Rapport de MAD

Kai HUANG et Ni CHEN

09/01/2017

Introduction

Le but de notre projet est de chercher un modèle de régression multiple qui sert à prévoir les valeurs des maisons dans la banlieue de Boston en fonction de certains critères, et de chercher des composantes plus importantes par rapport aux autres en utilisant le PCA (Principle Component Analysis). Dans notre data, il y a 506 observations et 14 variables. Il n'y a ni des données manquantes, ni des valeurs aberrantes. Nous avons 1 variables qualitatives et 13 variables qualitatives. Veillez trouver ci-dessous des explications des variable:

1. CRIM (quantitative): per capita crime rate by town.
2. ZN (quantitative): proportion of residential land zoned for lots over 25,000 sq.ft.
3. INDUS (quantitative): proportion of non-retail business acres per town.
4. CHAS (qualitative): Charles River dummy variable (= 1 if tract bounds river; 0 otherwise).
5. NOX: (quantitative) nitric oxides concentration (parts per 10 million).
6. RM: (quantitative) average number of rooms per dwelling.
7. AGE (quantitative): proportion of owner-occupied units built prior to 1940.
8. DIS (quantitative): weighted distances to five Boston employment centres.
9. RAD (quantitative): index of accessibility to radial highways.
10. TAX (quantitative): full-value property-tax rate per 10,000 dollars.
11. PTRATIO (quantitative): pupil-teacher ratio by town.
12. B (quantitative): 1000(Bk - 0.63)ˆ2 where Bk is the proportion of blacks by town.
13. LSTAT (quantitative): lower status of the population.
14. MEDV (quantitative): Median value of owner-occupied homes in 1000's.

Afin de réaliser notre objectif, nous séparons notre projet en 4 parties.

La première partie: analyse descriptive. Nous allons étudier les caractéristiques numériques des données pour mieux connaître des données. Pour les variables quantitatives, nous allons calculer leur moyennes, médians, minimums, maximums et tracer leurs boîtes de moustache. De plus, il est nécessaire de comparer les varibales 2-à-2 pour trouver leur covariances, corrélations et dépendances entre eux. Pour les varibales qualitatives, nous les examinons un par un de relations avec la variable objective MEDV.

La deuxième partie: modélisation. Dans cette partie. Nous utiliserons la régression multiple pour bien trouver la relation entre le prix de logement dans la banlieue du Boston et les variables. D'autre part, nous somme convaincus que nous devons faire la sélection de stepwise afin de sélectionner des variables en fonction de certaines critères et utiliser la méthode de pénalisation pour balancer le résidu et la variance.

La troisième partie: analyse en composantes principales. Nous utilisons la méthode PCA (Principle Component Analysis) sur les 10 variables quantitatives (sauf de MEDV) pour réduire le nombre des variables des prédicteurs et obtenir les variables les plus utiles.

Descriptive Analysis

D'abord, nous lisons les datas et nommons les colonnes. Ca nous permet de mieux manipuler les variables dans les analyses suivantes.

```
datahousing <- read.table("housingdata.txt")
colnames(datahousing) <- c("crim", "zn", "indus", "chas", "nox", "rm", "age", "dis",
                           "rad", "tax", "ptratio", "b", "lstat","medv" )
```

Dans notre data, la variable de réponse est la variable medv. Du coup, nous allons calculer sa moyenne, sa variance, etc. . .

```
mean(datahousing$medv)
```

```
## [1] 22.53281
```

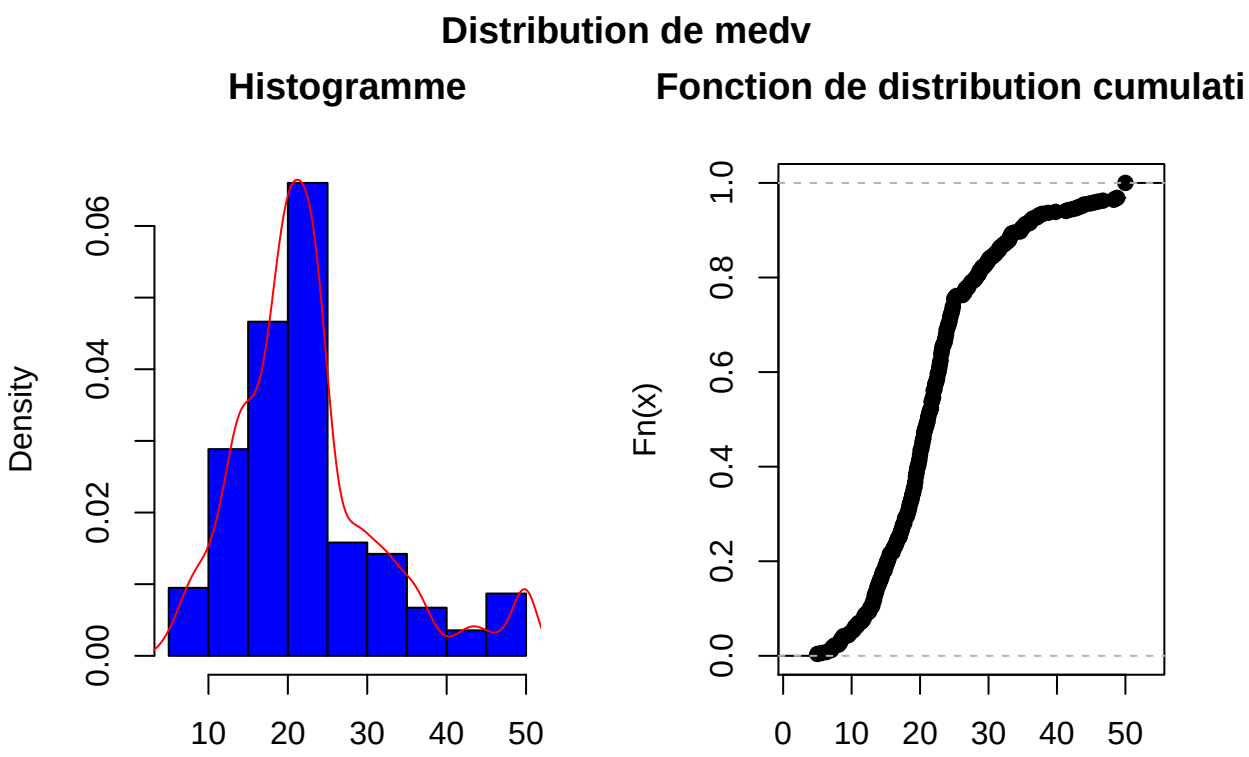
```
median(datahousing$medv)
```

```
## [1] 21.2
```

```
var(datahousing$medv)
```

```
## [1] 84.58672
```

```
par(mfrow = c(1,2))
hist(datahousing$medv, freq=FALSE, col="blue", main="Histogramme",xlab="")
lines(density(datahousing$medv), col="red")
plot(ecdf(datahousing$medv), main="Fonction de distribution cumulative",xlab="")
title(outer=TRUE, main="\n Distribution de medv")
```



Les résultats nous disent que sa moyenne est environs 22 (ça veut dire que la moyenne de medv est 22,000$). De plus, la variance est grande et il nous dit que le medv est dispersé. Les valeurs varient en fonction des plusieurs facteurs, par exemple, la criminalité de la ville, l'âge de la ville, le rapport des élèves et des professeurs, etc...

Comme nous pouvons le voir sur la figure, la distribution de medv est similaire de la distribution gaussienne. Et la plus part de prix de maison est concentré entre 10 et 30 milles dollars.

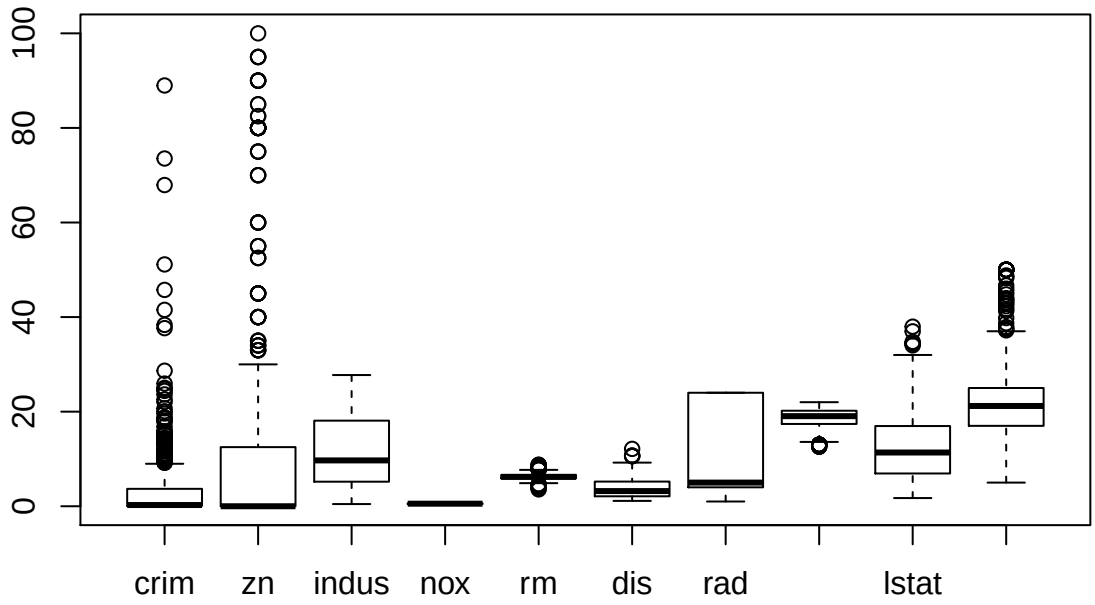
Analyses descriptives univariées

Pour les variables quantitatives, nous allons calculer leur moyennes, médians, minimums, maximums et tracer leurs boîtes de moustache.

