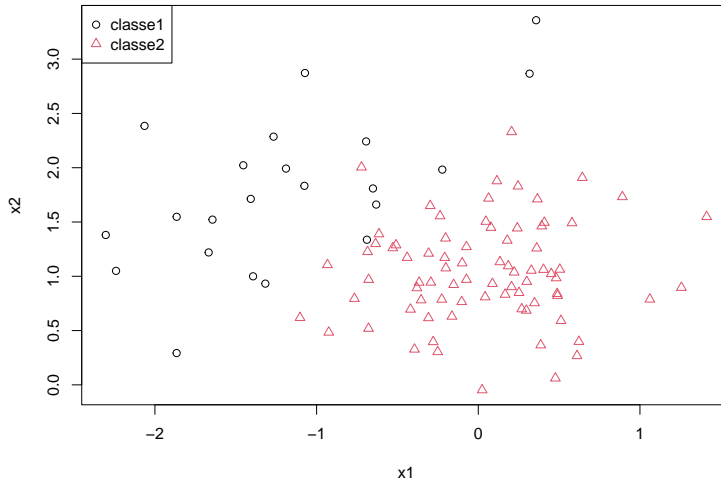


TP7 : régression logistique

Exercice 1. On utilise dans cet exercice les données synth_train.txt et synth_test.txt.

```
train <- read.table(file="../data/synth_train.txt", header=TRUE)
Xtrain <- train[,-1]
Ytrain <- train$y
plot(Xtrain, pch=Ytrain, col=Ytrain)
legend("topleft", legend=c("classe1", "classe2"), pch=1:2, col=1:2)
```



1. En régression logistique la variable de sortie Y prend ses valeurs dans $\{0, 1\}$. Recoder la variable de sortie Y pour avoir : "0=classe 1" et "1=classe 2."

```
train[,1] <- train[,1] - 1
```

2. On fait l'hypothèse paramétrique suivante :

$$\mathbb{P}(Y = 1 | X = (x_1, x_2)) = \frac{\exp(\beta_0 + \beta_1 x_1 + \beta_2 x_2)}{1 + \exp(\beta_0 + \beta_1 x_1 + \beta_2 x_2)},$$

Estimer sur les données d'apprentissage les paramètres inconnu $(\beta_0, \beta_1, \beta_2)$ à l'aide la fonction `glm`.

```
g <- glm(y ~ x1 + x2, data=train, family=binomial)
summary(g)$coefficients
```

3. Interpréter les coefficients β_1 et β_2 ainsi estimés.

```
exp(coef(g)[-1])
1/exp(coef(g)[-1])
```

4. Prédire la classe de la nouvelle observation $x = (0, 1)$ calculant le score logit (score linéaire)

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2$$

Vérifier que x est bien affecté à la classe 2.

```
x <- c(0,1)
beta <- coef(g)
logit <- c(1,x) %*% beta
# score positif donc on prédit Y=1 (classe 2)
```

5. Prédire la classe de la nouvelle observation $x = (-1, 2)$ en calculant cette fois le score

$$p = \mathbb{P}[Y = 1|X = x]$$

Vérifier que x est bien affecté à la classe 1.

```
x <- c(-1,2)
beta <- coef(g)
logit <- c(1,x) %*% beta
prob <- exp(logit)/(1+exp(logit))
# score < 0.5 donc on prédit Y=0 (classe 1)
```

6. Utiliser la méthode `predict` de la classe `glm` pour retrouver les scores obtenus aux deux questions précédentes. Cette fonction permet-elle de prédire directement les classes ?

```
x <- data.frame(x1=c(0,-1), x2=c(1,2))
predict(g, newdata=x)
predict(g, newdata=x, type="response")
```

7. Prédire les données d'apprentissage et calculer le taux d'erreur d'apprentissage.

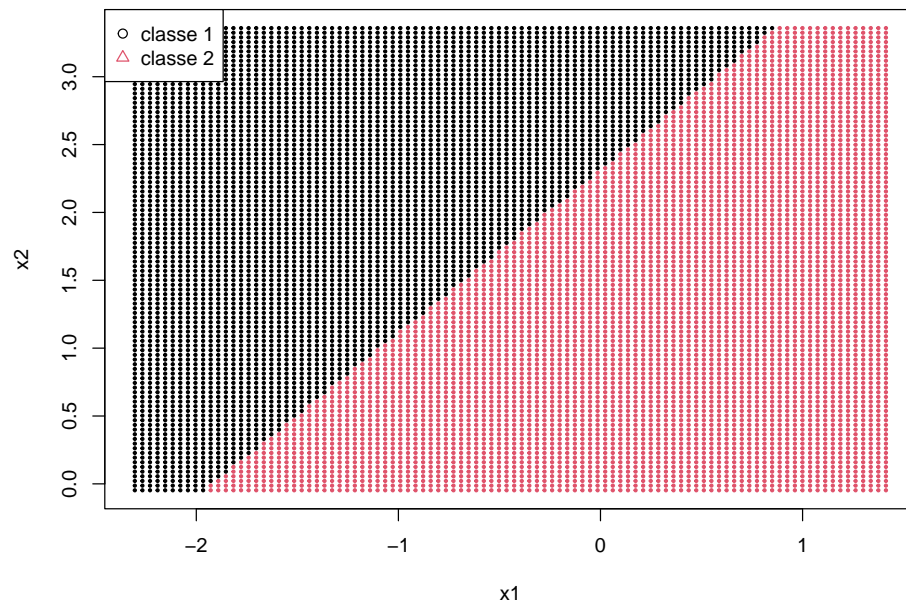
```
prob <- predict(g, newdata=Xtrain, type="response")
pred <- 1*(prob >= 0.5) + 0*(prob < 0.5)
sum(pred != Ytrain)/length(Ytrain)
```

