cart correction

Muyao GUO

2025-10-11

rm(list=objects()); graphics.off() library(kernlab) data(spam)

set.seed(12345) test = sample(1:length(spam$type),1536) train = -test train = spam[train,] #3065 observations test = spam[test,] #1536 observations

— RF ————————————————————————————-

```
library(kernlab)
data(spam)
dim(spam)
```

```
## [1] 4601   58
```

```
str(spam)
```

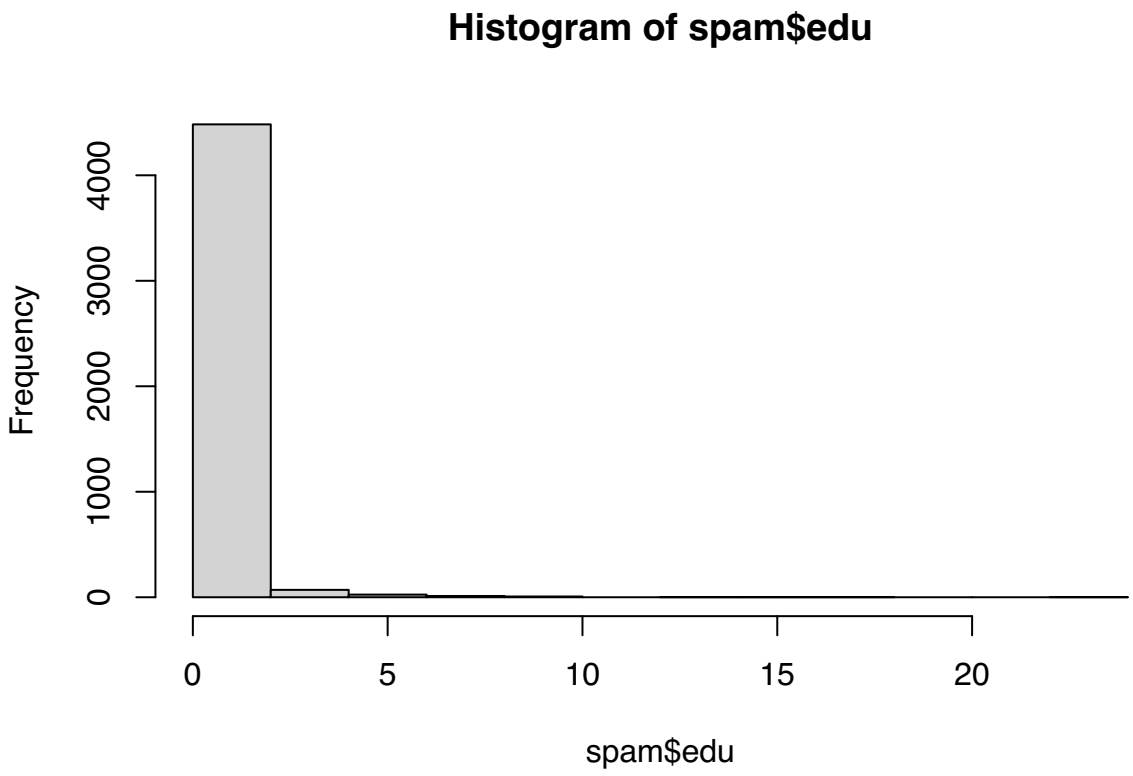
```
## 'data.frame':    4601 obs. of  58 variables:
##  $ make             : num  0 0.21 0.06 0 0 0 0 0 0.15 0.06 ...
##  $ address          : num  0.64 0.28 0 0 0 0 0 0 0 0.12 ...
##  $ all              : num  0.64 0.5 0.71 0 0 0 0 0 0.46 0.77 ...
##  $ num3d            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ our              : num  0.32 0.14 1.23 0.63 0.63 1.85 1.92 1.88 0.61 0.19 ...
##  $ over             : num  0 0.28 0.19 0 0 0 0 0 0 0.32 ...
##  $ remove           : num  0 0.21 0.19 0.31 0.31 0 0 0 0.3 0.38 ...
##  $ internet         : num  0 0.07 0.12 0.63 0.63 1.85 0 1.88 0 0 ...
##  $ order            : num  0 0 0.64 0.31 0.31 0 0 0 0.92 0.06 ...
##  $ mail             : num  0 0.94 0.25 0.63 0.63 0 0.64 0 0.76 0 ...
##  $ receive          : num  0 0.21 0.38 0.31 0.31 0 0.96 0 0.76 0 ...
##  $ will             : num  0.64 0.79 0.45 0.31 0.31 0 1.28 0 0.92 0.64 ...
##  $ people           : num  0 0.65 0.12 0.31 0.31 0 0 0 0 0.25 ...
##  $ report           : num  0 0.21 0 0 0 0 0 0 0 0 ...
##  $ addresses        : num  0 0.14 1.75 0 0 0 0 0 0 0.12 ...
##  $ free             : num  0.32 0.14 0.06 0.31 0.31 0 0.96 0 0 0 ...
##  $ business         : num  0 0.07 0.06 0 0 0 0 0 0 0 ...
##  $ email            : num  1.29 0.28 1.03 0 0 0 0.32 0 0.15 0.12 ...
##  $ you              : num  1.93 3.47 1.36 3.18 3.18 0 3.85 0 1.23 1.67 ...
##  $ credit           : num  0 0 0.32 0 0 0 0 0 3.53 0.06 ...
##  $ your             : num  0.96 1.59 0.51 0.31 0.31 0 0.64 0 2 0.71 ...
##  $ font             : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ num000           : num  0 0.43 1.16 0 0 0 0 0 0 0.19 ...
```

```