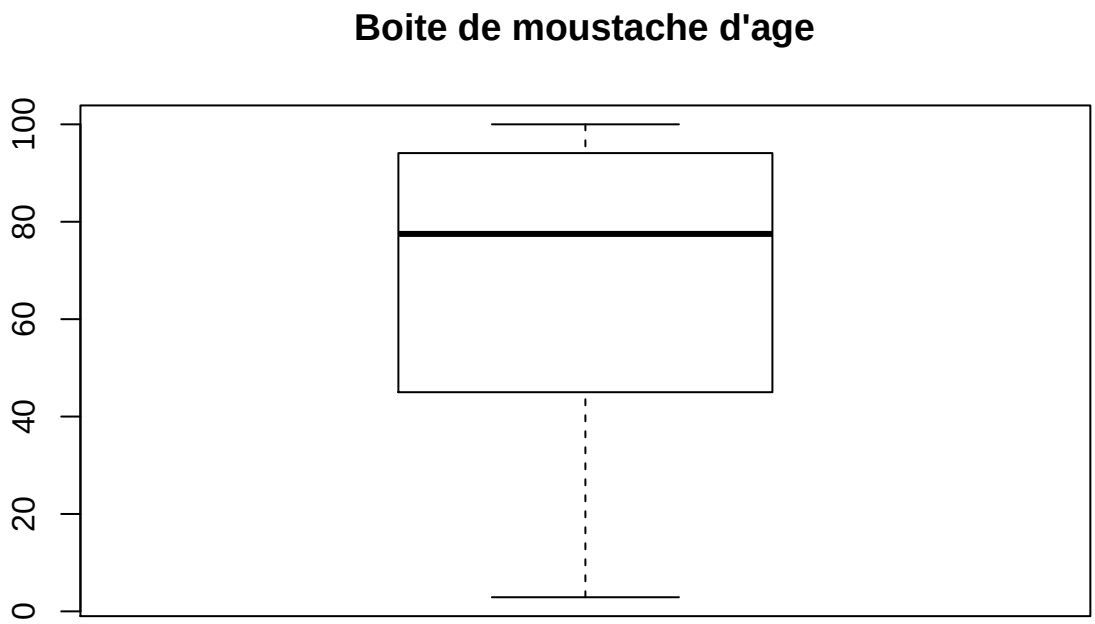
```
summary(datahousing)
```

```
##       crim                zn              indus            chas        
##  Min.   : 0.00632   Min.   :  0.00   Min.   : 0.46   Min.   :0.00000  
##  1st Qu.: 0.08204   1st Qu.:  0.00   1st Qu.: 5.19   1st Qu.:0.00000  
##  Median : 0.25651   Median :  0.00   Median : 9.69   Median :0.00000  
##  Mean   : 3.61352   Mean   : 11.36   Mean   :11.14   Mean   :0.06917  
##  3rd Qu.: 3.67708   3rd Qu.: 12.50   3rd Qu.:18.10   3rd Qu.:0.00000  
##  Max.   :88.97620   Max.   :100.00   Max.   :27.74   Max.   :1.00000  
##       nox               rm              age              dis        
##  Min.   :0.3850   Min.   :3.561   Min.   :  2.90   Min.   : 1.130  
##  1st Qu.:0.4490   1st Qu.:5.886   1st Qu.: 45.02   1st Qu.: 2.100  
##  Median :0.5380   Median :6.208   Median : 77.50   Median : 3.207  
##  Mean   :0.5547   Mean   :6.285   Mean   : 68.57   Mean   : 3.795  
##  3rd Qu.:0.6240   3rd Qu.:6.623   3rd Qu.: 94.08   3rd Qu.: 5.188  
##  Max.   :0.8710   Max.   :8.780   Max.   :100.00   Max.   :12.127  
##       rad              tax           ptratio            b        
##  Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   :  0.32  
##  1st Qu.: 4.000   1st Qu.:279.0   1st Qu.:17.40   1st Qu.:375.38  
##  Median : 5.000   Median :330.0   Median :19.05   Median :391.44  
##  Mean   : 9.549   Mean   :408.2   Mean   :18.46   Mean   :356.67  
##  3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:396.23  
##  Max.   :24.000   Max.   :711.0   Max.   :22.00   Max.   :396.90  
##      lstat            medv      
##  Min.   : 1.73   Min.   : 5.00  
##  1st Qu.: 6.95   1st Qu.:17.02  
##  Median :11.36   Median :21.20  
##  Mean   :12.65   Mean   :22.53  
##  3rd Qu.:16.95   3rd Qu.:25.00  
##  Max.   :37.97   Max.   :50.00
```

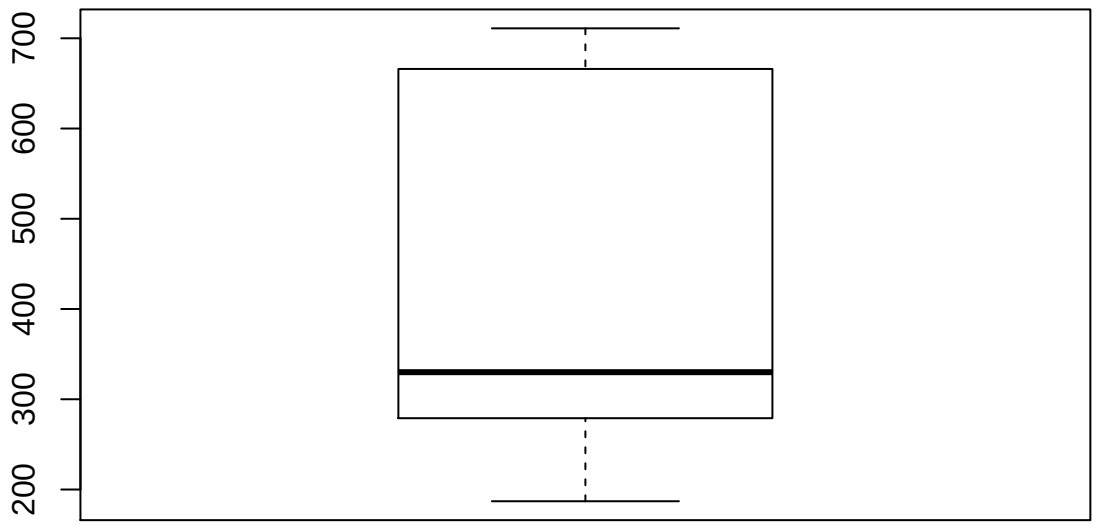
```
boxplot(datahousing[, c(-4,-7, -10, -12)])
```



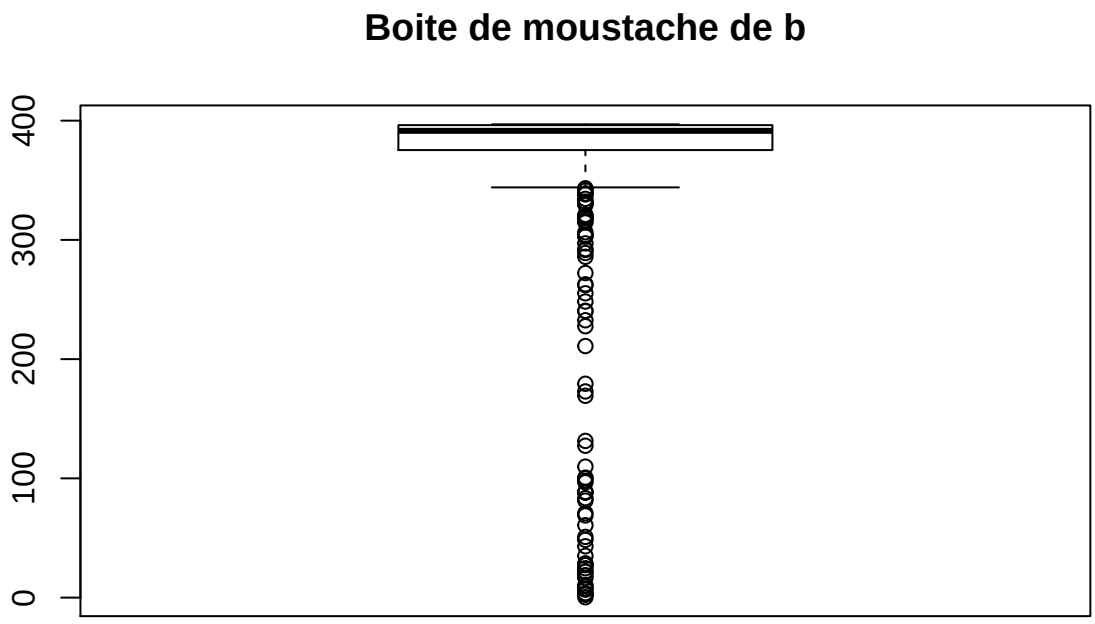

```r
boxplot(datahousing$age, main = "Boite de moustache d'âge")
```




```r
boxplot(datahousing$tax, main = "Boite de moustache de tax")
```




```
boxplot(datahousing$b, main = "Boite de moustache de b")
```

