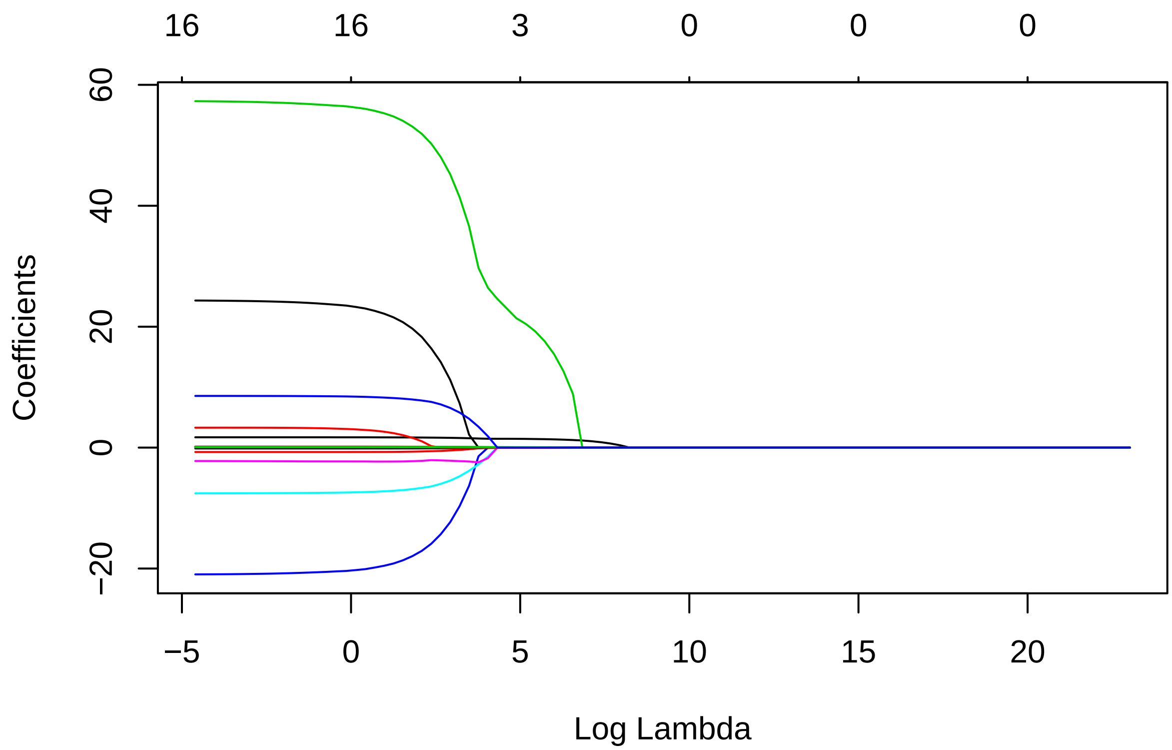
## Question No 1 c) 2.

```r
Q1c1_xtest <- model.matrix(Apps~.,data = data_test)[,-2]
Q1c1_ytest <- data_test$Apps
ridge.pred = predict(ridge.mod, s=bestlam1, newx = Q1c1_xtest)

mean((ridge.pred-Q1c1_ytest)^2)
```

