

Projet Stats

Analyse des données

Statistiques descriptives

Le jeu de données est composé de 768 batiments. Pour chaque bâtiment, nous possédons les 10 variables suivantes :

C'est normal que la liste soit comme ça, c'est latex qui fait des listes correctes après. * Quantitatives : + `Relative.compactness` : compacité relative à un cube de même volume ($RC = 6 \times V^{0.66} \times A^{-1}$ avec V et A respectivement le volume et l'aire totale du bâtiment (murs, toit, sol)) + `Surface.area` : surface totale (sol + murs + toit) + `Wall.area` : surface de mur extérieurs + `Roof.area` : surface de toit + `Overall.height` : hauteur du bâtiment (soit 3.5 soit 7 m) + `Glazing.area` : Surface vitrée (en pourcentage de la surface au sol) + `Load` : Énergie consommée (quantité à prédire) * Catégorielles : + `orientation` : orientation de la maison (Nord, Sud, Est, Ouest) + `Glazing.area.distr` : Distribution des vitres (uniforme, ou plus d'un côté) + `Energy.efficiency` : Indicateur de l'énergie consommées (de A (meilleur) à G (pire))

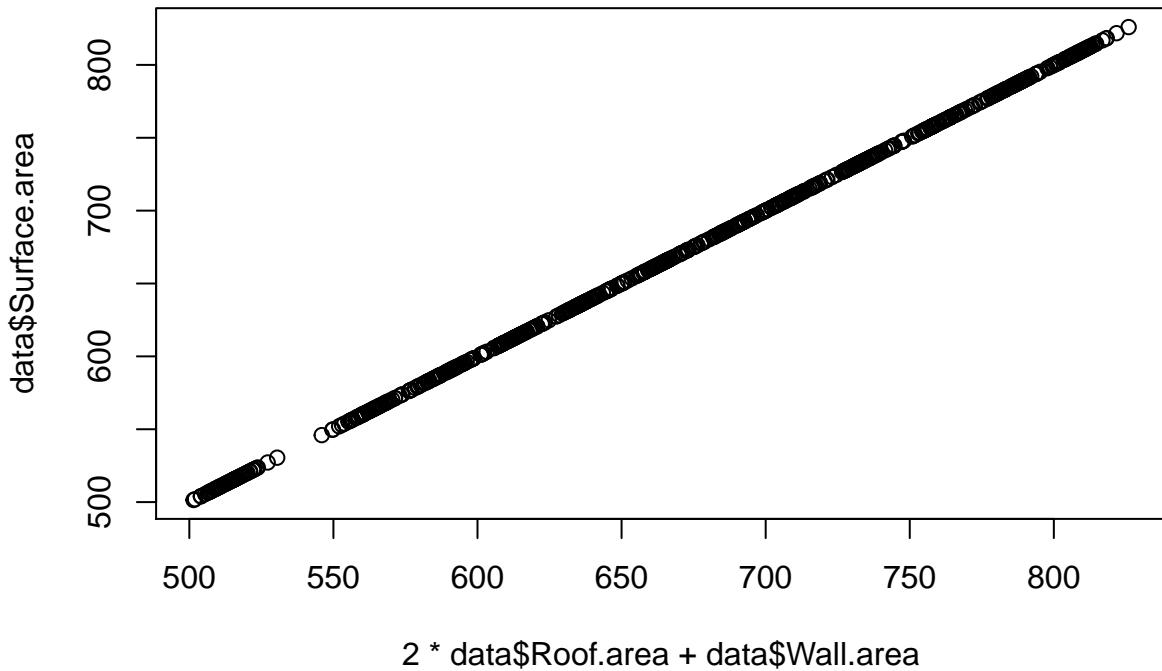
```
# Lecture jeu de données
setwd("/home/leo/GoogleDrive/Cours/INSA_4A/Projet")
data = read.table("DataEnergy-Student.csv", header = TRUE, sep = ",")
# Mise sous forme de facteur des données catégorielles
data$Energy.efficiency = as.factor(data$Energy.efficiency)
data$Glazing.area.distr = as.factor(data$Glazing.area.distr)
data$orientation = as.factor(data$orientation)
# Correction du jeu de données
data$Glazing.area[which(data$Glazing.area.distr == 0)] = 0
# Informations sur le jeu de données
summary(data)
# Séparation des variables quantitatives et qualitatives
quanti = c(1:5, 7, 9)
quali = c(6, 8, 10)

##   Relative.compactness   Surface.area      Wall.area      Roof.area
## Min.    :0.6125       Min.   :501.4       Min.   :234.3       Min.   :105.3
## 1st Qu.:0.6779       1st Qu.:598.7       1st Qu.:291.8       1st Qu.:137.4
## Median :0.7517       Median :673.1       Median :315.8       Median :183.3
## Mean    :0.7645       Mean    :671.3       Mean    :318.3       Mean    :176.5
## 3rd Qu.:0.8350       3rd Qu.:744.6       3rd Qu.:343.0       3rd Qu.:220.5
## Max.    :0.9912       Max.    :826.0       Max.    :425.8       Max.    :225.8
##
##   Overall.height orientation Glazing.area Glazing.area.distr Energy
## Min.    :3.50   East     :192   Min.   :0.0000  0: 48           Min.   :10.21
## 1st Qu.:3.50   North   :192   1st Qu.:0.1031  1:144           1st Qu.:29.36
## Median :5.25   South   :192   Median :0.2475  2:144           Median :41.76
## Mean    :5.25   West    :192   Mean    :0.2344  3:144           Mean    :46.92
## 3rd Qu.:7.00   North   :192   3rd Qu.:0.3912  4:144           3rd Qu.:64.33
## Max.    :7.00   South   :192   Max.    :0.4270  5:144           Max.    :94.84
##
##   Energy.efficiency
```

```

##  A:208
##  B:109
##  C: 80
##  D: 79
##  E:109
##  F:102
##  G: 81
# pairs(data[, quanti], pch = '.', cex = 0.1, cex.labels = 0.55)
plot(2*data$Roof.area + data$Wall.area, data$Surface.area)

```



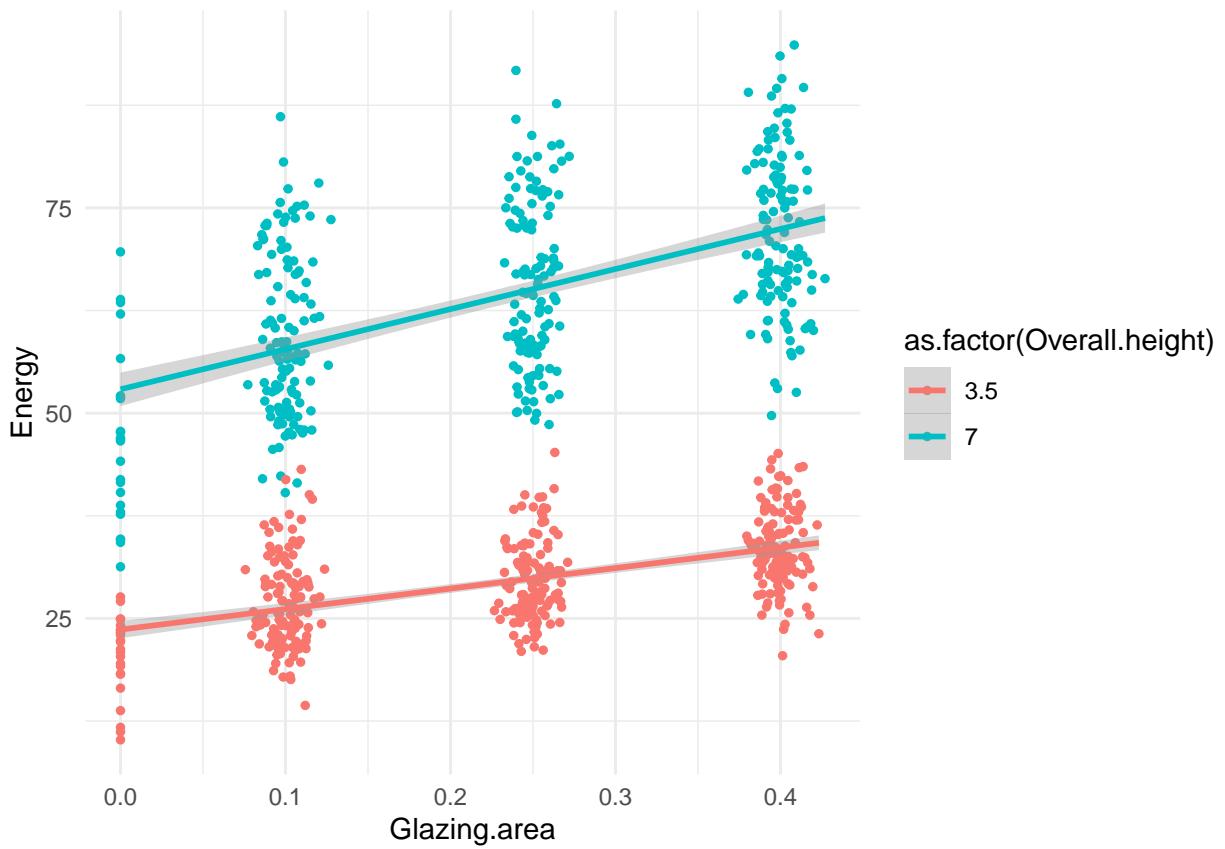
Même nombre de batiments faisant face à chaque orientation. Même nombre de batiments avec les différentes répartitions de fenêtres, sauf les batiments sans fenêtres.

```

ggplot(data) +
  aes(x = Glazing.area, y = Energy, colour = as.factor(Overall.height)) +
  geom_point(size = 1L) +
  theme_minimal() + geom_smooth(method = "lm")

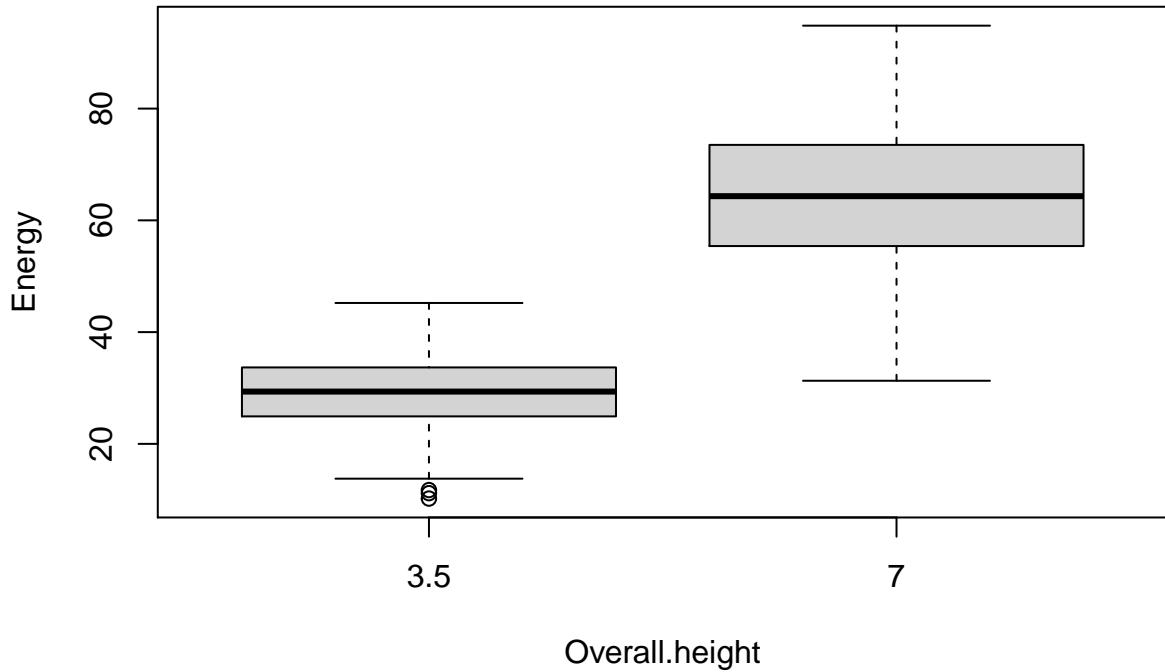
```

```
## `geom_smooth()` using formula 'y ~ x'
```



On voit que la surface vitrée à une influence sur l'énergie consommée.

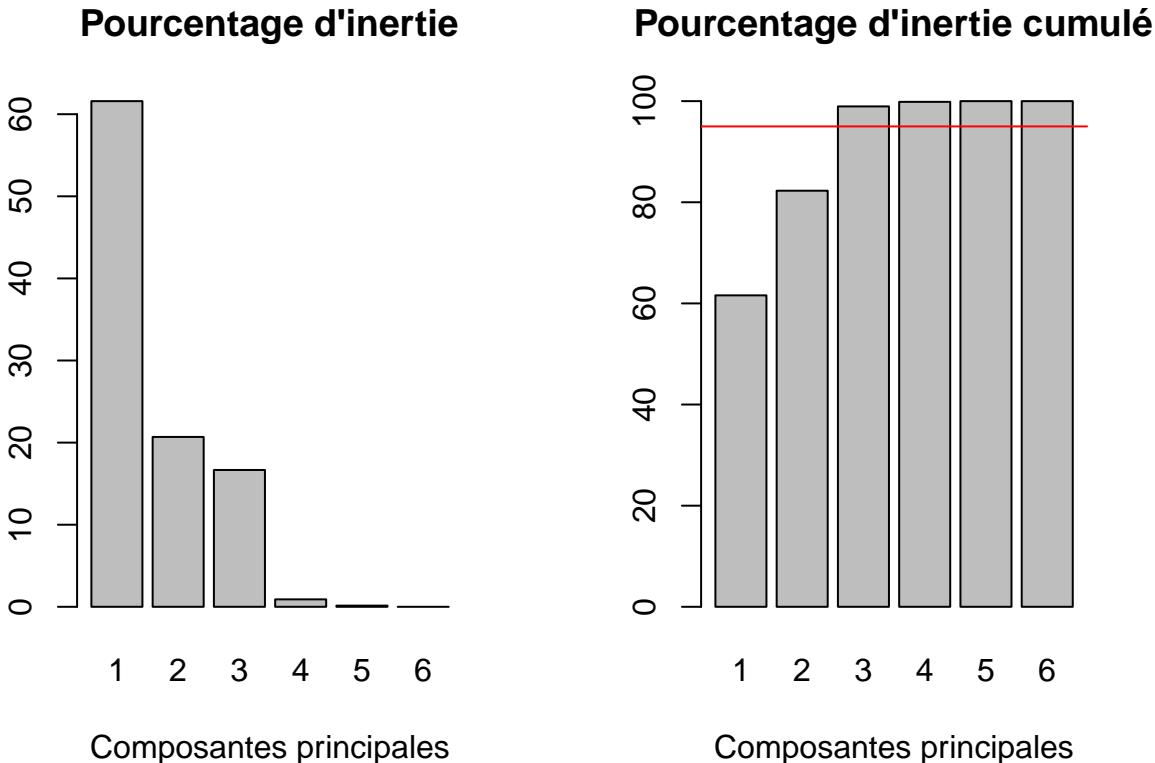
```
boxplot(Energy ~ Overall.height, data = data)
```



Analyse en Composantes Principales (ACP)

On souhaite réaliser une ACP afin de mener une analyse en dimension plus faible. Nous allons pour cela utiliser le package FactoMineR qui contient toutes les fonctions nécessaires à la réalisation de notre ACP. On cherche d'abord combien d'axes principaux nous allons avoir besoin.

```
library("FactoMineR")
par(mfrow=c(1,2))
res.acp <- PCA(data,scale.unit=T,quali.sup=c(6,8,10),quanti.sup=9,ncp=8, graph=F)
barplot(res.acp$eig[, "percentage of variance"], main="Pourcentage d'inertie",
        names.arg = seq(1,6), xlab = "Composantes principales")
barplot(res.acp$eig[, "cumulative percentage of variance"],
        main="Pourcentage d'inertie cumulé",
        names.arg = seq(1,6), xlab = "Composantes principales")
abline(h = 95, col = "red")
```



```
print(paste("Pourcentage d'inertie expliquée par les trois premiers axes :",
            res.acp$eig[, "cumulative percentage of variance"][3]))
```

```
## [1] "Pourcentage d'inertie expliquée par les trois premiers axes : 98.951976800162"
```

On voit sur le premier graphe que l'inertie portée par les trois premiers axes principaux est prépondérante par rapport aux autres. D'après le graphe de pourcentage d'inertie cumulée, ils expliquent presque 99% de l'inertie. Les deux premiers axes ne portent eux que 82% de l'inertie. Ne choisir que deux axes effacerait trop d'informations. Nous allons donc poursuivre notre analyse sur les trois premiers axes principaux.

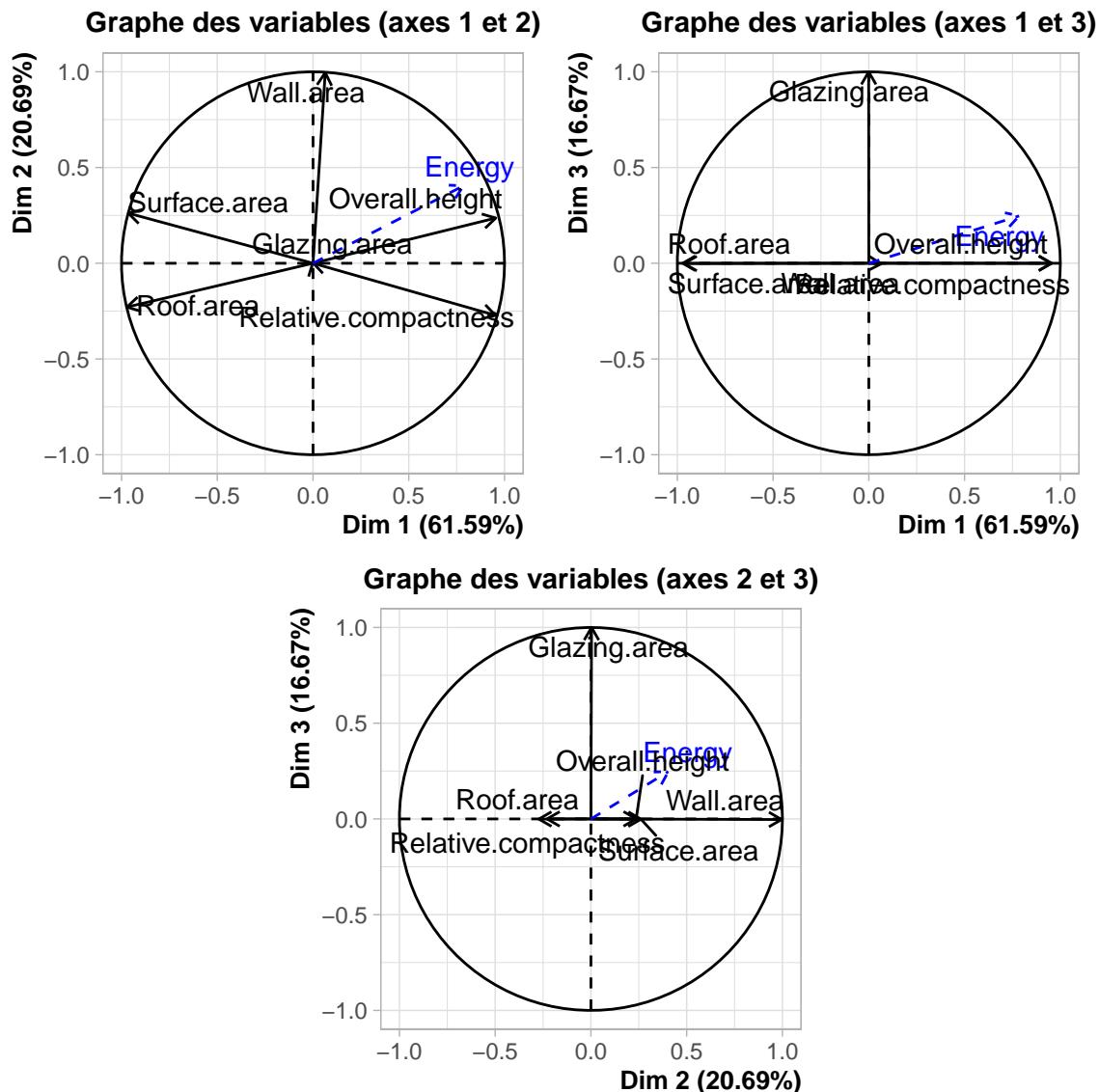
Les graphes des variables nous donnent des informations très intéressantes sur les variables corrélées.

```
gg1 = plot.PCA(res.acp, choix="var", axes = c(1,2), new.plot = FALSE,
                title = "Graphe des variables (axes 1 et 2)")
gg2 = plot.PCA(res.acp, choix="var", axes = c(1,3), new.plot = FALSE,
                title = "Graphe des variables (axes 1 et 3)")
```

```

gg3 = plot.PCA(res.acp, choix="var", axes = c(2,3), new.plot = FALSE,
               title = "Graphe des variables (axes 2 et 3)")
layout_matrix <- matrix(c(1, 1, 2, 2, 4, 3, 3, 4), nrow = 2, byrow = TRUE)
grid.arrange(gg1, gg2, gg3, layout_matrix = layout_matrix)

```



Les variables `Relative.compactness`, `Overall.height`, `Surface.area` et `Roof.area` semblent être plutôt portées par le premier axe principal (en positif pour les deux premières, négatif pour les deux autres).

Le second axe porte principalement la variable `Wall.area`, et le troisième la variable `Glazing.area`.

Finalement, les deux premiers axes ont plutôt trait à la forme du bâtiment (surface au sol pour le premier et surface murée pour le second), tandis que le troisième axe correspond à la surface vitrée.

Ainsi, l'énergie dépensée dépend d'abord de la forme du bâtiment, et ensuite de la surface vitrée.

Nous pouvons également observer les graphes des individus. Nous représentons les différentes classes énergétiques par couleur.

```

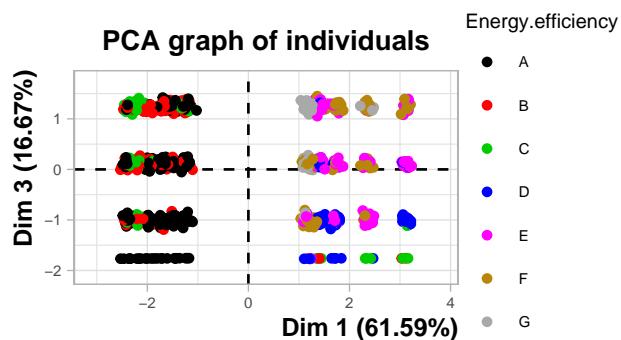
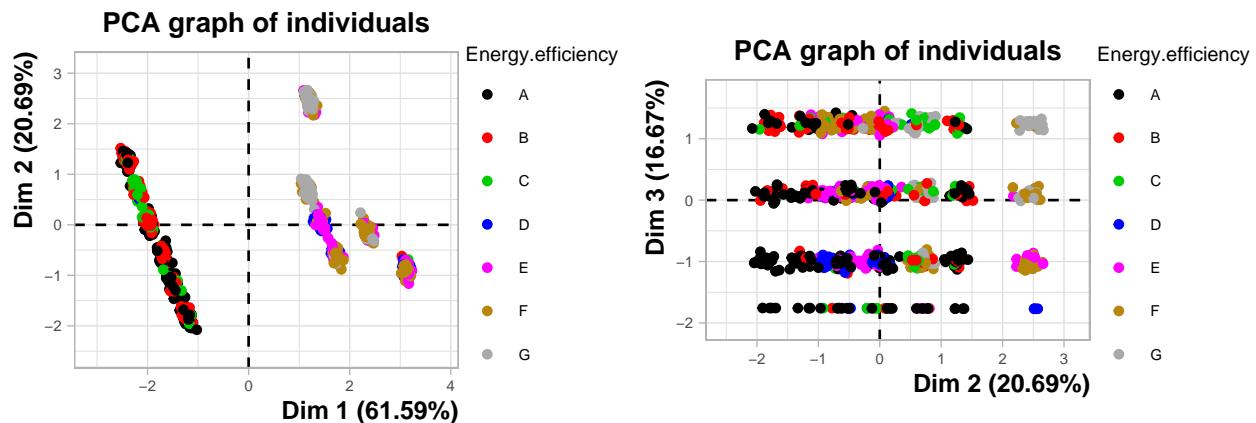
gg1 = plot(res.acp, choix="ind", axes = c(1,2), invisible="quali",
           habillage="Energy.efficiency", label = "none", new.plot = FALSE) +

```

```

theme(text = element_text(size=8))
gg2 = plot(res.acp, choix="ind", axes = c(2,3), invisible="quali",
           habillage="Energy.efficiency", label = "none", new.plot = FALSE) +
  theme(text = element_text(size=8))
gg3 = plot(res.acp, choix="ind", axes = c(1,3), invisible="quali",
           habillage="Energy.efficiency", label = "none", new.plot = FALSE) +
  theme(text = element_text(size=8))
grid.arrange(gg1,gg2,gg3,layout_matrix = layout_matrix)

```



On peut voir sur le premier et le troisième graphique que l'axe 1 marque une claire séparation entre les classes A, B, C et les classes D, E, F, G.