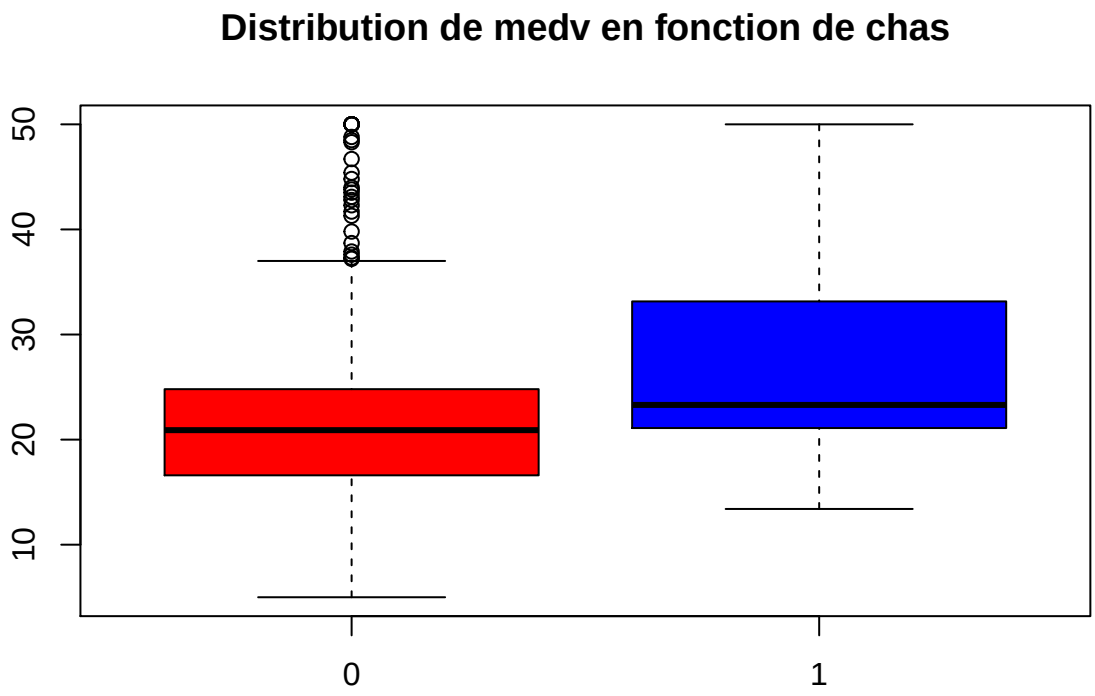


Nous voyons que les variables sont concentrés autour de leurs moyennes respectives. En moyenne, 80 pour-cents des résidences sont créés avant 1940. Et la moyenne de la variable b qui indique la proportion des noirs dans la ville est environ 400.

Analyses descriptives bivariées

Dans notre data, il existe qu'une seule variable qualitative. Afin de savoir son impact sur la variable de réponse, nous utiliserons la boîte de moustache.

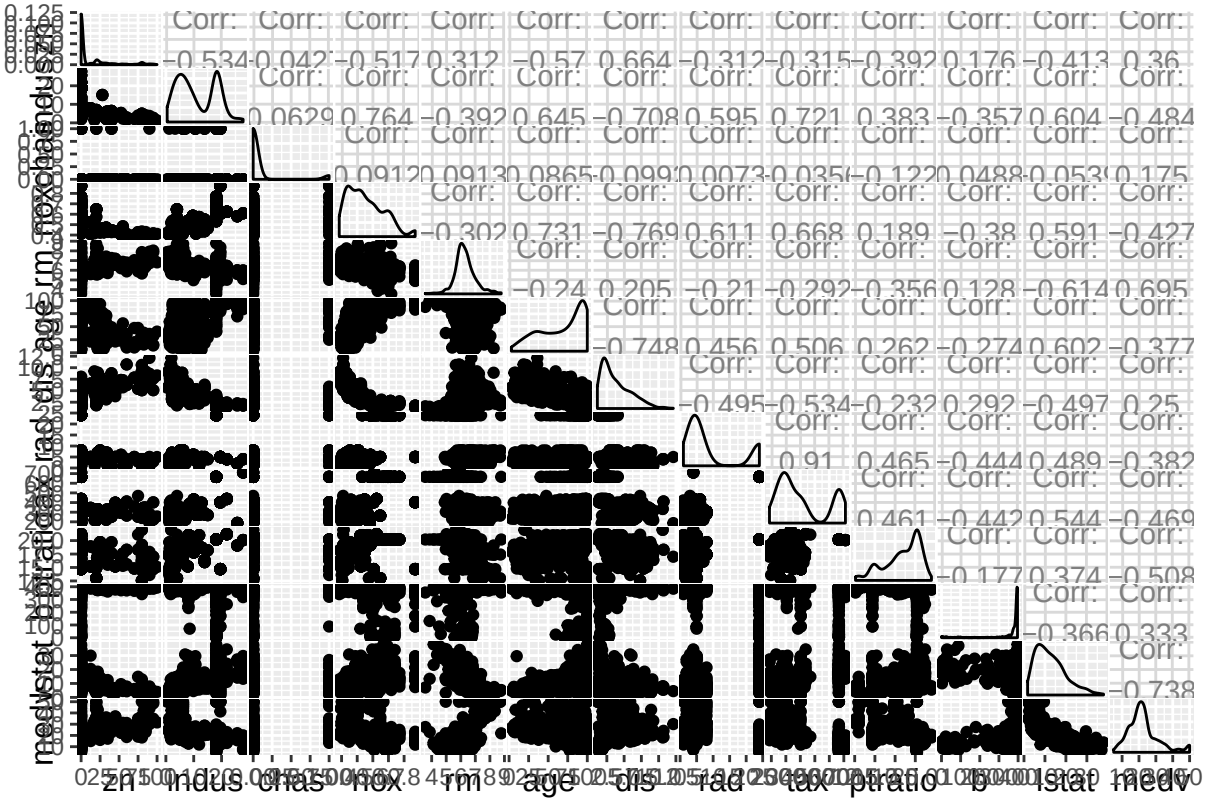
```
factor_chas <- factor(datahousing$chas)
plot(factor_chas, datahousing$medv, col=c('red', 'blue'))
title("Distribution de medv en fonction de chas")
```



Il nous dit que le prix de maison dans les villes avec la rivière coûtent plus chères que ceux sans rivière.

Pour les variables quantitatives, nous visons à étudier les relations entré les 12 variables de prédicteurs et leur relation avec la variable medv.

```
library(GGally)
ggpairs(datahousing[, -datahousing$chas])
```



Nous voyons que certaines variables sont liées avec des autres variables(ici, si le coefficient de corrélation est supérieur que 0.7, nous pensons que les deux variables sont liées).

1) La variable nos et la variable indus: Ils ont une relation linéaire. Et la relation est positive. La proportion d'acre non commercial contribue à la concentration d'oxynitrure.

2) La variable indus et la variable tax: Ils ont une relation positive, mais nous ne pouvons pas déduire la relation précise entre les deux en fonction de la graph.

3) La variable nos et la variable age: Selon le graph, nous voyons qu'ils sont très corrélés et ils suivent la loi log(x).

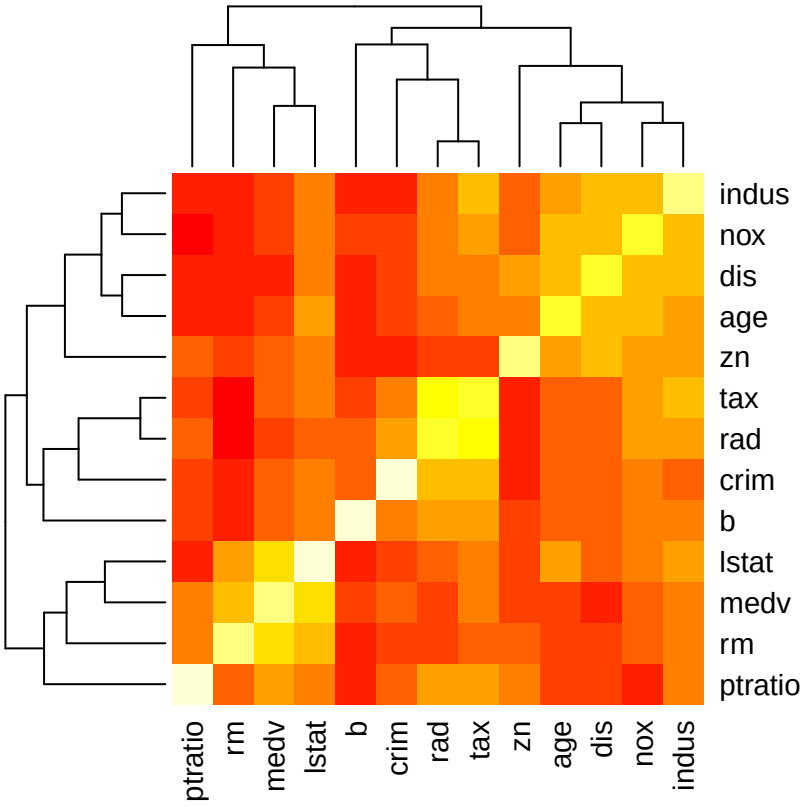
4) La variable nos et la variable dis: Nous voyons qu'ils sont liés et ils suivent la loi 1/x.

5) La variable age et la variable dis: Ils ont une relation négative.

6) La variable rad et la variable tax: Ils ont une relation positive.