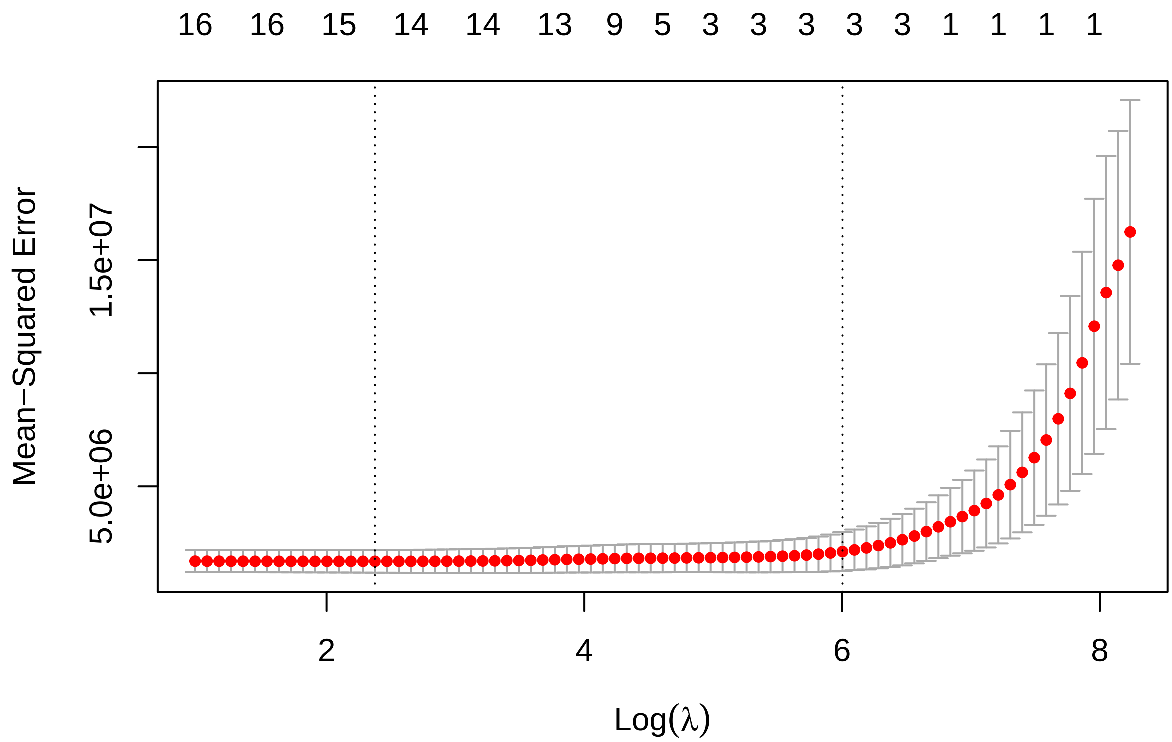
```
## [1] 968601.9
```

## Question No 1 d) 1.

```
grid <-10^seq(10,-2,length=100)
lasso.mod = glmnet(Q1c1_x, Q1c1_y, alpha = 1, lambda = grid)

plot(lasso.mod, xvar = "lambda")
```




```
cv.out <- cv.glmnet(Q1c1_x, Q1c1_y, alpha = 1)
plot(cv.out)
```




```r
bestlam2 <- cv.out$lambda.min
bestlam2
```

```
## [1] 10.75268
```

## Question No 1 d) 2.

```
lasso.pred = predict(lasso.mod, s=bestlam2, newx = Q1c1_xtest)
mean((lasso.pred-Q1c1_ytest)^2)
```

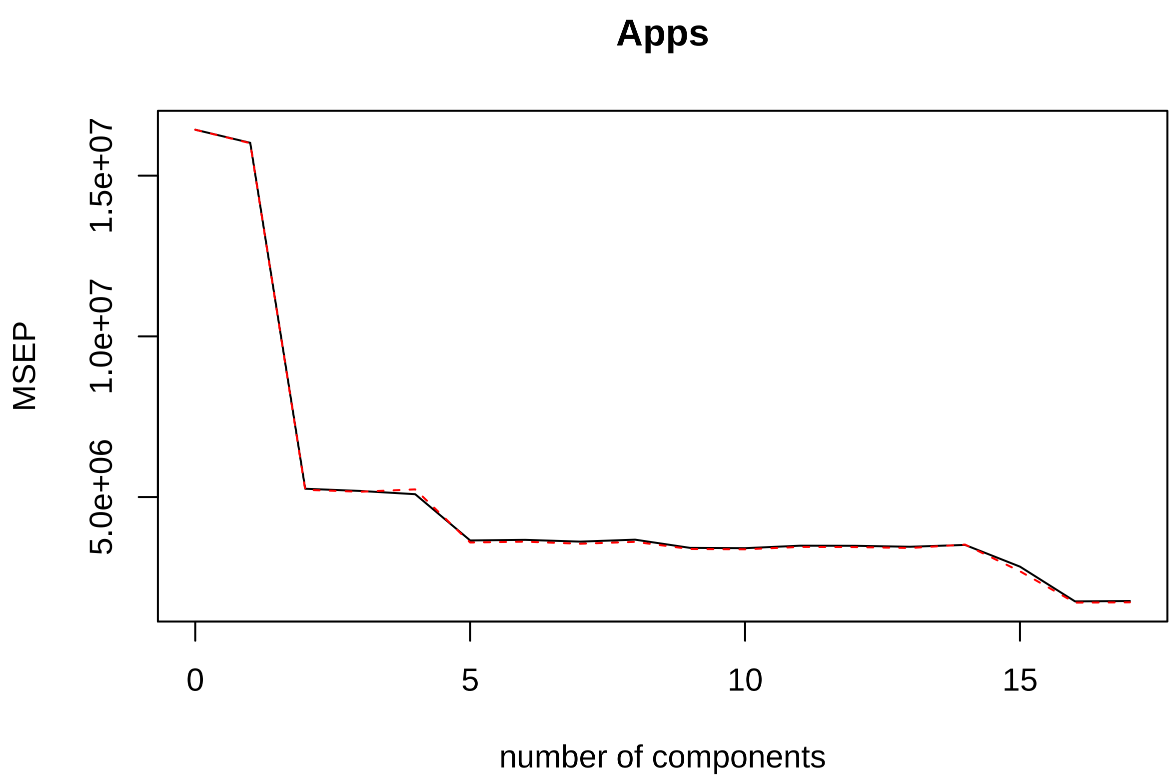
```
## [1] 1041081
```

## Question No 1 e) 1.

```
library(pls)
```

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

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

```
pcr.fit <- pcr(Apps~., data = data_train, scale = TRUE, validation = "CV")
validationplot(pcr.fit,val.type = "MSEP")
```




On peut voir que le MSEP est au plus bas à 16 composantes. Cependant, le MSE ne change pas vraiment entre 5 et 16 variables, il est donc pertinent de choisir 5 variables.