8. Représenter la frontière de décision de la méthode de régression logistique à partir de la grille de points du TP1.

```
#construction de la grille de points
a <- seq(from=min(Xtrain$x1), to=max(Xtrain$x1), length.out=100)
b <- seq(from=min(Xtrain$x2), to=max(Xtrain$x2), length.out=100)
grille <- NULL
for (i in a){
  grille <- rbind(grille, cbind(i,b))
}
colnames(grille) <- c("x1", "x2")

prob <- predict(g, newdata=data.frame(grille), type="response")
pred_grille <- 1*(prob >= 0.5) + 0*(prob <= 0.5)
plot(grille, pch = 20, col = pred_grille+1, cex = 0.5,
     main="Frontière de décision en régression logistique")
legend("topleft", legend=c("classe 1", "classe 2"), pch=1:2, col=1:2, bg="white")
```

Frontière de décision en régression logistique



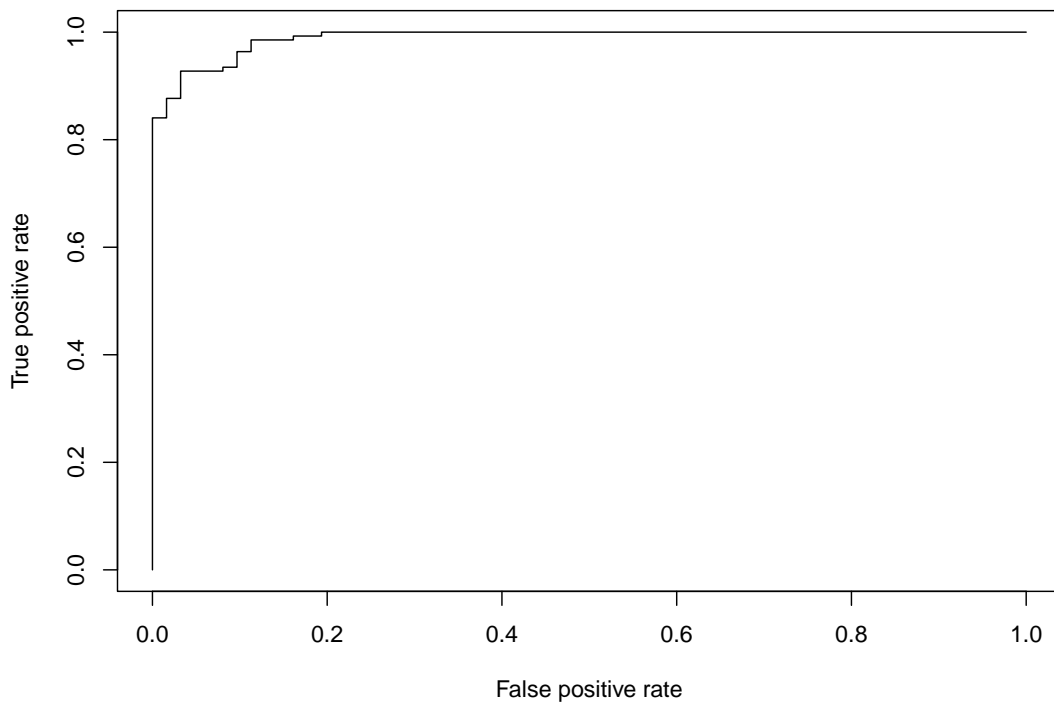
9. Charger le jeu de données test et calculer le taux d'erreur test.

```
test <- read.table(file="../data/synth_test.txt", header=TRUE)
Xtest <- test[,-1]
Ytest <- test$y
```

```
Ytest <- Ytest - 1
prob <- predict(g, newdata=Xtest, type="response")
pred <- 1*(prob >= 0.5) + 0*(prob <= 0.5)
sum(pred != Ytest)/length(Ytest)
```

10. Tracer la courbe ROC et calculer le critère AUC des données test.

```
library(ROCR)
toustab <- prediction(prob, Ytest, label.ordering=c("0", "1"))
perf <- performance(toustab, "tpr", "fpr")
plot(perf)
```



```
performance(toustab, "auc")@y.values[[1]]
```

Exercice 2. On reprend le jeu de données où 1260 exploitations agricoles saines ou défailtantes sont décrites par $p = 22$ critères économiques et financiers.

```
load("../data/Desbois_complet.rda")
```

```
dim(data)
colnames(data)
X <- data[,-1]
Y <- data$DIFF
```

On considère le modèle de régression logistique suivant

$$\mathbb{P}(Y = 1|X = x) = \frac{\exp(\beta_0 + \beta^T x)}{1 + \exp(\beta_0 + \beta^T x)} \quad (1)$$

où

- la variable à expliquer Y est la variable difficulté de paiement (0=sain et 1=défaillant),
- les p variables explicatives $X = (X^1, \dots, X^p)$ sont les $p = 22$ ratios financier,
- $\beta = (\beta_1, \dots, \beta_p)^T$ est le vecteur des coefficients associés à ces p variables explicatives.