Nous voyons que le coefficient entre rm et medv et supérieur que 0.7, du coup, nous pensons que la valeur de logement dépend du nombre moyen par logement. De plus, la population avec un mauvais salaire a un impact négative sur la valeur des logement dans cette ville.

Nous faisons aussi le heatmap.



Il nous donne la même chose.

Modélisation

Régression linéaire

Dans cette partie, nous allons chercher des modèles en utilisant des méthodes divers. D'abord, nous créons deux modèles null et full en utilisant la régression linéaire. Et puis, nous les comparons en utilisant la fonction anova. De plus, nous allons étudier le meilleur modèle.

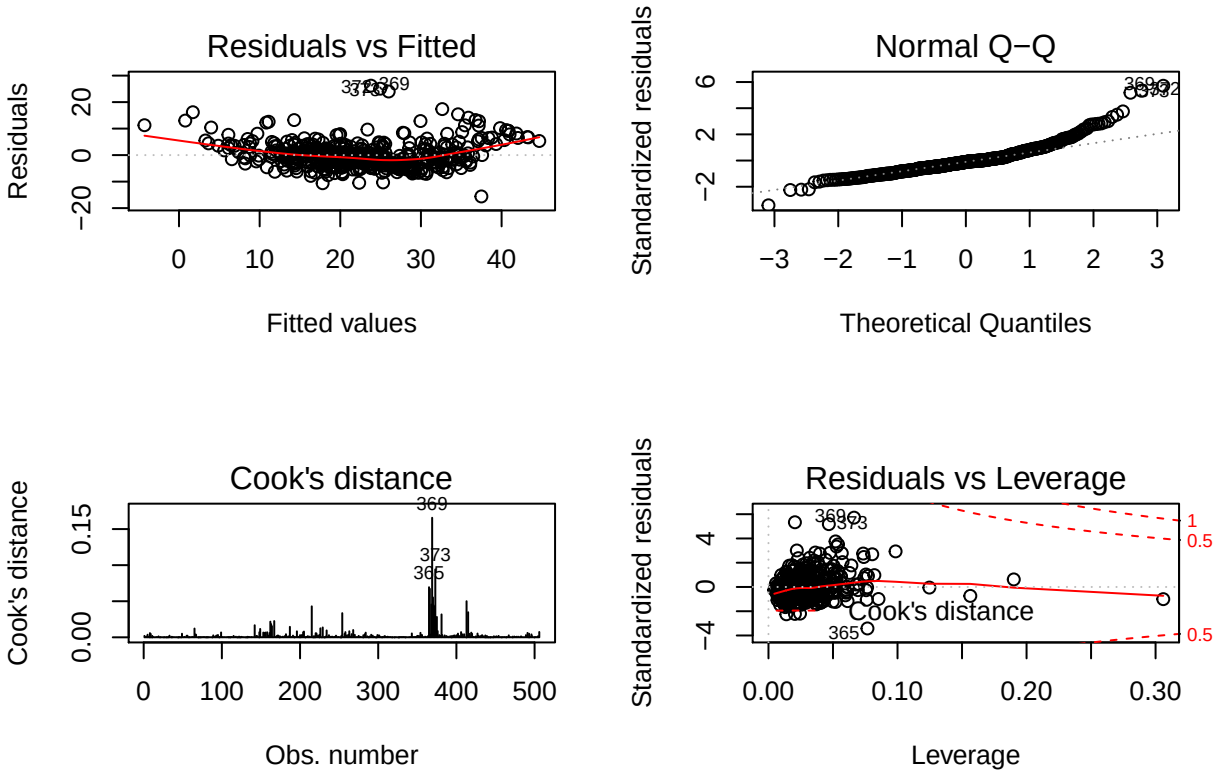
```
null <- lm(medv ~ 1, datahousing)
full <- lm(medv ~ ., datahousing)

anova(null, full)
```

```
## Analysis of Variance Table
##
## Model 1: medv ~ 1
## Model 2: medv ~ crim + zn + indus + chas + nox + rm + age + dis + rad +
##     tax + ptratio + b + lstat
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)
## 1    505 42716
## 2    492 11079 13     31638 108.08 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(full)
```

```
##
## Call:
## lm(formula = medv ~ ., data = datahousing)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.595  -2.730  -0.518   1.777  26.199
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
## crim        -1.080e-01  3.286e-02  -3.287 0.001087 **
## zn           4.642e-02  1.373e-02   3.382 0.000778 ***
## indus        2.056e-02  6.150e-02   0.334 0.738288
## chas         2.687e+00  8.616e-01   3.118 0.001925 **
## nox         -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
## rm           3.810e+00  4.179e-01   9.116  < 2e-16 ***
## age          6.922e-04  1.321e-02   0.052 0.958229
## dis         -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
## rad          3.060e-01  6.635e-02   4.613 5.07e-06 ***
## tax         -1.233e-02  3.760e-03  -3.280 0.001112 **
## ptratio     -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
## b            9.312e-03  2.686e-03   3.467 0.000573 ***
## lstat       -5.248e-01  5.072e-02 -10.347  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.745 on 492 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7338
## F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

```r
par(mfrow=c(2,2))
plot(full, which=c(1,2,4,5))
```



Le modèle null contient aucune variable, par contre, le modèle full contient tous les variables. Comme prévu, le modèle full est mieux que le modèle null parce que son RSS(residual sum of squares) est plus petit. Selon le graph, nous voyons que quand les fitted values sont petits ou grands, les résidus ne sont pas concentré autour de zéro, du coup, nous allons faire une transformation de log. De plus, les distances de cook des observations sont raisonnables, cependant, nous éliminons la 365ème, la 369ème et la 373ème observations.

```
datahousing=datahousing[c(-365, -369, -373),]

null <- lm( log(medv) ~ 1, datahousing)
full <- lm( log(medv) ~ ., datahousing)
```

Sélection de stepwise

D'abord, nous pensons à séparer les datas originaux en deux matrices. Une matrice contient 4 cinquièmes des datas originaux et nous allons l'utiliser pour chercher un modèle pertinent, l'autre matrice contient 1 cinquième des datas originaux et il sert à tester le modèle trouvé. Afin de maintenir l'aleatoireté de data, nous ne pouvons pas prendre les 4 premiers cinquièmes de data car nous perdrons l'aleatoireté de data et nous n'aurons pas un bon modèle. Du coup, nous utilisons la fonction "sample" qui prend des observations aléatoirement. Nous aurons une matrice pour chercher le modèle et une matrice pour tester le modèle.

Ici, nous pensons à chercher un modèle à partir du modèle full. A chaque étape, nous essayons d'ajouter et d'élimiter une variable en fonction de la critère AIC (ou BIC) jusqu'à un moment où la critère ne change pas beaucoup quand nous ajouter et limiter n'importe quelle variable. A la fin, nous aurons deux modèles qui prévoient le medv.

```r
n = nrow(datahousing)
test <- sample(1:n, round(n)/5)
datahousing.train <- datahousing[-test, ]
datahousing.test <- datahousing[test, ]

m <- nrow(datahousing.train)
lower = terms(medv ~ 1, data=datahousing.train)
upper = terms(medv ~ ., data=datahousing.train)

scope <- list(lower, upper)
model.AIC <- step(full, scope, direction='both', trace=TRUE)
```

```
## Start:  AIC=-1692.34
## log(medv) ~ crim + zn + indus + chas + nox + rm + age + dis +
##     rad + tax + ptratio + b + lstat
##
##           Df Sum of Sq    RSS     AIC
## - age      1    0.0011 16.453 -1694.3
## - indus    1    0.0398 16.491 -1693.1
## <none>                 16.452 -1692.3
## - zn       1    0.1155 16.567 -1690.8
## - chas     1    0.2540 16.706 -1686.6
## - b        1    0.5458 16.997 -1677.9
## - tax      1    0.6131 17.065 -1675.9
## - nox      1    0.7879 17.240 -1670.8
## - rad      1    0.8804 17.332 -1668.1
## - dis      1    1.1804 17.632 -1659.5
## - rm       1    1.6043 18.056 -1647.5
## - ptratio  1    1.8246 18.276 -1641.4
## - crim     1    2.1379 18.590 -1632.9
## - lstat    1    5.8451 22.297 -1541.4
##
## Step:  AIC=-1694.31
## log(medv) ~ crim + zn + indus + chas + nox + rm + dis + rad +
##     tax + ptratio + b + lstat
##
##           Df Sum of Sq    RSS     AIC
## - indus    1    0.0398 16.493 -1695.1
## <none>                 16.453 -1694.3
## - zn       1    0.1204 16.573 -1692.6
## - chas     1    0.2529 16.706 -1688.6
## - b        1    0.5448 16.998 -1679.9
## - tax      1    0.6154 17.068 -1677.8
## - nox      1    0.8679 17.321 -1670.5
## - rad      1    0.8920 17.345 -1669.8
## - dis      1    1.2597 17.713 -1659.2
## - rm       1    1.6710 18.124 -1647.7
## - ptratio  1    1.8448 18.298 -1642.8
## - crim     1    2.1385 18.591 -1634.8
## - lstat    1    6.7485 23.201 -1523.4
##
## Step:  AIC=-1695.09
## log(medv) ~ crim + zn + chas + nox + rm + dis + rad + tax + ptratio +
##     b + lstat
```