## Question No 1 e) 2.

```
pcr.pred <-predict(pcr.fit, Q1c1_xtest, ncomp=5)
mean((pcr.pred-Q1c1_ytest)^2)
```

```
## [1] 1656059
```

```
pcr.fit <- pcr(Apps~., data= data, scale = F, ncomp = 5)
summary(pcr.fit)
```

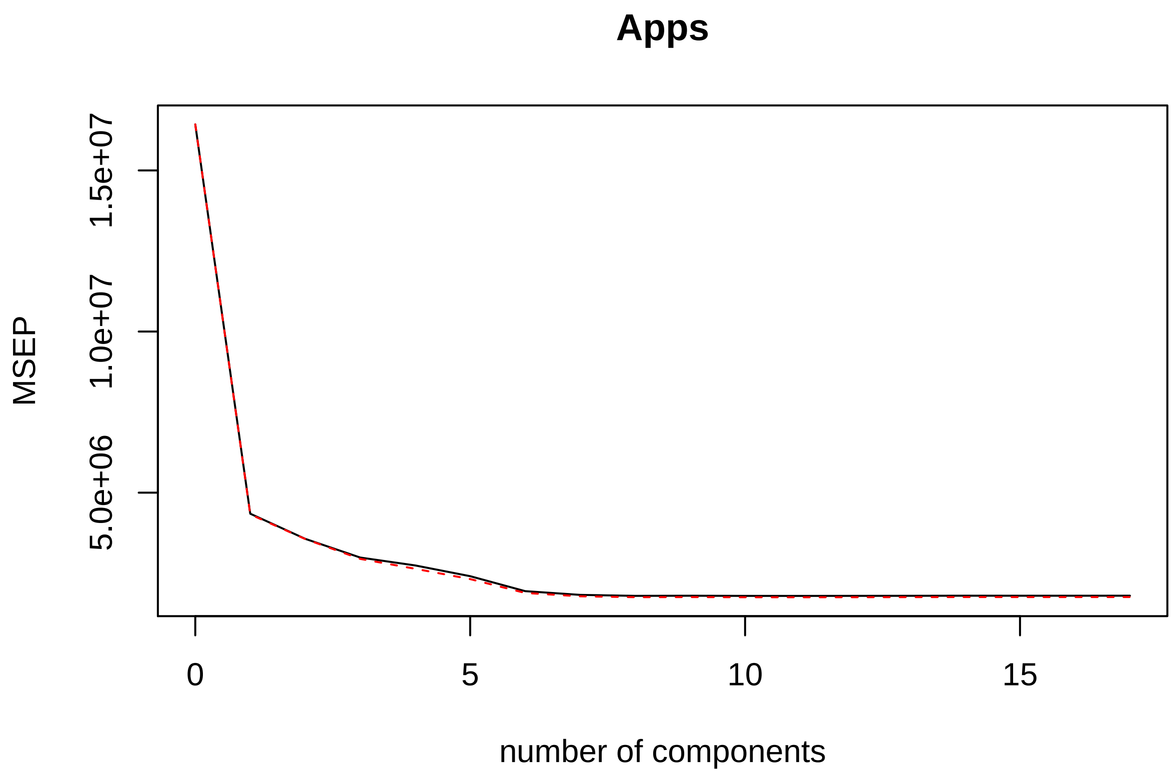
```
## Data:    X dimension: 777 17
##  Y dimension: 777 1
## Fit method: svdpc
## Number of components considered: 5
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps
## X     48.6069    87.55    95.36    97.39    98.65
## Apps   0.1659    78.04    79.05    81.51    90.98
```

# Question No 1 f) 1.

```r
library(pls)
pls.fit <- plsr(Apps~., data = data_train, scale = TRUE, validation = "CV")
summary(pls.fit)
```

```
## Data:    X dimension: 400 17
##  Y dimension: 400 1
## Fit method: kernelpls
## Number of components considered: 17
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            4053     2085     1889     1728     1656     1552     1394
## adjCV         4053     2080     1887     1716     1624     1522     1374
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        1352     1341     1342      1340      1340      1340      1341
## adjCV     1335     1325     1327      1325      1325      1325      1326
##        14 comps  15 comps  16 comps  17 comps
## CV         1342      1342      1342      1342
## adjCV      1327      1327      1327      1327
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       25.46    44.36    62.15    65.16    69.98    73.56    76.74    80.19
## Apps    75.24    81.82    86.04    90.40    92.13    92.65    92.73    92.76
##       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X       83.03     85.33     87.52     90.25     92.61     94.15     96.94
## Apps    92.78     92.79     92.80     92.80     92.80     92.80     92.80
##       16 comps  17 comps
## X        98.46     100.0
## Apps     92.80      92.8
```

```r
validationplot(pls.fit, val.type = "MSEP")
```




Le nombre de composantes est 3 lorsque le MSEP est relativement au plus bas et ne change presque plus en y ajoutant d'autres variables.