##  $ money            : num  0 0.43 0.06 0 0 0 0 0 0.15 0 ...
##  $ hp               : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ hpl              : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ george           : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ num650           : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ lab              : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ labs             : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ telnet           : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ num857           : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ data             : num  0 0 0 0 0 0 0 0 0.15 0 ...
##  $ num415           : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ num85            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ technology       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ num1999          : num  0 0.07 0 0 0 0 0 0 0 0 ...
##  $ parts            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ pm               : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ direct           : num  0 0 0.06 0 0 0 0 0 0 0 ...
##  $ cs               : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ meeting          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ original         : num  0 0 0.12 0 0 0 0 0 0.3 0 ...
##  $ project          : num  0 0 0 0 0 0 0 0 0 0.06 ...
##  $ re               : num  0 0 0.06 0 0 0 0 0 0 0 ...
##  $ edu              : num  0 0 0.06 0 0 0 0 0 0 0 ...
##  $ table            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ conference       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ charSemicolon    : num  0 0 0.01 0 0 0 0 0 0 0.04 ...
##  $ charRoundbracket : num  0 0.132 0.143 0.137 0.135 0.223 0.054 0.206 0.271 0.03 ...
##  $ charSquarebracket: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ charExclamation  : num  0.778 0.372 0.276 0.137 0.135 0 0.164 0 0.181 0.244 ...
##  $ charDollar       : num  0 0.18 0.184 0 0 0 0.054 0 0.203 0.081 ...
##  $ charHash         : num  0 0.048 0.01 0 0 0 0 0 0.022 0 ...
##  $ capitalAve       : num  3.76 5.11 9.82 3.54 3.54 ...
##  $ capitalLong      : num  61 101 485 40 40 15 4 11 445 43 ...
##  $ capitalTotal     : num  278 1028 2259 191 191 ...
##  $ type             : Factor w/ 2 levels "nonspam","spam": 2 2 2 2 2 2 2 2 2 2 ...
```

```
table(spam$type)
```

```
##
## nonspam    spam
##    2788    1813
```