```
##
##           Df Sum of Sq    RSS     AIC
## <none>                  16.493 -1695.1
## - zn       1    0.1073 16.600 -1693.8
## - chas     1    0.2769 16.770 -1688.7
## - b        1    0.5356 17.028 -1681.0
## - tax      1    0.6013 17.094 -1679.1
## - nox      1    0.8314 17.324 -1672.3
## - rad      1    0.8566 17.349 -1671.6
## - dis      1    1.4295 17.922 -1655.3
## - rm       1    1.6379 18.131 -1649.5
## - ptratio  1    1.8055 18.298 -1644.8
## - crim     1    2.1630 18.656 -1635.1
## - lstat    1    6.7088 23.201 -1525.4
```

```
model.BIC <- step(full, scope, direction='both', k=log(m), trace=TRUE)
```

```
## Start:  AIC=-1636.36
## log(medv) ~ crim + zn + indus + chas + nox + rm + age + dis +
##     rad + tax + ptratio + b + lstat
##
##           Df Sum of Sq    RSS     AIC
## - age      1    0.0011 16.453 -1642.3
## - indus    1    0.0398 16.491 -1641.1
## - zn       1    0.1155 16.567 -1638.8
## <none>                  16.452 -1636.4
## - chas     1    0.2540 16.706 -1634.7
## - b        1    0.5458 16.997 -1625.9
## - tax      1    0.6131 17.065 -1624.0
## - nox      1    0.7879 17.240 -1618.8
## - rad      1    0.8804 17.332 -1616.1
## - dis      1    1.1804 17.632 -1607.5
## - rm       1    1.6043 18.056 -1595.5
## - ptratio  1    1.8246 18.276 -1589.5
## - crim     1    2.1379 18.590 -1580.9
## - lstat    1    5.8451 22.297 -1489.4
##
## Step:  AIC=-1642.32
## log(medv) ~ crim + zn + indus + chas + nox + rm + dis + rad +
##     tax + ptratio + b + lstat
##
##           Df Sum of Sq    RSS     AIC
## - indus    1    0.0398 16.493 -1647.1
## - zn       1    0.1204 16.573 -1644.7
## <none>                  16.453 -1642.3
## - chas     1    0.2529 16.706 -1640.6
## - b        1    0.5448 16.998 -1631.9
## - tax      1    0.6154 17.068 -1629.8
## - nox      1    0.8679 17.321 -1622.5
## - rad      1    0.8920 17.345 -1621.8
## - dis      1    1.2597 17.713 -1611.2
## - rm       1    1.6710 18.124 -1599.7
## - ptratio  1    1.8448 18.298 -1594.9
## - crim     1    2.1385 18.591 -1586.8
## - lstat    1    6.7485 23.201 -1475.4
```

```
##
## Step:  AIC=-1647.1
## log(medv) ~ crim + zn + chas + nox + rm + dis + rad + tax + ptratio +
##     b + lstat
##
##           Df Sum of Sq    RSS     AIC
## - zn       1    0.1073 16.600 -1649.8
## <none>                 16.493 -1647.1
## - chas     1    0.2769 16.770 -1644.7
## - b        1    0.5356 17.028 -1637.0
## - tax      1    0.6013 17.094 -1635.1
## - nox      1    0.8314 17.324 -1628.4
## - rad      1    0.8566 17.349 -1627.6
## - dis      1    1.4295 17.922 -1611.3
## - rm       1    1.6379 18.131 -1605.5
## - ptratio  1    1.8055 18.298 -1600.8
## - crim     1    2.1630 18.656 -1591.1
## - lstat    1    6.7088 23.201 -1481.4
##
## Step:  AIC=-1649.84
## log(medv) ~ crim + chas + nox + rm + dis + rad + tax + ptratio +
##     b + lstat
##
##           Df Sum of Sq    RSS     AIC
## <none>                 16.600 -1649.8
## - chas     1    0.2759 16.876 -1647.5
## - tax      1    0.5264 17.126 -1640.1
## - b        1    0.5392 17.139 -1639.8
## - rad      1    0.8169 17.417 -1631.7
## - nox      1    0.9030 17.503 -1629.2
## - dis      1    1.4211 18.021 -1614.5
## - rm       1    1.8102 18.410 -1603.8
## - crim     1    2.0927 18.693 -1596.1
## - ptratio  1    2.3942 18.994 -1588.1
## - lstat    1    6.6884 23.288 -1485.5
```

```
summary(model.AIC)
```

```
##
## Call:
## lm(formula = log(medv) ~ crim + zn + chas + nox + rm + dis +
##     rad + tax + ptratio + b + lstat, data = datahousing)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.73613 -0.09140 -0.01669  0.09296  0.87386
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.8688757  0.2005990  19.287  < 2e-16 ***
## crim        -0.0101867  0.0012694  -8.025 7.56e-15 ***
## zn           0.0009361  0.0005239   1.787  0.07457 .
## chas         0.0973767  0.0339131   2.871  0.00426 **
## nox         -0.6820816  0.1371028  -4.975 9.04e-07 ***
## rm           0.1134584  0.0162480   6.983 9.43e-12 ***
```

```
## dis         -0.0471506  0.0072277  -6.524 1.71e-10 ***
## rad          0.0124236  0.0024601   5.050 6.24e-07 ***
## tax         -0.0005527  0.0001306  -4.231 2.78e-05 ***
## ptratio     -0.0367331  0.0050103  -7.332 9.45e-13 ***
## b            0.0004133  0.0001035   3.993 7.52e-05 ***
## lstat       -0.0265023  0.0018753 -14.132  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1833 on 491 degrees of freedom
## Multiple R-squared:  0.8009, Adjusted R-squared:  0.7964
## F-statistic: 179.5 on 11 and 491 DF,  p-value: < 2.2e-16
```

```
summary(model.BIC)
```

```
##
## Call:
## lm(formula = log(medv) ~ crim + chas + nox + rm + dis + rad +
##     tax + ptratio + b + lstat, data = datahousing)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.73350 -0.09827 -0.01640  0.09437  0.88074
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.8799314  0.2009499  19.308  < 2e-16 ***
## crim        -0.0099766  0.0012668  -7.876 2.19e-14 ***
## chas         0.0971894  0.0339884   2.859  0.00442 **
## nox         -0.7071230  0.1366884  -5.173 3.36e-07 ***
## rm           0.1178828  0.0160939   7.325 9.86e-13 ***
## dis         -0.0406742  0.0062672  -6.490 2.10e-10 ***
## rad          0.0120988  0.0024589   4.920 1.18e-06 ***
## tax         -0.0005072  0.0001284  -3.950 8.96e-05 ***
## ptratio     -0.0397781  0.0047221  -8.424 4.03e-16 ***
## b            0.0004147  0.0001037   3.998 7.38e-05 ***
## lstat       -0.0264598  0.0018793 -14.080  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1837 on 492 degrees of freedom
## Multiple R-squared:  0.7996, Adjusted R-squared:  0.7955
## F-statistic: 196.3 on 10 and 492 DF,  p-value: < 2.2e-16
```

Le résume des deux modèle nous dit que le modèle model.AIC a un R^2 0.8009 avec une valeur p qui est presque zéro et que le modèle model.BIC a un R^2 0.7996 avec une valeur p qui est presque zéro. Les deux modèles sont tous pertinents, d'après moi, le modèle model.BIC est préférable parce que les qualités des deux modèles sont presque même et le model.BIC contient 10 variables (par contre, le model.AIC contient 11 variables, donc, le model.BIC est plus interprétable).

Méthode de pénalisation.

Régression de ridge

Nous faisons la régression de ridge pour chercher un modèle pertinent. Et puis, nous traçons le chemin de la régression de ridge.

```
library(Matrix)
```

```
## Warning: package 'Matrix' was built under R version 3.3.2
```

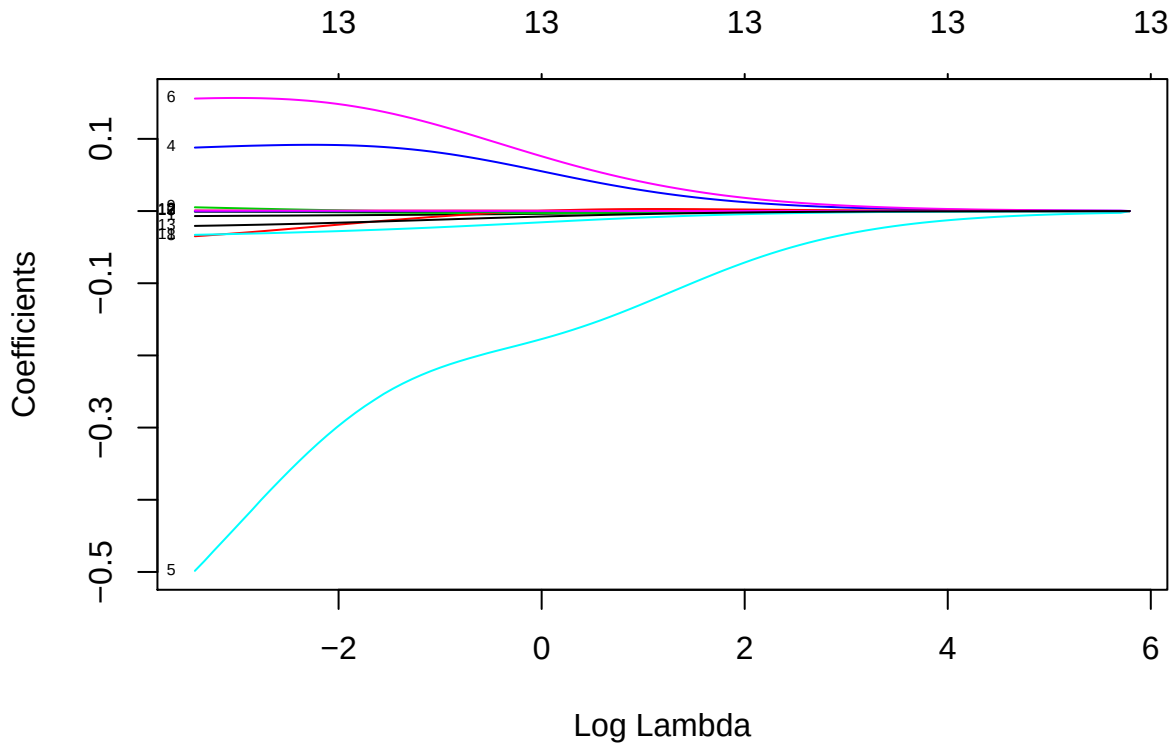
```
library(foreach)
```