1. Estimer à l'aide la fonction `glm` les coefficients $(\beta_0, \beta_1, \dots, \beta_p)$ de ce modèle de régression logistique. Quelles sont les variables qui expliquent significativement la défaillance d'une exploitation agricole ?

```
g <- glm(y~.,data=data.frame(X,y=Y), family=binomial)
summary(g)$coefficients
```

2. Comparer le taux d'erreur de la régression logistique et le taux d'erreur de l'analyse discriminante linéaire à partir de $B = 50$ découpages aléatoires des données (945 observations d'apprentissage et 315 observations test à chaque découpage).

```
library(MASS)
library(klaR)

B <- 50
err_test_lda <- rep(NA,B)
err_test_glm <- rep(NA,B)

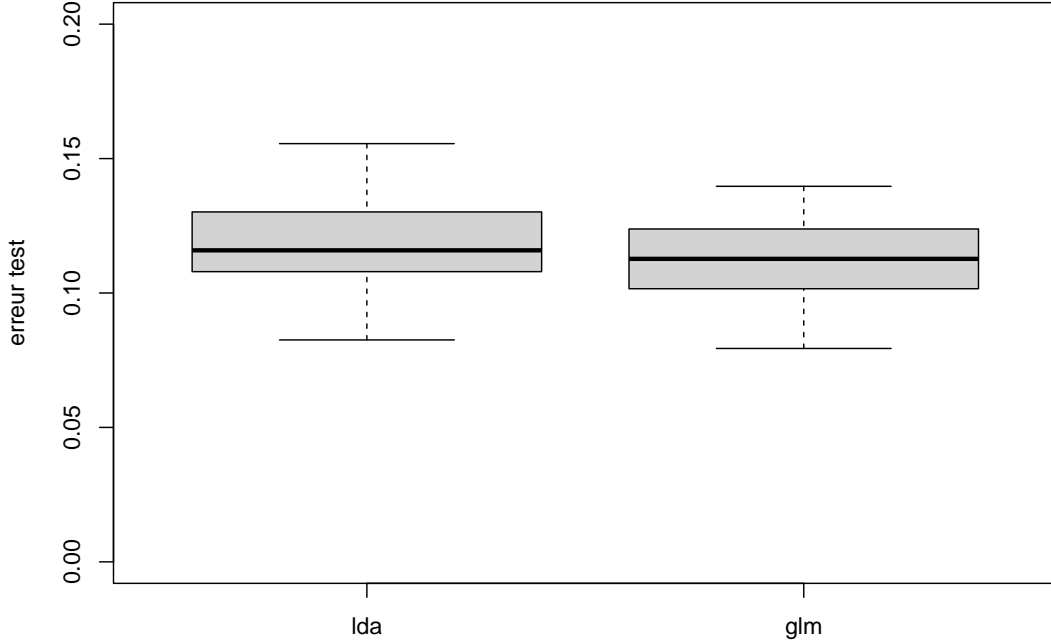
for (b in 1:B)
{
  tr <- sample(1:nrow(X), 945)
  Xtrain <- X[tr,]
  Ytrain <- Y[tr]
  Xtest <- X[-tr,]
  Ytest <- Y[-tr]

  # Apprentissage
  #lda
  g1 <- lda(Xtrain, grouping=Ytrain)
  #logistique
  g2 <- glm(y~.,data=data.frame(Xtrain,y=Ytrain), family=binomial)

  # Test
  #lda
  pred <- predict(g1, Xtest)$class
  err_test_lda[b] <- sum(pred!=Ytest)/length(Ytest)
  #logistique
  prob <- predict(g2, Xtest, type="response")
  pred <- 1*(prob>=0.5) + 0*(prob<0.5)
  err_test_glm[b] <- sum(pred!=Ytest)/length(Ytest)
}

err_test <- data.frame(lda=err_test_lda,
                      glm=err_test_glm)
boxplot(err_test, main="50 découpages apprentissage/test",
        ylab="erreur test", ylim=c(0,0.2))
```

50 découpages apprentissage/test



Exercice 3. On continue avec les données de l'exercice 2 mais on estime cette fois les paramètres du modèle (1) avec la fonction `glmnet` du package `glmnet`¹. La fonction `glmnet` estime les coefficients de la régression logistique par maximisation de la log-vraisemblance pénalisée :

$$\max_{(\beta_0, \beta) \in \mathbb{R}^{p+1}} \ell(\beta_0, \beta) - \lambda \left((1 - \alpha) \frac{\|\beta\|_2^2}{2} + \alpha \|\beta\|_1 \right) \quad (2)$$

où

— $\ell(\beta_0, \beta)$ est la log-vraisemblance qui s'écrit en régression logistique :

$$\ell(\beta_0, \beta) = \sum_{i=1}^n y_i (\beta_0 + x_i^T \beta) - \log(1 + \exp(\beta_0 + x_i^T \beta)) \quad (3)$$

— $(1 - \alpha) \frac{\|\beta\|_2^2}{2} + \alpha \|\beta\|_1$ est la pénalité **elastic-net**.

