Sélection de variables

Analyse multidimensionnelle appliquée

Léo Belzile

HEC Montréal

automne 2022

Présentation des données

Exemple de base de données marketing (par ex., organisme de charité). Cibler les clients pour l'envoi d'un catalogue.

But: maximiser les levées de fonds

1. Envoyer un échantillon de produits au coût de 10$ à un groupe échantillon.
2. Construire un modèle de prédiction pour déterminer à qui envoyer le produit parmi tous les clients.

Population cible

Clients qui ont

- plus de 18 ans,
- au moins un an d'historique avec l'entreprise et
- qui ont effectué au moins un achat au cours de la dernière année.

Regroupements:

- 1K personnes dans l'échantillon d'apprentissage,
- 100K personnes pour l'ensemble des autres clients.

Liste des variables

- `yachat`, une variable binaire qui indique si le client a acheté quelque chose dans le catalogue égale à 1 si oui et 0 sinon;
- `ymontant`, le montant de l'achat si le client a acheté quelque chose;
- `x1`: sexe de l'individu, soit homme (0) ou femme (1);
- `x2`: l'âge (en année);
- `x3`: variable catégorielle indiquant le revenu, soit moins de 35 000$ (1), entre 35 000$ et 75 000$ (2) ou plus de 75 000$ (3);
- `x4`: variable catégorielle indiquant la région où habite le client (de 1 à 5);
- `x5`: couple : la personne est elle en couple (0=non, 1=oui);
- `x6`: nombre d'année depuis que le client est avec la compagnie;
- `x7`: nombre de semaines depuis le dernier achat;
- `x8`: montant (en dollars) du dernier achat;
- `x9`: montant total (en dollars) dépensé depuis un an;
- `x10`: nombre d'achats différents depuis un an.