```
## Warning: package 'foreach' was built under R version 3.3.2
```

```
library(glmnet)
```

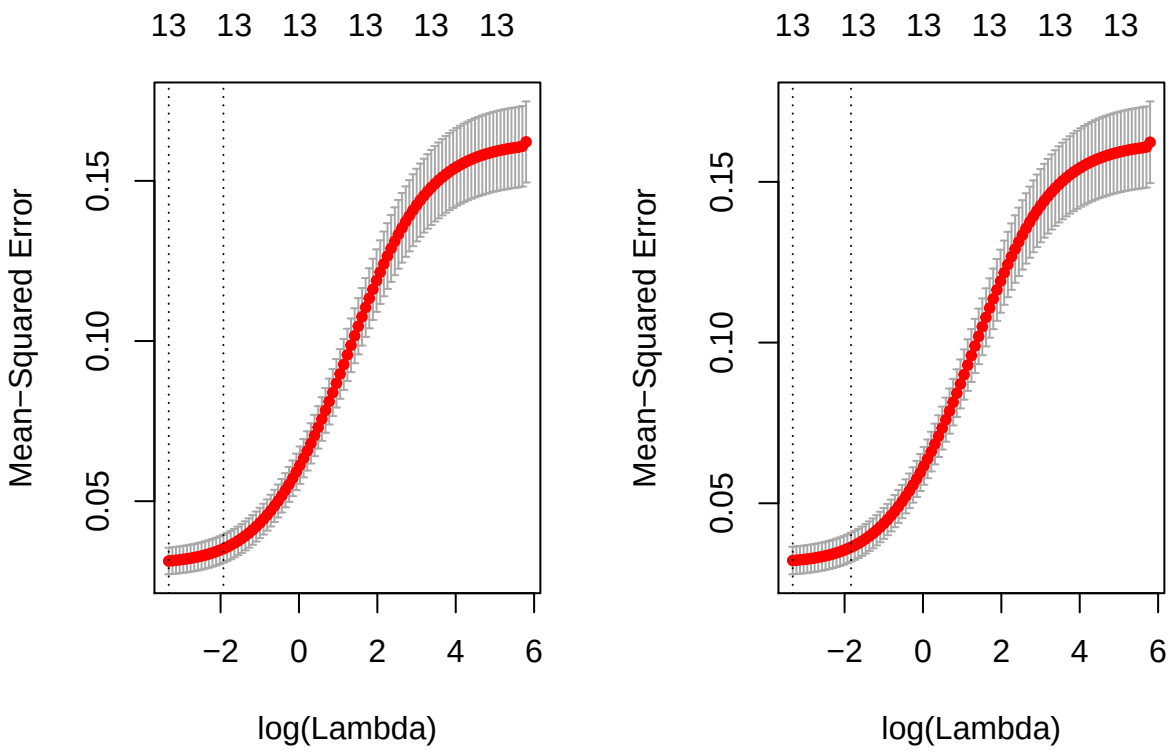
```
## Loaded glmnet 2.0-5
```

```
x <- as.matrix(datahousing.train[, -14])
y <- log(datahousing.train$medv)
ridge <- glmnet(x,y,alpha=0)

plot(ridge, xvar="lambda",label=TRUE)
```



Selon le graph, nous pouvons voir que quand log(lambda) augmente, les coefficients des variables baissent et deviennent zéro à la fin. Quand on utilise la régression de ridge, on utilise l'argument lambda pour contrôler la complexité du modèle. Si lambda est trop petit, le modèle séra possible sur-ajustement, si lambda est trop grand, on perdra trop d'information. Afin de trouver un bon lambda, maintenant on faire la validation croisée.

```
ridge.10cv <- cv.glmnet(x,y,nfolds=10, alpha=0, grouped=FALSE)
ridge.loo <- cv.glmnet(x,y,nfolds=n, alpha=0, grouped=FALSE)
par(mfrow=c(1,2))
plot(ridge.10cv)
plot(ridge.loo)
```

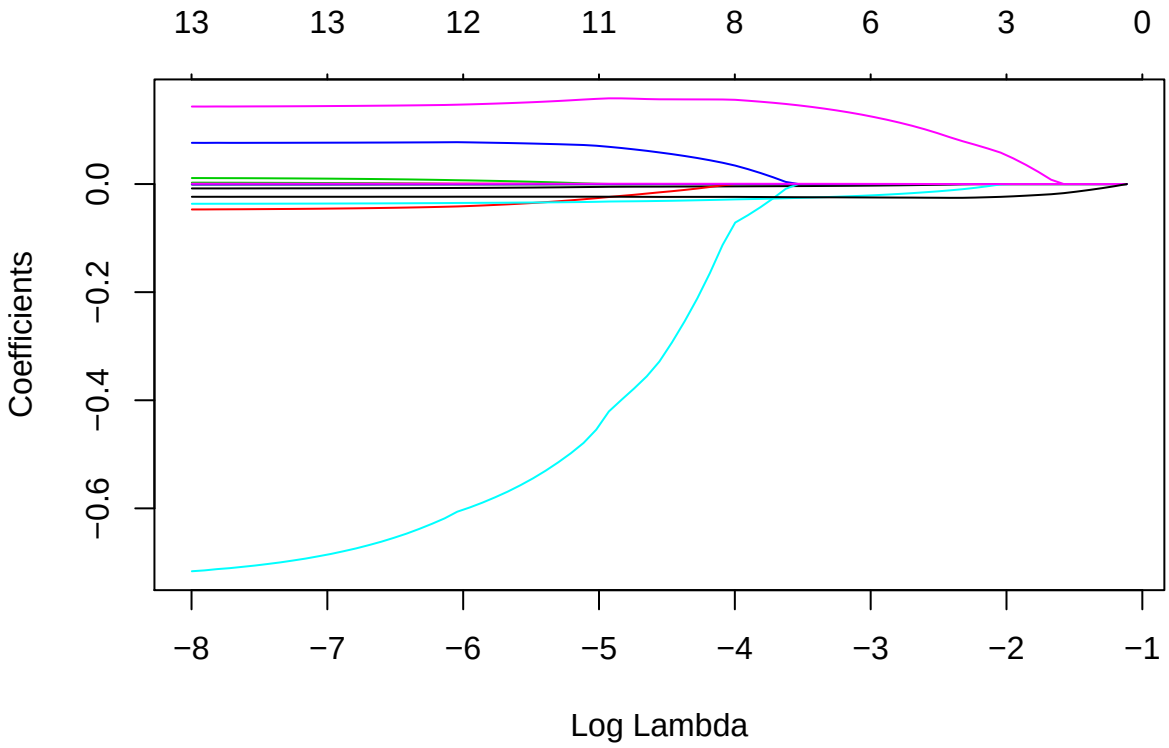


Le graph à gauche correspond à la validation croisée 10, Le graph à droit correspond à la validation croisée n. Dans chaque graph, il y a deux lignes droite. La ligne à gauche nous donne le lambda.min. Le lambda.min permet de minimiser le moyenne d'erreur de la validation croisée. L'autre ligne nous donne le lambda.1se, il nous donne le modèle le plus régularisé et l'erreur est inférieur qu'une valeur fixé. Quand on fait la validation croisée 10, le meilleur log(lambda) est -3,8, et quand on fait la validation croisée n, le meilleur log(lambda) est -3,8 aussi.

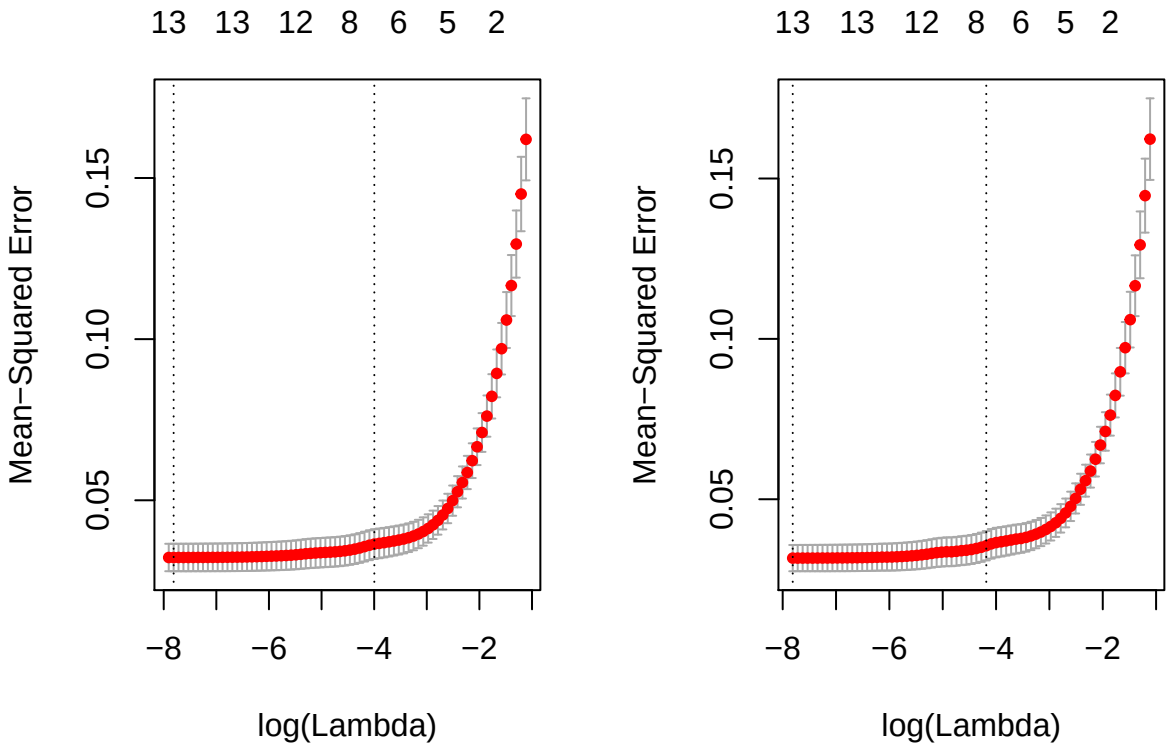
Régression de lasso

On fait la même chose avec la régression de lasso.