La pénalité elastic-net est un mélange des pénalités ridge et lasso contrôlé par le paramètre $\alpha \in [0, 1]$:

- Si $\alpha = 0$, la pénalité elastic-net est une pénalité ridge ($\|\beta\|_2^2$). Cette pénalité permet d'éviter les trop grandes valeurs de $\hat{\beta}$ et de contrôler la variance de cet estimateur (au risque d'augmenter son biais). Une pénalité ridge ne permet pas en revanche de sélectionner des variables.
- Si $\alpha = 1$, la pénalité elastic-net est une pénalité lasso ($\|\beta\|_1$). Cette pénalité permet de mettre à 0 certains coefficients de $\hat{\beta}$ et donc de sélectionner des variables.

1. https://web.stanford.edu/~hastie/glmnet/glmnet_alpha.html

Le paramètre λ ($\lambda \geq 0$) est un paramètre de régularisation à fixer et qui permet de contrôler l'impact de la pénalité :

- Si $\lambda = 0$, les coefficients du modèle (1) sont estimés par maximum de vraisemblance et la fonction `glmnet` effectue une régression logistique classique.
- Plus λ augmente, plus on pénalise la vraisemblance et si :
 - $\alpha = 0$, λ est le paramètre de régularisation de la régression logistique ridge,
 - $\alpha = 1$, λ est le paramètre de régularisation de la régression logistique lasso.

```
load("../data/Desbois_complet.rda")
X <- data[,-1]
Y <- data$DIFF
```

1. Démontrer (3).
2. Estimer à l'aide la fonction `glmnet` les coefficients $(\beta_0, \beta_1, \dots, \beta_p)$ du modèle de régression logistique (1) pour retrouver les résultats de la fonction `glm` à l'exercice 2. Ces résultats sont-ils parfaitement identiques? Pourquoi?

```
library(glmnet)
g <- glmnet(as.matrix(X), as.factor(Y), family="binomial",
            lambda=0,
            standardize = FALSE)
coef(g)
```

Exercice 4. On continue avec les données sur les exploitations agricoles.

```
load("../data/Desbois_complet.rda")
X <- data[,-1]
Y <- data$DIFF
```

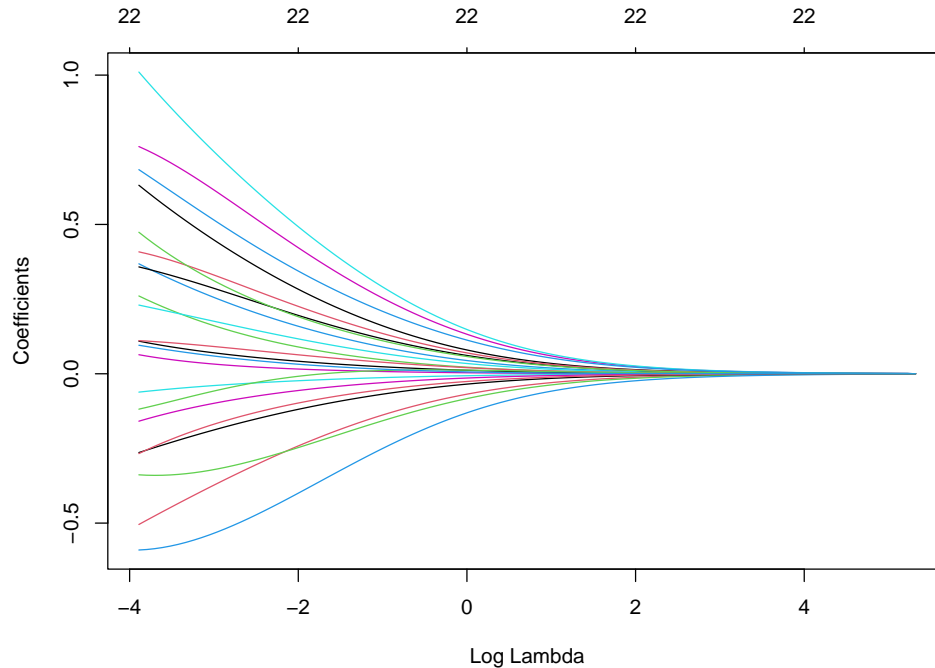
L'objectif cette fois est d'estimer les coefficient du modèle (1) par **régression logistique ridge** ce qui nécessite de fixer la valeur d'un paramètre de régularisation λ . On utilise pour cela la fonction `glmnet`.

1. Que fait le code ci-dessous?

```
g <- glmnet(as.matrix(X), as.factor(Y), family="binomial",
            alpha=0,
            standardize = FALSE)
```

2. Expliquer les 3 axes du graphique ci-dessous.

```
plot(g, xvar = "lambda")
```



3. Comment est définie la grille de valeurs de λ ?

```
g$lambda
log(g$lambda)
# lambda.min=lambda.max*lambda.min.ratio
# par defaut lambda.min.ratio = 0.001 si nobs < nvar et 0.01 sinon
# lambda.max=+ petite valeur de lambda pour laquelle tous les coefficients sont nuls, dédui
# puis grille de nlambda valeurs en lambda.min et lambda.max
# ici lambda.min=0.001*204387
# attention, découpage non linéaire. A voir si log ?
```

Vérifier la relation en la plus grande et la plus petite valeur de cette grille.

```
lambda.max <- g$lambda[1]
lambda.min <- g$lambda[100]
0.0001*lambda.max
```

4. Que fait le code ci-dessous ?

```
g$beta[,c(1, 50, 100)]
```

5. Utiliser la méthode `coef` pour comparer les coefficients obtenus avec une grande valeur de λ ($\lambda = 200$) et une petite valeur de λ ($\lambda = 0.02$). Vérifier que plus la valeur de λ est grande plus la norme $\|\beta\|_2$ est petite.

