TP R 2: Solution exercice 2

1. Données midwest

```
# install.packages('ggplot2') #Pour installer le package
library(ggplot2)

par(mfrow=c(1,2)) #Deux graphiques sur la même ligne

plot(midwest$area,midwest$poptotal,
     pch=19,
     xlab='Surface',ylab='Population',
     col=midwest$inmetro+1, # inmetro=0=rural=noir, inmetro=1=urbain=rouge
     main='Contées américaines'
     )
legend(x='topleft',pch=19,col=1:2,legend=c('Rural','Urbain'),box.lty=0)

plot(midwest$area,midwest$poptotal,
     pch=as.numeric(as.factor(midwest$state)),
     col=midwest$inmetro+1,
     xlab='Surface', ylab='Population',
     main='Contées américaines'
     )
legend(x='topleft',pch=c(1:5,1,1),
       col=c(rep(1,5),1:2),
       legend=c(levels(as.factor(midwest$state)),'Rural','Urbain'),
       box.lty=0)
```

2. Données adult

2.1 Importation des données

```
url.data <- 'https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data'
adult = read.table(url.data, sep = ",", na.strings = " ?") # attention à l'espace avant le ?
```

On peut trouver le nom des variables dans le fichier `adult.names`:

```
names(adult) = c("age","workclass","fnlwgt","education",
                 "education.num","marital.status","occupation",
                 "relationship","race","sex","capital.gain",
                 "capital.loss","hours.per.week","native.country","class")
attach(adult)
```

2.2 Description des variables

Pour savoir si une variable est quantitative ou qualitative, on trouve sa classe. En effet, le mode n'est pas utile à ce fin, parce que le mode d'un facteur est `numeric` exactement comme pour les variables numériques.

```
sapply(adult,class) # sapply() applique une fonction à tous les elements d'un vecteur ou liste et rend,
```

```
##            age      workclass         fnlwgt      education  education.num
##      "integer"       "factor"      "integer"       "factor"      "integer"
## marital.status     occupation   relationship           race            sex
##       "factor"       "factor"       "factor"       "factor"       "factor"
##   capital.gain   capital.loss hours.per.week native.country          class
##      "integer"      "integer"      "integer"       "factor"       "factor"
```

2.2.1 Variables quantitatives

```
quant=names(adult)[sapply(adult,class)=='integer']
```

Pour chacune des variables `age, capital.gain, capital.loss, hours.per.week, fnlwgt, education.num` nous calculons min, moyenne, médiane, max, et écart type.

Par exemple:

```
summary(age,na.rm=T)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   17.00   28.00   37.00   38.58   48.00   90.00
```

```
min(age,na.rm=T)
```

```
## [1] 17
```

```
round(mean(age,na.rm=T),1)
```

```
## [1] 38.6
```

```
median(age,na.rm=T)
```

```
## [1] 37
```

```
max(age,na.rm=T)
```

```
## [1] 90
```

```
sd(age,na.rm=T)
```

```
## [1] 13.64043
```

Il est préférable de faire une table avec toutes les variables. En particulier on peut travailler directement à partir du sous data frame des variables quantitatives et appliquer la fonction de notre choix (`min()`, `median()`...) à chacune de ses colonnes à l'aide de `sapply()`. En effet, `sapply(X,FUN)` applique la fonction `FUN` à tous les éléments de `X` (aux colonnes si `X` est un data frame) et, si possible, rend le résultat dans un vecteur.

```
minimum=sapply(adult[,quant],min)
moyenne=round(sapply(adult[,quant], mean),2)
maximum=sapply(adult[,quant], max)
ecart_type=round(sapply(adult[,quant], sd),2)
res=data.frame(minimum,moyenne,maximum,ecart_type)
res
```

```
##                minimum   moyenne maximum ecart_type
## age                 17     38.58      90      13.64
## fnlwgt           12285 189778.37 1484705  105549.98
```

```
## education.num        1     10.08      16       2.57
## capital.gain         0   1077.65   99999    7385.29
## capital.loss         0     87.30    4356     402.96
## hours.per.week       1     40.44      99      12.35
```