# Question No 1 f) 2.

```
pls.pred <-predict(pls.fit, Q1c1_xtest, ncomp=3)
mean((pls.pred-Q1c1_ytest)^2)
```

```
## [1] 1376568
```

```
pls.fit <- plsr(Apps~., data = data, scale = TRUE, ncomp = 3)
summary(pls.fit)
```

```
## Data:    X dimension: 777 17
##  Y dimension: 777 1
## Fit method: kernelpls
## Number of components considered: 3
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps
## X       25.76    40.33    62.59
## Apps    78.01    85.14    87.67
```

## Question No 1 g)

| Modèle | MSE | Paramètre optimal | Classement |
|---|---|---|---|
| Meilleurs sous ensembles | $1.6900391 \times 10^6$ | $\ell = 4$ | 3 |
| Approche ridge | $9.6860189 \times 10^5$ | $\lambda$ =377.55 | 5 |
| Approche Lasso | $1.0410806 \times 10^6$ | $\lambda$ =10.75 | 2 |
| Approche PCR | $1.6560585 \times 10^6$ | $M = 5$ | 4 |
| Approche PLS | $1.3765675 \times 10^6$ | $M = 3$ | 1 |

Les MSE des méthodes sont relativement identiques, sauf que la méthode Ridge est la plus basse. Elle est donc la meilleure en terme de MSE. Au niveau des paramètres optimaux, on constate que les paramètres sont entre 3, 4 et 5 pour les méthodes de pls, subsets et pcr. Idéalement le moins de paramètres pour un même MSE est un meilleur choix. Pour ce qui est des approches de Ridge et Lasso, on peut voir que Lasso possède un paramètre lambda beaucoup plus petit que la méthode Ridge, il est donc meilleur. Pour ce qui est des prédictions, le modèle Ridge est un bon prédicteur étant donné son faible MSE, cependant, la complexité du modèle laisse supposer que le modèle est sur-entraîné. Donc un bon compromis serait l'approche PLS, dont le MSE est relativement bas et la complexité du modèle est limitée à 3 paramètres.

## Question No 7 p.262 a)

$$
\begin{aligned}
L(\theta \mid \beta) &= p(\beta \mid \theta) \\
&= p(\beta_1 \mid \theta) \times \cdots \times p(\beta_n \mid \theta) \\
&= \prod_{i=1}^{n} p(\beta_i \mid \theta) \\
&= \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{Y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j X_{ij})}{2\sigma^2}\right) \\
&= \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} \left[Y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j X_{ij})\right]^2\right)
\end{aligned}
$$

## Question No 7 p.262 b)

On a comme fonction à priori

$$p(\beta) = \frac{1}{2b}\exp(-|\beta|/b)$$

La fonction à postériori est donnée par

$$f(\beta \mid X, Y) \propto f(Y \mid X, \beta)p(\beta \mid X) = f(Y \mid X, \beta)p(\beta)$$

Donc en substituant les valeurs de a) dans la fonction à postériori, on obtient

$$\begin{aligned}
f(Y \mid X, \beta)p(\beta) &= \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left[Y_i - (\beta_0 + \sum_{j=1}^{p}\beta_j X_{ij})\right]^2\right)\left(\frac{1}{2b}\exp(-|\beta|/b)\right) \\
&= \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \left(\frac{1}{2b}\right)\exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left[Y_i - (\beta_0 + \sum_{j=1}^{p}\beta_j X_{ij})\right]^2 - \frac{|\beta|}{b}\right)
\end{aligned}$$

# Question No 7 p.262 c)

Dire que l'estimateur de Lasso pour $\beta$ est le mode pour la distribution a postériori revient à dire que la valeur la plus probable de $\beta$ dépend d'un certain $\lambda$.

$$\begin{aligned}
\log f(Y \mid X, \beta) p(\beta) &= \log \left[ \left( \frac{1}{\sigma\sqrt{2\pi}} \right)^n \left( \frac{1}{2b} \right) \exp \left( -\frac{1}{2\sigma^2} \sum_{i=1}^{n} \left[ Y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j X_{ij}) \right]^2 - \frac{|\beta|}{b} \right) \right] \\
&= \log \left[ \left( \frac{1}{\sigma\sqrt{2\pi}} \right)^n \left( \frac{1}{2b} \right) \right] - \left( \frac{1}{2\sigma^2} \sum_{i=1}^{n} \left[ Y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j X_{ij}) \right]^2 + \frac{|\beta|}{b} \right)
\end{aligned}$$

On maximise ensuite cette fonction à postériori

$$\arg\max_{\beta}\ f(\beta \mid X, Y) = \arg\max_{\beta}\ \log \left[ \left( \frac{1}{\sigma\sqrt{2\pi}} \right)^n \left( \frac{1}{2b} \right) \right] - \left( \frac{1}{2\sigma^2} \sum_{i=1}^{n} \left[ Y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j X_{ij}) \right]^2 + \frac{|\beta|}{b} \right)$$

Comme on maximise la différence de deux valeurs, on revient à minimiser la valeur négative en terme de $\beta$.

$$\begin{aligned}
&= \arg\min_{\beta}\ \frac{1}{2\sigma^2} \sum_{i=1}^{n} \left[ Y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j X_{ij}) \right]^2 + \frac{|\beta|}{b} \\
&= \arg\min_{\beta}\ \frac{1}{2\sigma^2} \sum_{i=1}^{n} \left[ Y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j X_{ij}) \right]^2 + \frac{1}{b} \sum_{j=1}^{p} |\beta_j| \\
&= \arg\min_{\beta}\ \frac{1}{2\sigma^2} \left( \sum_{i=1}^{n} \left[ Y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j X_{ij}) \right]^2 + \frac{2\sigma^2}{b} \sum_{j=1}^{p} |\beta_j| \right)
\end{aligned}$$

Dans cette équation, si on remplace $\lambda = 2\sigma^2/b$, on obtient l'équation suivante:

$$\begin{aligned}
&= \arg\min_{\beta}\ \sum_{i=1}^{n} \left[ Y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j X_{ij}) \right]^2 + \lambda \sum_{j=1}^{p} |\beta_j| \\
&= \arg\min_{\beta}\ \text{RSS} + \lambda \sum_{j=1}^{p} |\beta_j|
\end{aligned}$$

Ce qui correspond à l'équation 6.7 du ISL. Comme mentionné en page 227 d'ISL: Si la distribution à postériori provient d'une distribution de Laplace avec moyenne zéro et un paramètre “scale” b, alors le mode de $\beta$ est donné par la solution de Lasso lorsque $\lambda = 2\sigma^2/b$.