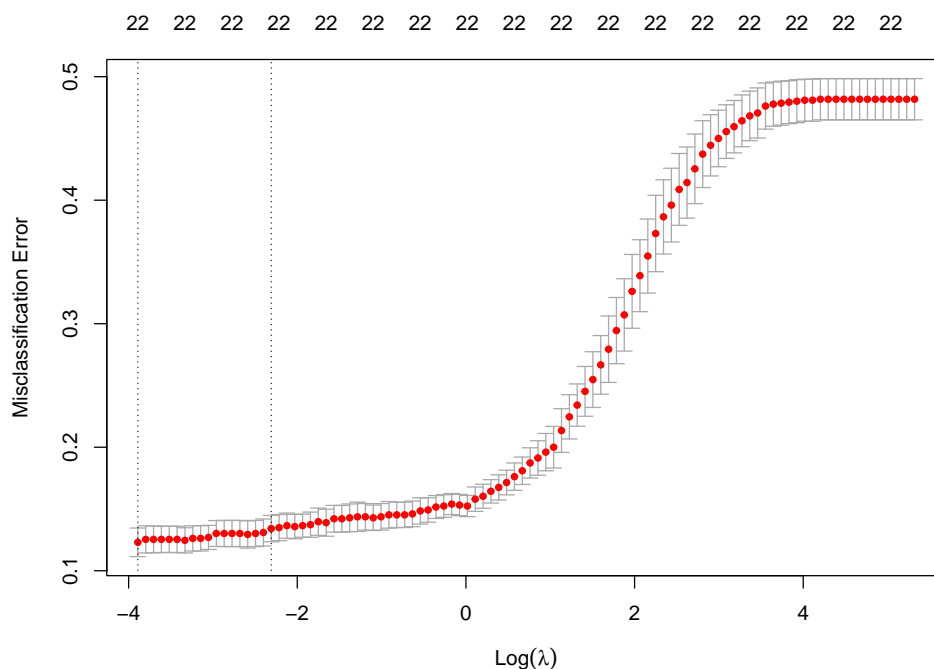
```
coef(g,s=204.3) # coefficients du modèle pour lambda=200
beta <- coef(g,s=200)[-1] # sans beta0
sqrt(sum(beta^2)) # norme L2
coef(g,s=0.02) # coefficients du modèle pour lambda=0.02
beta <- coef(g,s=0.02)[-1] # sans beta0
sqrt(sum(beta^2)) # norme L2
```


6. On veut maintenant fixer la valeur du paramètre de régularisation λ par validation croisée. Pour cela on utilise la fonction `glmnet.cv`. Que fait le code ci-dessous ? Quel est par défaut le type de validation croisée implémenté ?

```
g <- cv.glmnet(as.matrix(X), as.factor(Y), family="binomial",
              alpha=0,
              standardize=FALSE,
              type.measure="class")
```

7. Expliquer les 3 axes du graphique ci-dessous. A quoi correspondent les points rouges et les intervalles ?

```
plot(g)
```



8. La première ligne verticale du graphique correspond à une valeur particulière de λ appelée `lambda.min` (dans les sorties de la fonction `glmnet.cv`) et la seconde correspond à une valeur particulière appelée `lambda.1se`. Ces deux valeurs de λ sont définies comme suit :

- `lambda.min` minimise l'erreur (moyenne) de validation croisée.
- `lambda.1se` est la plus grande valeur de λ dont l'erreur (moyenne) de validation croisée est inférieure à l'erreur optimale (de "lambda.min") plus une fois son écart-type.

Quelles sont les valeurs de `lambda.min` et `lambda.1se` pour les données sur les exploitations agricoles ? Vérifier qu'elles correspondent bien aux deux lignes verticales du graphique.