Les maisons dont la consommation énergétique la plus faible possèdent des coordonnées négatives pour la première composante. Ces maisons sont celles qui ont une compacité relative plus faible (plus basses et avec une surface au sol plus élevée), comme on peut le voir sur les graphiques suivants :

```

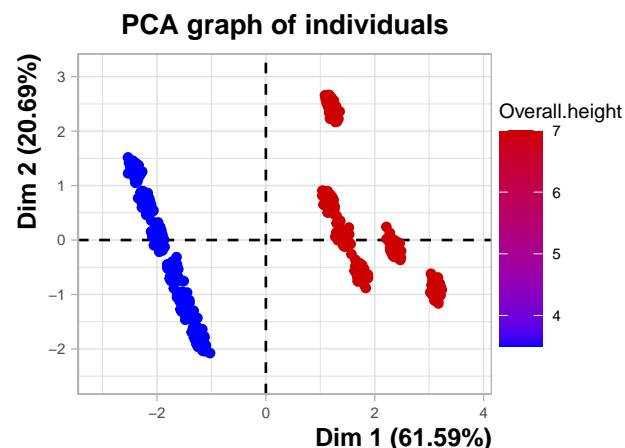
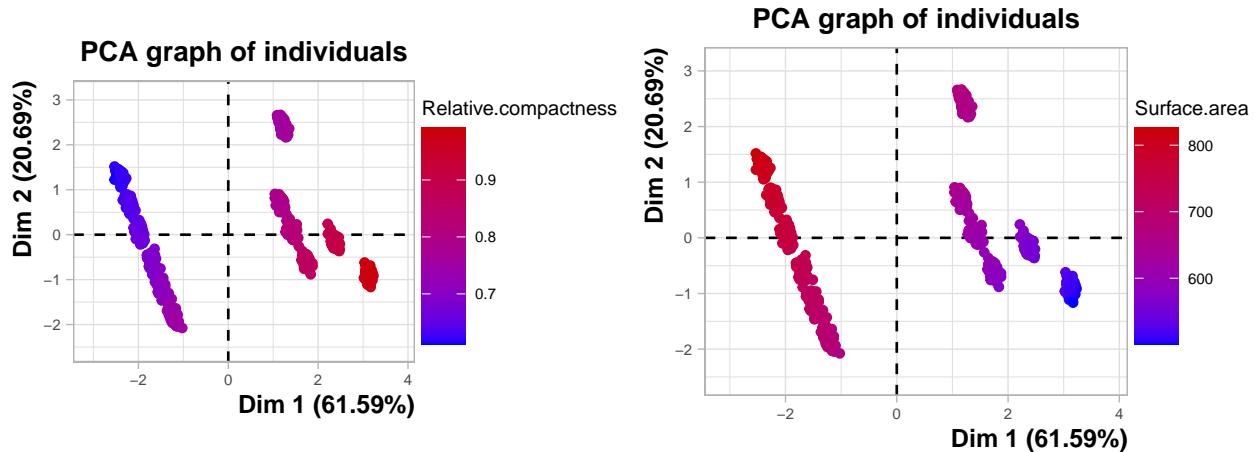
gg1 = plot(res.acp, choix="ind", axes = c(1,2), invisible="quali",
           habillage="Relative.compactness", label = "none", new.plot = FALSE) +
  theme(text = element_text(size=8))
gg2 = plot(res.acp, choix="ind", axes = c(1,2), invisible="quali",
           habillage="Surface.area", label = "none", new.plot = FALSE) +
  theme(text = element_text(size=8))

```

```

gg3 = plot(res.acp, choix="ind", axes = c(1,2), invisible="quali",
           habillage="Overall.height", label = "none") +
  theme(text = element_text(size=8))
grid.arrange(gg1,gg2,gg3,layout_matrix = layout_matrix)

```



Ainsi, les maisons plus étalées sur le sol possèdent les meilleures performances énergétiques.

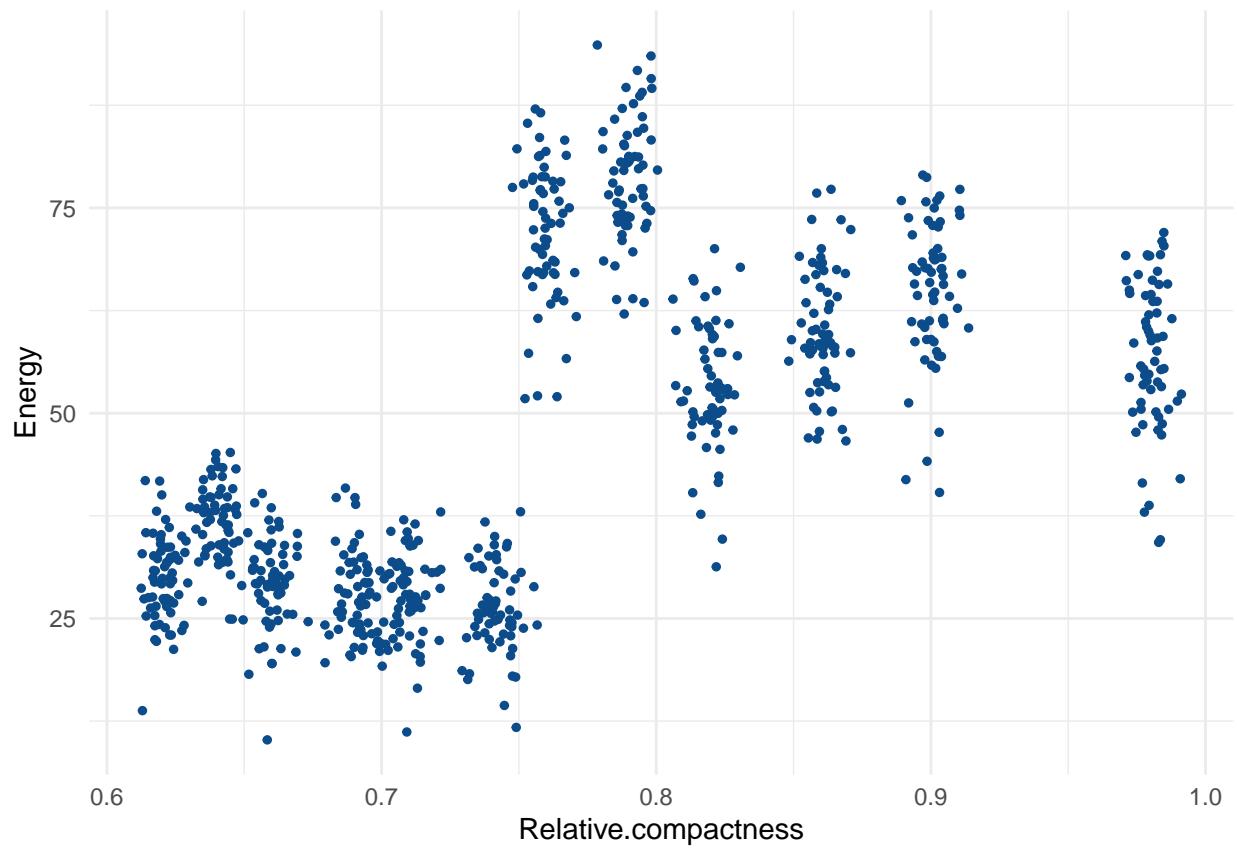
Clustering de variables

Analyse visuelle

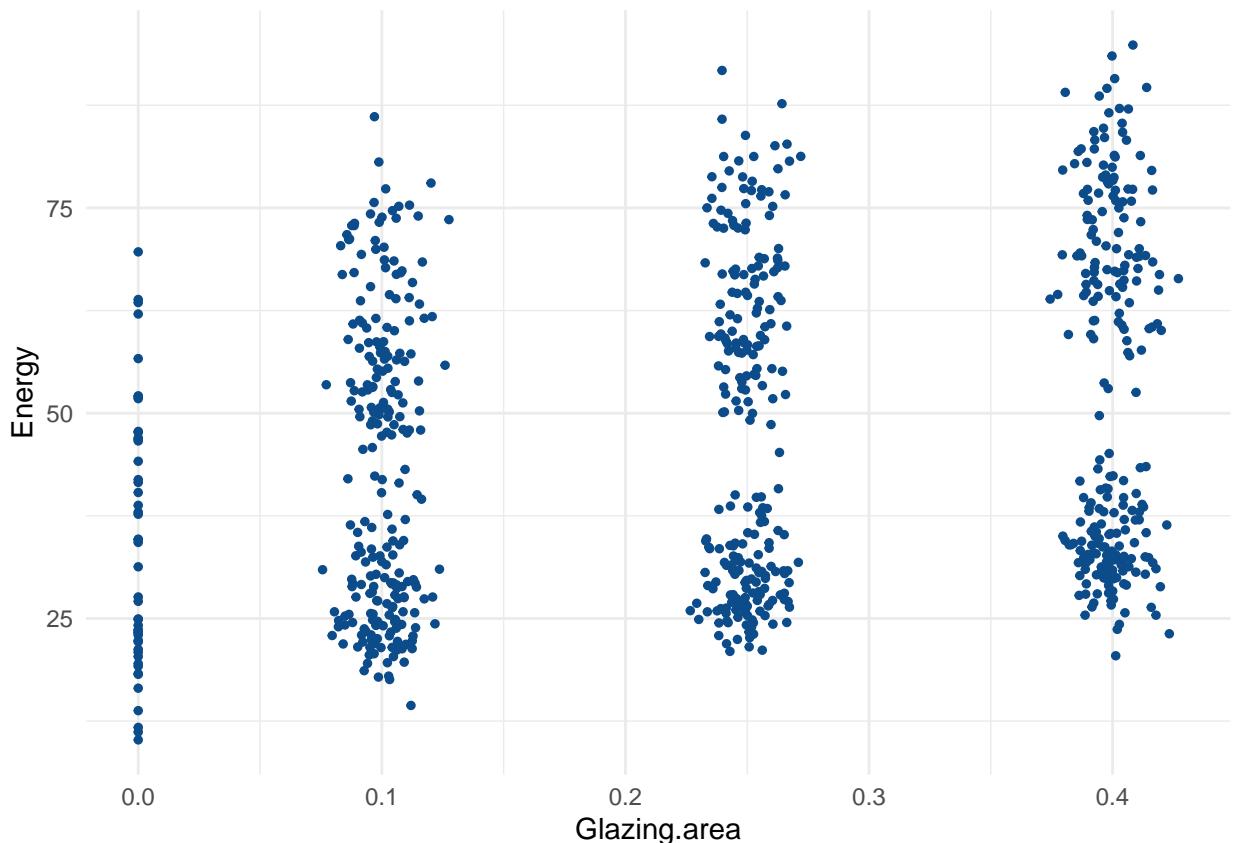
```

ggplot(data) +
  aes(x = Relative.compactness, y = Energy) +
  geom_point(size = 1L, colour = "#0c4c8a") +
  theme_minimal()

```



```
ggplot(data) +
  aes(x = Glazing.area, y = Energy) +
  geom_point(size = 1L, colour = "#0c4c8a") +
  theme_minimal()
```



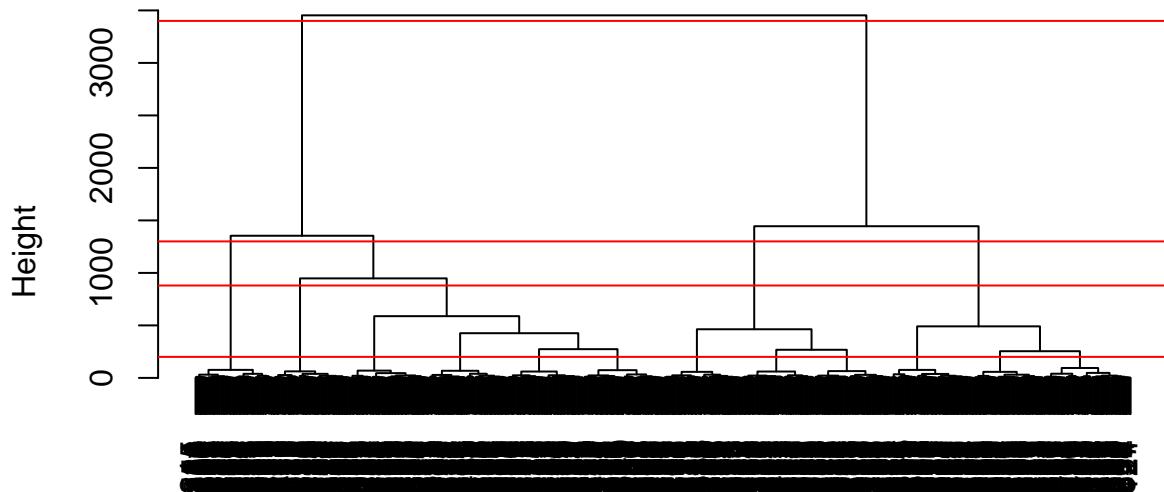
Visuellement, c'est selon la compacité relative et la surface vitrée que le distingue le mieux des groupes. Selon la compacité relative, on constate principalement deux groupes. On peut également voir 12 groupes mais ceux-ci sont moins évidents. Selon la surface vitrée, on distingue principalement 4 groupes.

Clustering hiérarchique

Après cette analyse visuelle, nous allons utiliser des méthodes de clustering pour confirmer ou infirmer nos observations. Commençons par le clustering hiérarchique.

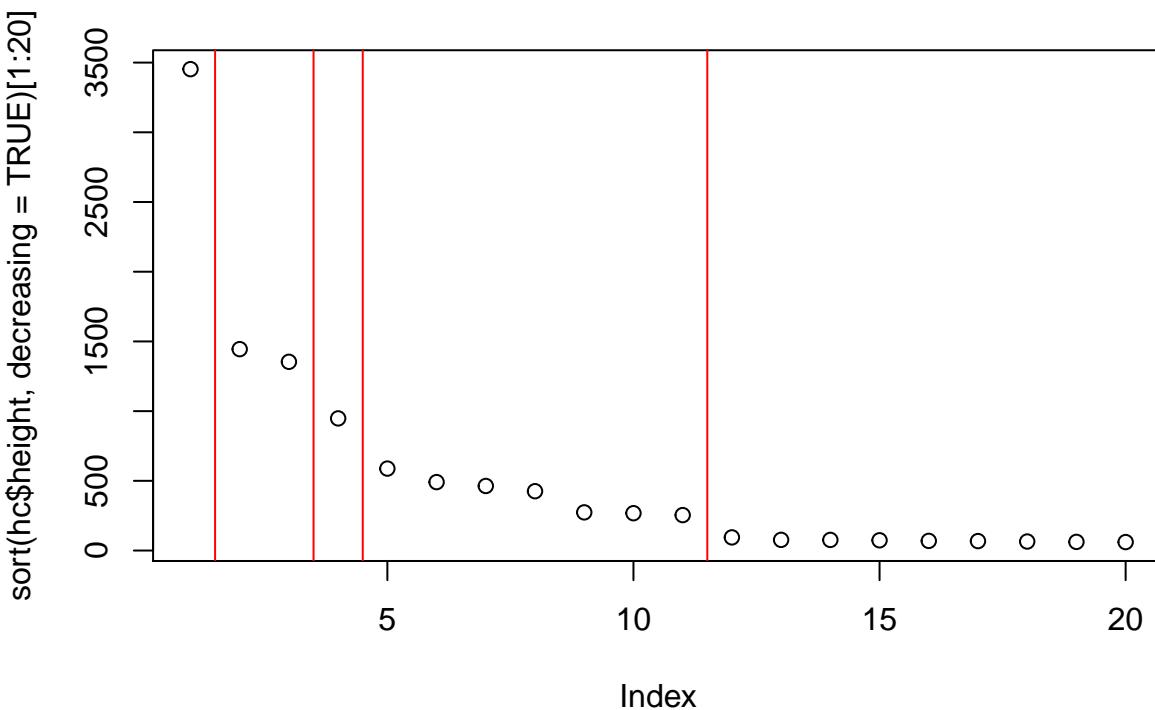
```
hc = hclust(dist(data[c(1,2,3,4,5,7)]), method = "ward.D2")
plot(hc)
abline(h=3400,col="red")
abline(h=1300,col="red")
abline(h=880,col="red")
abline(h=200,col="red")
```

Cluster Dendrogram



```
dist(data[c(1, 2, 3, 4, 5, 7)])
hclust (*, "ward.D2")
```

```
plot(sort(hc$height, decreasing = TRUE)[1:20])
abline(v=1.5,col="red")
abline(v=3.5,col="red")
abline(v=4.5,col="red")
abline(v=11.5,col="red")
```

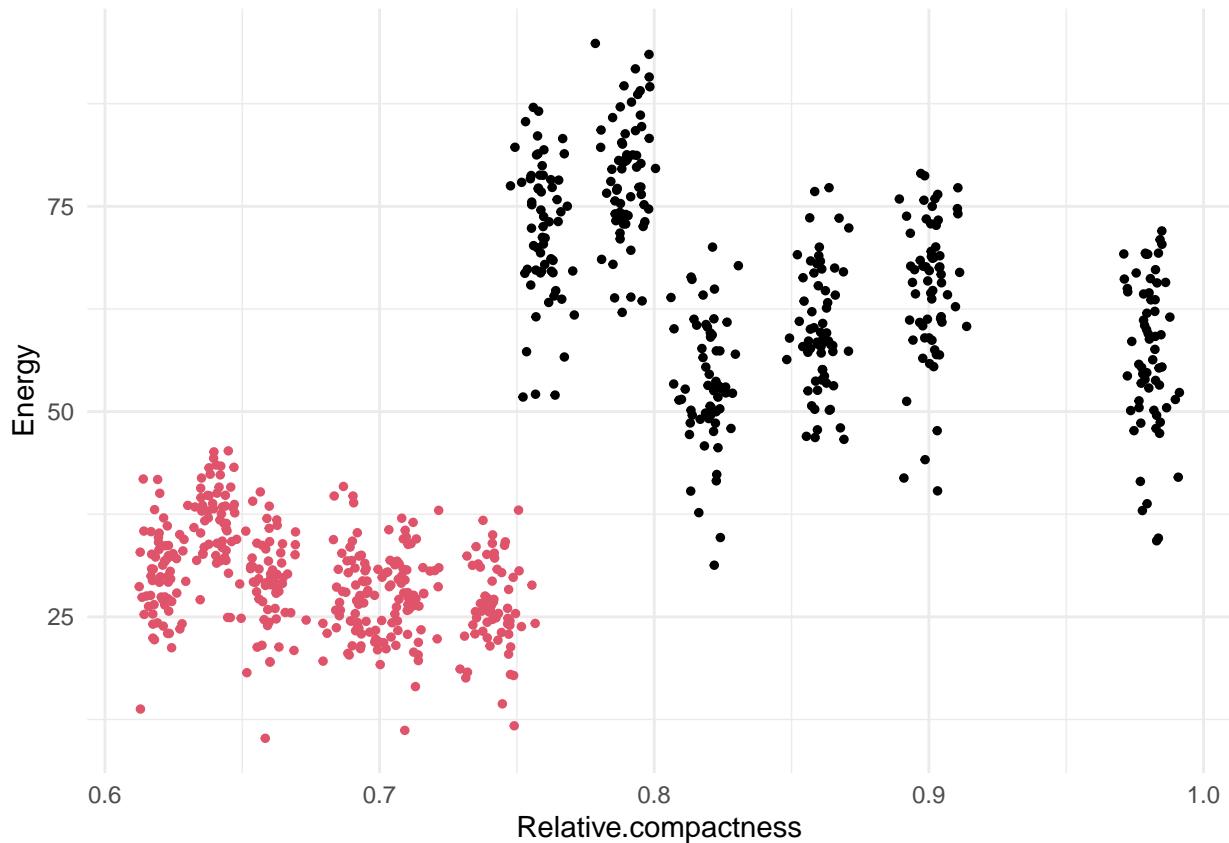


Sur le dendrogramme, deux classes apparaissent clairement. On peut aussi distinguer 4,5 ou 12 classes. Sur le graphique de la variance inter-classe, on observe un saut important au passage à 2 classes. On retrouve également les résultats que l'on a obtenu avec le dendrogramme en observant des sauts aux passages à 4, 5 et 12 classes.

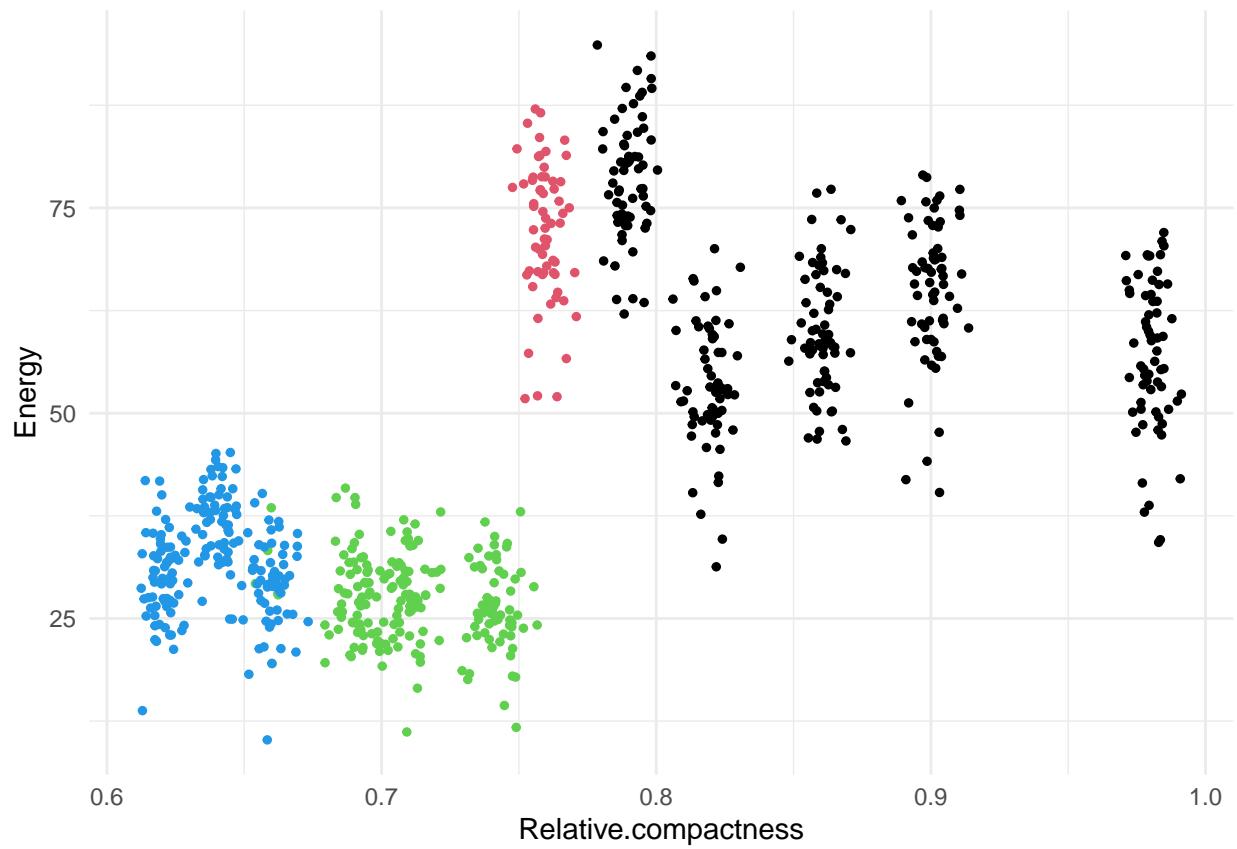
Ces premiers outils semblent confirmer les nombres de classes qui semblent pertinents : 2, 4 et 12.