# Question No 7 p.262 d)

La fonction à postériori selon une distribution normale de moyenne 0 et de variance c est:

$$f(\beta \mid X, Y) \propto f(Y \mid X, \beta)p(\beta \mid X) = f(Y \mid X, \beta)p(\beta)$$

Alors que la fonction de densité est donnée par:

$$p(\beta) = \prod_{i=1}^{p} p(\beta_i) = \prod_{i=1}^{p} \frac{1}{\sqrt{2c\pi}} \exp\left(-\frac{\beta_i^2}{2c}\right) = \left(\frac{1}{\sqrt{2c\pi}}\right)^p \exp\left(-\frac{1}{2c}\sum_{i=1}^{p}\beta_i^2\right)$$

Donc, en combinant les deux équations, on obtient:

$$\begin{aligned} f(Y \mid X, \beta)p(\beta) &= \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left[Y_i - (\beta_0 + \sum_{j=1}^{p}\beta_j X_{ij})\right]^2\right) \left(\frac{1}{\sqrt{2c\pi}}\right)^p \exp\left(-\frac{1}{2c}\sum_{i=1}^{p}\beta_i^2\right) \\ &= \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \left(\frac{1}{\sqrt{2c\pi}}\right)^p \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left[Y_i - (\beta_0 + \sum_{j=1}^{p}\beta_j X_{ij})\right]^2 - \frac{1}{2c}\sum_{i=1}^{p}\beta_i^2\right) \end{aligned}$$

# Question No 7 p.262 e)

Dire que l'estimateur de Ridge pour $\beta$ est le mode et la moyenne pour la distribution a postériori revient à dire que la valeur la plus probable de $\beta$ dépend d'un certain $\lambda$. Ainsi, on commence par simplifier la dernière équation avec un log.

$$\begin{aligned}
\log f(Y \mid X,\beta)p(\beta) &= \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \left(\frac{1}{\sqrt{2c\pi}}\right)^p \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left[Y_i - (\beta_0 + \sum_{j=1}^{p}\beta_j X_{ij})\right]^2 - \frac{1}{2c}\sum_{i=1}^{p}\beta_i^2\right) \\
&= \log\left[\left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \left(\frac{1}{\sqrt{2c\pi}}\right)^p\right] - \left(\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left[Y_i - (\beta_0 + \sum_{j=1}^{p}\beta_j X_{ij})\right]^2 + \frac{1}{2c}\sum_{i=1}^{p}\beta_i^2\right)
\end{aligned}$$

On cherche à maximiser la fonction à postériori :

$$\arg\max_{\beta}\ f(\beta \mid X,Y) = \arg\max_{\beta}\ \log\left[\left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \left(\frac{1}{\sqrt{2c\pi}}\right)^p\right] - \left(\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left[Y_i - (\beta_0 + \sum_{j=1}^{p}\beta_j X_{ij})\right]^2 + \frac{1}{2c}\sum_{i=1}^{p}\beta_i^2\right)$$

Comme la maximisation revient à faire la différence de 2 valeurs, on peut simplement minimiser la valeur négative selon $\beta$:

$$\begin{aligned}
&= \arg\min_{\beta}\left(\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left[Y_i - (\beta_0 + \sum_{j=1}^{p}\beta_j X_{ij})\right]^2 + \frac{1}{2c}\sum_{i=1}^{p}\beta_i^2\right) \\
&= \arg\min_{\beta}\left(\frac{1}{2\sigma^2}\right)\left(\sum_{i=1}^{n}\left[Y_i - (\beta_0 + \sum_{j=1}^{p}\beta_j X_{ij})\right]^2 + \frac{\sigma^2}{c}\sum_{i=1}^{p}\beta_i^2\right)
\end{aligned}$$