On peut utiliser la fonction `kable()` du package `knitr` pour afficher cette table au format `Markdown`:

```
require(knitr) # même chose que library(knitr)
```

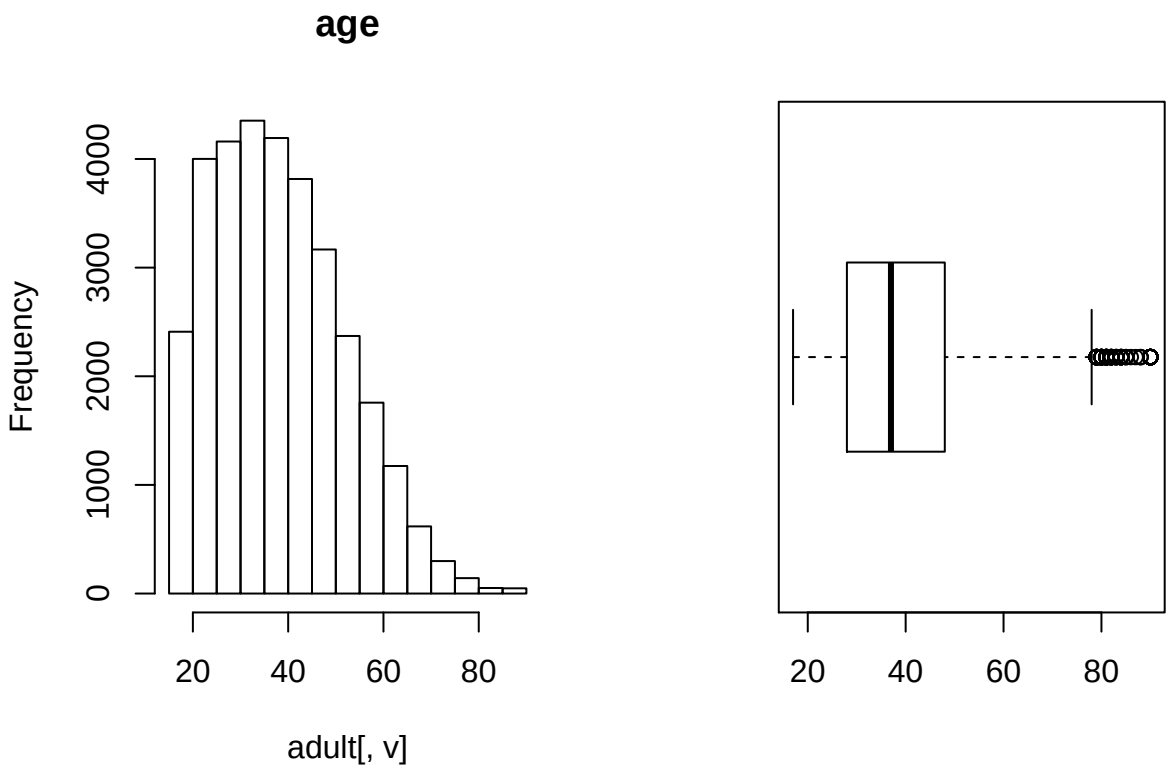
```
## Warning: package 'knitr' was built under R version 3.4.4
```