On réalise donc dans un premier temps un découpage en 2, 4 et 12 classes selon le clustering hiérarchique. On compare ces résultats avec nos observations selon plusieurs variables explicatives.

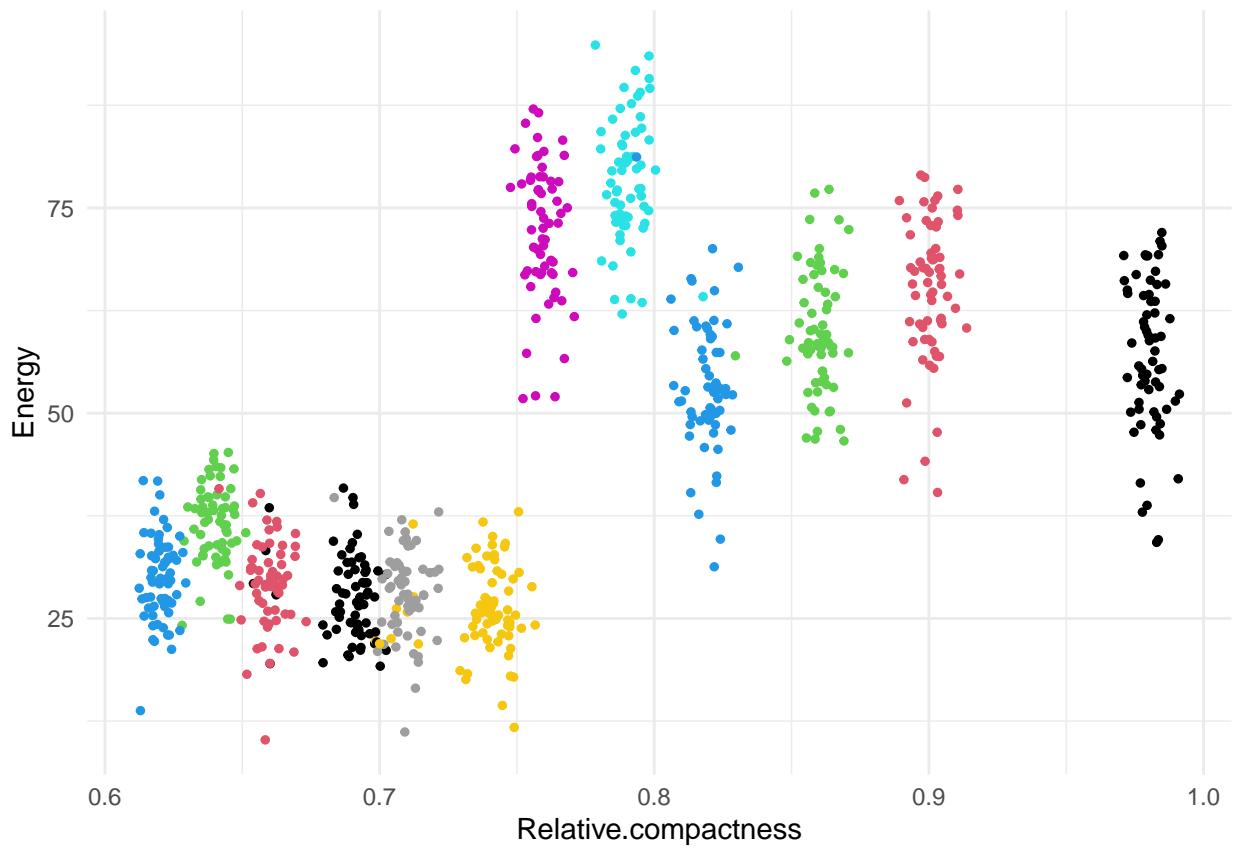
```
class2 = cutree(hc, k = 2)
ggplot(data) +
  aes(x = Relative.compactness, y = Energy) +
  geom_point(size = 1L, colour = class2) +
  theme_minimal()
```



```
class4 = cutree(hc, k = 4)
ggplot(data) +
  aes(x = Relative.compactness, y = Energy) +
  geom_point(size = 1L, colour = class4) +
  theme_minimal()
```

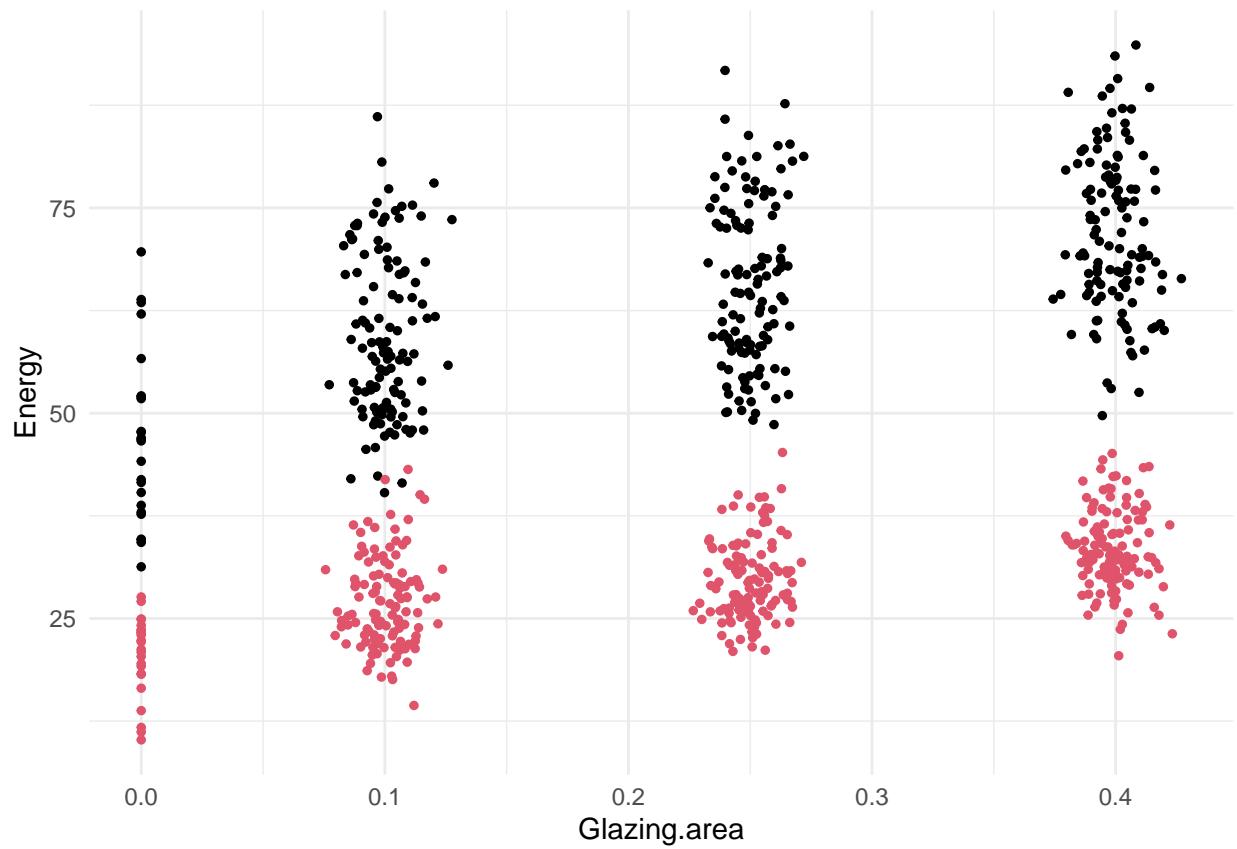


```
class12 = cutree(hc, k = 12)
ggplot(data) +
  aes(x = Relative.compactness, y = Energy) +
  geom_point(size = 1L, colour = class12) +
  theme_minimal()
```

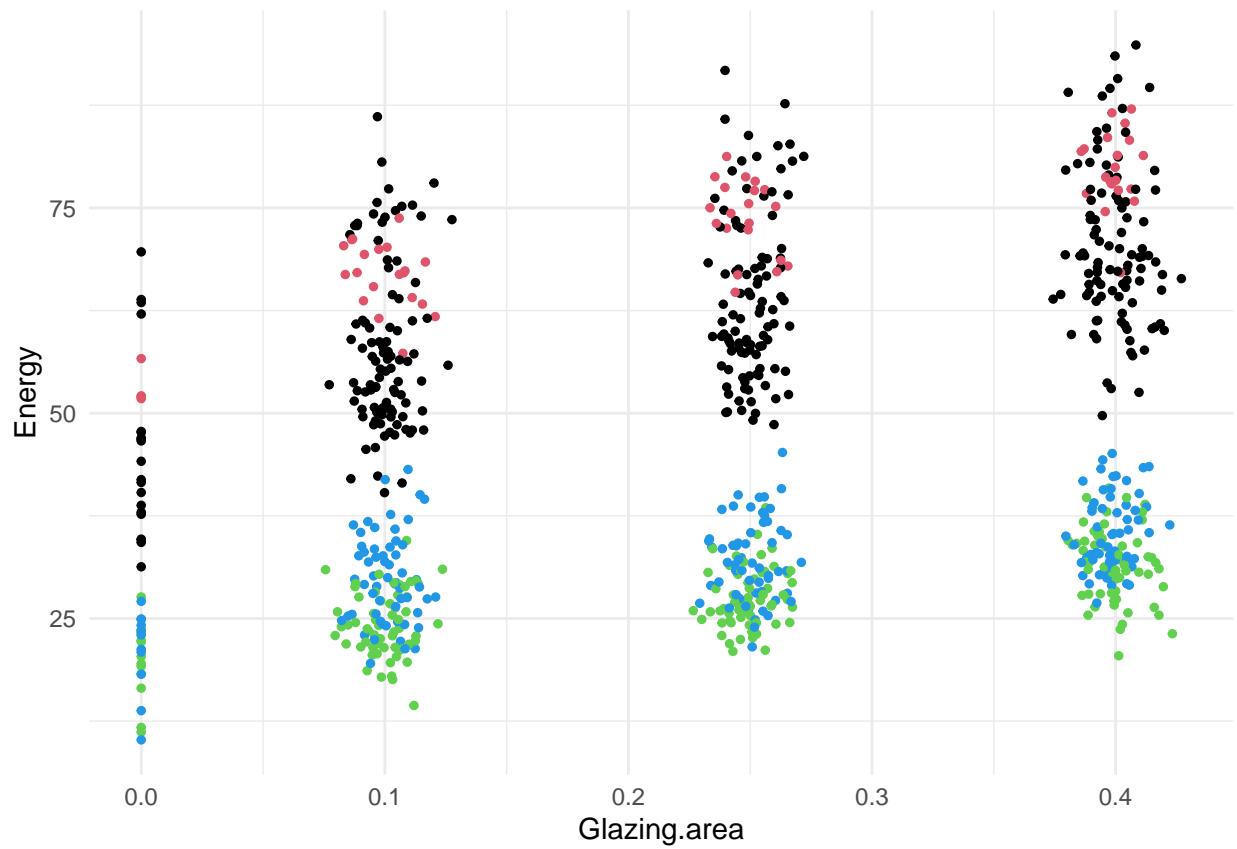


Visuellement selon la compacité relative, le découpage en 2 groupes et celui en 12 groupes semblent très pertinents. Celui en 4 groupes ne semble pas s'expliquer par la compacité relative.

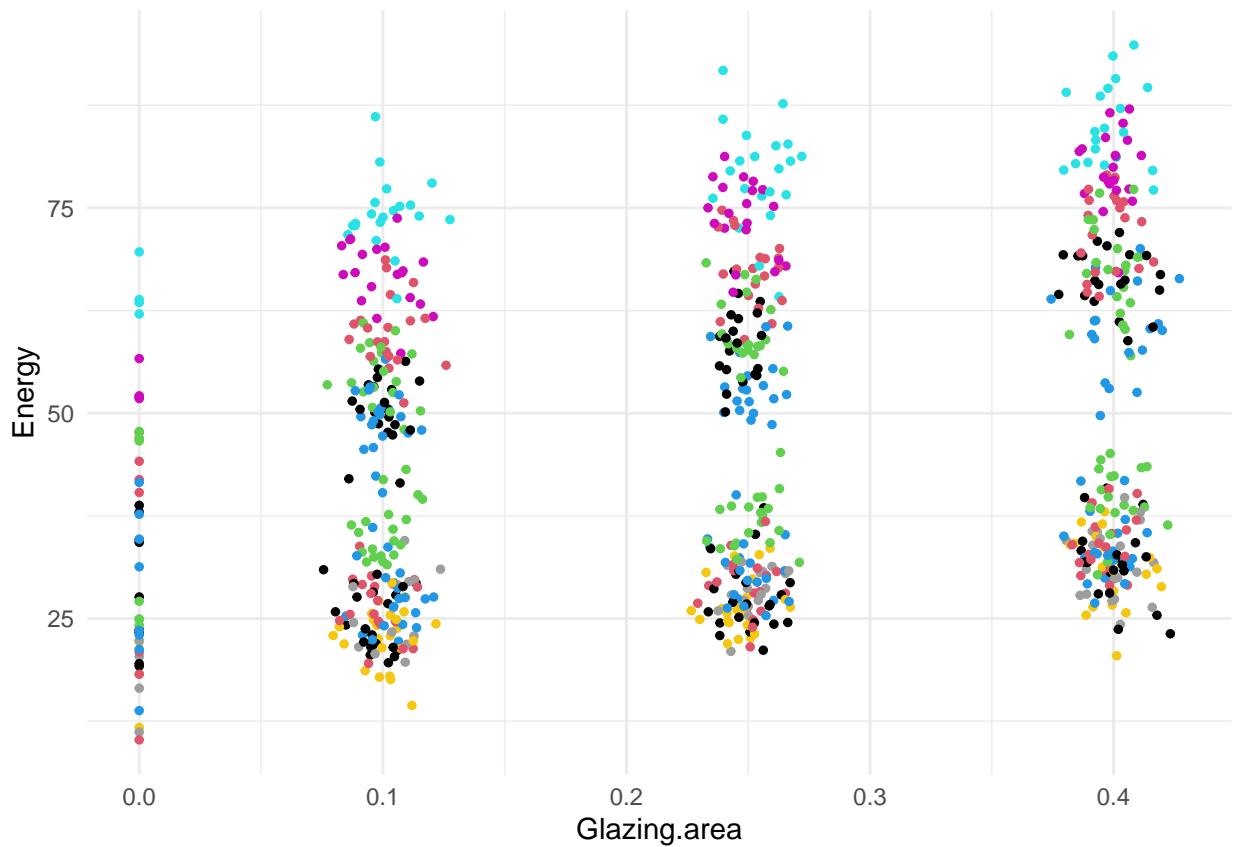
```
class2 = cutree(hc, k = 2)
ggplot(data) +
  aes(x = Glazing.area, y = Energy) +
  geom_point(size = 1L, colour = class2) +
  theme_minimal()
```



```
class4 = cutree(hc, k = 4)
ggplot(data) +
  aes(x = Glazing.area, y = Energy) +
  geom_point(size = 1L, colour = class4) +
  theme_minimal()
```

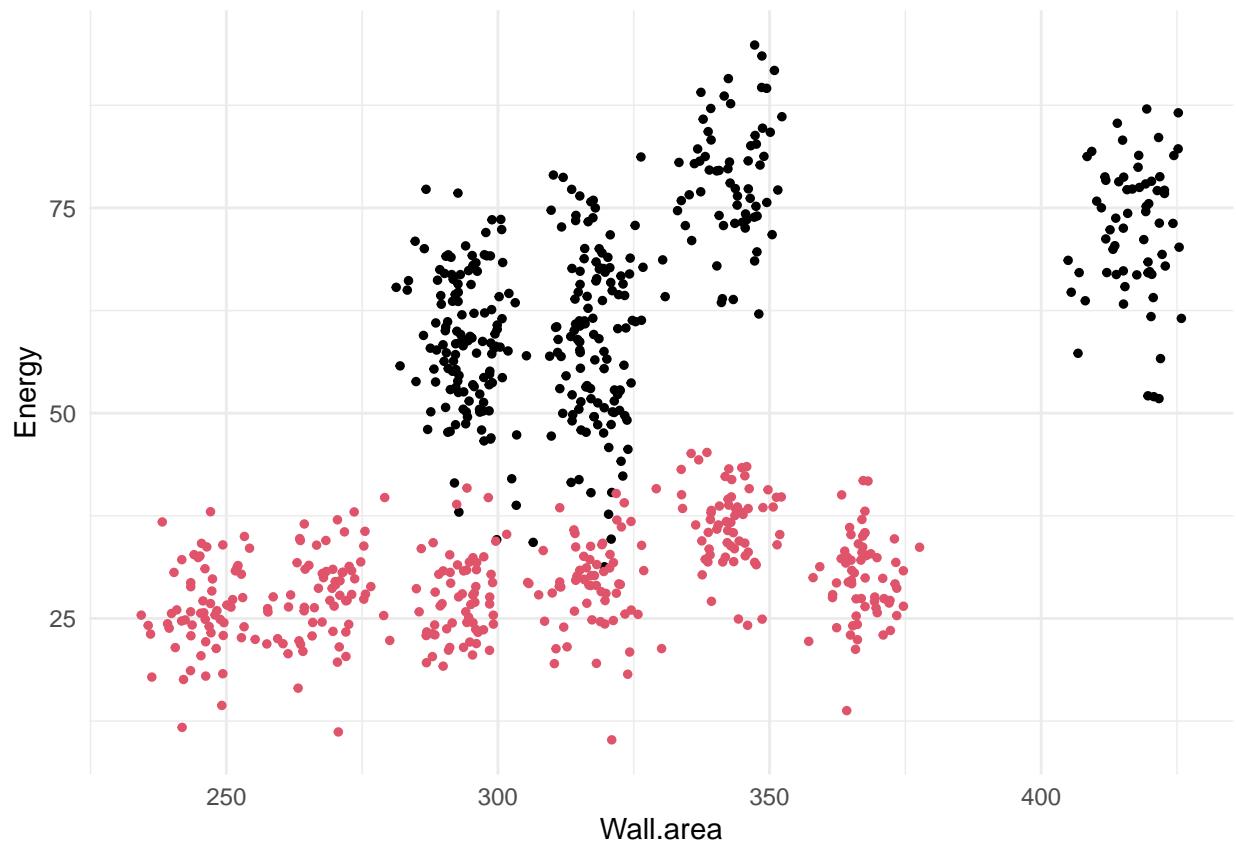


```
class12 = cutree(hc, k = 12)
ggplot(data) +
  aes(x = Glazing.area, y = Energy) +
  geom_point(size = 1L, colour = class12) +
  theme_minimal()
```

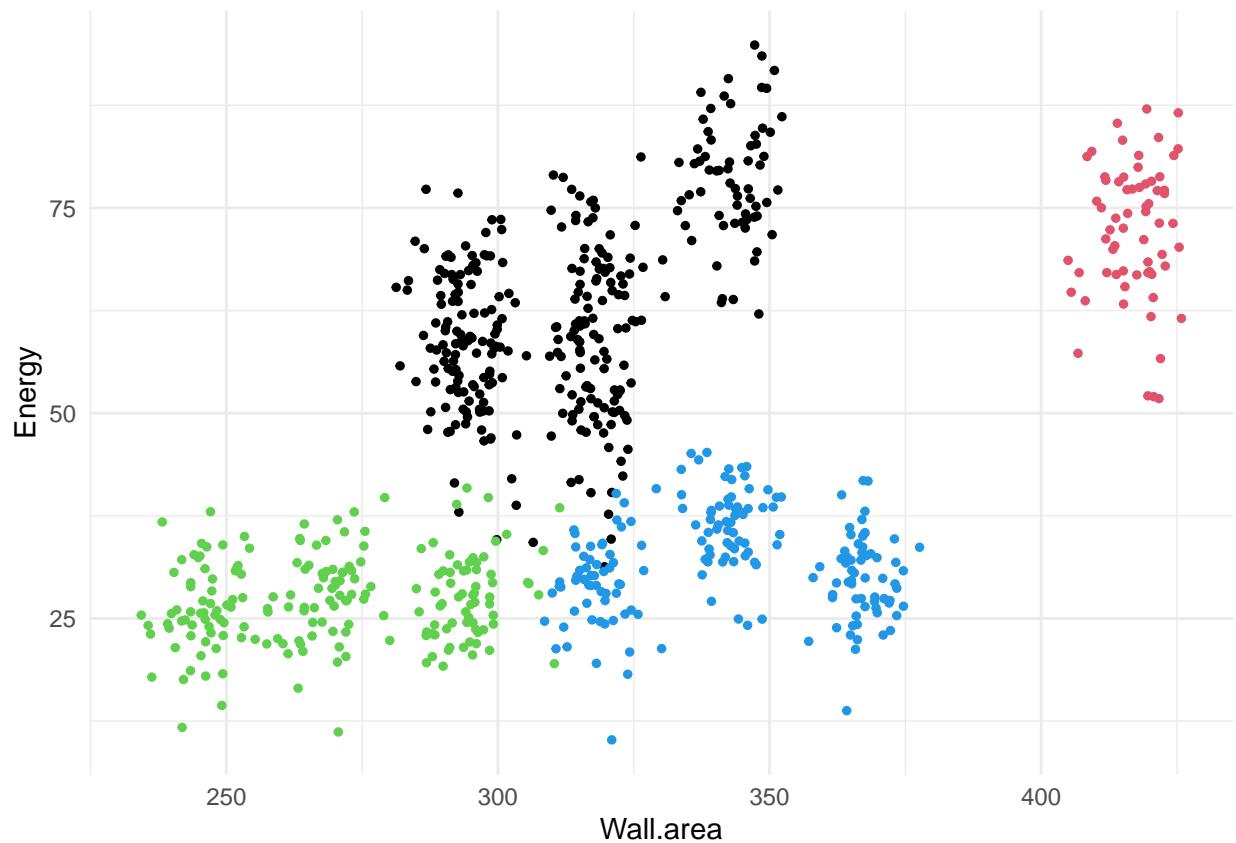


Selon la surface vitrée, seul le découpage en 2 classes semble pertinent. On aurait pu s'attendre à un découpage en 4 classes pertinents sur ce graphique mais ce n'est pas le cas.