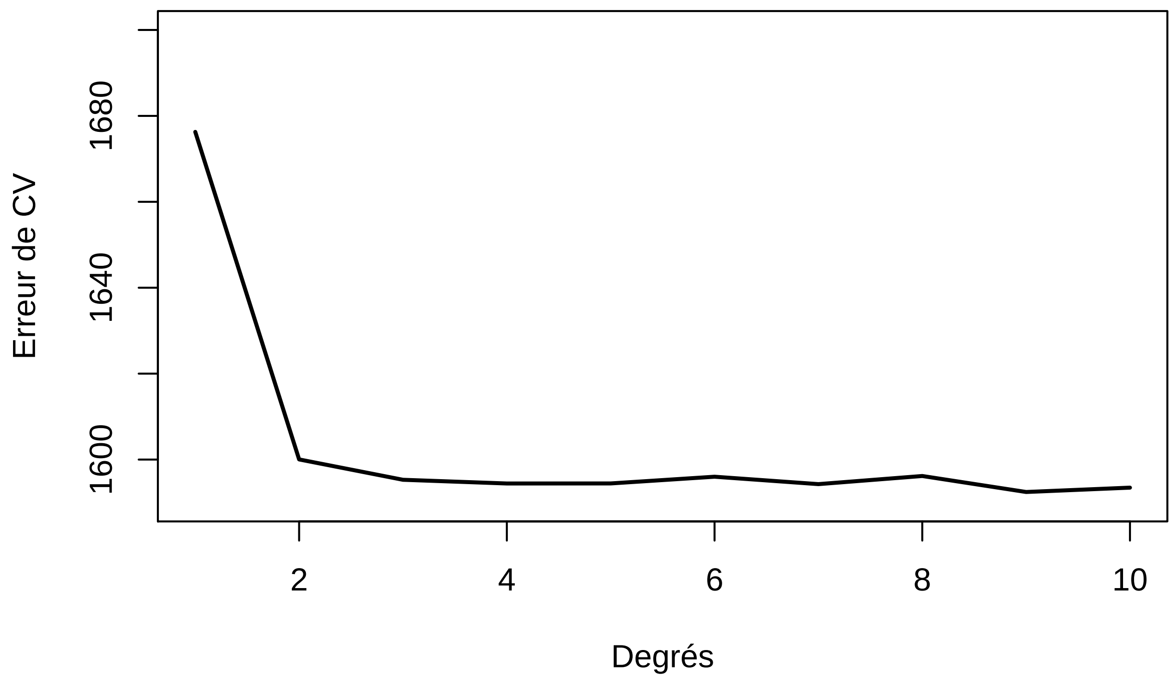
Et si on remplace $\lambda = \sigma^2/c$, nous obtenons:

$$\begin{aligned}
&= \arg\min_{\beta}\left(\frac{1}{2\sigma^2}\right)\left(\sum_{i=1}^{n}\left[Y_i - (\beta_0 + \sum_{j=1}^{p}\beta_j X_{ij})\right]^2 + \lambda\sum_{i=1}^{p}\beta_i^2\right) \\
&= \arg\min_{\beta}\ \mathrm{RSS} + \lambda\sum_{i=1}^{p}\beta_i^2
\end{aligned}$$

Ce qui correspond à l'équation 6.5 du ISL. Comme mentionné en page 227 d'ISL: Si la distribution à postériori provient d'une loi normale de moyenne 0 et de variance c, alors le mode de $\beta$ est donné par la solution de Ridge lorsque $\lambda = \sigma^2/c$.

# Question No 6 p.299 a)

```
library(ISLR)
library(boot)
modele.deltas = rep(NA, 10)
for (i in 1:10) {
  glm.fit = glm(wage~poly(age, i), data=Wage)
  modele.deltas[i] = cv.glm(Wage, glm.fit, K=10)$delta[2]
}

plot(1:10, modele.deltas, xlab="Degrés", ylab="Erreur de CV", type="l", pch=20, lwd=2, ylim=c(1590, 170
```




On peut voir que le premier minimum intéressant se trouve à 3 degrés, par la suite, l'erreur reste relativement stable malgré l'ajout de degrés supplémentaires.