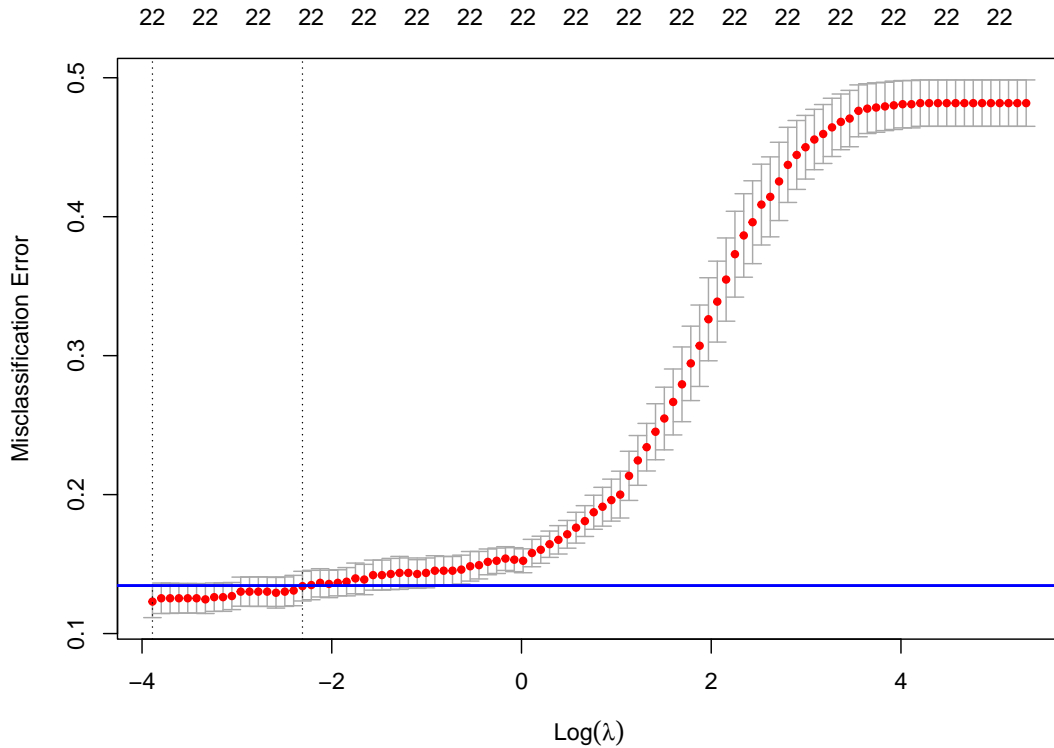
```
g$lambda.min
log(g$lambda.min)
g$lambda.1se
log(g$lambda.1se)
```

9. Retrouver `lambda.min` à partir de la sortie `cvm` (cross validation mean).
10. Calculer le seuil permettant de trouver `lambda.1se` à partir de la sortie `cvstd` (cross validation standard deviation) et ajouter au graphique une ligne horizontale bleue en fonction de ce seuil.

```

# Seuil pour la règle du 1se
seuil <- g$cvup[which.min(g$cvm)]
plot(g)
abline(h=seuil, col="blue", lwd=2)

```

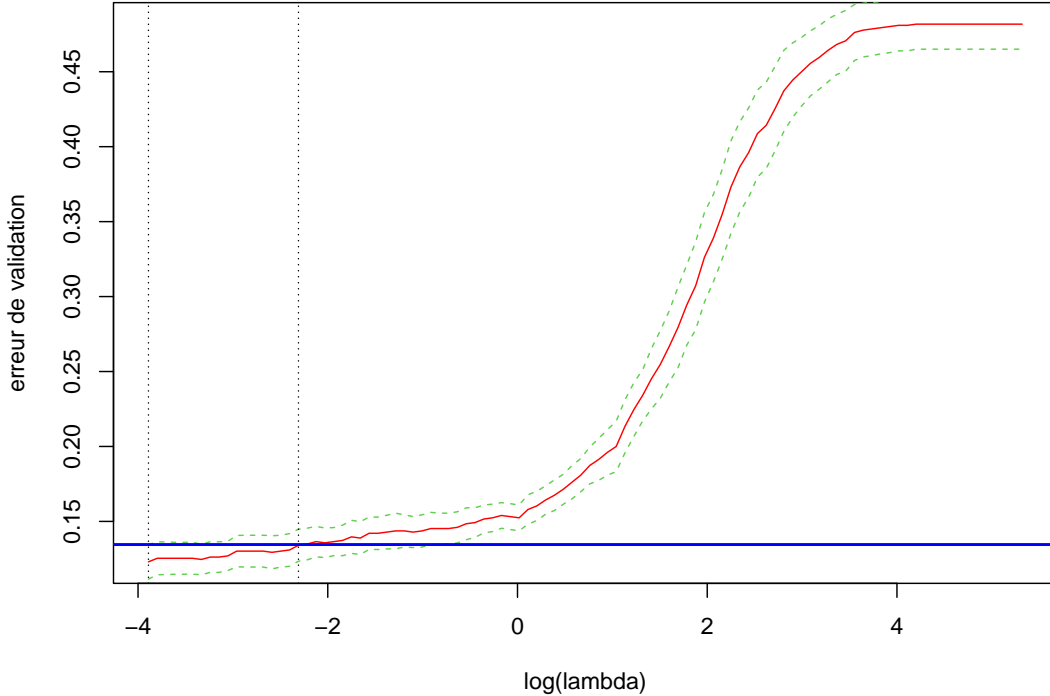


11. Construire ensuite le même graphique sans utiliser la méthode `plot`.

```

abs <- log(g$lambda[100:1])
plot(cbind(abs,g$cvm[100:1]), type="l", col="red", xlab="log(lambda)", ylab="erreur de vali
points(cbind(abs,g$cvup[100:1]), type="l", col=3, lty=2)
points(cbind(abs,g$cvlo[100:1]), type="l", col=3, lty=2)
abline(h=seuil, col="blue", lwd=2)
abline(v=log(g$lambda.min), lty=3)
abline(v=log(g$lambda.1se), lty=3)

```



12. Laquelle des deux valeurs de λ régularise plus ?
13. On choisit `lambda.min` comme paramètre de régularisation. Utiliser la méthode `predict` pour prédire la classe de la première exploitation agricole et estimer sa probabilité à posteriori d'être défaillante.

```
i <- 1 # première exploitation agricole
predict(g, newx = as.matrix(X[i,, drop=FALSE]),
       s = "lambda.min", type="class")
predict(g, newx = as.matrix(X[i,, drop=FALSE]),
       s = "lambda.min", type="response")
```

Exercice 5. On continue avec les données sur les exploitations agricoles.