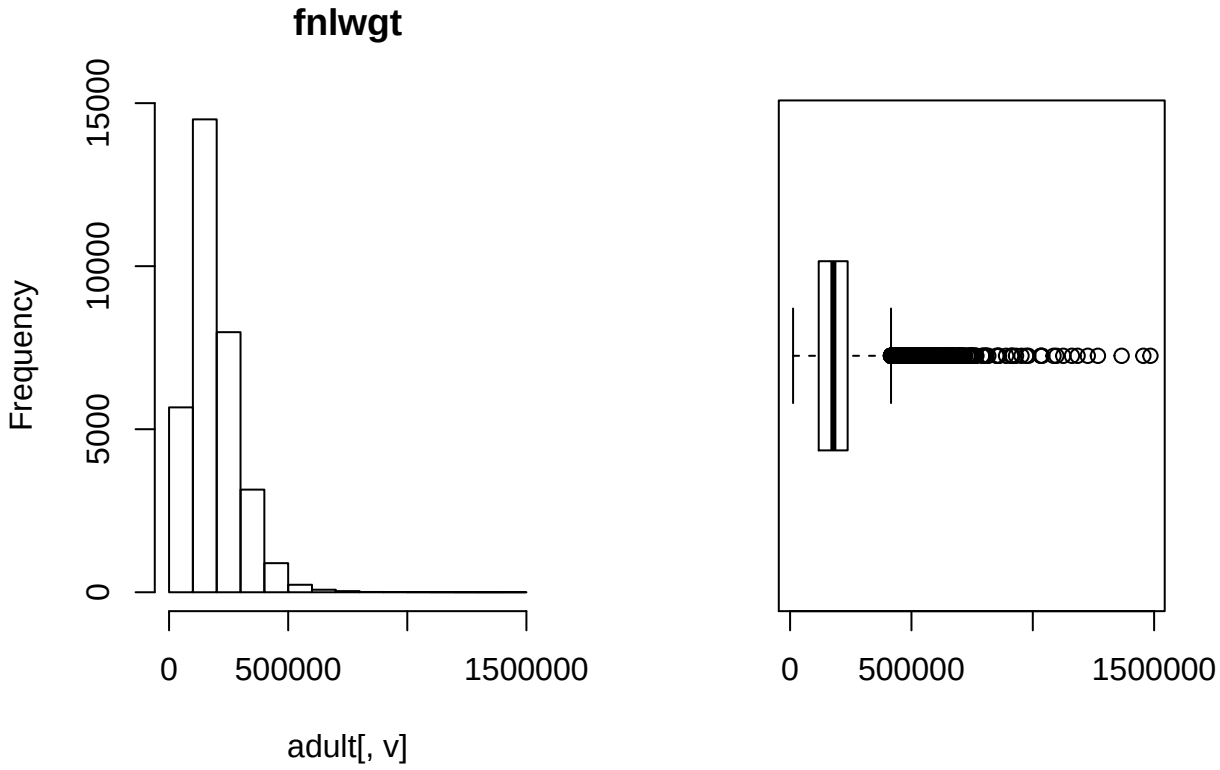
```
kable(res)
```

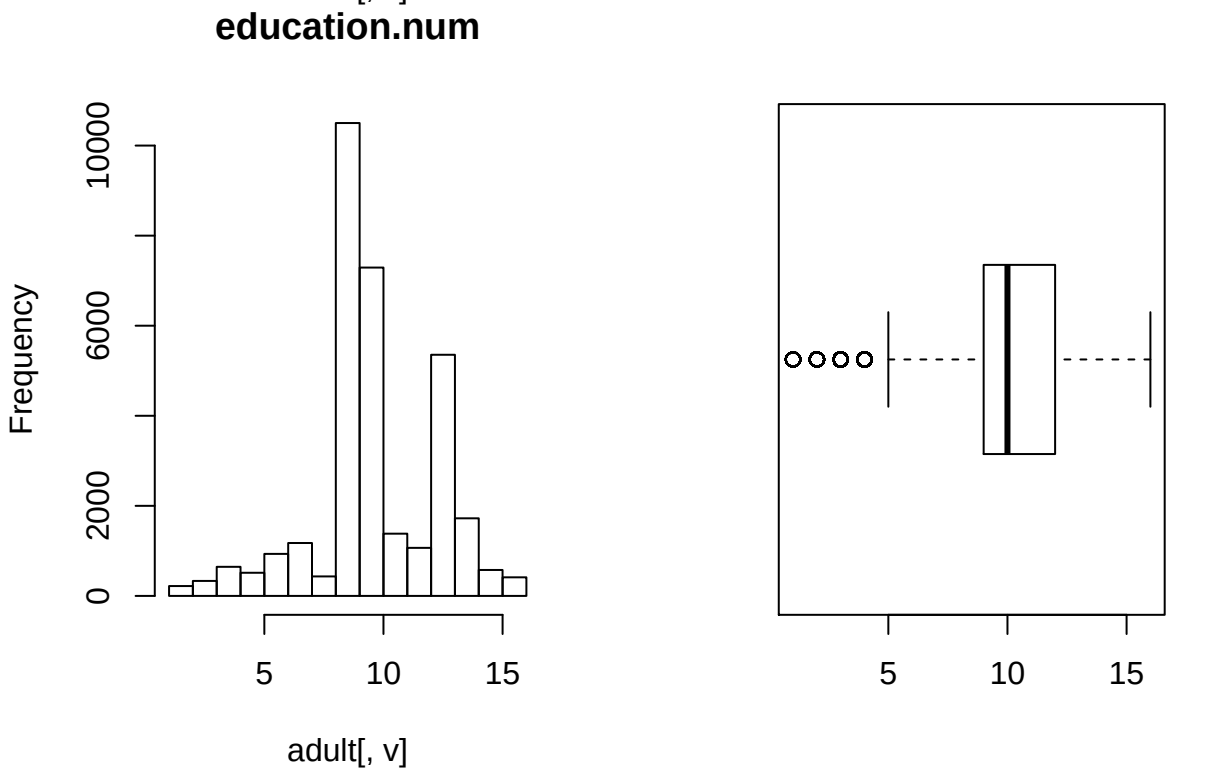
	minimum	moyenne	maximum	ecart_type
age	17	38.58	90	13.64
fnlwgt	12285	189778.37	1484705	105549.98
education.num	1	10.08	16	2.57
capital.gain	0	1077.65	99999	7385.29
capital.loss	0	87.30	4356	402.96
hours.per.week	1	40.44	99	12.35