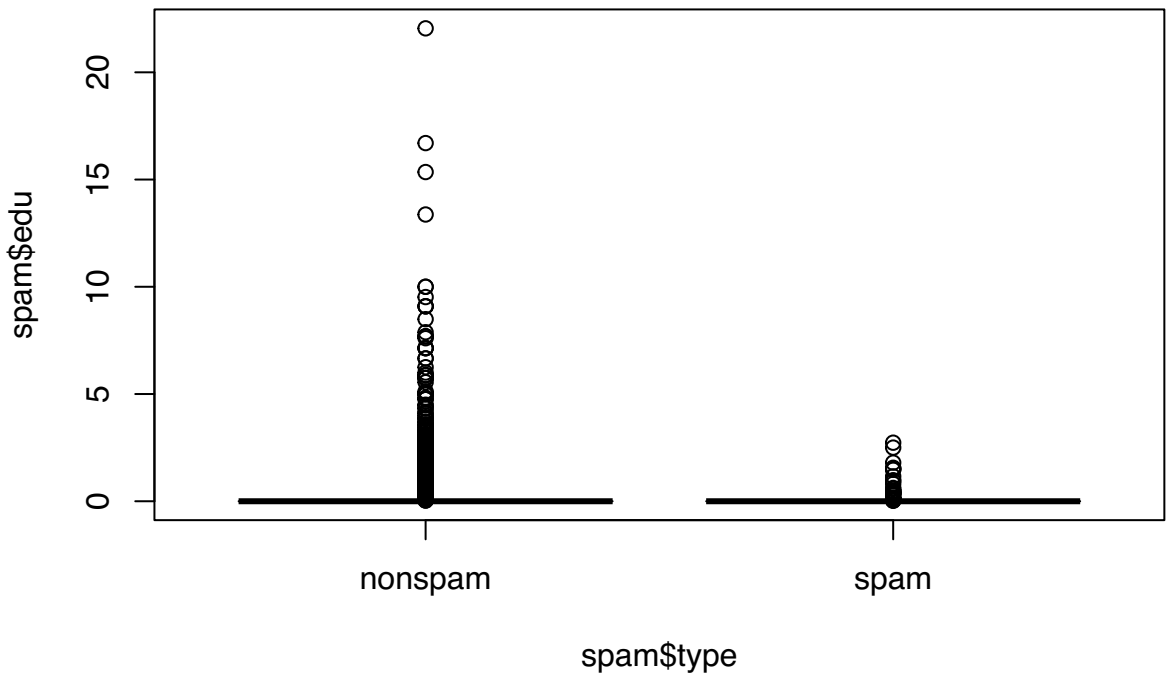
```
hist(spam$edu)
```



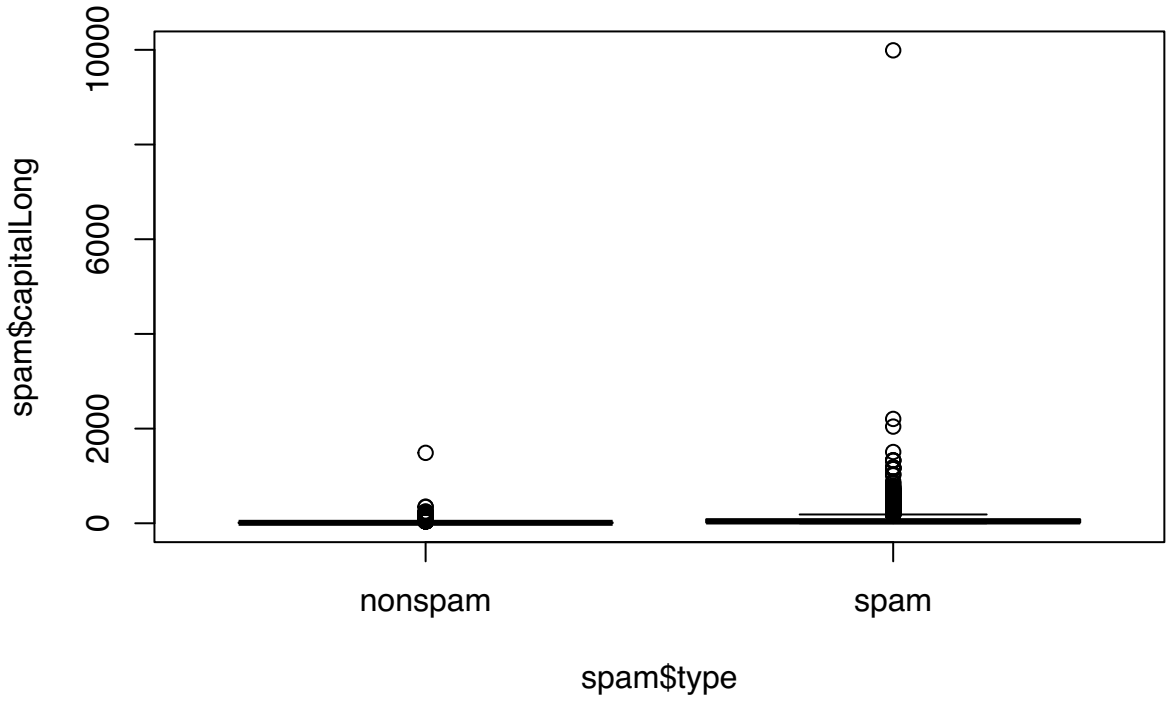

```
summary(spam$edu)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.0000  0.1798  0.0000 22.0500
```

```
boxplot(spam$edu~spam$type)
```




```
boxplot(spam$capitalLong~spam$type)
```




```
library(randomForest)
```

```
## randomForest 4.7-1.2

## Type rfNews() to see new features/changes/bug fixes.
```

```
set.seed(9170)
test = sample(1:length(spam$type),1533)
train = -test
train = spam[train, ]
test = spam[test,]
fit.rf=randomForest(type~., data=train)
fit.rf
```

```
##
## Call:
##  randomForest(formula = type ~ ., data = train)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 4.95%
## Confusion matrix:
##         nonspam spam class.error
## nonspam    1789   60  0.03244997
## spam         92 1127  0.07547170
```

fit.rf 中的参数

```
names(fit.rf)
```

RF在训练过程中每增加一棵树后的OOB误差(袋外误差)

```
##  [1] "call"            "type"            "predicted"       "err.rate"
##  [5] "confusion"       "votes"           "oob.times"       "classes"
##  [9] "importance"      "importanceSD"    "localImportance" "proximity"
## [13] "ntree"           "mtry"            "forest"          "y"
## [17] "test"            "inbag"           "terms"
```

Erreur OOB

estimateur

```
pred.rf=predict(fit.rf,type="class")
mean(pred.rf!=train$type)
```

直接返回预测类别

"prob" → prob

```
## [1] 0.04954368
```

Erreur training

```
pred.rf=predict(fit.rf,newdata=train,type="class")
mean(pred.rf!=train$type)
```