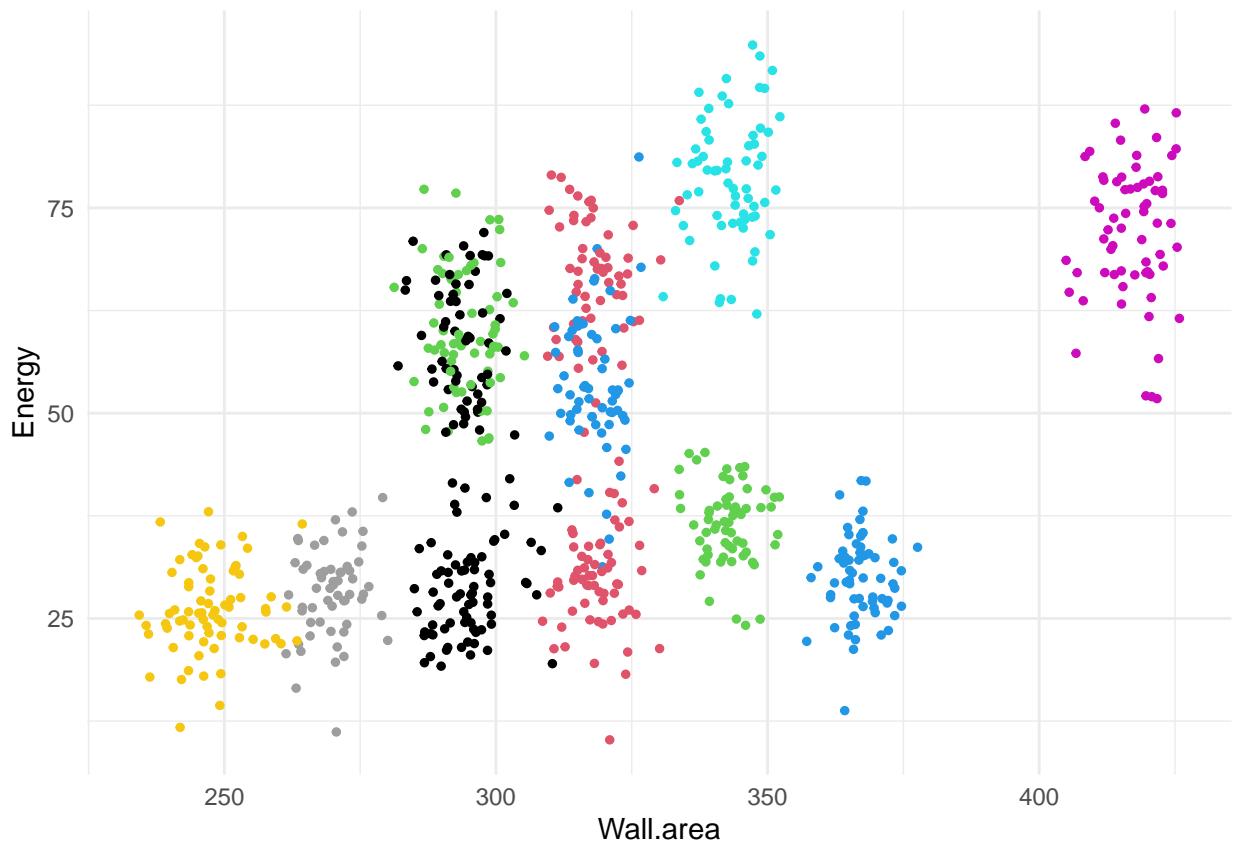
```
class2 = cutree(hc, k = 2)
ggplot(data) +
  aes(x = Wall.area, y = Energy) +
  geom_point(size = 1L, colour = class2) +
  theme_minimal()
```



```
class4 = cutree(hc, k = 4)
ggplot(data) +
  aes(x = Wall.area, y = Energy) +
  geom_point(size = 1L, colour = class4) +
  theme_minimal()
```



```
class12 = cutree(hc, k = 12)
ggplot(data) +
  aes(x = Wall.area, y = Energy) +
  geom_point(size = 1L, colour = class12) +
  theme_minimal()
```



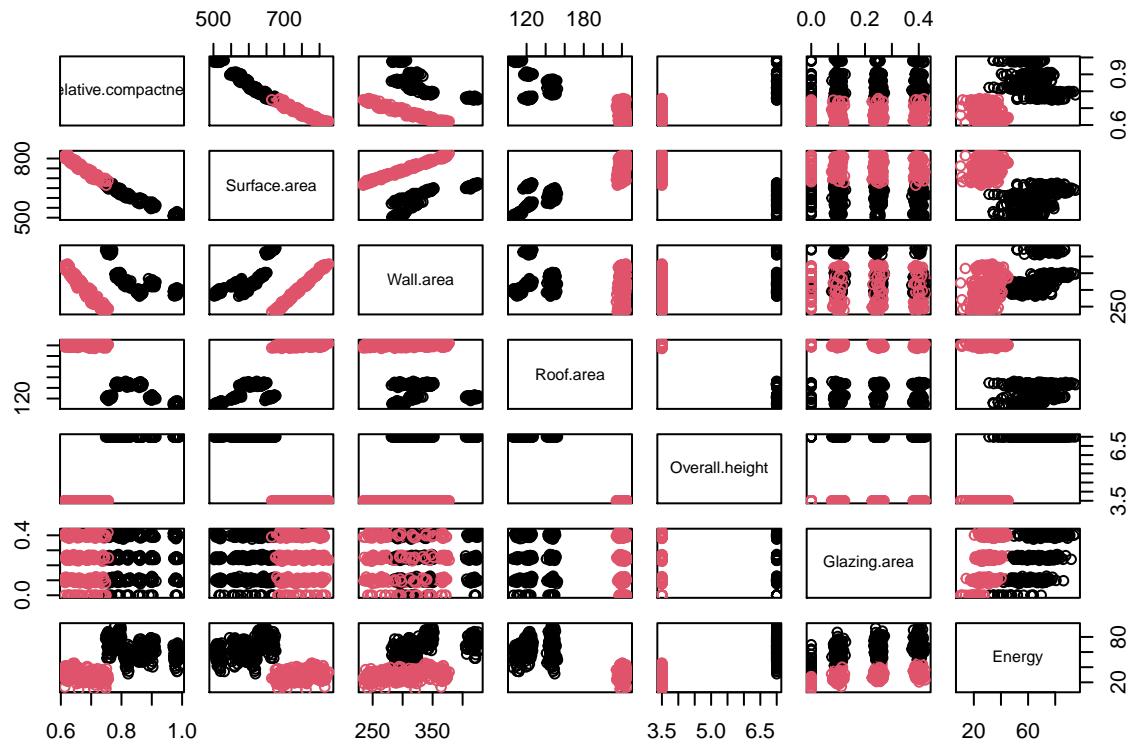
Finalement, le découpage en 4 classes semble pertinent selon la surface des murs.

k-means

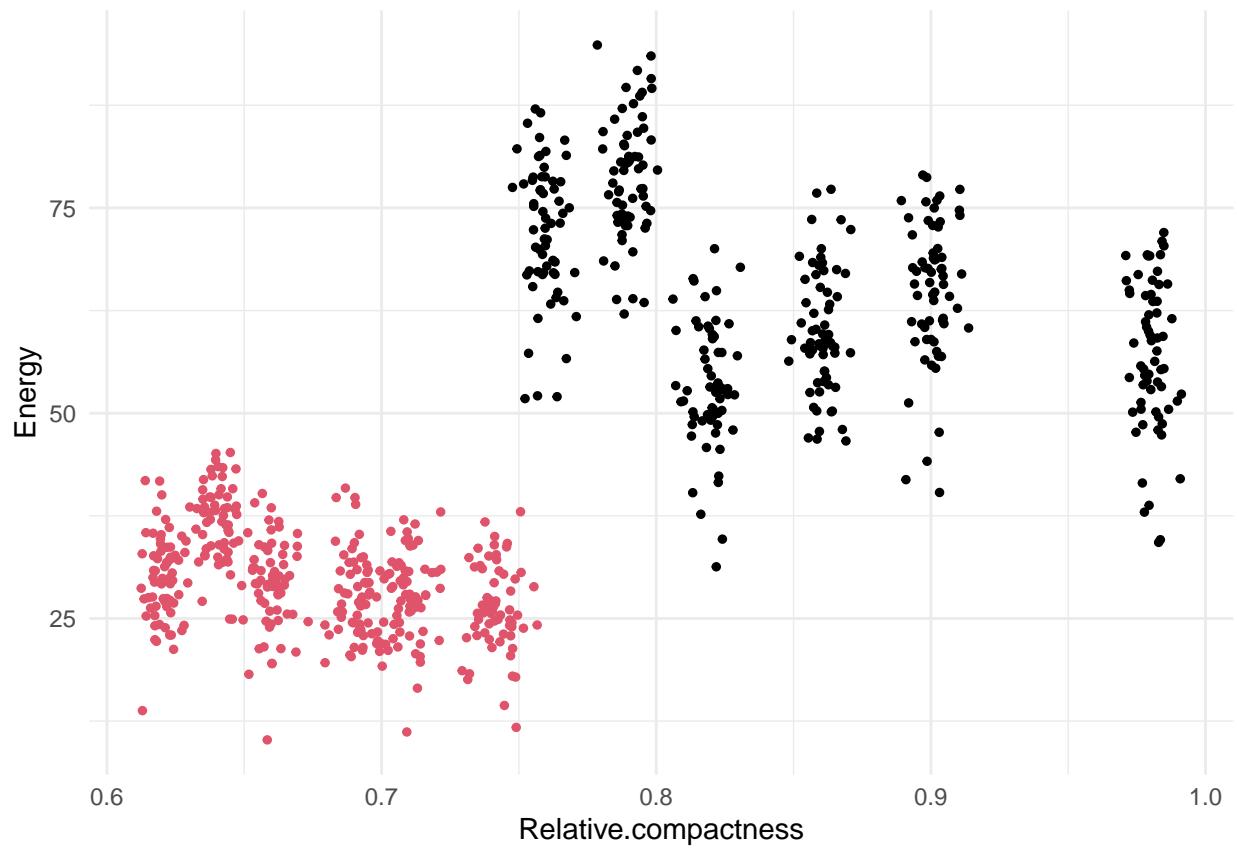
On utilise à présent un autre outil : k-means. Comparons les résultats avec ceux du clustering hiérarchique.

```
# k-means à 2 classes
kmres2 = kmeans(data[,c(1:5,7)], centers = 2)
kmclus2 = kmres2$cluster

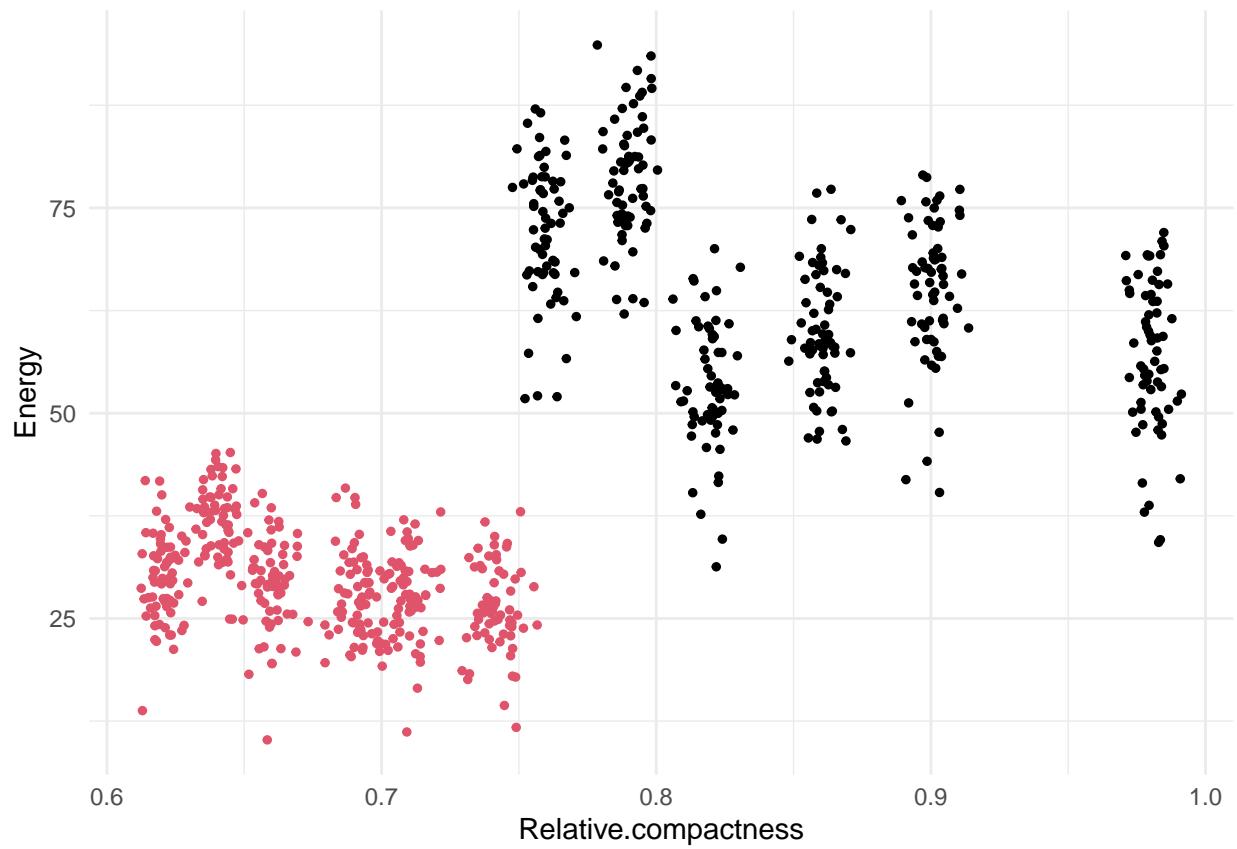
pairs(data[,quanti], col = kmclus2)
```



```
ggplot(data) +
  aes(x = Relative.compactness, y = Energy) +
  geom_point(size = 1L, colour = kmclus2) +
  theme_minimal()
```



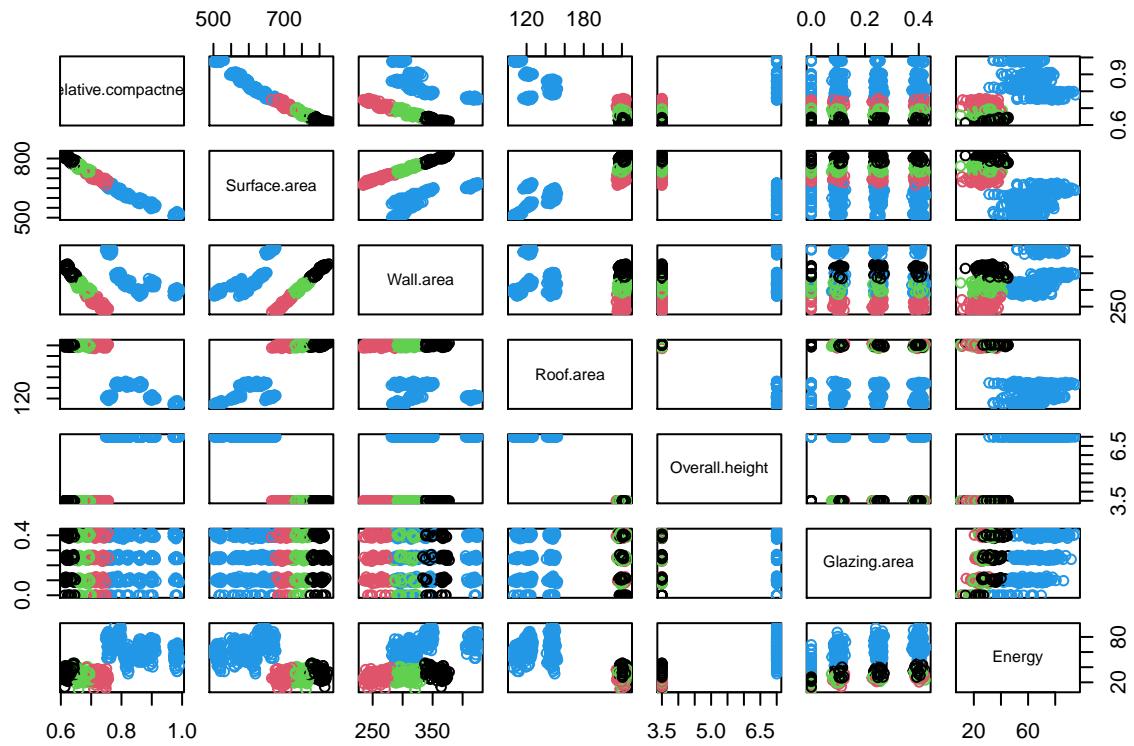
```
# clustering hierarchique à 2 classes
ggplot(data) +
  aes(x = Relative.compactness, y = Energy) +
  geom_point(size = 1L, colour = class2) +
  theme_minimal()
```



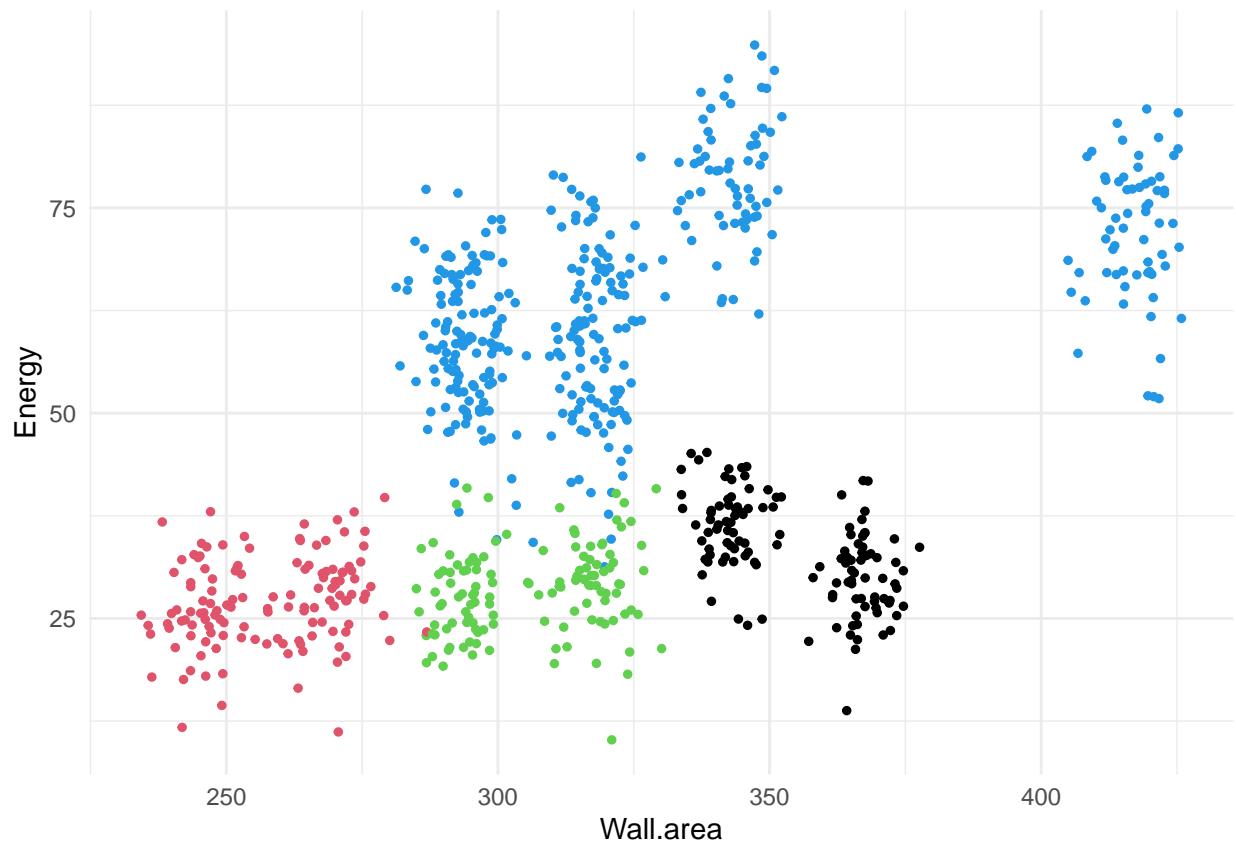
Pour le découpage à deux classes, les deux méthodes fournissent exactement les mêmes résultats.

```
# k-means à 4 classes
kmres4 = kmeans(data[,c(1:5,7)], centers = 4)
kmclus4 = kmres4$cluster

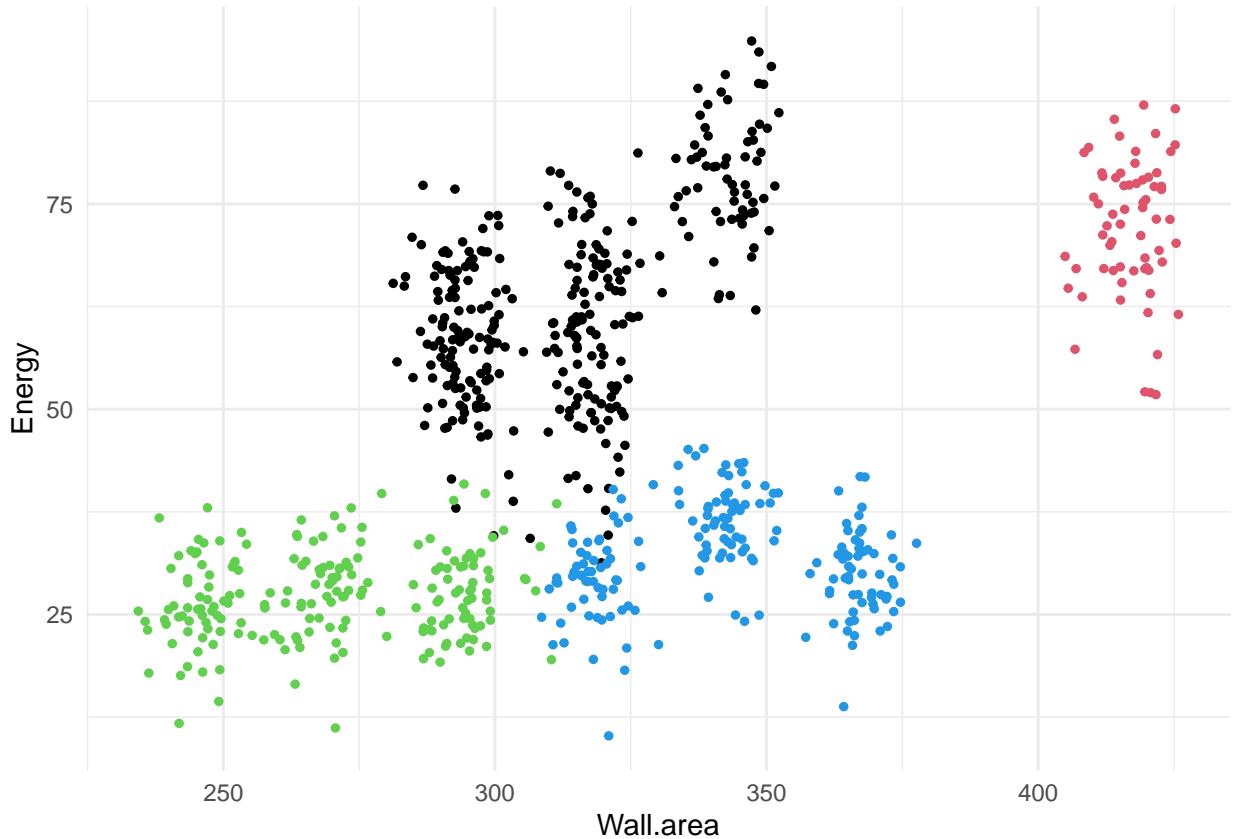
pairs(data[,quanti], col = kmclus4)
```



```
ggplot(data) +
  aes(x = Wall.area, y = Energy) +
  geom_point(size = 1L, colour = kmclus4) +
  theme_minimal()
```



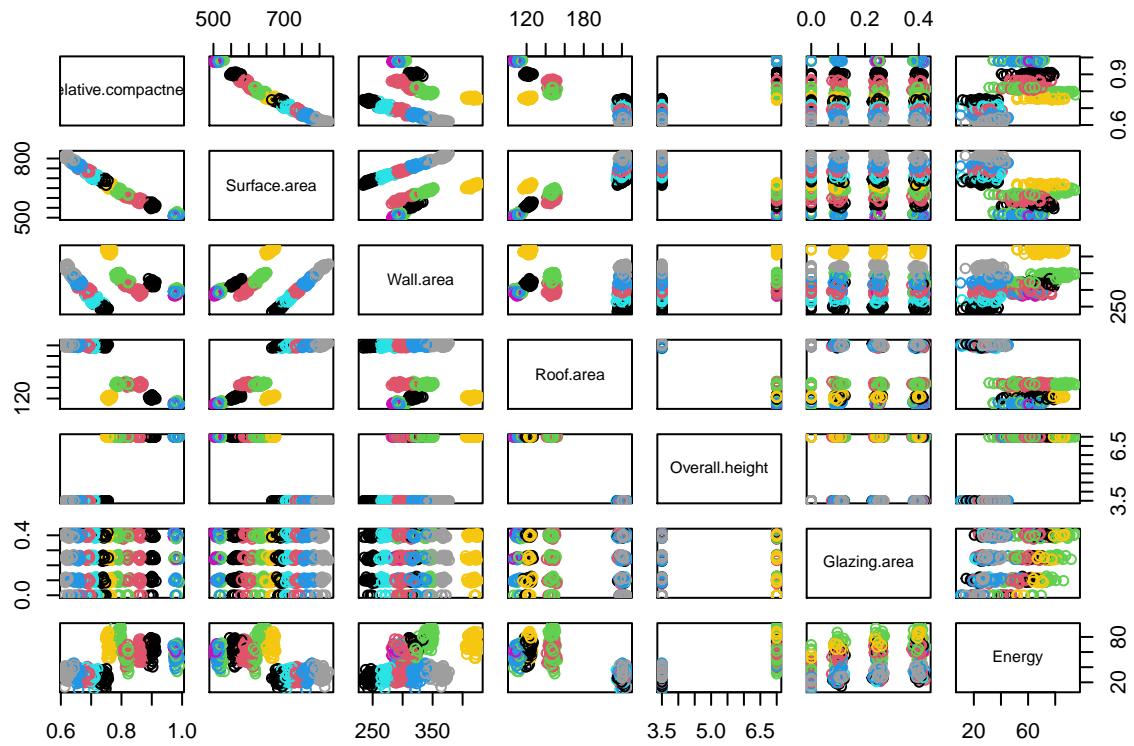
```
# clustering hierarchique à 4 classes
ggplot(data) +
  aes(x = Wall.area, y = Energy) +
  geom_point(size = 1L, colour = class4) +
  theme_minimal()
```