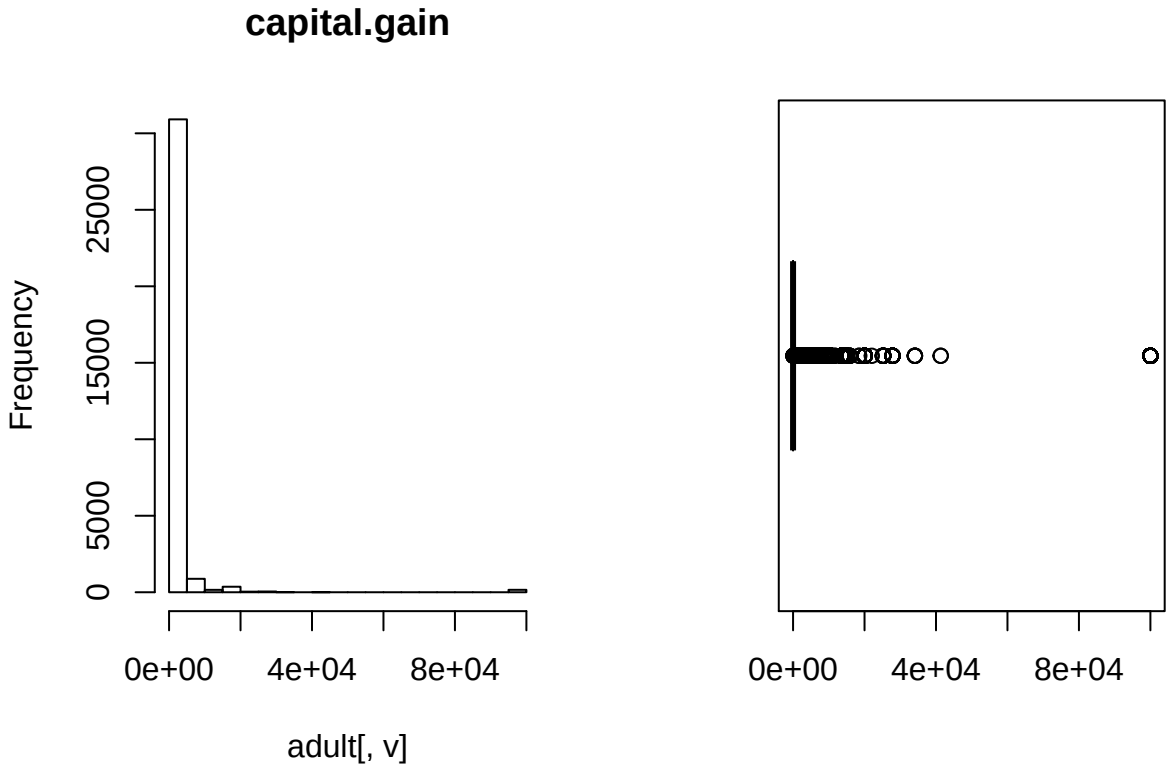
Pour chaque variable on affiche l'histogramme et le boxplot:

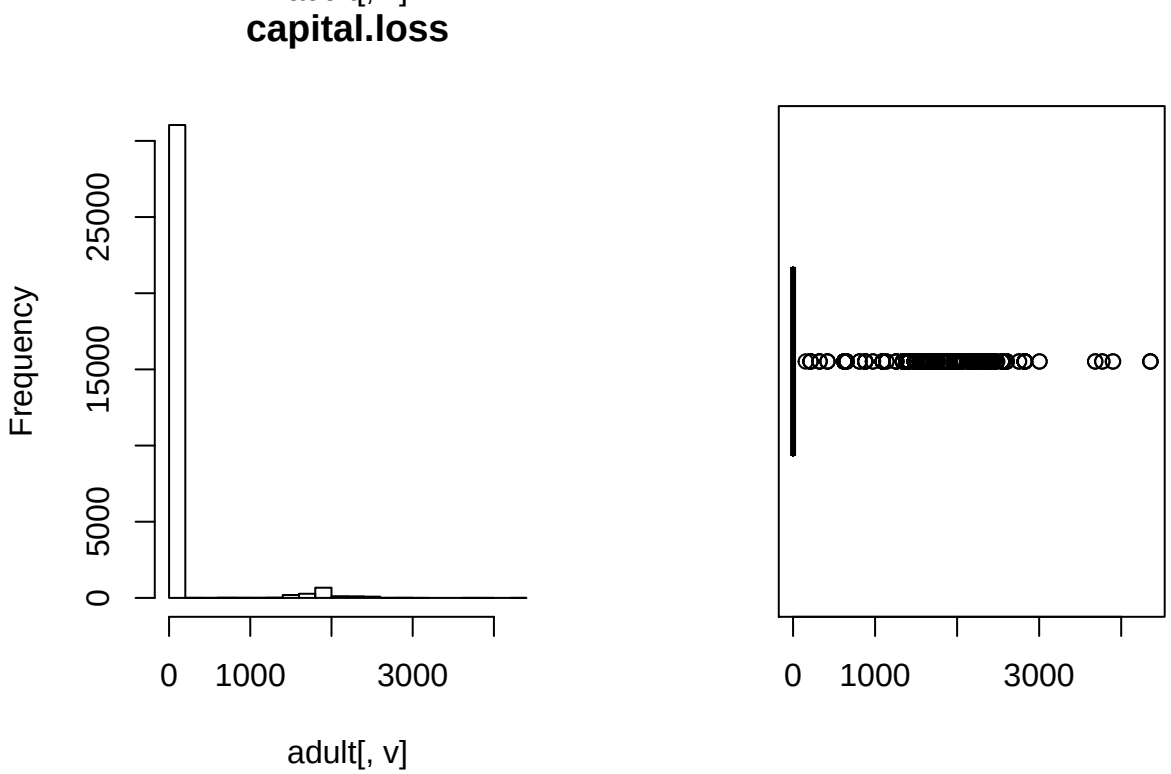
```
for(v in quant){
  par(mfrow=c(1,2)) # Pour avoir 1 ligne avec 2 graphiques
  hist(adult[,v],main=v)
  boxplot(adult[,v],horizontal=T)
}
```