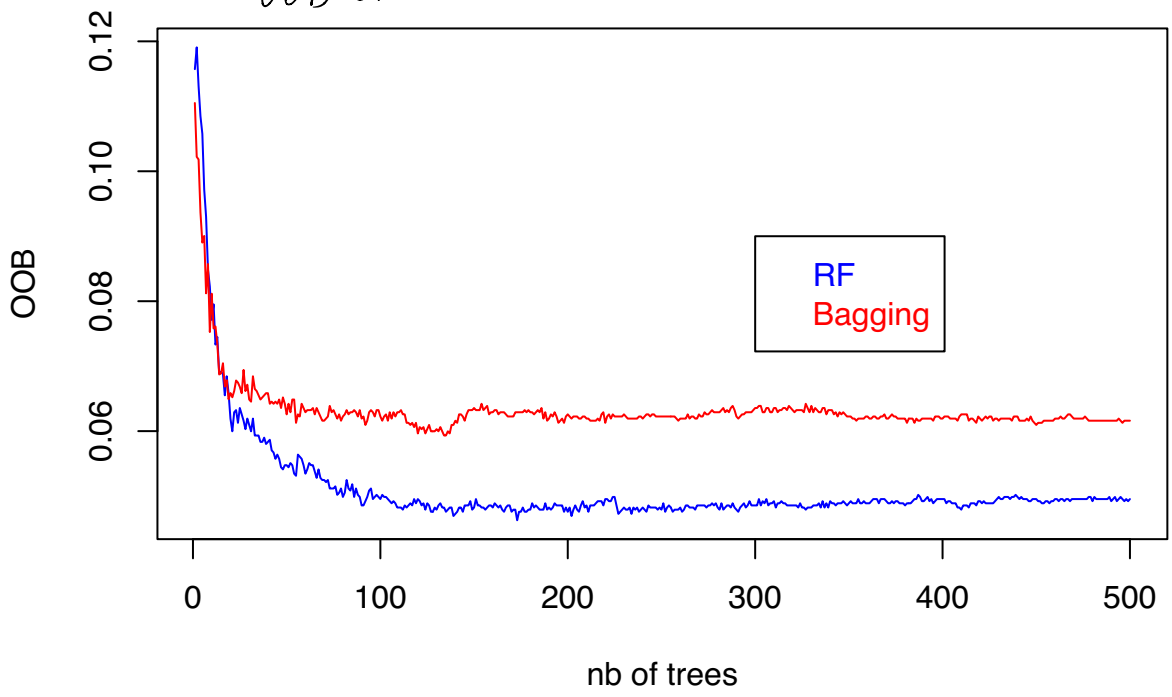
```
## [1] 0.003585398
```

```
# calibrartion de mtry et ntree
plot(1:fit.rf$ntree,fit.rf$err.rate[,1],ylab="OOB",xlab ="nb of trees",type="l", col="blue")
fit.bag=randomForest(type~., data=train, mtry=57)
lines(1:fit.bag$ntree,fit.bag$err.rate[,1],col="red")
legend(x=300,y=0.09,c("RF","Bagging"),text.col=c("blue","red"))
```

分裂时考虑的特征数

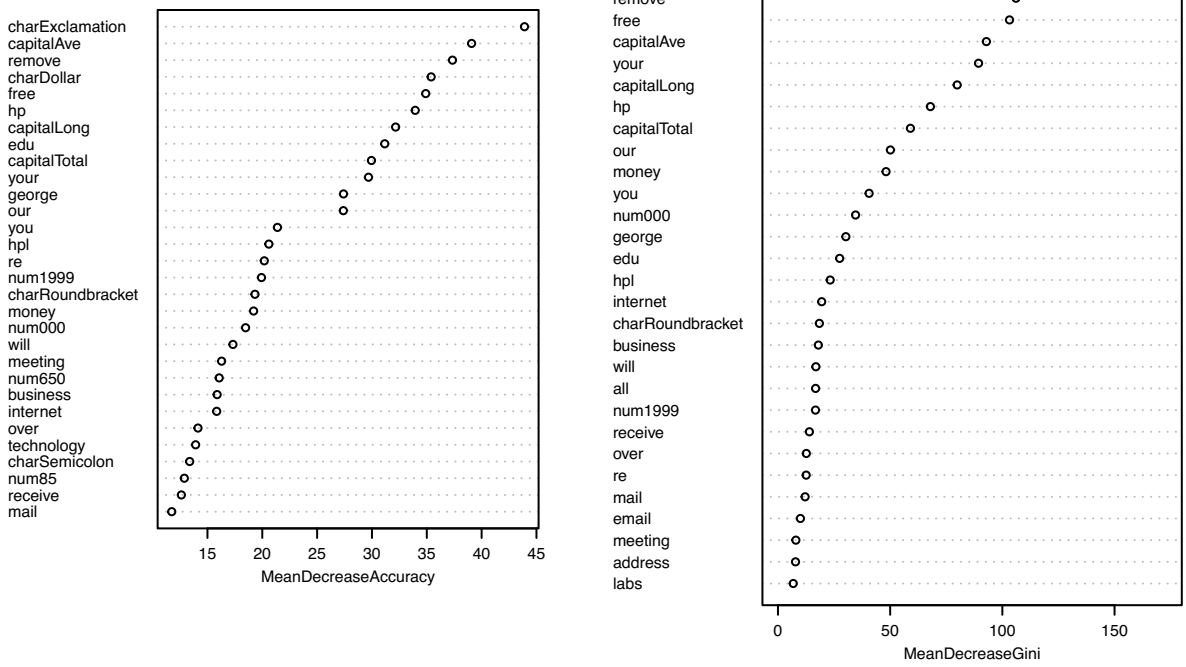
RF会自动进行内部交叉验证(OOB误差),不用手动划分 traing. testing