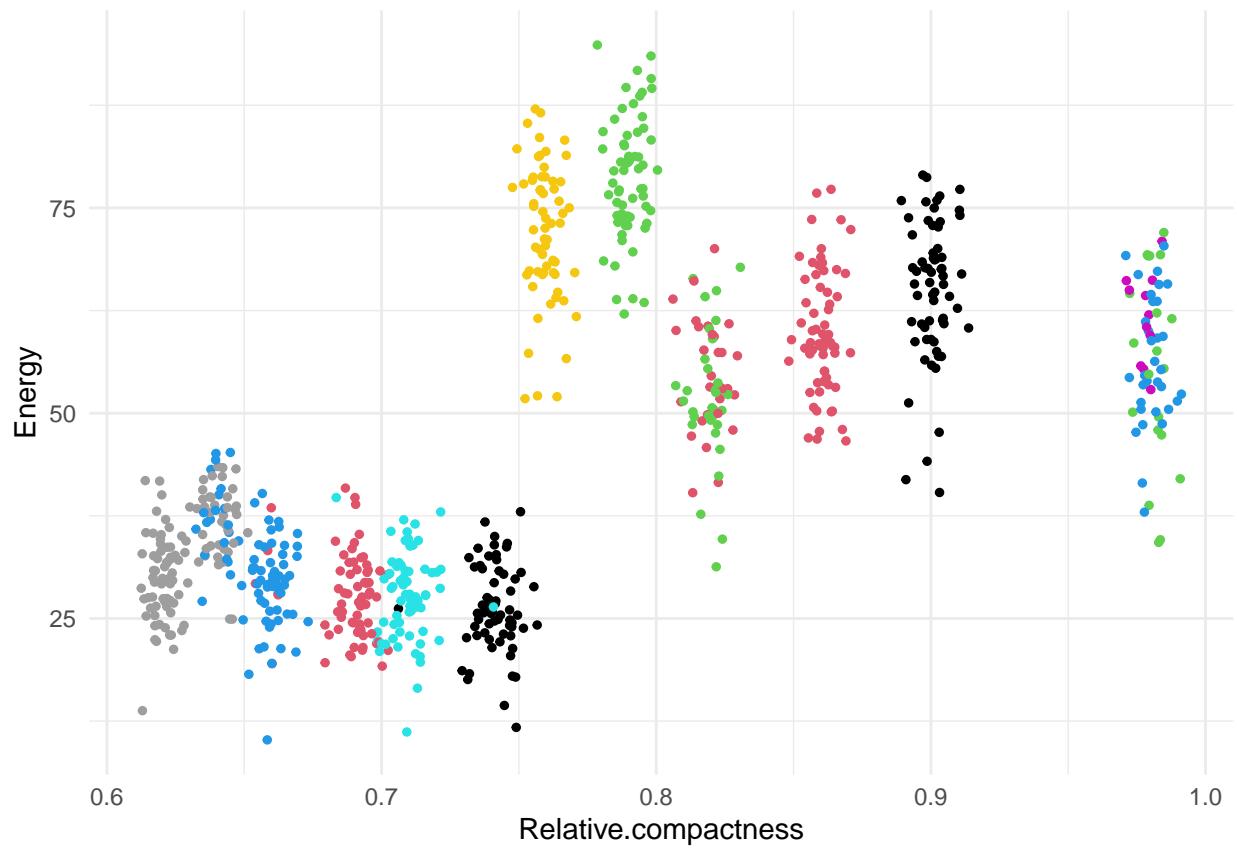
Après plusieurs exécutions, k-means ne donne presque jamais le même résultat que le clustering hiérarchiques. Alors que le clustering hiérarchiques semble créer des classes pertinentes selon la surface des murs, le k-means réalise souvent un découpage moins pertinent. En effet, le découpage issu de k-means consiste souvent à séparer en deux les données, puis à séparer en trois un des deux groupes obtenus de façon assez arbitraire. Le découpage en 4 classes avec k-means est donc peu pertinent.

```
# k-means à 12 classes
kmres12 = kmeans(data[,c(1:5,7)], centers = 12)
kmclus12 = kmres12$cluster

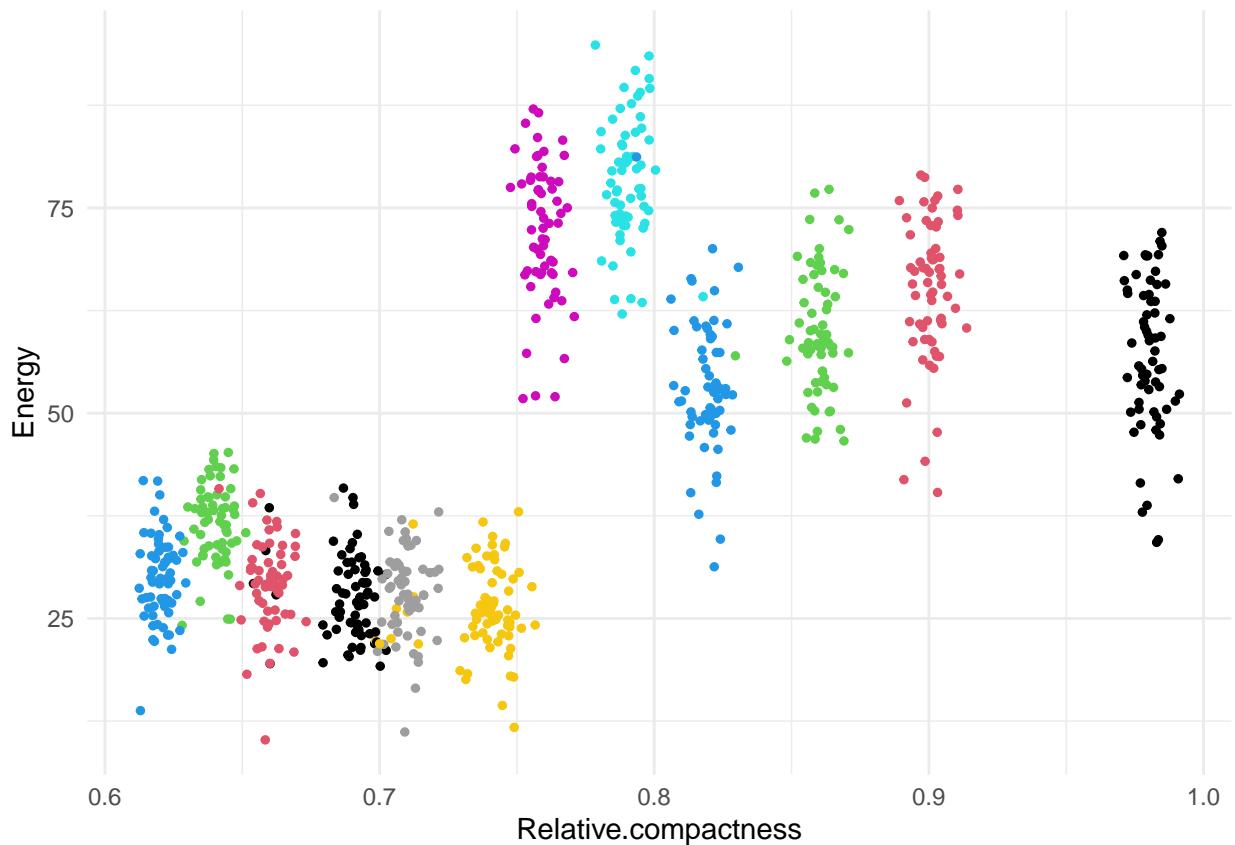
pairs(data[,quanti], col = kmclus12)
```



```
ggplot(data) +
  aes(x = Relative.compactness, y = Energy) +
  geom_point(size = 1L, colour = kmclus12) +
  theme_minimal()
```



```
# clustering hierarchique à 12 classes
ggplot(data) +
  aes(x = Relative.compactness, y = Energy) +
  geom_point(size = 1L, colour = class12) +
  theme_minimal()
```



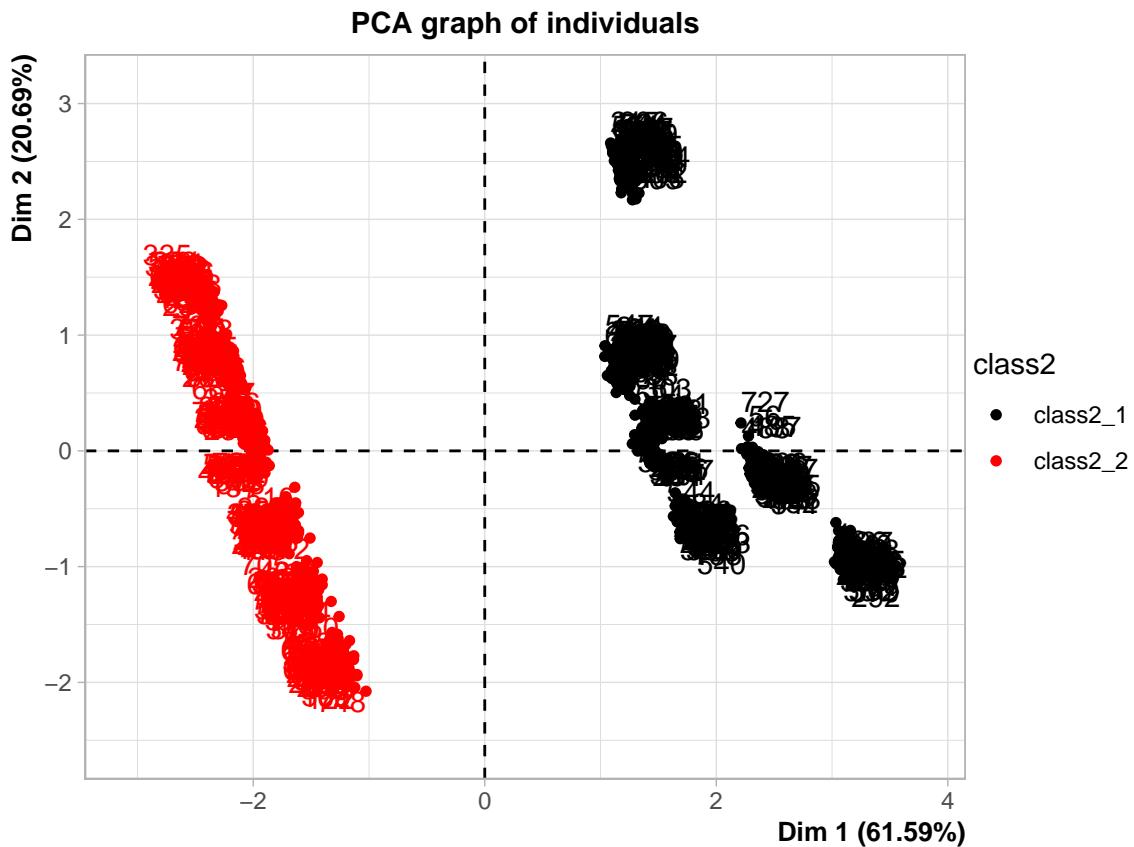
Selon les exécutions, k-means donne ici des résultats assez proches du clustering hiérarchique.

Visualisation des classes sur les axes de l'ACP

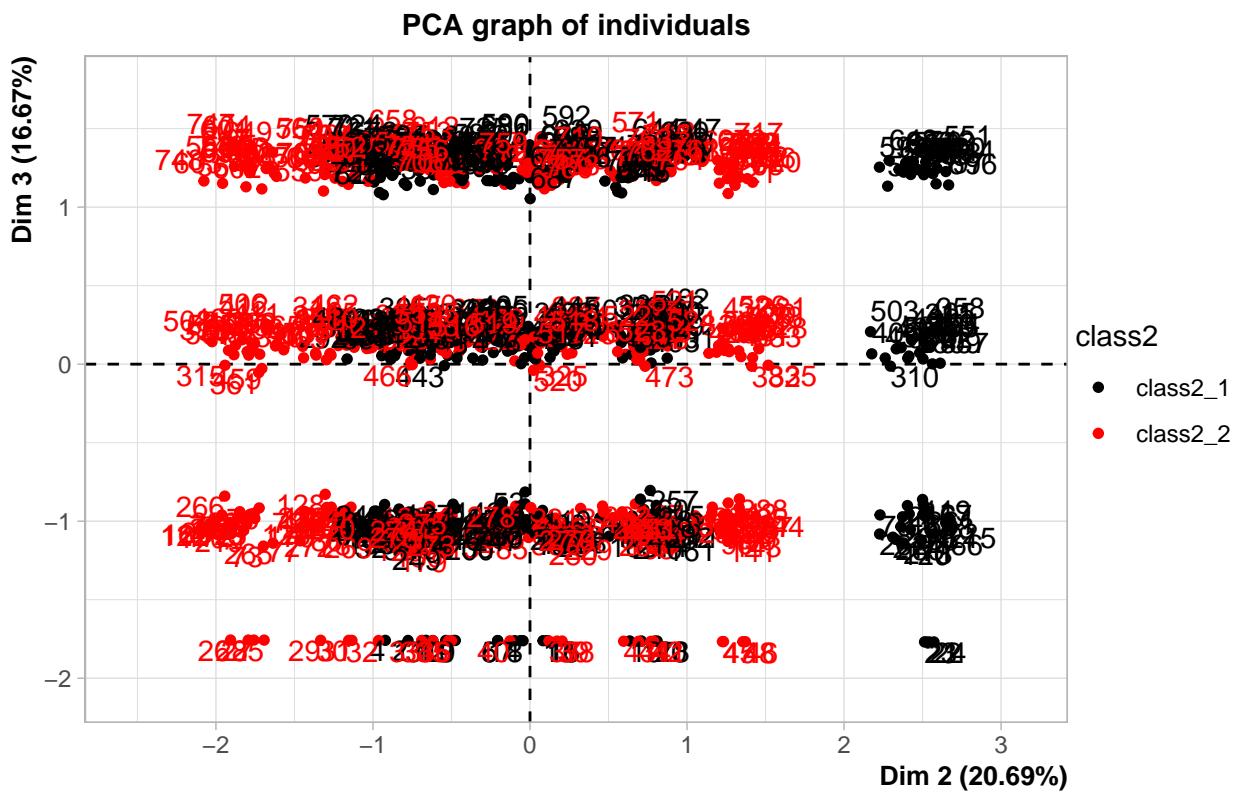
On utilise ici les classes obtenues par clustering hiérarchiques car celles-ci semblaient plus naturelles visuellement.

```
data$class2 = as.factor(class2)
data$class4 = as.factor(class4)
data$class12 = as.factor(class12)
res.acp <- PCA(data,scale.unit=T,quali.sup=c(6,8,10,11,12,13),quanti.sup=9,ncp=8, graph=F)

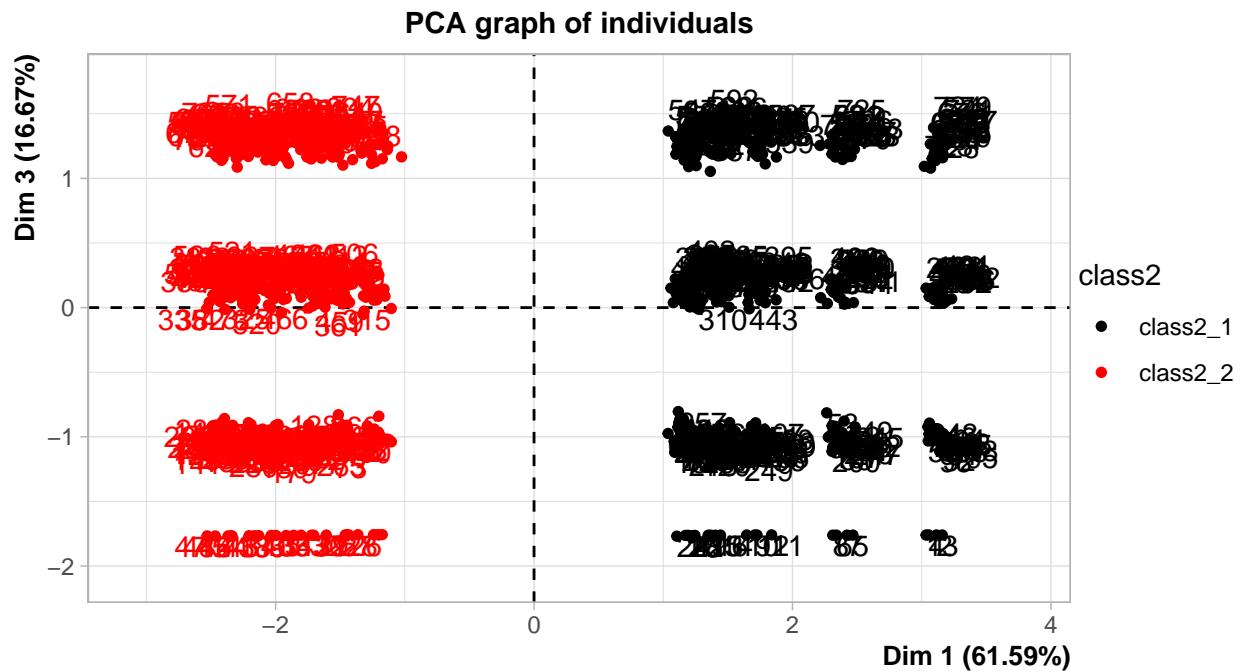
plot(res.acp, choix="ind", axes = c(1,2), invisible="quali", habillage="class2")
```



```
plot(res.acp, choix="ind", axes = c(2,3), invisible="quali", habillage="class2")
```

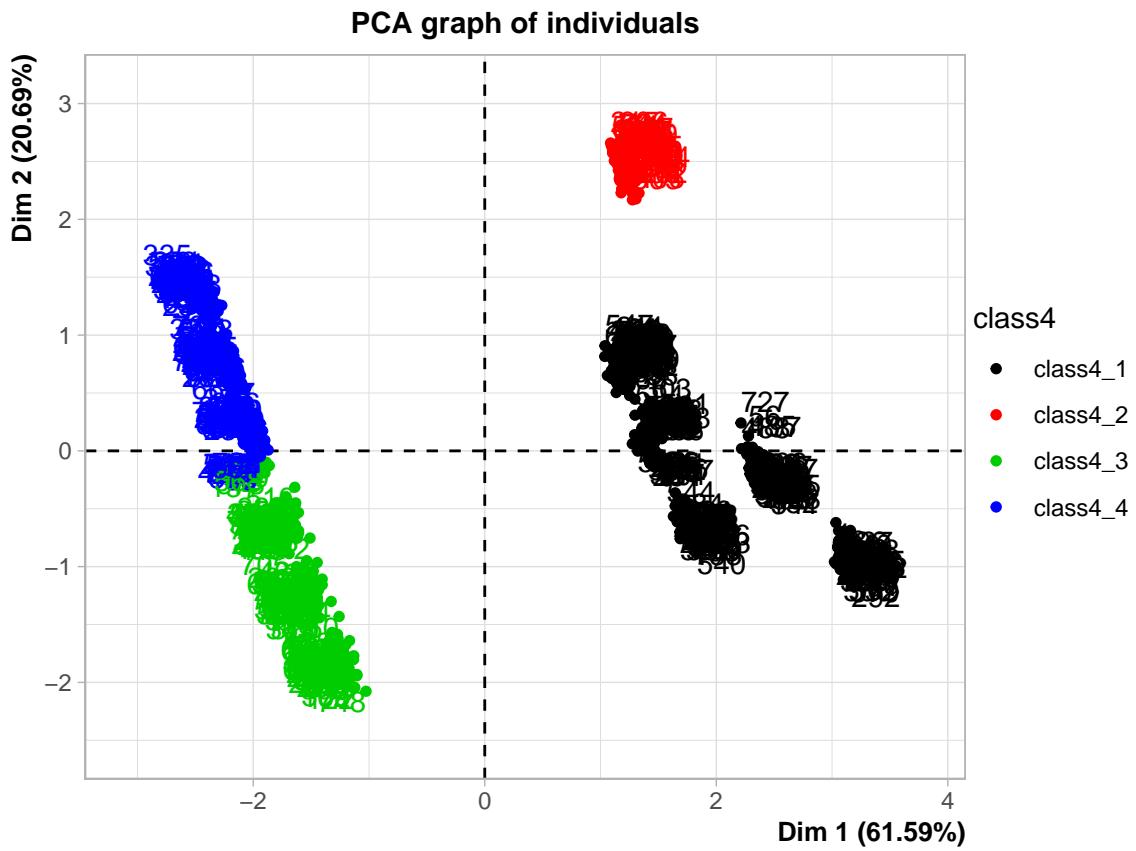


```
plot(res.acp, choix="ind", axes = c(1,3), invisible="quali", habillage="class2")
```

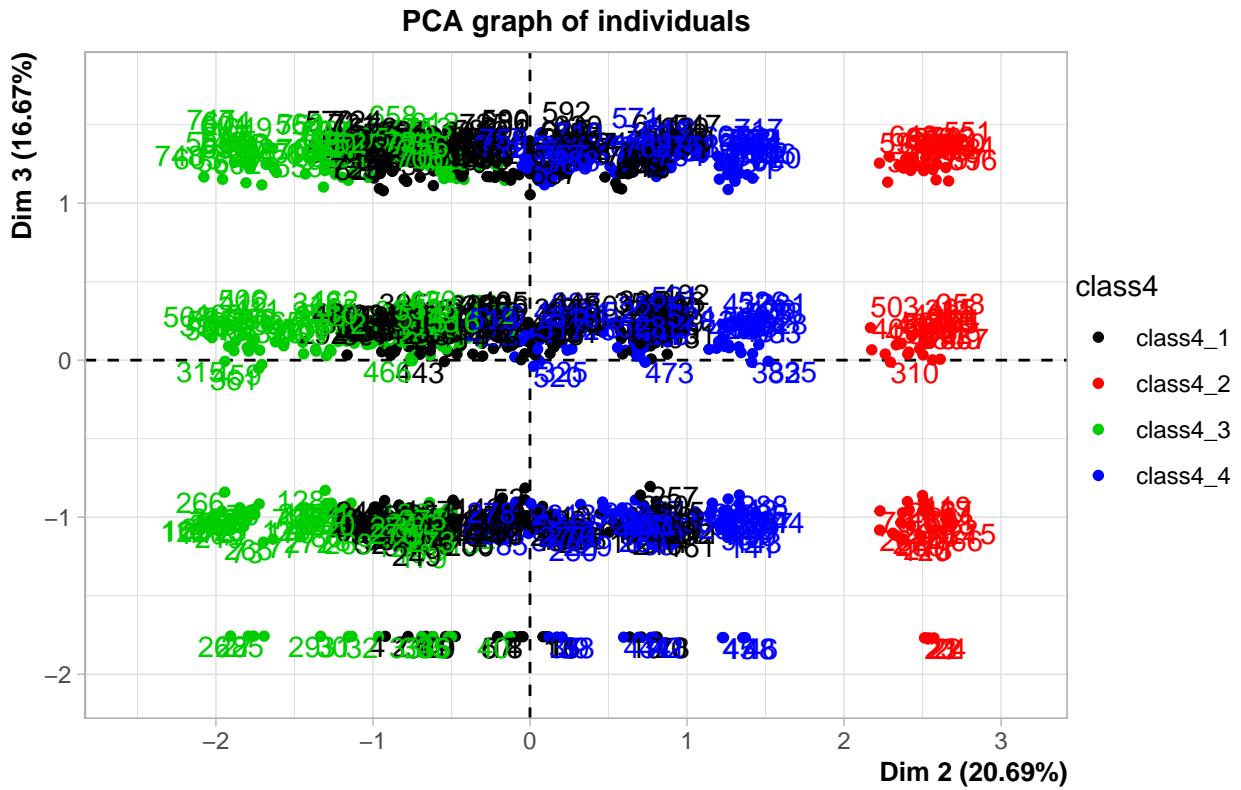


On voit ici que les données sont bien découpées en 2 classes selon le premier axe. Cela donne une bonne confiance en le découpage en 2 classes.