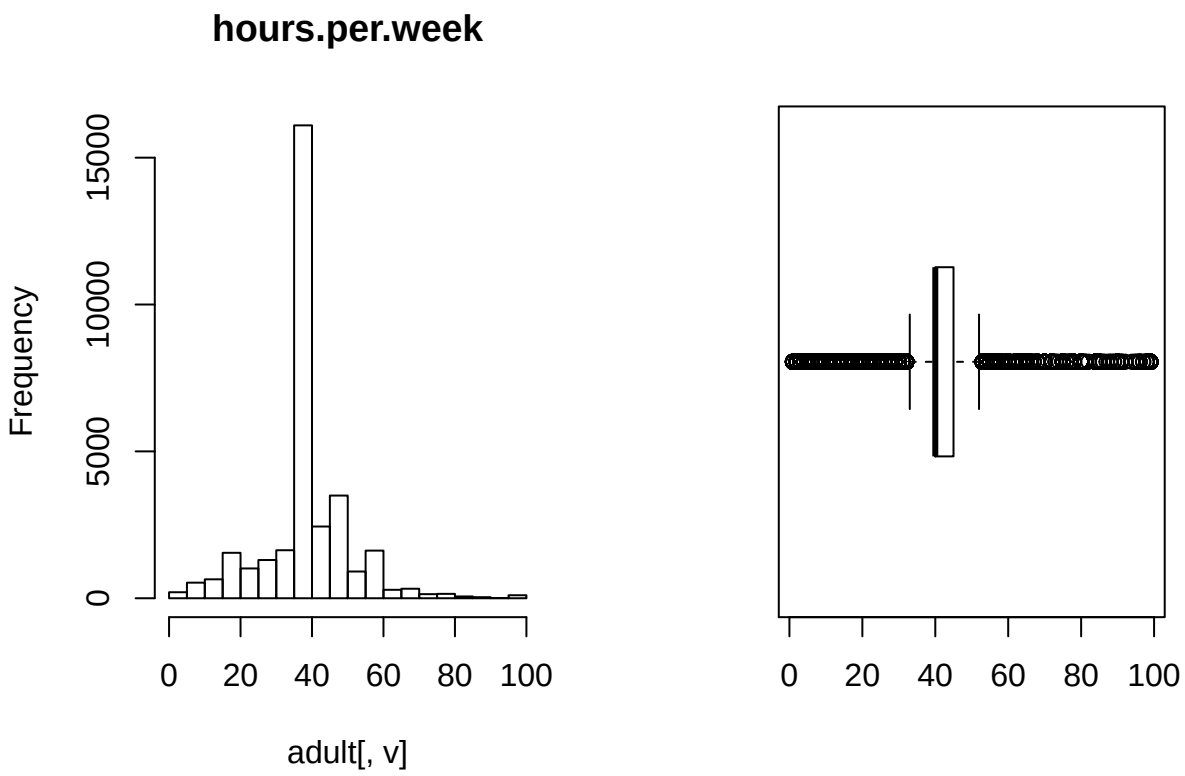



fnlwgt
Frequency
0
5000
10000
15000
0
500000
1500000
adult[, v]
0
500000
1500000


education.num
Frequency
0
2000
6000
10000
5
10
15
adult[, v]
5
10
15


capital.gain
Frequency
0
5000
15000
25000
0e+00
4e+04
8e+04
adult[, v]
0e+00
4e+04
8e+04


capital.loss
Frequency
0
5000
15000
25000
0
1000
3000
adult[, v]
0
1000
3000



Pour réinitialiser les paramètres graphiques:

```
dev.off()
```

```
## null device
##           1
```