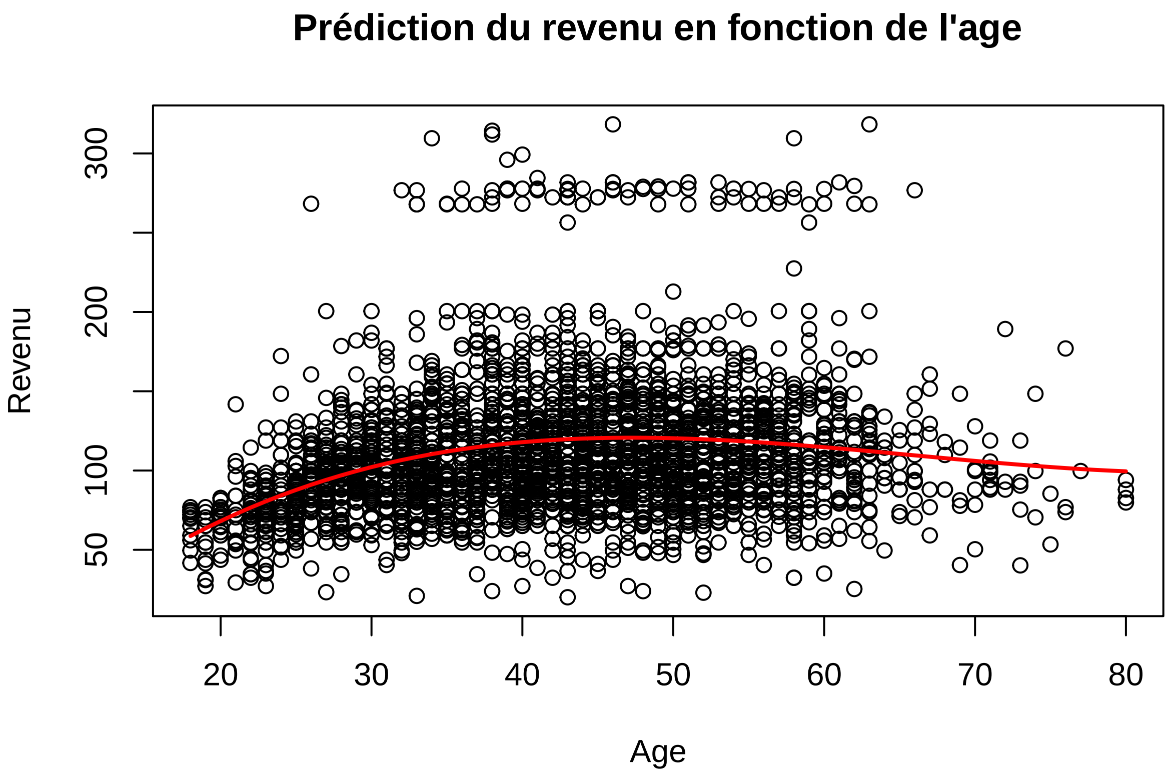
```
fit.1 = lm(wage~poly(age, 1), data=Wage)
fit.2 = lm(wage~poly(age, 2), data=Wage)
fit.3 = lm(wage~poly(age, 3), data=Wage)
fit.4 = lm(wage~poly(age, 4), data=Wage)
fit.5 = lm(wage~poly(age, 5), data=Wage)
fit.6 = lm(wage~poly(age, 6), data=Wage)
fit.7 = lm(wage~poly(age, 7), data=Wage)
fit.8 = lm(wage~poly(age, 8), data=Wage)
fit.9 = lm(wage~poly(age, 9), data=Wage)
fit.10 = lm(wage~poly(age, 10), data=Wage)
```

```r
anova(fit.1, fit.2, fit.3, fit.4, fit.5, fit.6, fit.7, fit.8, fit.9, fit.10)
```

```
## Analysis of Variance Table
##
## Model  1: wage ~ poly(age, 1)
## Model  2: wage ~ poly(age, 2)
## Model  3: wage ~ poly(age, 3)
## Model  4: wage ~ poly(age, 4)
## Model  5: wage ~ poly(age, 5)
## Model  6: wage ~ poly(age, 6)
## Model  7: wage ~ poly(age, 7)
## Model  8: wage ~ poly(age, 8)
## Model  9: wage ~ poly(age, 9)
## Model 10: wage ~ poly(age, 10)
##    Res.Df     RSS Df Sum of Sq        F    Pr(>F)
## 1    2998 5022216
## 2    2997 4793430  1    228786 143.7638 < 2.2e-16 ***
## 3    2996 4777674  1     15756   9.9005  0.001669 **
## 4    2995 4771604  1      6070   3.8143  0.050909 .
## 5    2994 4770322  1      1283   0.8059  0.369398
## 6    2993 4766389  1      3932   2.4709  0.116074
## 7    2992 4763834  1      2555   1.6057  0.205199
## 8    2991 4763707  1       127   0.0796  0.777865
## 9    2990 4756703  1      7004   4.4014  0.035994 *
## 10   2989 4756701  1         3   0.0017  0.967529
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

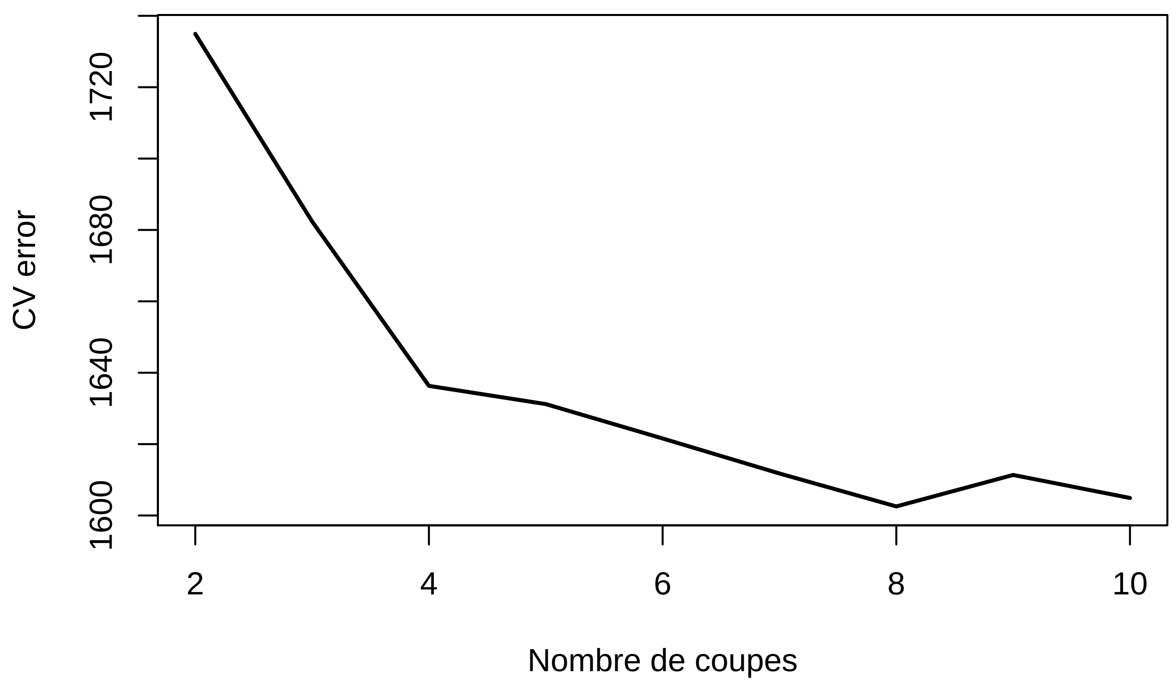
En vérifiant à l'aide d'un ANOVA on constate qu'au delà de 3 degrés sont réellement importants pour rester sous le seuil de confiance de 0.01, autrement, on passe à un seuil de 0.1 avec 4 degrés ou 0.05 avec 9 degrés. L'utilisation de 3 degrés est un bon compromis entre un modèle simple et un niveau de confiance acceptable.