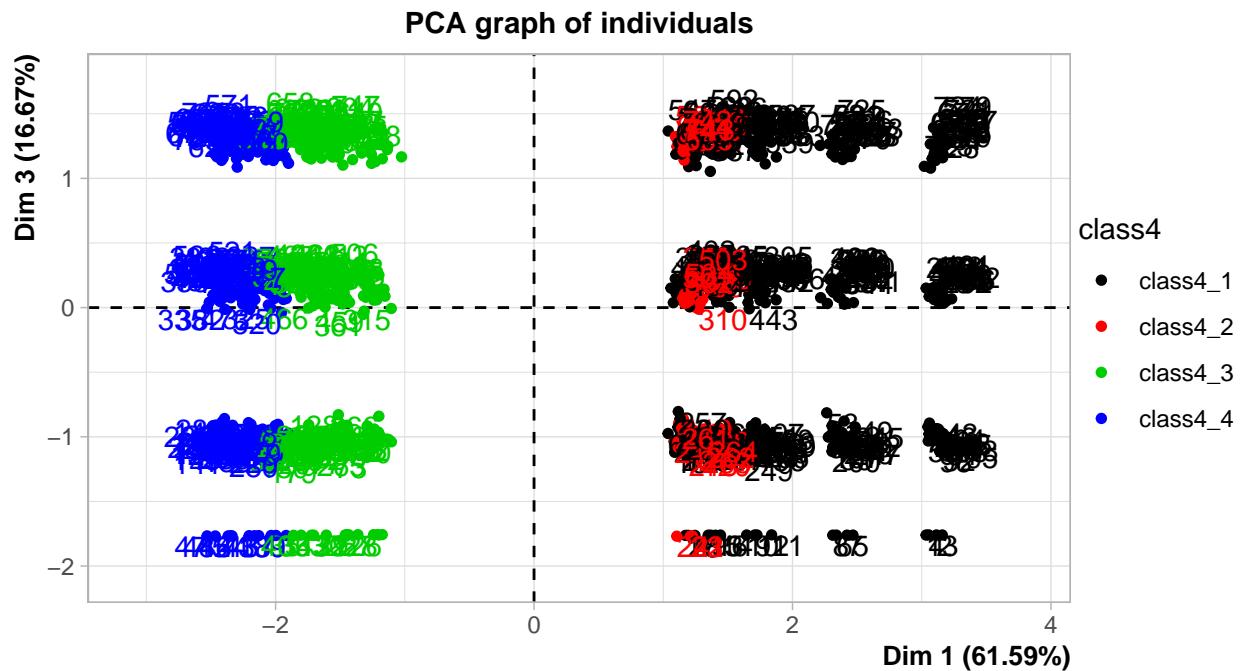
```
plot(res.acp, choix="ind", axes = c(1,2), invisible="quali", habillage="class4")
```



```
plot(res.acp, choix="ind", axes = c(2,3), invisible="quali", habillage="class4")
```

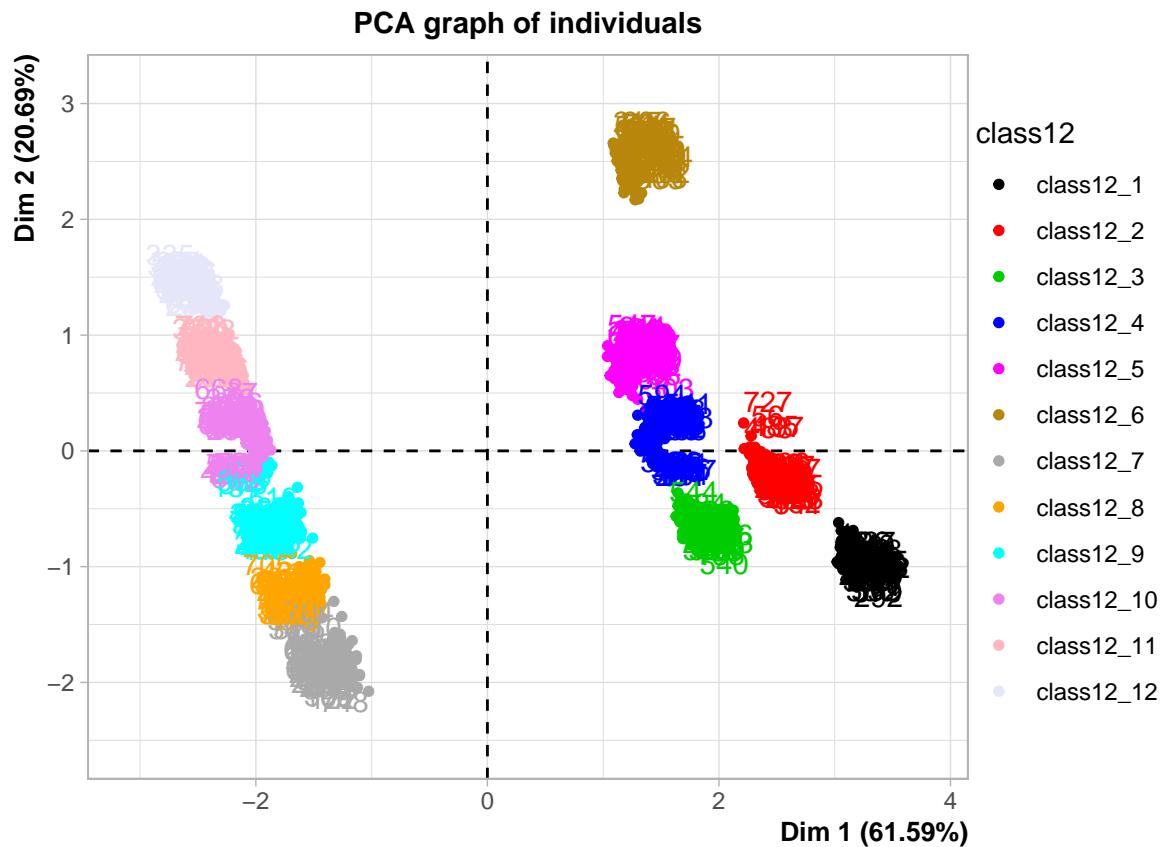


```
plot(res.acp, choix="ind", axes = c(1,3), invisible="quali", habillage="class4")
```

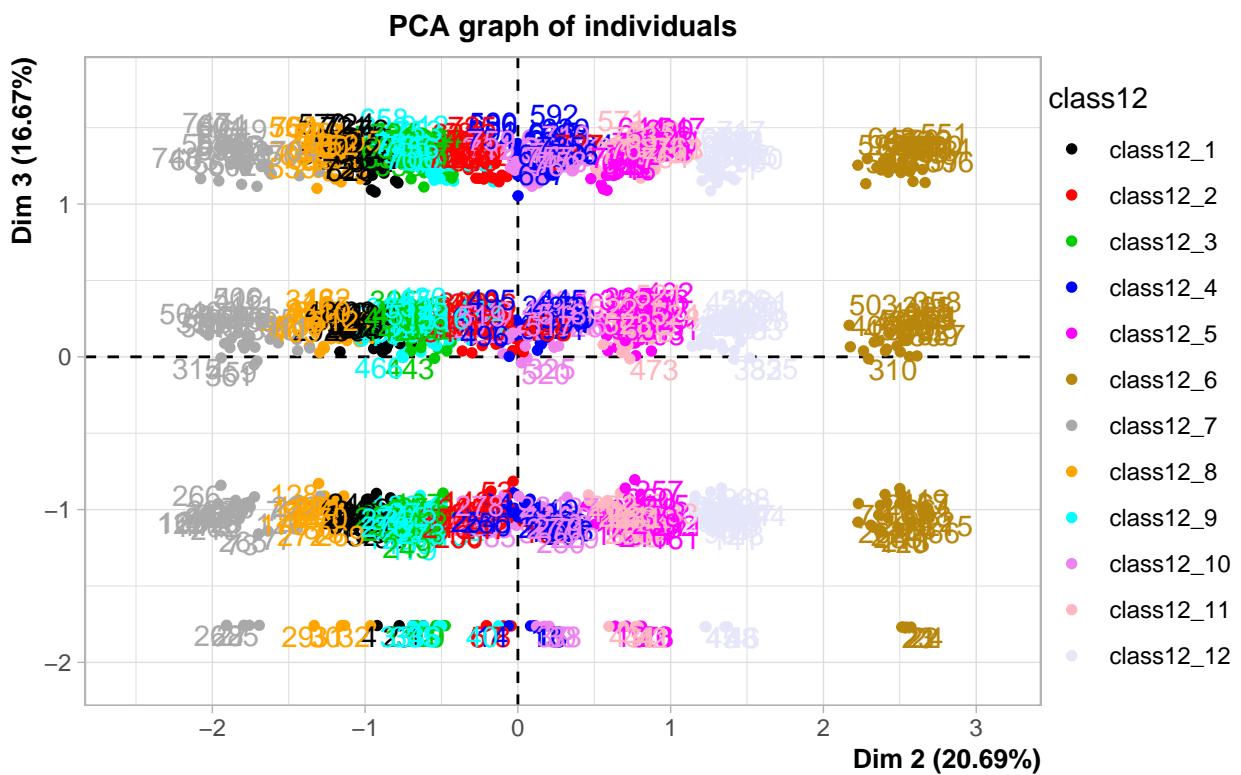


Le découpage en 4 classes se fait selon les axes 1 et 2, même si il est moins évident que celui en 2 classes. (notamment sur les classes 3 et 4 ci-dessus).

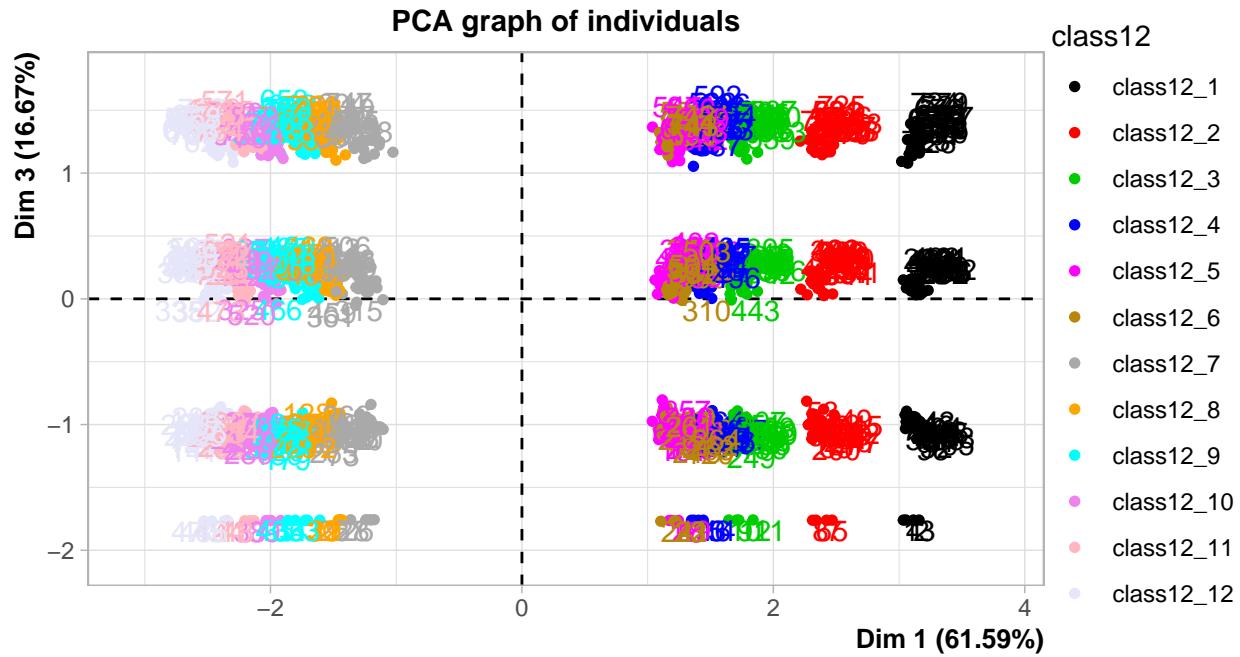
```
plot(res.acp, choix="ind", axes = c(1,2), invisible="quali", habillage="class12")
```



```
plot(res.acp, choix="ind", axes = c(2,3), invisible="quali", habillage="class12")
```



```
plot(res.acp, choix="ind", axes = c(1,3), invisible="quali", habillage="class12")
```



Ici, on peut avoir la même observation que pour le découpage en 4 classes : le découpage est moins net que celui en 2 classes et se fait selon les axes 1 et 2.

Modèles linéaires

Modèle fonction des variables quantitatives

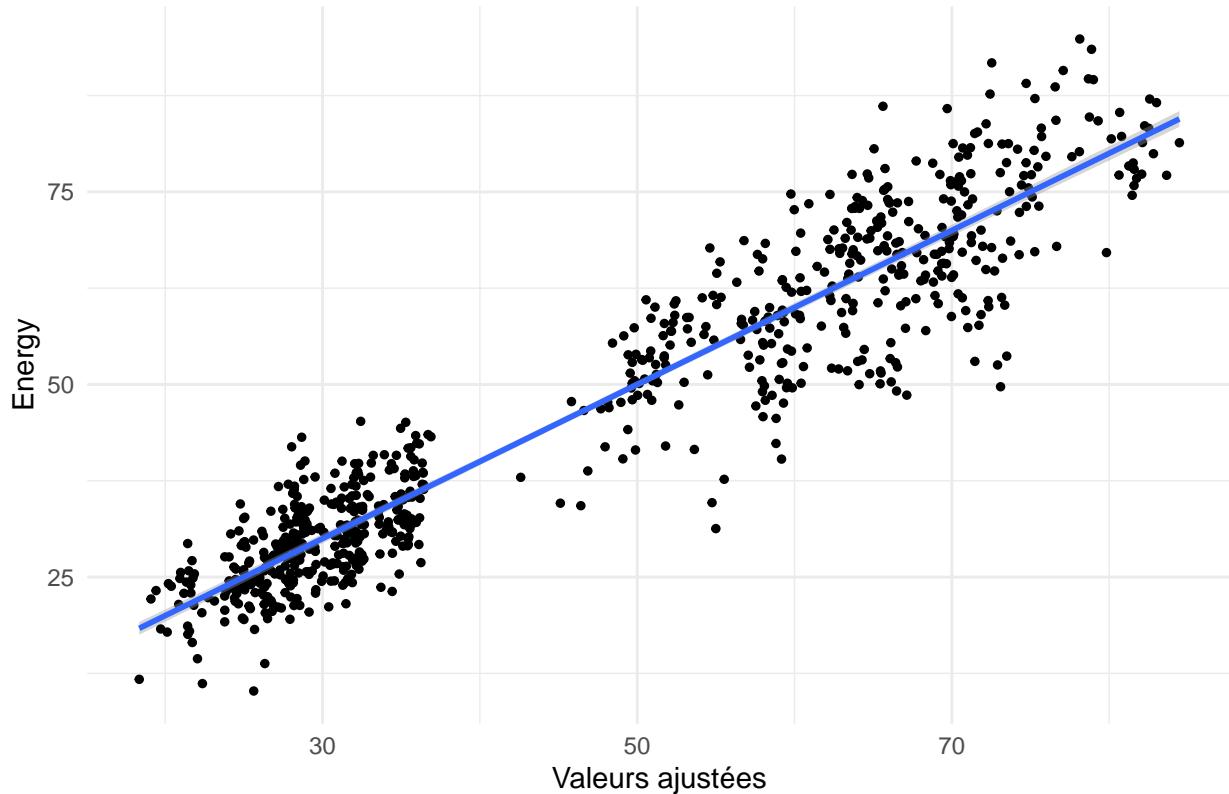
Nous étudierons dans un premier temps des modèles linéaires expliquant la variable quantitative `Energy` en fonction des variables *quantitatives* uniquement. Nous écarterons cependant la variable `Roof.area`, car nous avons vu qu'elle était combinaison linéaire des variables `Wall.area` et `Surface.area`. La conserver rendrait notre modèle singulier.

Modèle avec interactions

Nous essayons un premier modèle contenant les variables explicatives et leurs interactions.

```
model_quanti_complet = lm(Energy ~ . - Roof.area)^2, data = data[,quanti])
r.sq = summary(model_quanti_complet)$r.squared ; paste("R^2 =", r.sq)
data$fitted_quanti_complet = model_quanti_complet$fitted.values
ggplot(data) +
  aes(x = fitted_quanti_complet, y = Energy) +
  geom_point(size = 1L) + geom_smooth(method = "lm", formula = "y~x") +
  labs(title = "Regression linéaire de `Energy` en fonction des variables quantitatives",
       x = "Valeurs ajustées") +
  theme_minimal()
```

Regression linéaire de 'Energy' en fonction des variables quantitatives



```
## [1] "R² = 0.89516910409479"
```

Le modèle obtenu semble déjà bien passer au sein des données : on trouve un R^2 de 0.8952.

Cependant, ce modèle conserve beaucoup de variables, on peut donc se demander si elles sont toutes pertinentes. Nous allons donc essayer de n'en conserver que certaines.

Selection de variables par critère BIC

Nous allons réaliser dans un premier temps une sélection de variables par critère BIC.

```
modselect_bic_back = stepAIC(model_quanti_complet, trace=FALSE, direction=c("backward"), k=log(nrow(data)))
paste("Energy ~", paste(dimnames(modselect_bic_back$qr$qr)[[2]][-1], collapse=" + "))
```

```
## [1] "Energy ~ Relative.compactness + Surface.area + Wall.area + Overall.height + Glazing.area + Relat
```

Le modèle sélectionné conserve nos 5 des variables quantitatives, mais supprime certaines de leurs interactions. Afin de s'assurer que l'on peut simplifier notre modèle, on réalise un test de sous-modèle entre le modèle complet et le modèle sélectionné.

```
anova(modselect_bic_back, model_quanti_complet)
r.sq = summary(modselect_bic_back)$r.squared ; paste("R² =", r.sq)
```

```
## Analysis of Variance Table
##
## Model 1: Energy ~ Relative.compactness + Surface.area + Wall.area + Overall.height +
##           Glazing.area + Relative.compactness:Surface.area + Relative.compactness:Wall.area +
##           Relative.compactness:Overall.height + Relative.compactness:Glazing.area +
##           Wall.area:Glazing.area
## Model 2: Energy ~ ((Relative.compactness + Surface.area + Wall.area +
```

```

##      Roof.area + Overall.height + Glazing.area) - Roof.area)^2
##   Res.Df   RSS Df Sum of Sq    F Pr(>F)
## 1     757 31986
## 2     752 31723  5     262.5 1.2445 0.2864
## [1] "R^2 = 0.894301671567371"

```

On obtient une p-valeur de 0.2864. On ne rejette donc pas notre modèle sélectionné au seuil de 5 %. Ainsi, notre modèle possède un R^2 de 0.8943, ce qui est à peine plus faible que ce que nous avions avec toutes nos variables.

Selection de variable par regression regularisée

On peut également tenter de sélectionner nos variables par régression régularisée.

Problème sur régression généralisée : tau est nul, voir avec Cathy

```

# centrage et réduction des données
eng=scale(data["Energy"],center=T,scale=T)
f = as.formula(paste("Energy ~ (", paste(dimnames(data[,c(1,2,3,5,7)])[[2]], collapse=" + "), ")^2"))
expli=scale(model.matrix(f,data)[,-1],center=T,scale=T)

# création du tableau de tau
tau_seq <- seq(0, 0.002, by = 0.000001)

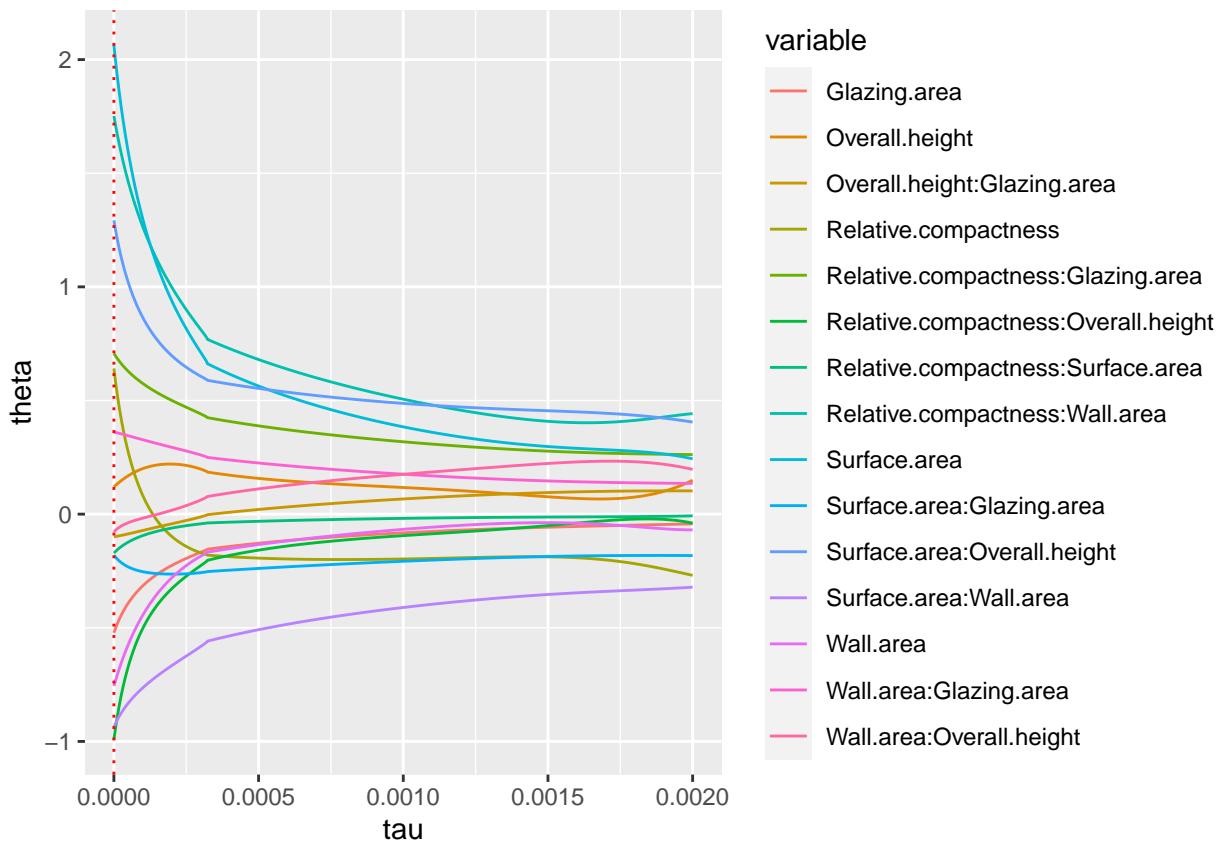
# regression ridge
fitridge <- glmnet(x = expli, y = eng, family = "gaussian", alpha = 0, lambda = tau_seq, type.measure=c("deviance"))

# récupération du tau minimum par validation croisée
ridge_cv = cv.glmnet(x = expli, y = eng, family = "gaussian", alpha = 0, lambda = tau_seq, type.measure=c("deviance"))
tau_min_ridge = ridge_cv$lambda.min
print(tau_min_ridge)

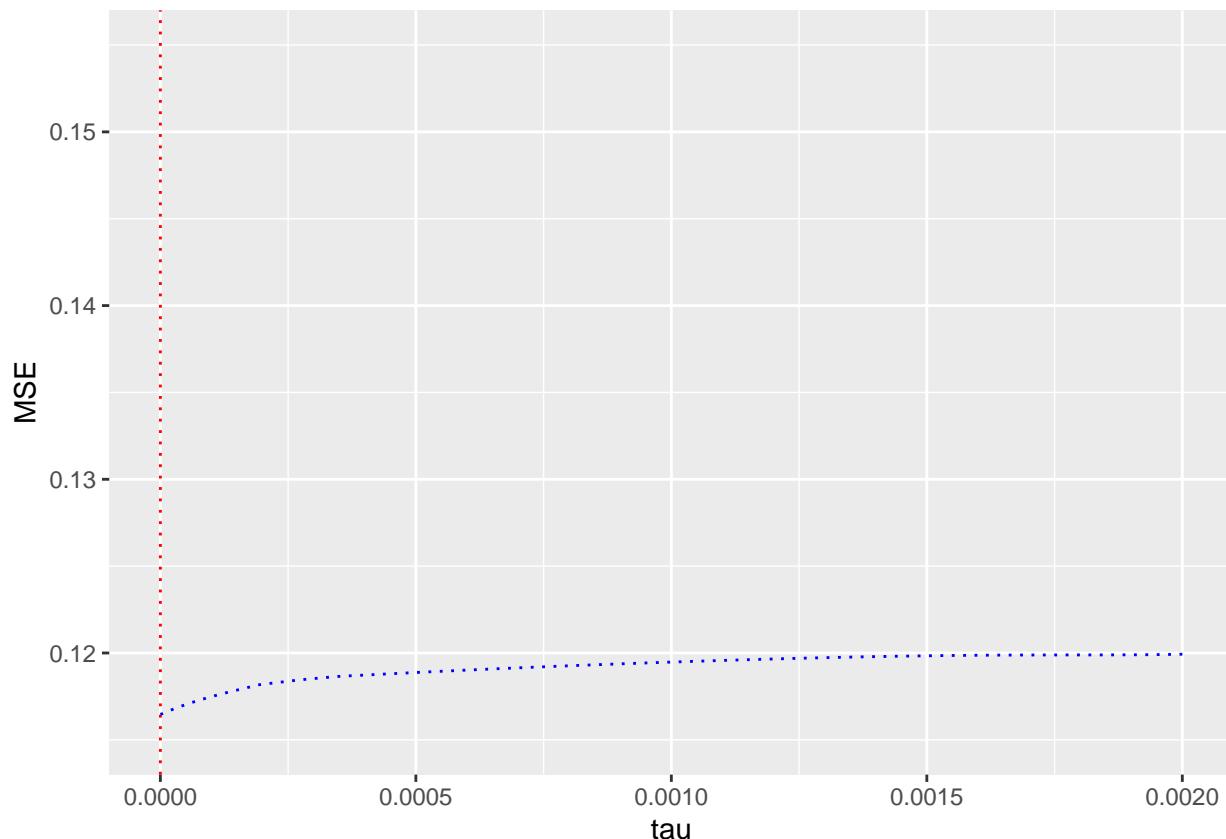
# affichage des estimations de tau
dfridge=data.frame(tau = rep(fitridge$lambda,ncol(expli)), theta=as.vector(t(fitridge$beta)),variable=rownames(expli))
ggplot(dfridge,aes(x=tau,y=theta,col=variable)) +
  geom_line() +
  geom_vline(xintercept = tau_min_ridge, linetype = "dotted", color = "red") +
  theme(legend.position="right")