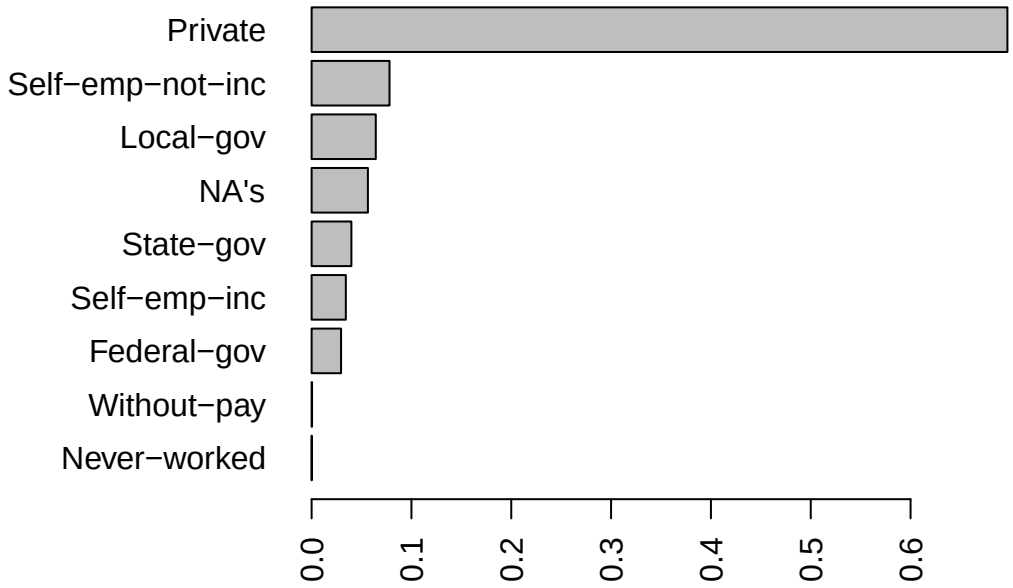
2.2.2 Variables qualitatives

```
qual=names(adult)[sapply(adult,class)=='factor']
qual
```

```
## [1] "workclass"      "education"      "marital.status" "occupation"
## [5] "relationship"   "race"           "sex"            "native.country"
## [9] "class"
```

Pour chaque variable qualitatives on affiche son diagramme à bâton et la table des comptages:

```
par(mar=c(5,12,4,2)+0.1)
# l'argument mar permet de gérer la dimension des marges.
# Ici on augmente la marge à gauche pour que toutes les labels des barres
# puissent être correctement affichés. Le default est mar=c(5, 4, 4, 2) + 0.1
barplot(
  sort( #sort() ordonne les éléments de l'argument en ordre croissant
    prop.table(summary(workclass))
  ),
  horiz=T,las=2,main='workclass') # las=2 pour avoir les labels en horizontal
```

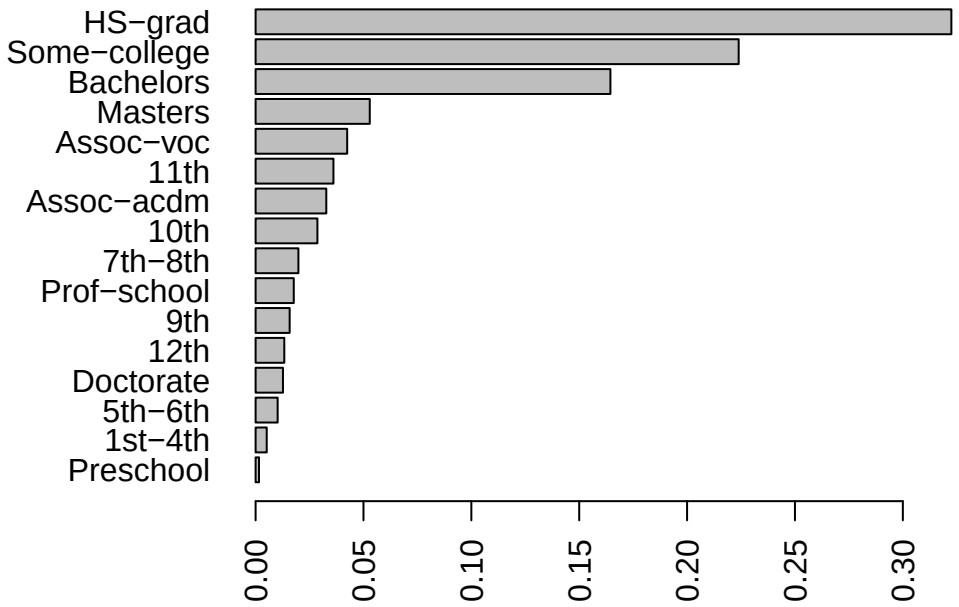



```
kable(as.data.frame(
  sort(summary(workclass),
       decreasing=T)),
  col.names='Effectifs',caption='Workclass')
```