```
lasso <- glmnet(x, y, alpha = 1)
plot(lasso, xvar="lambda")
```




```
lasso.10cv <- cv.glmnet(x,y,nfolds=10, grouped=FALSE)
lasso.loo <- cv.glmnet(x,y,nfolds=n , grouped=FALSE)
par(mfrow=c(1,2))
plot(lasso.10cv)
plot(lasso.loo)
```



Nous voyons que quand log(lambda) augmente, le nombre de coefficient diminue (c'est la différence entre la régression de ridge et la régression de lasso). Et le bon log(lambda) est -8.

Evaluation des modèles

Pour comparer les modèles, nous pensons à calculer la racine de l'erreur quadratique moyenne(RMSE).

```
test_x <- as.matrix(datahousing.test[, -14])
test_y <- log(datahousing.test$medv)
```

Pour le modèle full

```
error_full <- test_y - predict(full, datahousing.test)
rmse <- function(error_full)
  sqrt(mean((error_full)^2))
rmse(error_full)
```

```
## [1] 0.2152921
```

Pour les modèles model.AIC et model.BIC

```
# model.AIC

error_AIC <- test_y - predict(model.AIC, datahousing.test)
rmse <- function(error_AIC)
  sqrt(mean((error_AIC)^2))
rmse(error_AIC)
```

```
## [1] 0.2150709
```

```
# model.BIC

error_BIC <- test_y - predict(model.BIC, datahousing.test)
rmse <- function(error_BIC)
  sqrt(mean((error_BIC)^2))
rmse(error_BIC)
```

```
## [1] 0.2140178
```

Pour les modèles trouvés par la méthode de pénalisation

```
mse_ridge_min <- mean((test_y - predict(ridge, newx=test_x, s=ridge.10cv$lambda.min))^2)
rmse_ridge_min <- sqrt(mse_ridge_min)

mse_ridge_1se <- mean((test_y - predict(ridge, newx=test_x, s=ridge.10cv$lambda.1se))^2)
rmse_ridge_1se <- sqrt(mse_ridge_1se)

mse_lasso_min <- mean((test_y - predict(lasso, newx=test_x, s=lasso.10cv$lambda.min))^2)
rmse_lasso_min <- sqrt(mse_lasso_min)

mse_lasso_1se <- mean((test_y - predict(lasso, newx=test_x, s=lasso.10cv$lambda.1se))^2)
rmse_lasso_1se <- sqrt(mse_lasso_1se)

rmse_ridge_min
```

```
## [1] 0.2369492
```

```
rmse_ridge_1se
```

```
## [1] 0.2452799
```

```
rmse_lasso_min
```

```
## [1] 0.2309212
```

```
rmse_lasso_1se
```

```
## [1] 0.2490969
```

Nous voyons que les modèles trouvés par la sélection de stepwise sont mieux que les modèles trouvés par la méthode de pénalisation car leurs RSMEs sont plus petits. Du coup, nous incluons que le modèle linéaire model.BIC est le meilleur modèle. Et le modèle est suivant:

medv = exp(3,87993 - 0,00998crim + 0.09718chas - 0.70712nox + 0.11788rm - 0.04067dis + 0.01209rad - 0.00051tax - 0.03978ptratio + 0.00041b - 0.02646lstat)

Analyse en composantes principales

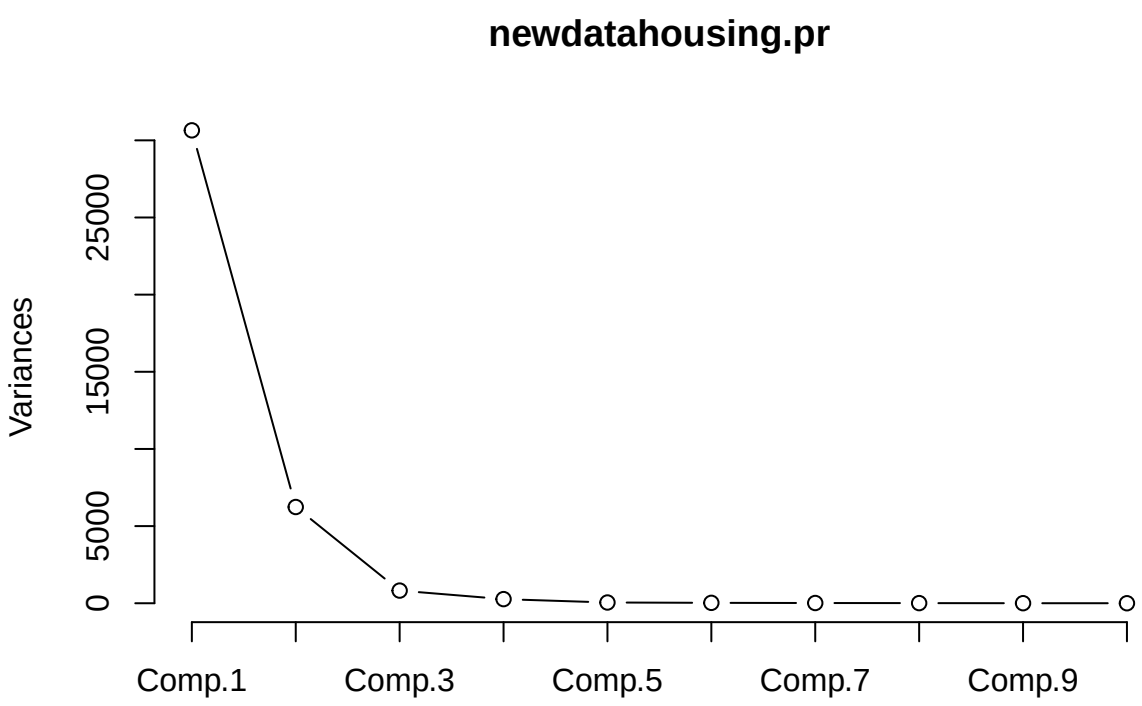
Nous utilisons la matrice de covariance des données datahousing (avec des variables actives) pour faire PCA. Et nous obtenons les caractéristiques principales et les projections des variables sur eux.

```
newdatahousing <- datahousing[,c(-14)]
newdatahousing.pr<-princomp(newdatahousing,cor=FALSE) #analyse par la matrix de covariance
summary(newdatahousing.pr,loadings=TRUE)
```

```
## Importance of components:
##                            Comp.1     Comp.2      Comp.3       Comp.4
```