```

Ridge



```
## [1] 0
## Warning: Removed 2001 row(s) containing missing values (geom_path).
## Warning: Removed 2001 row(s) containing missing values (geom_path).
```



```

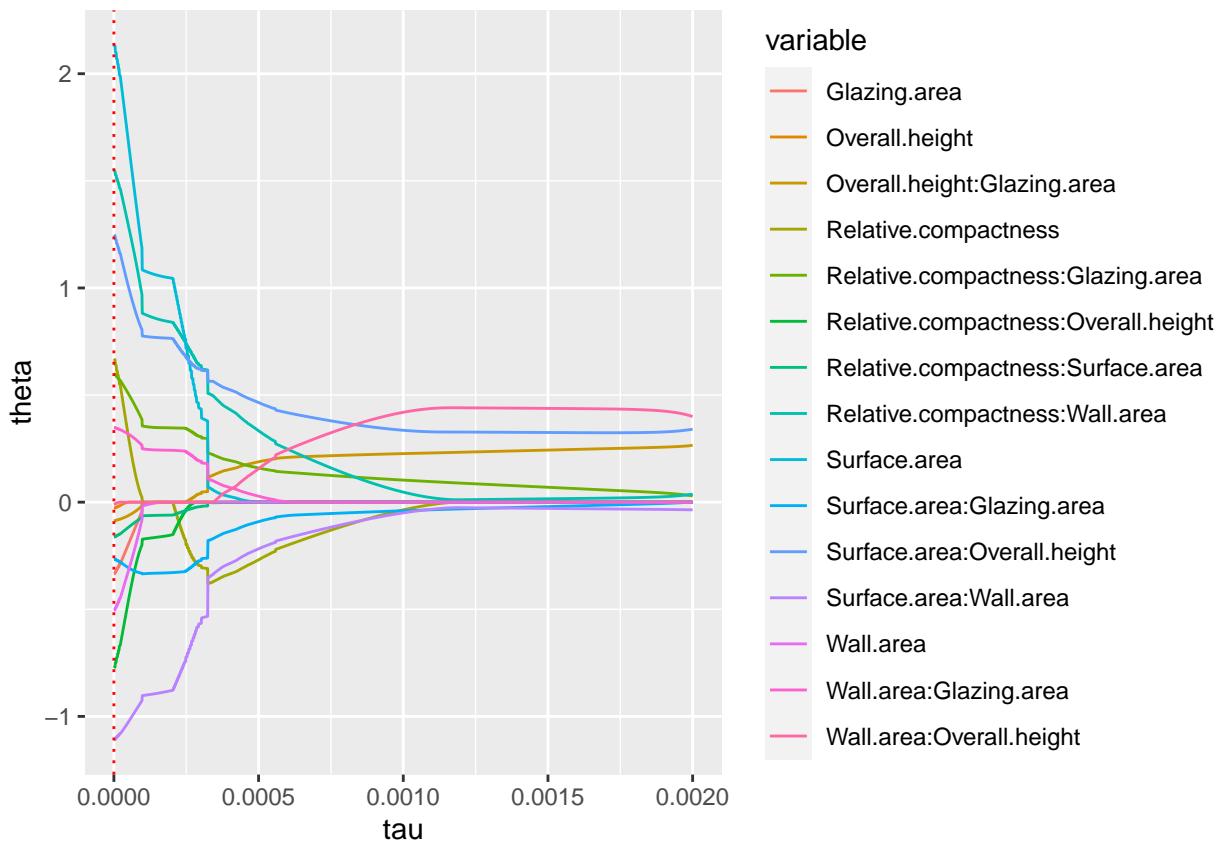
# regression lasso
fitlasso <- glmnet(x = expli, y = eng, family = "gaussian", alpha = 1, lambda = tau_seq, type.measure=c("mse"))

# récupération du tau minimum par validation croisée
lasso_cv = cv.glmnet(x = expli, y = eng, family = "gaussian", alpha = 1, lambda = tau_seq, type.measure="mse")
tau_min_lasso = lasso_cv$lambda.min
print(tau_min_lasso)

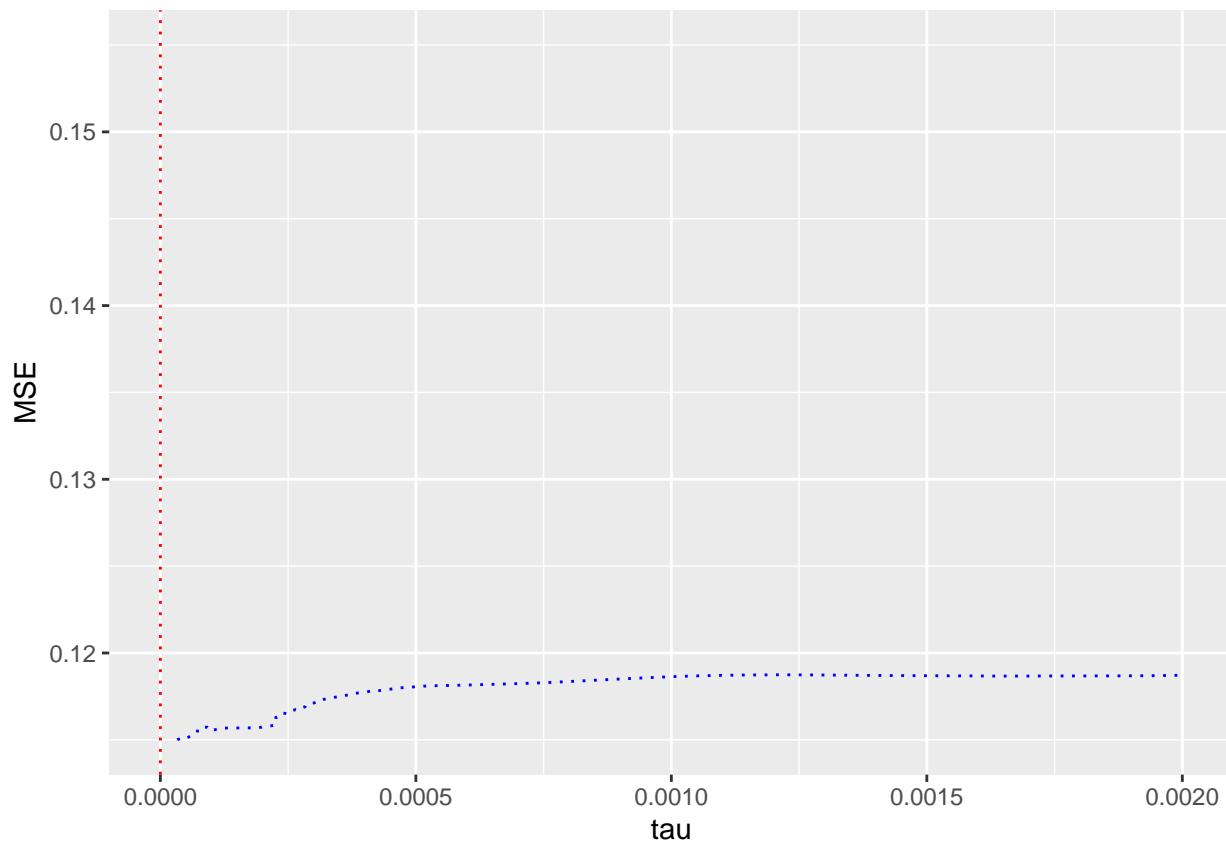
# affichage des estimations de tau
dflasso=data.frame(tau = rep(fitlasso$lambda,ncol(expli)), theta=as.vector(t(fitlasso$beta)),variable=rep(1:ncol(expli),each=1))
ggplot(dflasso,aes(x=tau,y=theta,col=variable)) +
  geom_line() +
  geom_vline(xintercept = tau_min_lasso, linetype = "dotted", color = "red") +
  theme(legend.position="right")

```

Lasso



```
## [1] 0
## Warning: Removed 2001 row(s) containing missing values (geom_path).
## Warning: Removed 33 row(s) containing missing values (geom_path).
## Warning: Removed 2001 row(s) containing missing values (geom_path).
```



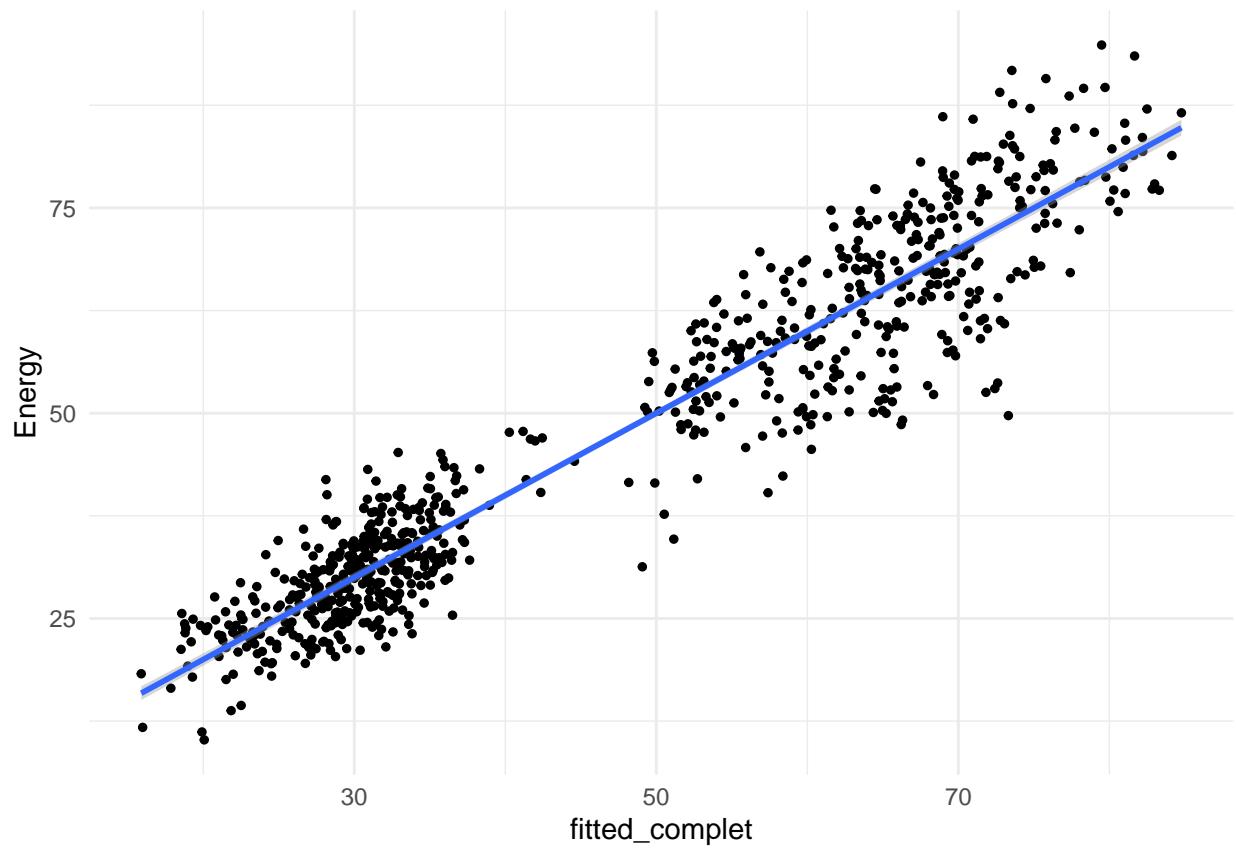
Model depending on all variables

```

model_complet = lm(Energy ~ . - Roof.area^2, data = data[,c(1:9)])
summary(model_complet)
data$fitted_complet = model_complet$fitted.values
ggplot(data) +
  aes(x = fitted_complet, y = Energy) +
  geom_point(size = 1L) +
  theme_minimal() + geom_smooth(method = "lm")

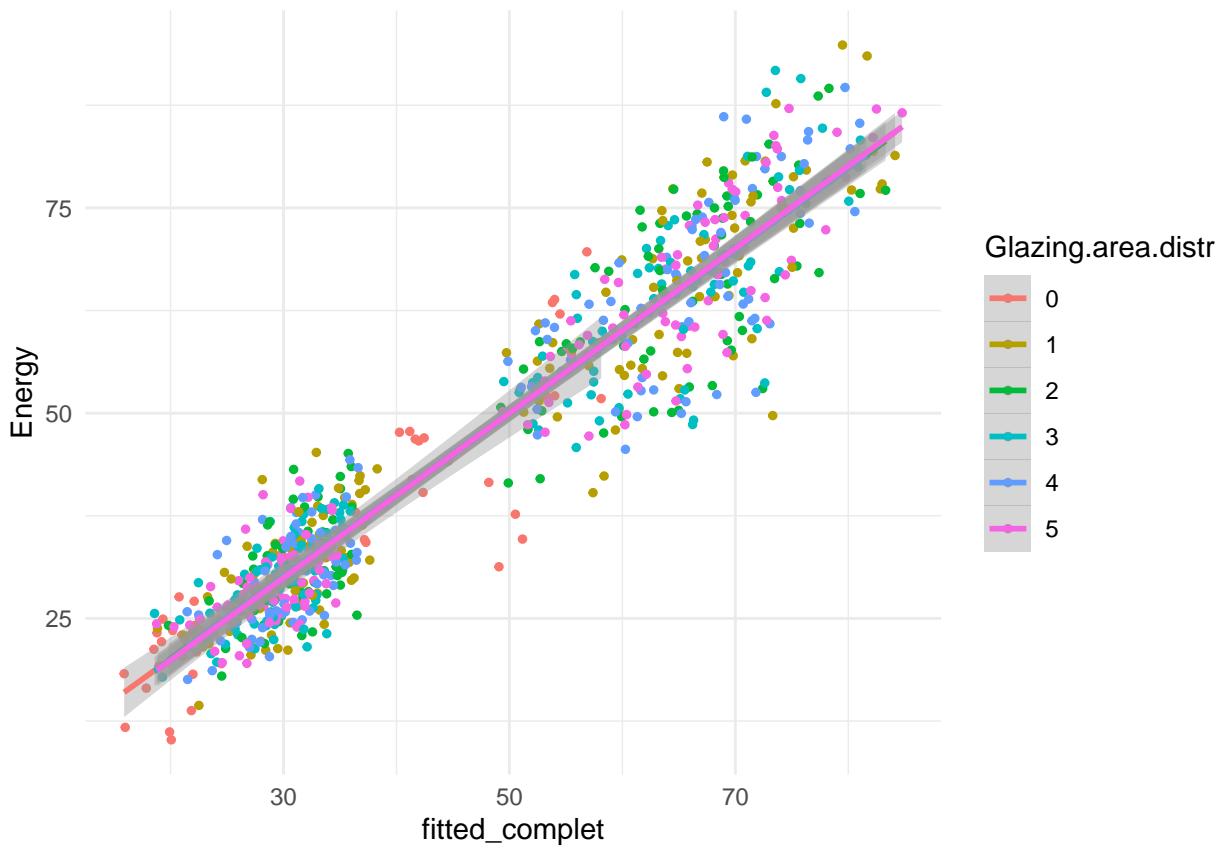
## `geom_smooth()` using formula 'y ~ x'

```



```
ggplot(data) +
  aes(x = fitted_complet, y = Energy, col = Glazing.area.distr) +
  geom_point(size = 1L) +
  theme_minimal() + geom_smooth(method = "lm")

## `geom_smooth()` using formula 'y ~ x'
```

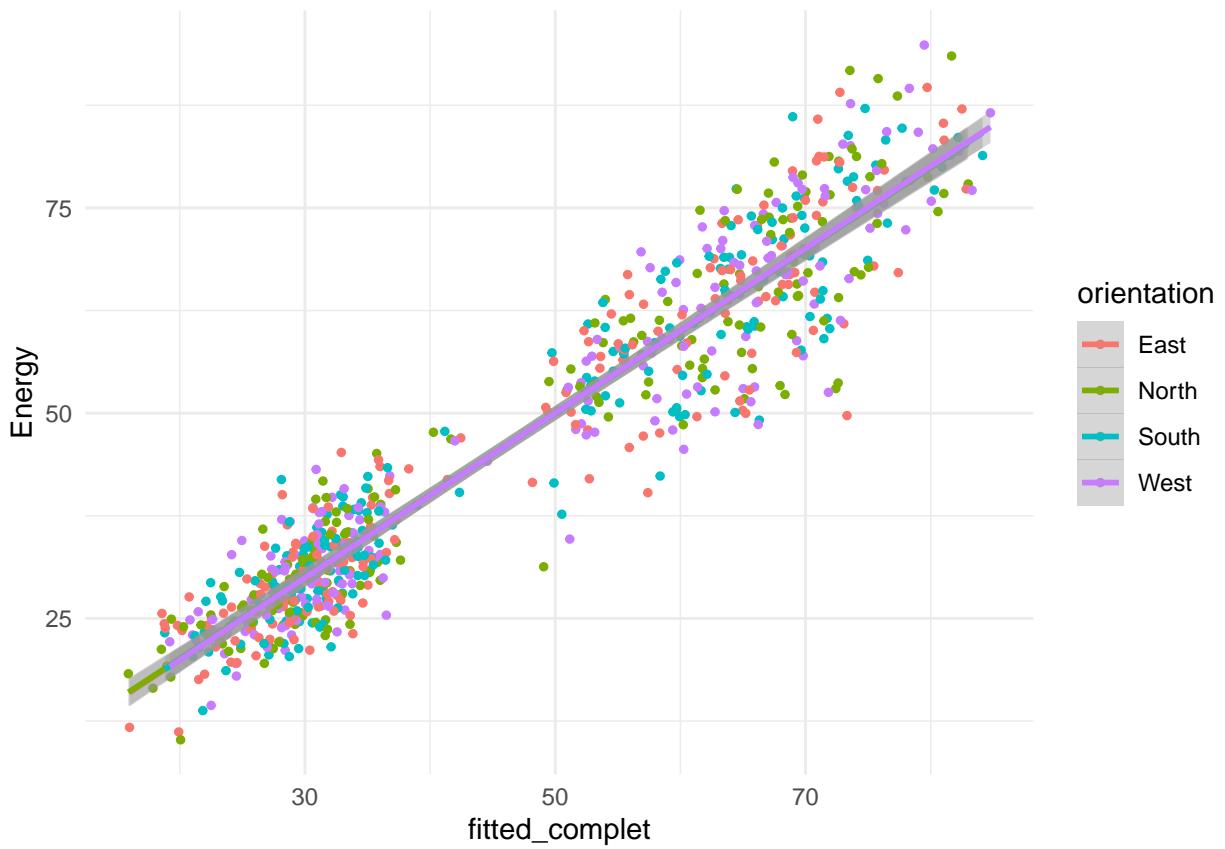


```

ggplot(data) +
  aes(x = fitted_complet, y = Energy, col = orientation) +
  geom_point(size = 1L) +
  theme_minimal() + geom_smooth(method = "lm")

## `geom_smooth()` using formula 'y ~ x'

```



```
##
## Call:
## lm(formula = Energy ~ (. - Roof.area)^2, data = data[, c(1:9)])
##
## Residuals:
##    Min     1Q   Median     3Q    Max 
## -23.606 -3.795  0.266  4.040 18.202 
##
## Coefficients: (1 not defined because of singularities)
##                               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)                 -3.503e+02  3.045e+02 -1.150 0.250446  
## Relative.compactness        6.018e+02  2.343e+02  2.569 0.010414  
## Surface.area                6.522e-01  2.846e-01  2.292 0.022229  
## Wall.area                   -1.218e+00  8.875e-01 -1.372 0.170433  
## Overall.height              4.298e+01  2.509e+01  1.713 0.087203  
## orientationNorth            -1.164e+01  8.791e+01 -0.132 0.894689  
## orientationSouth            2.343e+01  8.985e+01  0.261 0.794359  
## orientationWest             8.238e+01  8.781e+01  0.938 0.348522  
## Glazing.area                1.278e+01  2.662e+02  0.048 0.961726  
## Glazing.area.distr1         -1.383e+02  1.551e+02 -0.891 0.373119  
## Glazing.area.distr2         -9.461e+01  1.556e+02 -0.608 0.543254  
## Glazing.area.distr3         -1.570e+02  1.526e+02 -1.029 0.303815  
## Glazing.area.distr4         -1.047e+02  1.556e+02 -0.673 0.501231  
## Glazing.area.distr5         -1.072e+02  1.523e+02 -0.704 0.481924  
## Relative.compactness:Surface.area -1.340e+00  2.295e-01 -5.841 8e-09  
## Relative.compactness:Wall.area   2.127e+00  6.263e-01  3.397 0.000722  
## Relative.compactness:Overall.height -7.998e+01  2.105e+01 -3.799 0.000158
```