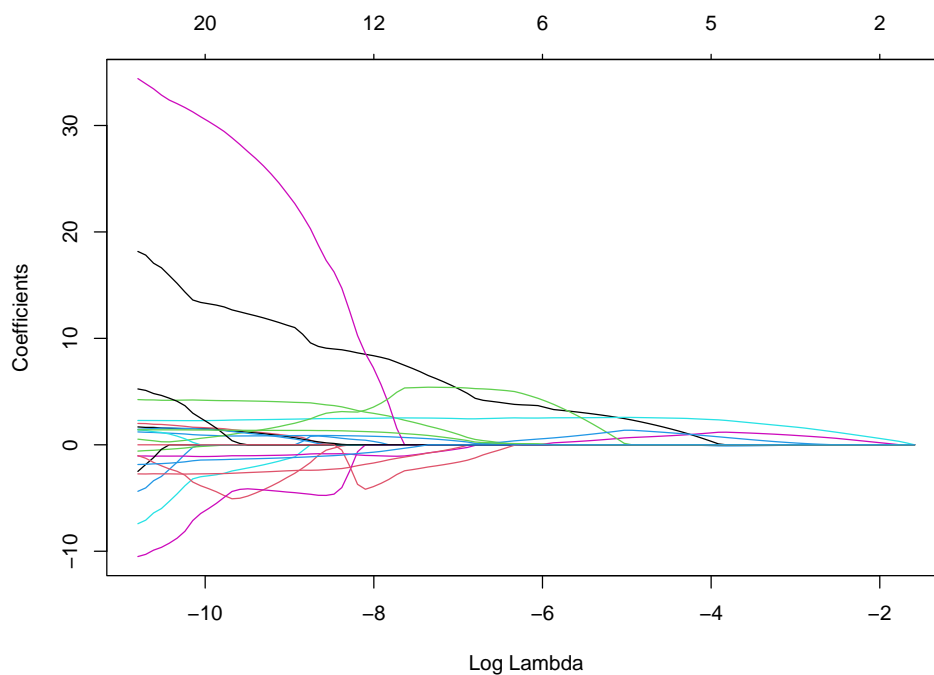
```
load("../data/Desbois_complet.rda")
X <- data[,-1]
Y <- data$DIFF
```

L'objectif est cette fois d'estimer les coefficients du modèle (1) par **régression logistique lasso** ce qui nécessite (comme en régression ridge) de fixer la valeur d'un paramètre de régularisation λ . On utilise là encore fonction `glmnet`.

1. Que fait le code ci-dessous ?

```
g <- glmnet(as.matrix(X), as.factor(Y), family="binomial",
           alpha=1,
           standardize = FALSE)
```

2. Expliquer les 3 axes du graphique ci-dessous.



3. Que fait le code ci-dessous ?

```
g$beta[,c(1, 50, 100)]
```

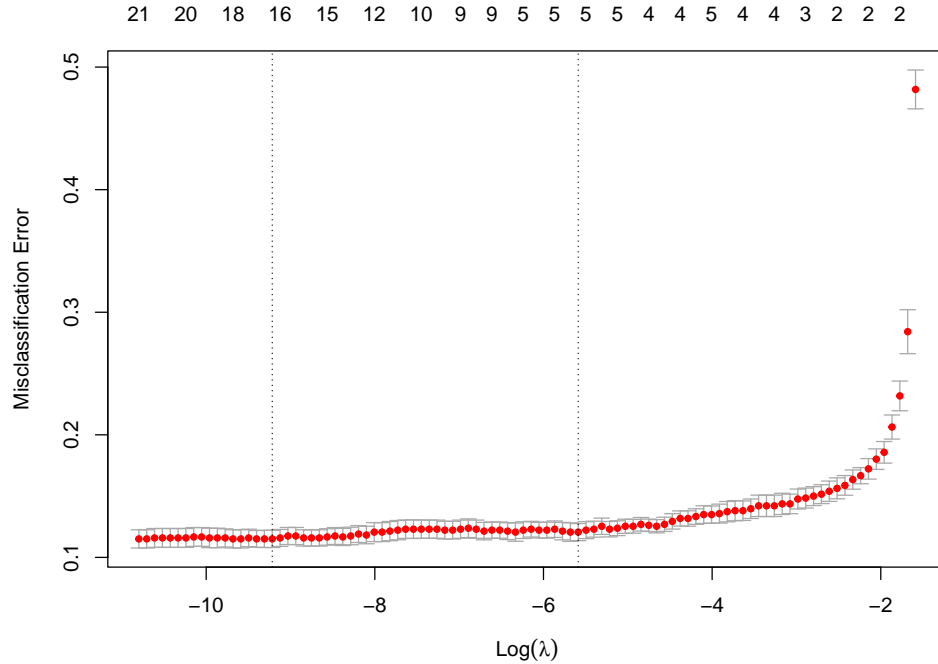
4. Utiliser la méthode `coef` pour comparer les coefficients obtenus avec une grande valeur de λ ($\lambda = 0.2$) et une petite valeur de λ ($\lambda = 0.002$). Vérifier que plus la valeur de λ est grande, plus le nombre de variables sélectionnées est petit.