Table 2: Workclass

	Effectifs
Private	22696
Self-emp-not-inc	2541
Local-gov	2093
NA's	1836
State-gov	1298
Self-emp-inc	1116
Federal-gov	960
Without-pay	14
Never-worked	7

```
par(mar=c(5,12,4,2)+0.1)
barplot(
  sort(
    prop.table(summary(education))
  ),
  horiz=T,las=2,main='education')
```




```
kable(as.data.frame(
  sort(summary(education),
       decreasing=T)),
  col.names='Effectifs',caption='Education')
```

Table 3: Education

	Effectifs
HS-grad	10501
Some-college	7291
Bachelors	5355
Masters	1723
Assoc-voc	1382
11th	1175
Assoc-acdm	1067
10th	933
7th-8th	646
Prof-school	576
9th	514
12th	433
Doctorate	413
5th-6th	333
1st-4th	168
Preschool	51

```
dev.off()
```

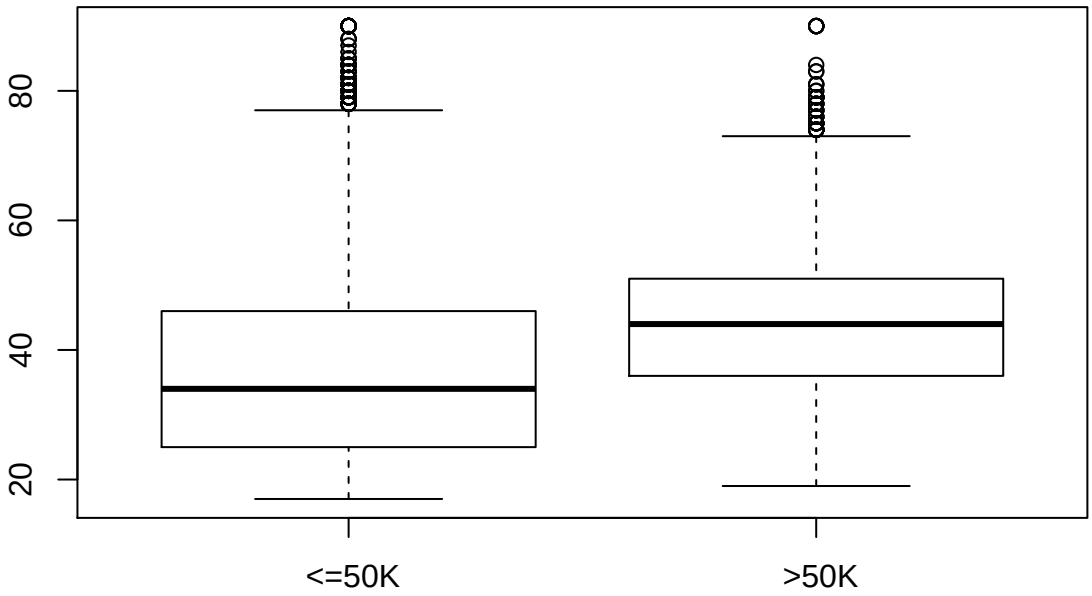
```
## null device
##           1
```

. . . et ainsi de suite pour les autres variables. . .

2.3 Lien entre les variables age et class

Pour chacune des modalité de class on regarde la distribution des ages:

```
boxplot(age~class)
```




```
haut_revenus=round(
  c(min(age[class==' >50K']), mean(age[class==' >50K']),
    max(age[class==' >50K']),sd(age[class==' >50K'])),
  2)

bas_revenus=round(
  c(min(age[class==' <=50K']), mean(age[class==' <=50K']),
    max(age[class==' <=50K']),sd(age[class==' <=50K'])),
  2)

res = as.data.frame(cbind(bas_revenus, haut_revenus),
            row.names=c('minimum','moyenne','maximum','écart type'))
kable(res,row.names=T,caption='age')
```

Table 4: age

	bas_revenus	haut_revenus
minimum	17.00	19.00
moyenne	36.78	44.25
maximum	90.00	90.00
écart type	14.02	10.52