OOB erreur ≈ test erreur

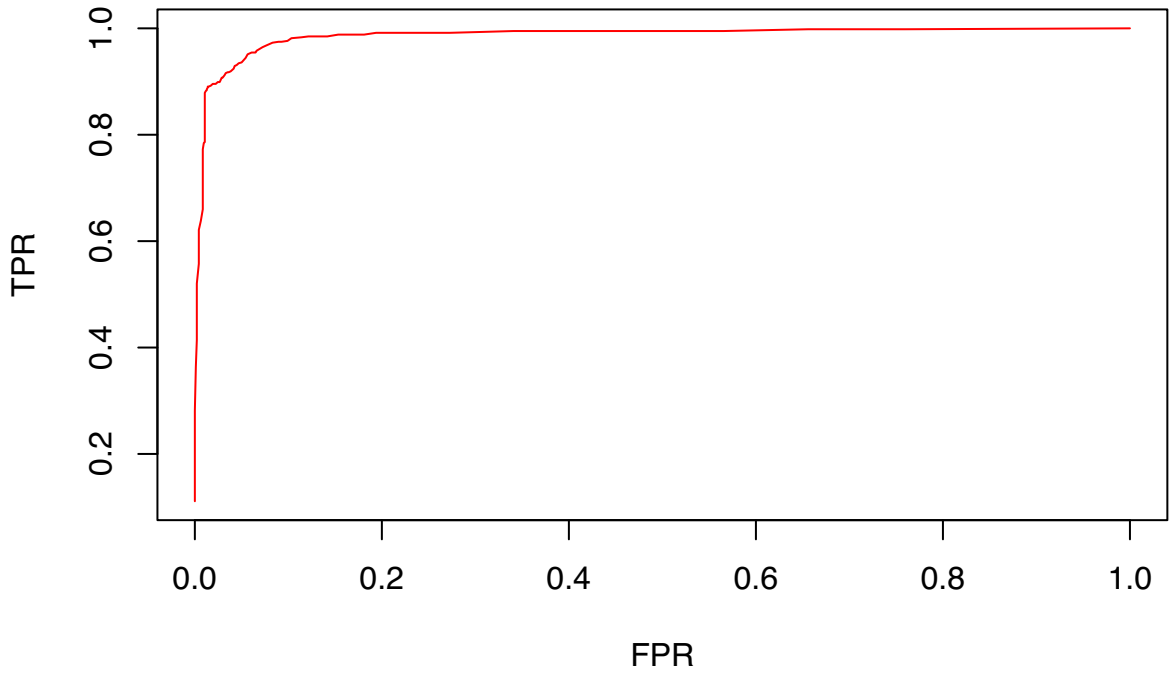



```
# on conserve mtry=7 et ntree=500

# importance des variables
fit.rf=randomForest(type~., data=train, importance=T)
varImpPlot(fit.rf,cex=.5)
```

fit.rf




```
# courbe ROC
s=seq(0,1,.01) #seuil de décision s
score.rf=predict(fit.rf,newdata=test,type="prob")
score.rf=score.rf[,2]
absc=numeric(length(s));ordo=numeric(length(s))

# Courbe Roc RF
for (i in 1:length(s)){
  ordo[i]=sum( score.rf>=s[i] & test$type=="spam")/sum(test$type=="spam")
  absc[i]=sum( score.rf>=s[i] & test$type=="nonspam")/sum(test$type=="nonspam")
}
plot(absc,ordo,col="red",type="l",xlab="FPR",ylab="TPR")
```




```
# à comparer avec les courbes ROC de l'arbre CART et de la régression logistique.
```