## Relative.compactness:orientationNorth	1.721e+00	4.739e+01	0.036	0.971049
## Relative.compactness:orientationSouth	-1.341e+01	4.888e+01	-0.274	0.783931
## Relative.compactness:orientationWest	-4.520e+01	4.720e+01	-0.958	0.338633
## Relative.compactness:Glazing.area	6.774e+01	1.459e+02	0.464	0.642603
## Relative.compactness:Glazing.area.distr1	9.149e+01	8.362e+01	1.094	0.274307
## Relative.compactness:Glazing.area.distr2	6.415e+01	8.424e+01	0.761	0.446628
## Relative.compactness:Glazing.area.distr3	9.985e+01	8.267e+01	1.208	0.227490
## Relative.compactness:Glazing.area.distr4	6.196e+01	8.402e+01	0.738	0.461057
## Relative.compactness:Glazing.area.distr5	7.095e+01	8.257e+01	0.859	0.390510
## Surface.area:Wall.area	1.726e-05	6.154e-04	0.028	0.977635
## Surface.area:Overall.height	7.588e-02	3.016e-02	2.516	0.012101
## Surface.area:orientationNorth	1.500e-02	8.007e-02	0.187	0.851480
## Surface.area:orientationSouth	-1.318e-02	8.189e-02	-0.161	0.872174
## Surface.area:orientationWest	-6.796e-02	7.983e-02	-0.851	0.394838
## Surface.area:Glazing.area	-9.404e-02	2.399e-01	-0.392	0.695195
## Surface.area:Glazing.area.distr1	8.470e-02	1.416e-01	0.598	0.549973
## Surface.area:Glazing.area.distr2	5.935e-02	1.420e-01	0.418	0.676218
## Surface.area:Glazing.area.distr3	1.064e-01	1.396e-01	0.763	0.445938
## Surface.area:Glazing.area.distr4	7.405e-02	1.422e-01	0.521	0.602786
## Surface.area:Glazing.area.distr5	5.764e-02	1.387e-01	0.416	0.677756
## Wall.area:Overall.height	-7.125e-02	2.973e-02	-2.397	0.016817
## Wall.area:orientationNorth	-1.949e-02	3.753e-02	-0.519	0.603677
## Wall.area:orientationSouth	-1.157e-02	3.840e-02	-0.301	0.763320
## Wall.area:orientationWest	8.745e-03	3.746e-02	0.233	0.815504
## Wall.area:Glazing.area	1.228e-01	1.101e-01	1.115	0.265124
## Wall.area:Glazing.area.distr1	5.075e-02	6.682e-02	0.759	0.447828
## Wall.area:Glazing.area.distr2	3.532e-02	6.716e-02	0.526	0.599129
## Wall.area:Glazing.area.distr3	3.966e-02	6.699e-02	0.592	0.554051
## Wall.area:Glazing.area.distr4	2.880e-02	6.701e-02	0.430	0.667504
## Wall.area:Glazing.area.distr5	6.111e-02	6.580e-02	0.929	0.353384
## Overall.height:orientationNorth	9.834e-01	1.854e+00	0.530	0.596063
## Overall.height:orientationSouth	-1.154e-02	1.872e+00	-0.006	0.995085
## Overall.height:orientationWest	-5.561e-01	1.858e+00	-0.299	0.764818
## Overall.height:Glazing.area	-1.955e+00	5.396e+00	-0.362	0.717300
## Overall.height:Glazing.area.distr1	-1.470e-01	3.308e+00	-0.044	0.964582
## Overall.height:Glazing.area.distr2	-1.105e-01	3.290e+00	-0.034	0.973231
## Overall.height:Glazing.area.distr3	2.814e-01	3.270e+00	0.086	0.931449
## Overall.height:Glazing.area.distr4	9.690e-01	3.290e+00	0.294	0.768476
## Overall.height:Glazing.area.distr5	1.293e-01	3.248e+00	0.040	0.968261
## orientationNorth:Glazing.area	-4.695e+00	5.627e+00	-0.834	0.404354
## orientationSouth:Glazing.area	-4.017e+00	5.587e+00	-0.719	0.472371
## orientationWest:Glazing.area	-4.731e+00	5.621e+00	-0.842	0.400313
## orientationNorth:Glazing.area.distr1	4.462e+00	3.341e+00	1.336	0.182059
## orientationSouth:Glazing.area.distr1	5.852e-01	3.353e+00	0.175	0.861493
## orientationWest:Glazing.area.distr1	4.525e-01	3.360e+00	0.135	0.892915
## orientationNorth:Glazing.area.distr2	4.948e+00	3.350e+00	1.477	0.140113
## orientationSouth:Glazing.area.distr2	6.536e-01	3.361e+00	0.194	0.845877
## orientationWest:Glazing.area.distr2	1.369e+00	3.369e+00	0.406	0.684629
## orientationNorth:Glazing.area.distr3	2.655e+00	3.345e+00	0.794	0.427735
## orientationSouth:Glazing.area.distr3	1.491e+00	3.349e+00	0.445	0.656404
## orientationWest:Glazing.area.distr3	-7.015e-01	3.371e+00	-0.208	0.835198
## orientationNorth:Glazing.area.distr4	2.099e+00	3.351e+00	0.626	0.531326
## orientationSouth:Glazing.area.distr4	8.512e-01	3.356e+00	0.254	0.799876
## orientationWest:Glazing.area.distr4	-2.320e+00	3.365e+00	-0.690	0.490739

	1.864e+00	3.373e+00	0.553	0.580778
## orientationNorth:Glazing.area.distr5	1.888e+00	3.373e+00	0.560	0.575920
## orientationSouth:Glazing.area.distr5	9.687e-01	3.377e+00	0.287	0.774315
## orientationWest:Glazing.area.distr5	1.151e+01	6.249e+00	1.842	0.065886
## Glazing.area:Glazing.area.distr1	4.415e+00	6.235e+00	0.708	0.479089
## Glazing.area:Glazing.area.distr2	4.555e+00	6.267e+00	0.727	0.467535
## Glazing.area:Glazing.area.distr3	4.506e+00	6.274e+00	0.718	0.472870
## Glazing.area:Glazing.area.distr4	NA	NA	NA	NA
## Glazing.area:Glazing.area.distr5	NA	NA	NA	NA
##				
## (Intercept)	*			
## Relative.compactness	*			
## Surface.area	*			
## Wall.area	.			
## Overall.height	.			
## orientationNorth				
## orientationSouth				
## orientationWest				
## Glazing.area				
## Glazing.area.distr1				
## Glazing.area.distr2				
## Glazing.area.distr3				
## Glazing.area.distr4				
## Glazing.area.distr5				
## Relative.compactness:Surface.area	***			
## Relative.compactness:Wall.area	***			
## Relative.compactness:Overall.height	***			
## Relative.compactness:orientationNorth				
## Relative.compactness:orientationSouth				
## Relative.compactness:orientationWest				
## Relative.compactness:Glazing.area				
## Relative.compactness:Glazing.area.distr1				
## Relative.compactness:Glazing.area.distr2				
## Relative.compactness:Glazing.area.distr3				
## Relative.compactness:Glazing.area.distr4				
## Relative.compactness:Glazing.area.distr5				
## Surface.area:Wall.area				
## Surface.area:Overall.height	*			
## Surface.area:orientationNorth				
## Surface.area:orientationSouth				
## Surface.area:orientationWest				
## Surface.area:Glazing.area				
## Surface.area:Glazing.area.distr1				
## Surface.area:Glazing.area.distr2				
## Surface.area:Glazing.area.distr3				
## Surface.area:Glazing.area.distr4				
## Surface.area:Glazing.area.distr5				
## Wall.area:Overall.height	*			
## Wall.area:orientationNorth				
## Wall.area:orientationSouth				
## Wall.area:orientationWest				
## Wall.area:Glazing.area				
## Wall.area:Glazing.area.distr1				
## Wall.area:Glazing.area.distr2				
## Wall.area:Glazing.area.distr3				

```

## Wall.area:Glazing.area.distr4
## Wall.area:Glazing.area.distr5
## Overall.height:orientationNorth
## Overall.height:orientationSouth
## Overall.height:orientationWest
## Overall.height:Glazing.area
## Overall.height:Glazing.area.distr1
## Overall.height:Glazing.area.distr2
## Overall.height:Glazing.area.distr3
## Overall.height:Glazing.area.distr4
## Overall.height:Glazing.area.distr5
## orientationNorth:Glazing.area
## orientationSouth:Glazing.area
## orientationWest:Glazing.area
## orientationNorth:Glazing.area.distr1
## orientationSouth:Glazing.area.distr1
## orientationWest:Glazing.area.distr1
## orientationNorth:Glazing.area.distr2
## orientationSouth:Glazing.area.distr2
## orientationWest:Glazing.area.distr2
## orientationNorth:Glazing.area.distr3
## orientationSouth:Glazing.area.distr3
## orientationWest:Glazing.area.distr3
## orientationNorth:Glazing.area.distr4
## orientationSouth:Glazing.area.distr4
## orientationWest:Glazing.area.distr4
## orientationNorth:Glazing.area.distr5
## orientationSouth:Glazing.area.distr5
## orientationWest:Glazing.area.distr5
## Glazing.area:Glazing.area.distr1
## Glazing.area:Glazing.area.distr2
## Glazing.area:Glazing.area.distr3
## Glazing.area:Glazing.area.distr4
## Glazing.area:Glazing.area.distr5
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.438 on 690 degrees of freedom
## Multiple R-squared: 0.9055, Adjusted R-squared: 0.895
## F-statistic: 85.87 on 77 and 690 DF, p-value: < 2.2e-16
model_reduit1 = lm(Energy ~ . - Roof.area - orientation, data = data[,c(1:9)])
anova(model_complet, model_reduit1)
summary(model_reduit1)

## Analysis of Variance Table
##
## Model 1: Energy ~ ((Relative.compactness + Surface.area + Wall.area +
##     Roof.area + Overall.height + orientation + Glazing.area +
##     Glazing.area.distr) - Roof.area)^2
## Model 2: Energy ~ (Relative.compactness + Surface.area + Wall.area + Roof.area +
##     Overall.height + orientation + Glazing.area + Glazing.area.distr) -
##     Roof.area - orientation
##   Res.Df   RSS Df Sum of Sq    F    Pr(>F)
## 1     690 28596

```

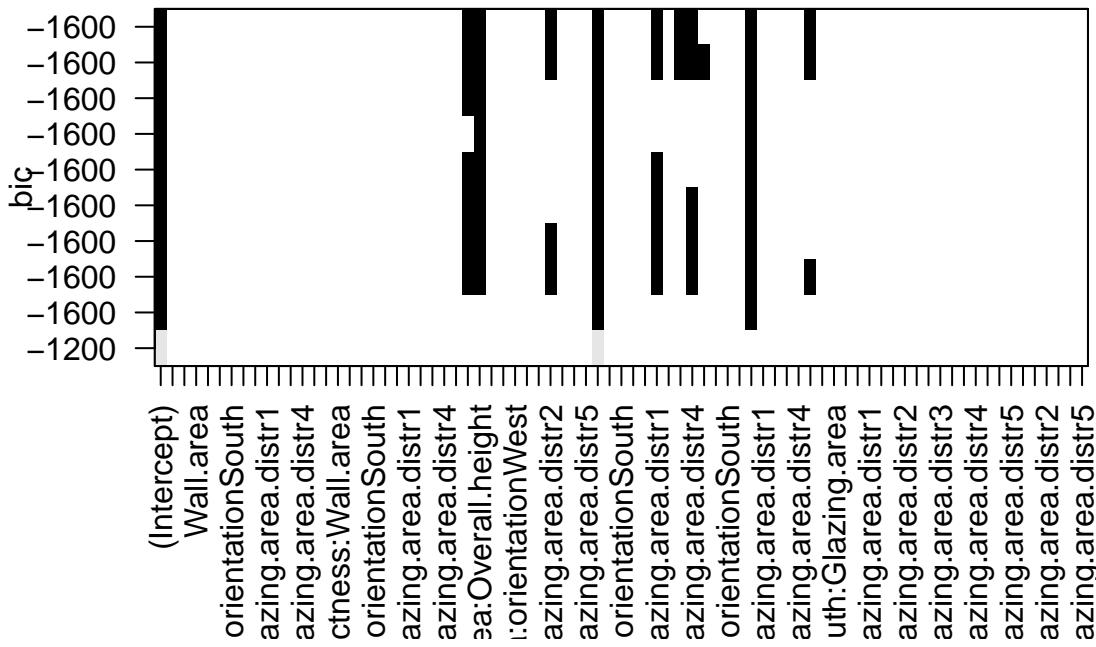
```

## 2      757 35863 -67    -7266.5 2.6169 5.048e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Call:
## lm(formula = Energy ~ . - Roof.area - orientation, data = data[,,
##   c(1:9)])
##
## Residuals:
##       Min     1Q Median     3Q    Max
## -21.2257 -4.3582 -0.0094  4.3013 25.6243
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)               67.90870  32.16901  2.111  0.03510 *
## Relative.compactness -75.98738  17.45481 -4.353 1.52e-05 ***
## Surface.area            -0.08022  0.02930 -2.738  0.00633 **
## Wall.area                0.08415  0.01402  6.003 3.01e-09 ***
## Overall.height           9.69799  0.68521 14.153 < 2e-16 ***
## Glazing.area             31.62185  2.09682 15.081 < 2e-16 ***
## Glazing.area.distr1      6.83971  1.26087  5.425 7.82e-08 ***
## Glazing.area.distr2      6.42294  1.26318  5.085 4.64e-07 ***
## Glazing.area.distr3      5.63231  1.26186  4.464 9.29e-06 ***
## Glazing.area.distr4      6.66705  1.26124  5.286 1.64e-07 ***
## Glazing.area.distr5      5.64903  1.26383  4.470 9.03e-06 ***
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.883 on 757 degrees of freedom
## Multiple R-squared:  0.8815, Adjusted R-squared:  0.8799
## F-statistic: 563.1 on 10 and 757 DF,  p-value: < 2.2e-16
selectf = regsubsets(Energy ~ (. - Roof.area)^2, data = data[,c(1:9)], nbest=1, nvmax=10, method="forward")

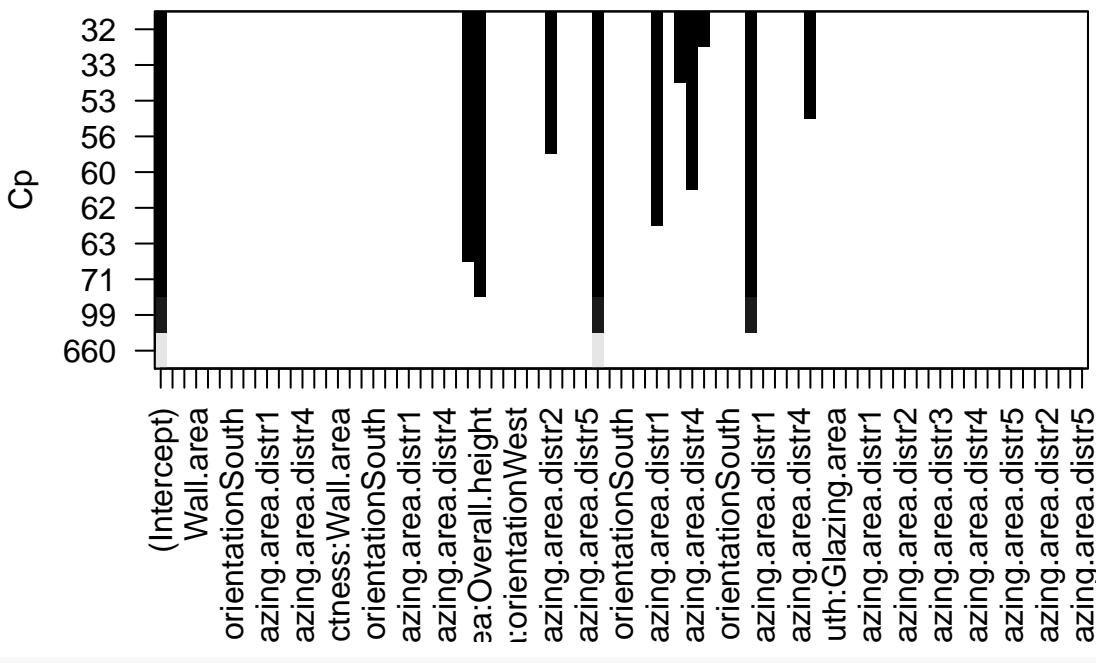
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax, force.in =
## force.in, : 1 linear dependencies found
plot(selectf, scale="bic", main="Forward BIC")

```

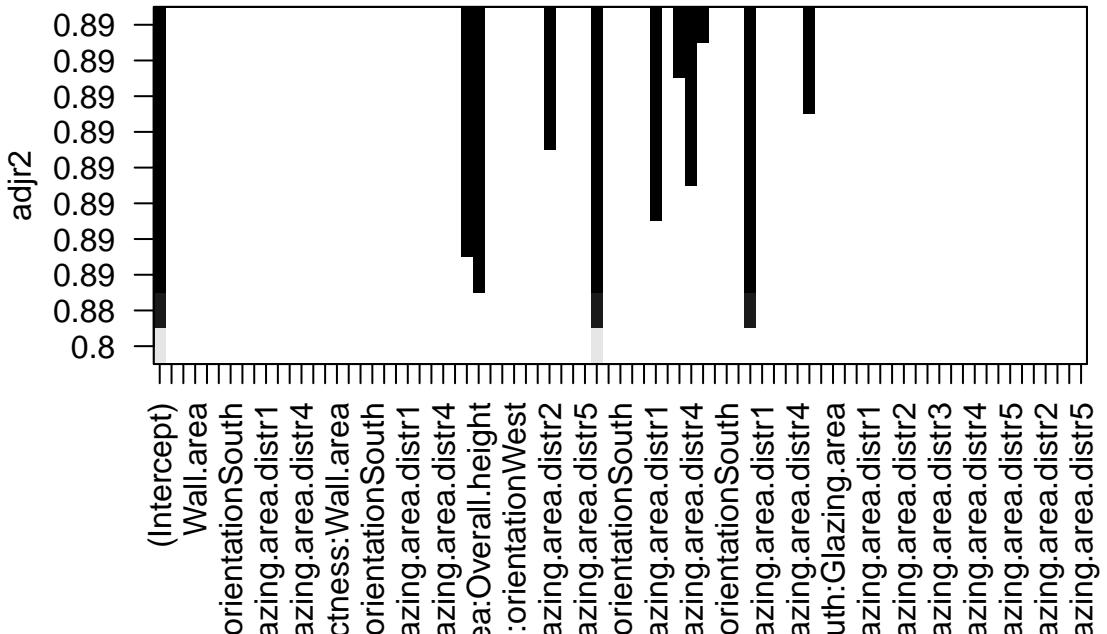
Forward BIC



Forward Cp



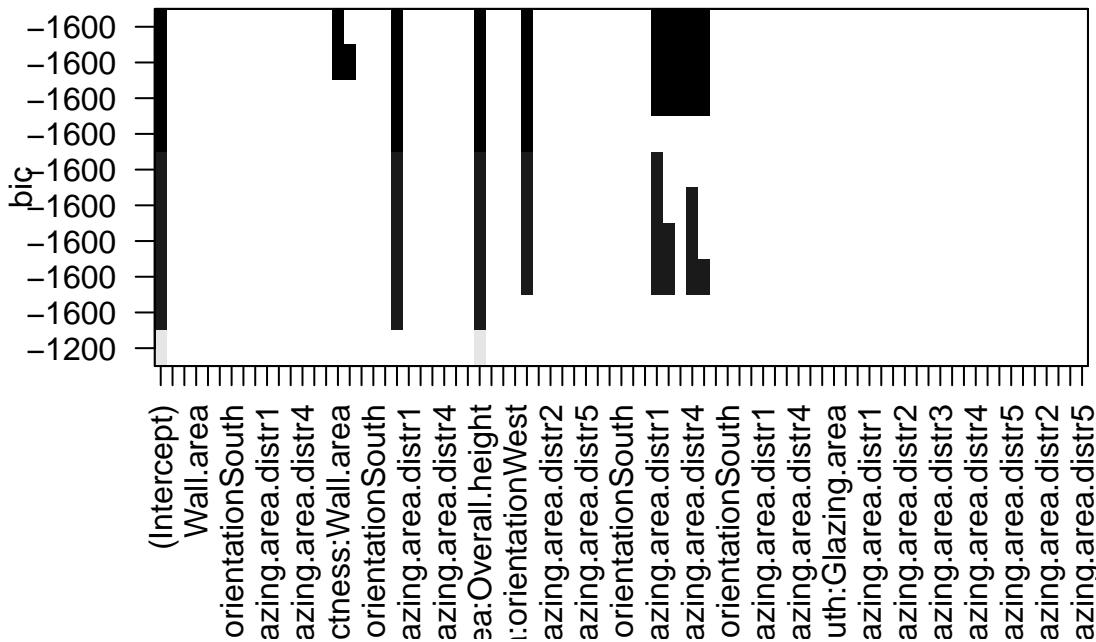
Forward adjusted R²



```
selectb = regsubsets(Energy ~ . - Roof.area)^2, data = data[,c(1:9)], nbest=1, nvmax=10, method="backward"

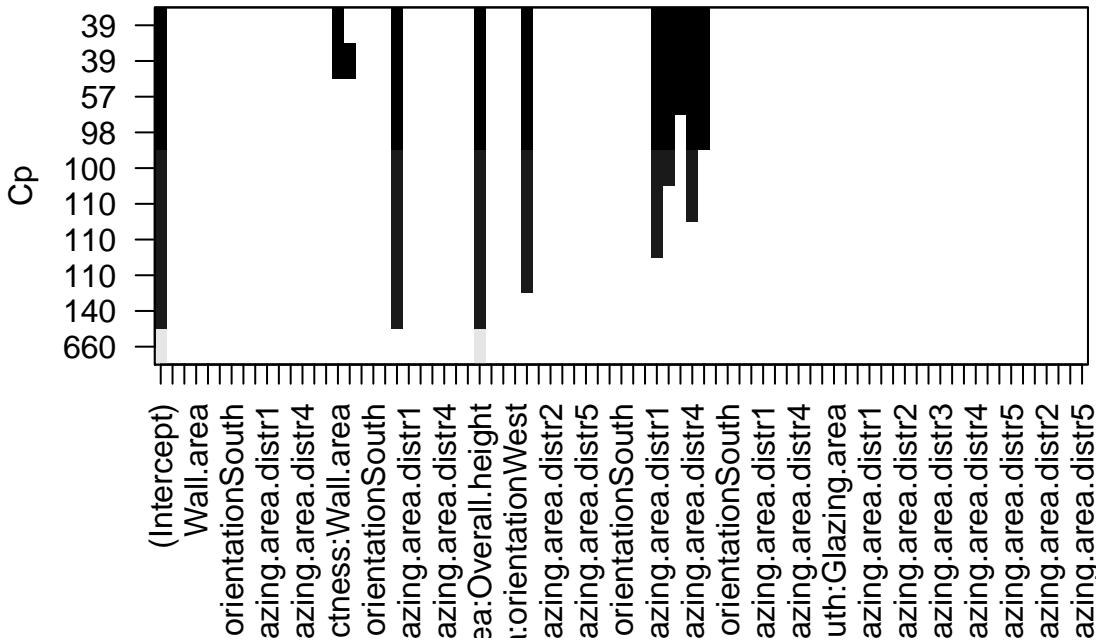
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax, force.in =
## force.in, : 1 linear dependencies found
plot(selectb, scale="bic", main="Backward BIC")
```

Backward BIC



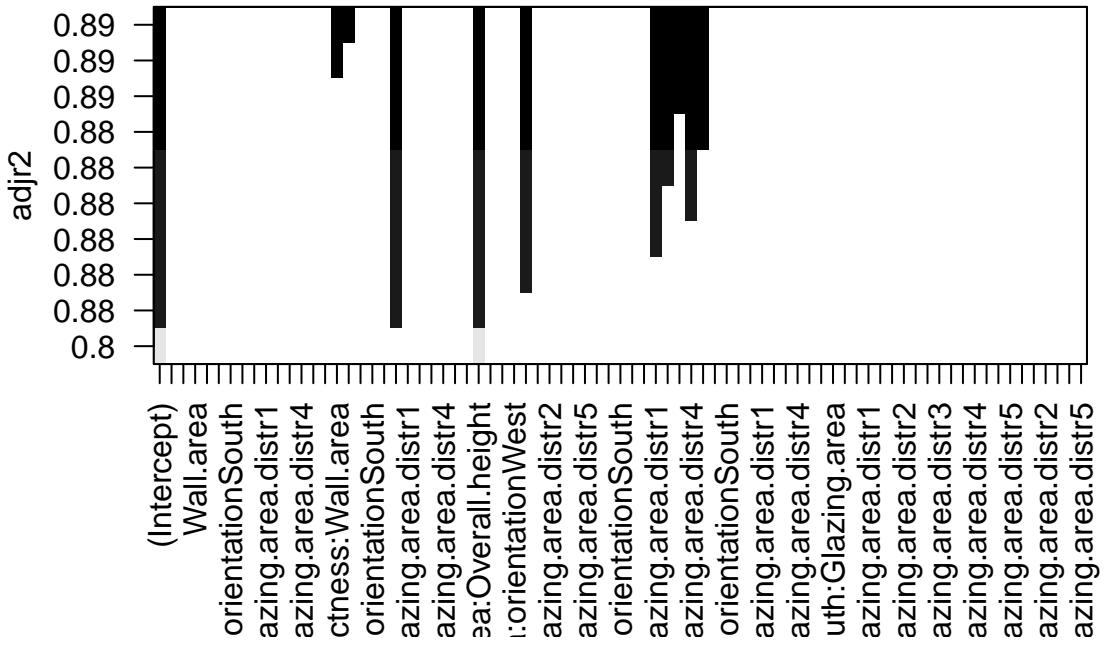
```
plot(selectb,scale="Cp",main="Backward Cp")
```

Backward Cp



```
plot(selectb,scale="adjr2",main="Backward adjusted R2")
```

Backward adjusted R²



```
# model_reduit2  
# anova()
```