```r
plot(wage~age, data=Wage, main = "Prédiction du revenu en fonction de l'age", xlab = "Age", ylab = "Reve
age.limites = range(Wage$age)
age.grid = seq(from=age.limites[1], to=age.limites[2])
lm.fit = lm(wage~poly(age, 3), data=Wage)
lm.pred = predict(lm.fit, data.frame(age=age.grid))
lines(age.grid, lm.pred, col="red", lwd=2)
```


Prédiction du revenu en fonction de l'age
Revenu
50
100
200
300
20
30
40
50
60
70
80
Age

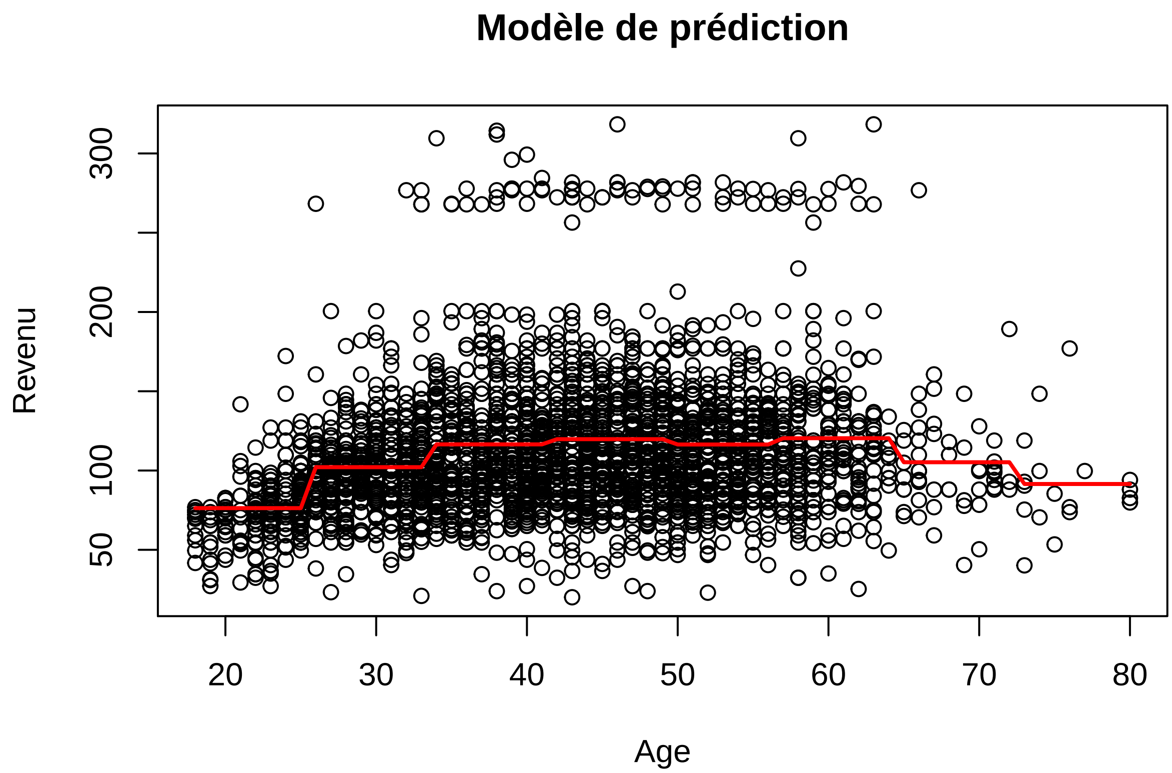
## Question No 6 p.299 b)

```
modele.cvs = rep(NA, 10)
for (i in 2:10) {
  Wage$age.cut = cut(Wage$age, i)
  lm.fit = glm(wage~age.cut, data=Wage)
  modele.cvs[i] = cv.glm(Wage, lm.fit, K=10)$delta[2]
}
plot(2:10, modele.cvs[-1], xlab="Nombre de coupes", ylab="CV error", type="l", pch=20, lwd=2)
```




On peut voir sur le graphique que le minimum le plus intéressant, se trouver à k = 8 coupures.

```
lm.fit = glm(wage~cut(age, 8), data=Wage)
age.limitess = range(Wage$age)
age.grid = seq(from=age.limites[1], to=age.limites[2])
lm.pred = predict(lm.fit, data.frame(age=age.grid))
plot(wage~age, data=Wage, main = "Modèle de prédiction", xlab = "Age", ylab = "Revenu")
lines(age.grid, lm.pred, col="red", lwd=2)
```


Modèle de prédiction
Revenu
300
200
100
50
20
30
40
50
60
70
80
Age