```
## Standard deviation     175.0619977 78.9756829 28.66077637 16.359253212
## Proportion of Variance   0.8047924  0.1637898  0.02157128  0.007027924
## Cumulative Proportion    0.8047924  0.9685822  0.99015352  0.997181441
##                             Comp.5       Comp.6       Comp.7       Comp.8
## Standard deviation     7.050655513 5.2192123470 4.0128071928 3.0839774427
## Proportion of Variance 0.001305447 0.0007153359 0.0004228601 0.0002497598
## Cumulative Proportion  0.998486887 0.9992022231 0.9996250832 0.9998748430
##                              Comp.9     Comp.10      Comp.11      Comp.12
## Standard deviation     1.811955e+00 1.088148757 4.883060e-01 2.393226e-01
## Proportion of Variance 8.621738e-05 0.000031094 6.261585e-06 1.504069e-06
## Cumulative Proportion  9.999611e-01 0.999992154 9.999984e-01 9.999999e-01
##                             Comp.13
## Standard deviation     5.517698e-02
## Proportion of Variance 7.994954e-08
## Cumulative Proportion  1.000000e+00
##
## Loadings:
##         Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8 Comp.9
## crim                                  0.956 -0.147  0.216  0.125
## zn                     0.633 -0.768
## indus                                        0.357  0.779 -0.491
## chas
## nox
## rm
## age                   -0.756 -0.637        -0.111
## dis                                                             -0.225
## rad                                   0.178 -0.202 -0.435 -0.854
## tax      0.948 -0.297
## ptratio                                                         -0.966
## b       -0.296 -0.955
## lstat                                 0.216  0.888 -0.376
##         Comp.10 Comp.11 Comp.12 Comp.13
## crim
## zn
## indus
## chas                     -1.000
## nox                              -1.000
## rm                0.994
## age
## dis      -0.965
## rad
## tax
## ptratio  0.231
## b
## lstat
```

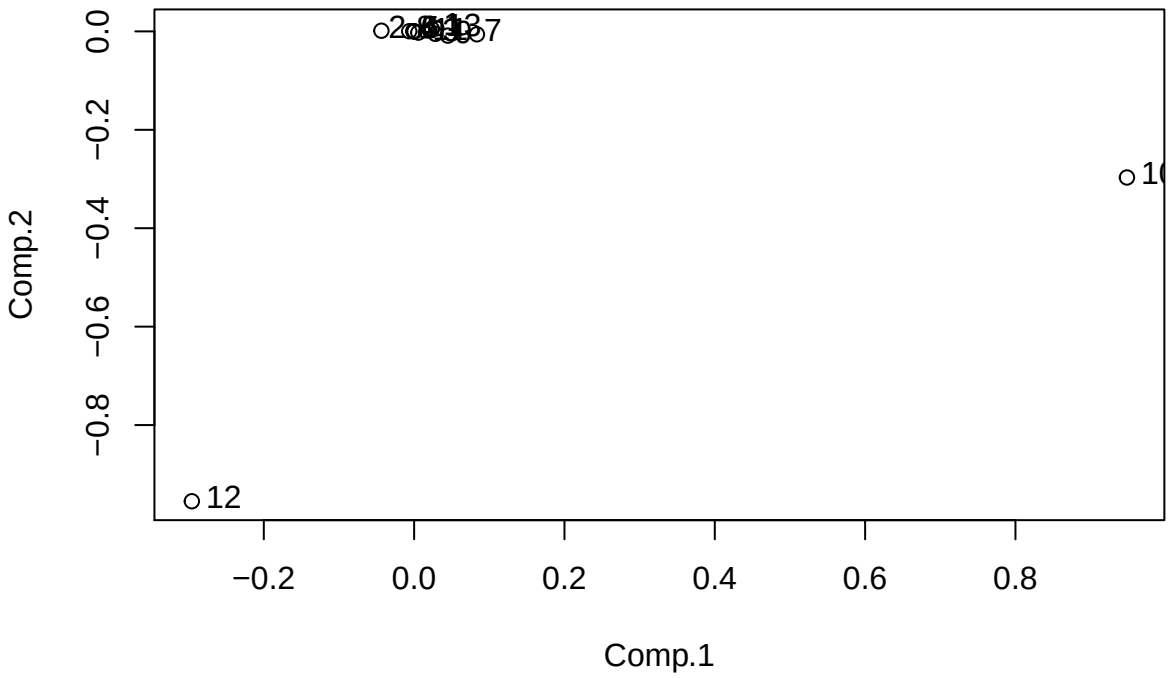
```
screeplot(newdatahousing.pr,type="lines")
```



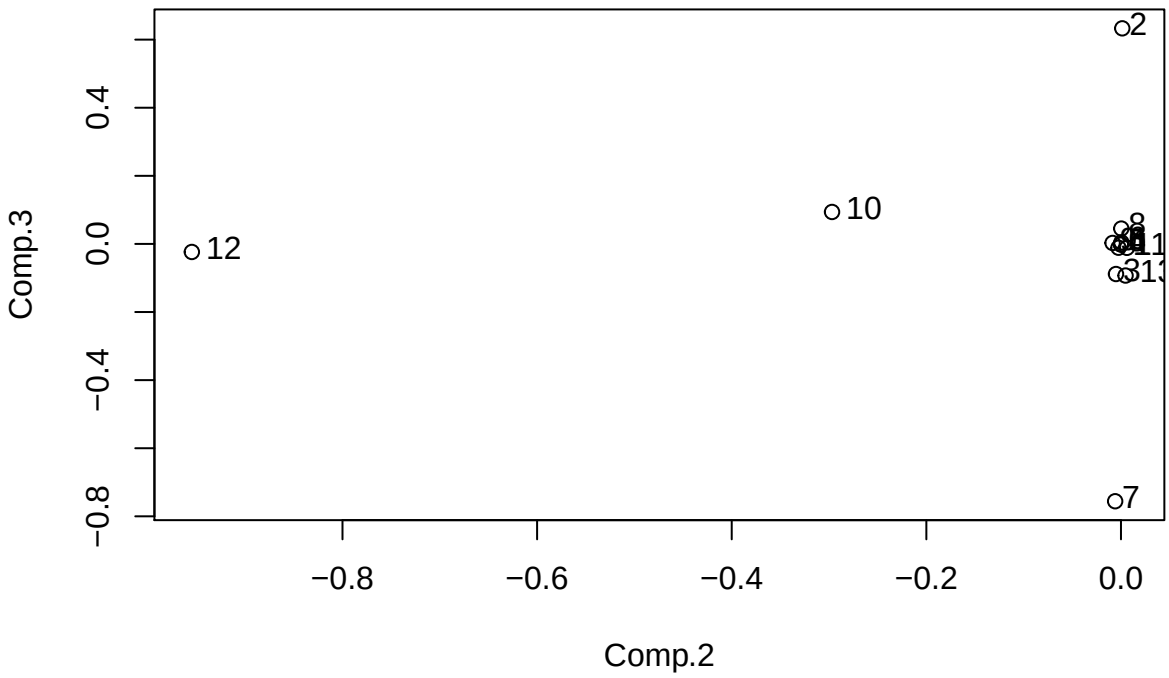
Selon le graph, nous choisissons les 3 premières composantes comme les composantes principales, parce que leur contribution d'accumulation arrive déjà 99%.

Nous traçons les variables par leur projections sur les composantes principales.

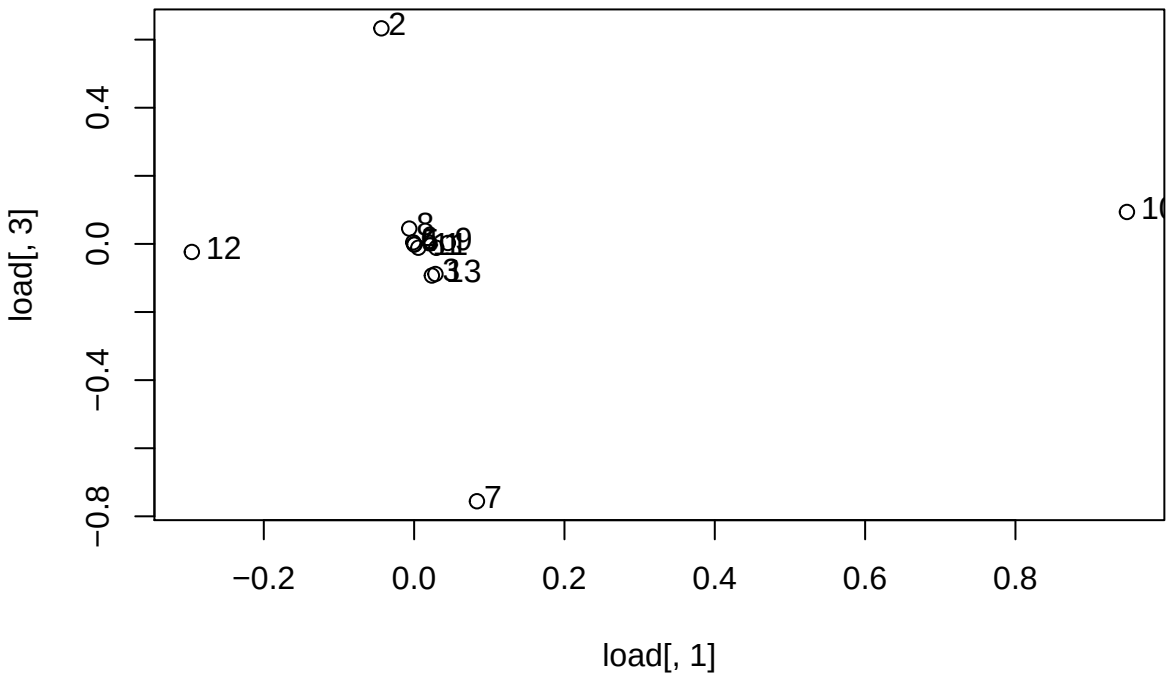
```
load<-loadings(newdatahousing.pr)
plot(load[,1:2])
text(load[,1],load[,2],adj=c(-0.4,0.3))
```




```
plot(load[,2:3])
text(load[,2],load[,3],adj=c(-0.4,0.3))
```




```
plot(load[,1],load[,3])
text(load[,1],load[,3],adj=c(-0.4,0.3))
```



Conclusion

Nous analysons les valeurs des maisons dans la banlieue de Boston avec différents méthodes et nous concluons que le valeur des maisons est principalement influencé par 8 facteurs:

- chas: si il y a rivière, la valeur des maisons est plus haut.
- influence positive(c-a-d: quand il est plus haut, la valeur des maisons est plus haut):

rm: average number of rooms per dwelling

rad: index of accessibility to radial highways

- influence negative(c-a-d: quand il est plus haut, la valeur des maisons est plus bas):

crim: per capita crime rate by town

nox: nitric oxides concentration (parts per 10 million)

dis: weighted distances to five Boston employment centres

ptratio: pupil-teacher ratio by town

lstat: % lower status of the population

Référence

http://mlr.cs.umass.edu/ml/datasets/Housing