5. On veut maintenant fixer la valeur du paramètre de régularisation λ par validation croisée. Pour cela on utilise la fonction `glmnet.cv`. Que fait le code ci-dessous ? Quel est par défaut le type de validation croisée implémenté ?

```
g <- cv.glmnet(as.matrix(X), as.factor(Y), family="binomial",  
              alpha=1,  
              standardize=FALSE,  
              type.measure="class")
```

6. Expliquer les 3 axes du graphique ci-dessous. A quoi correspondent les points rouges, les intervalles, les lignes pointillées verticales ?

```
plot(g)
```



7. Quelles sont les valeurs de `lambda.min` et `lambda.1se` ? Vérifier que ces valeurs correspondent bien aux deux lignes verticales du graphique.
8. Le modèle obtenu avec `lambda.1se` est-il beaucoup plus parcimonieux que le modèle obtenu avec `lambda.min` ? Est-il beaucoup moins bon ?
9. On choisit finalement `lambda.1se` comme paramètre de régularisation. Quelles sont les variables explicatives sélectionnées ?
10. Utiliser la méthode `predict` pour prédire la classe de la première exploitation agricole et estimer sa probabilité a posteriori d'être défailante.

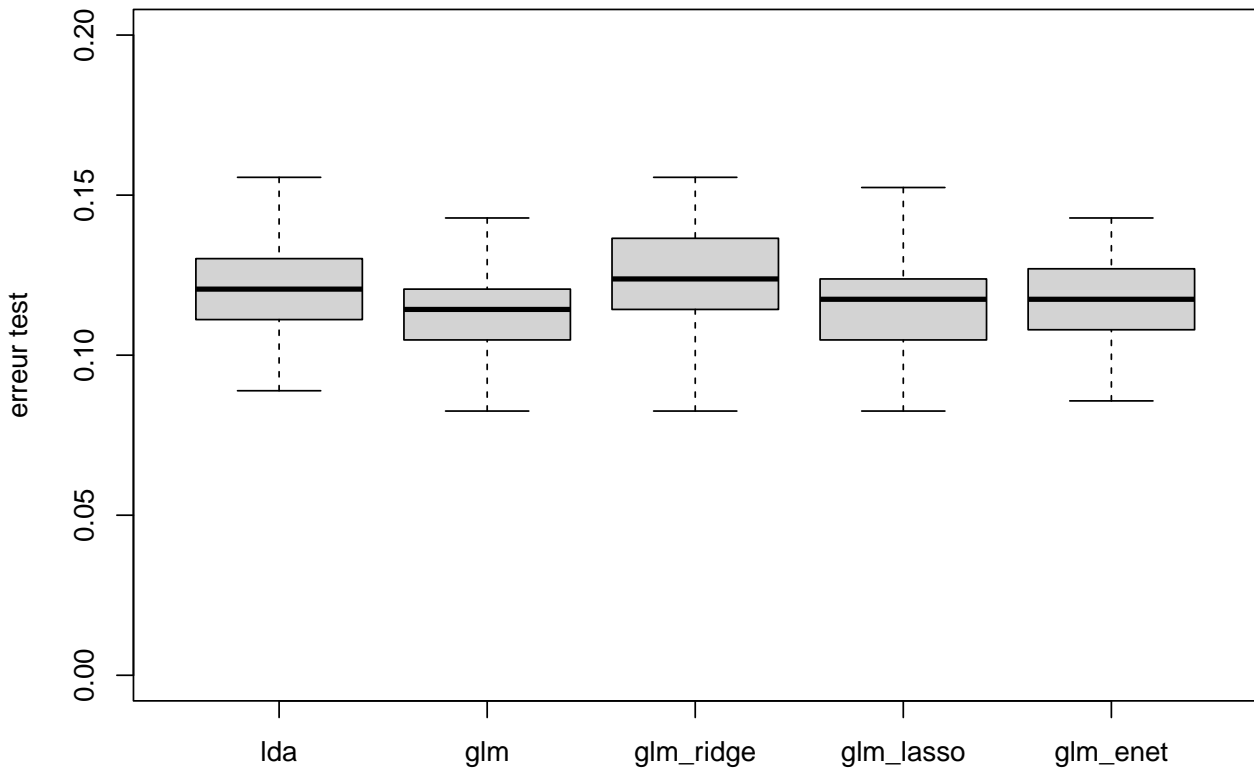
Exercice 6. On continue avec les données sur les exploitations agricoles.

```
load("../data/Desbois_complet.rda")
X <- data[,-1]
Y <- data$DIFF
```

On veut maintenant comparer différentes méthodes en estimant l'erreur de classification sur B découpages aléatoires des données en 945 observations d'apprentissage et 315 observations test. Les méthodes comparées sont :

- l'analyse discriminante linéaire,
- la régression logistique,
- la régression logistique ridge avec λ fixé par validation croisée 10-folds,
- la régression logistique lasso avec λ fixé par validation croisée 10-folds,
- la régression logistique elasticnet avec $\alpha = 0.9$ et λ fixé par validation croisée 10-folds.

50 découpages apprentissage/test



Conclure.