Aperçu des données

```r
data(dbm, package = "hecmulti")
str(dbm)
```

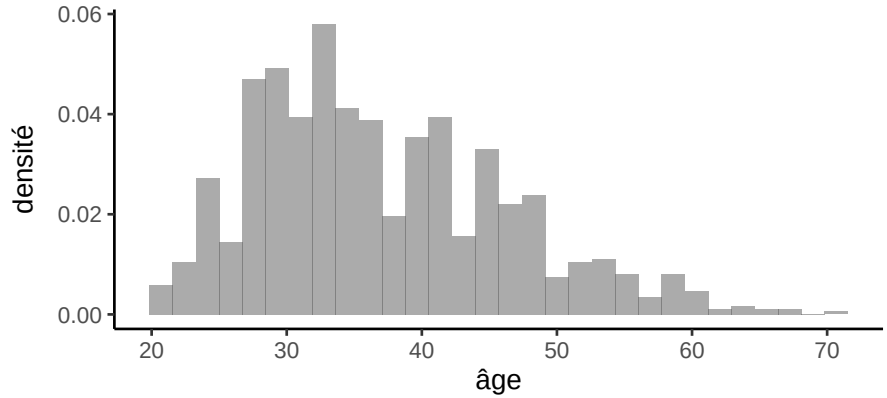
```
Classes 'tbl_df', 'tbl' and 'data.frame':   101000 obs. of  13 variables:
 $ x1       : int  1 1 0 0 1 1 0 0 0 1 ...
 $ x2       : num  42 59 52 32 38 63 35 32 26 32 ...
 $ x3       : Factor w/ 3 levels "1","2","3": 1 2 3 1 2 2 2 1 3 1 ...
 $ x4       : Factor w/ 5 levels "1","2","3","4",..: 3 3 5 1 5 5 1 3 1 5 ...
 $ x5       : int  1 1 1 0 0 1 1 0 0 0 ...
 $ x6       : num  8.6 8.6 1.4 10.7 9.1 9.4 10.6 4.8 4 10.3 ...
 $ x7       : num  8 9 9 42 5 1 6 5 48 9 ...
 $ x8       : num  49 70 120 31 30 28 59 70 73 55 ...
 $ x9       : num  159 123 434 110 55 102 593 298 83 90 ...
 $ x10      : num  5 5 8 3 3 8 10 6 2 3 ...
 $ yachat   : int  0 0 0 0 0 0 0 1 1 1 ...
 $ ymontant: num  NA NA NA NA NA NA NA 52 79 77 ...
 $ test     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

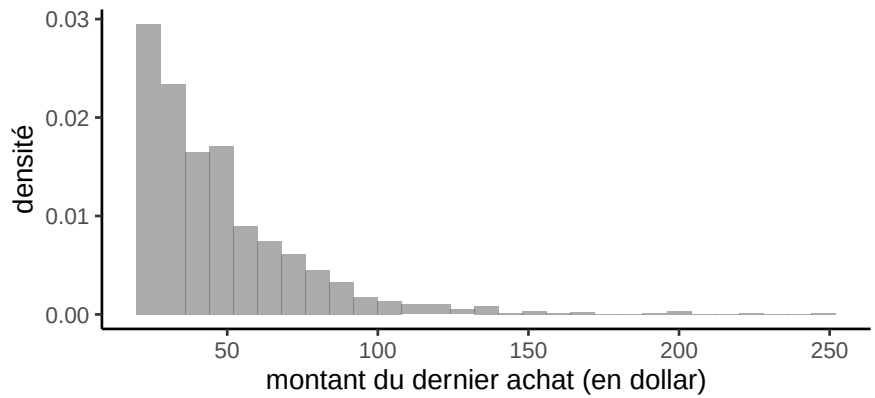
sexe	décompte
0	534
1	466

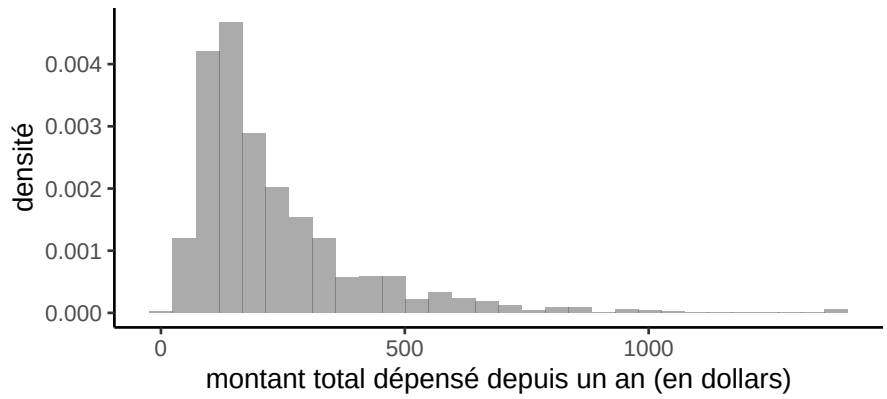
couple	décompte
0	575
1	425

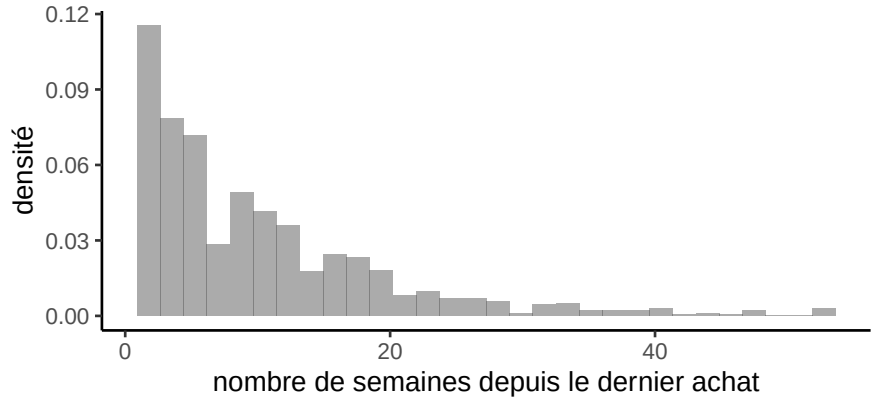
revenu	décompte
1	397
2	337
3	266

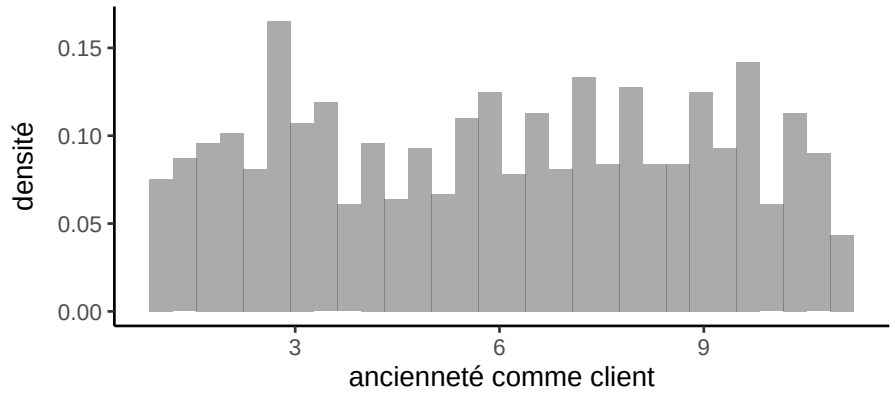
région	décompte
1	216
2	185
3	216
4	191
5	192

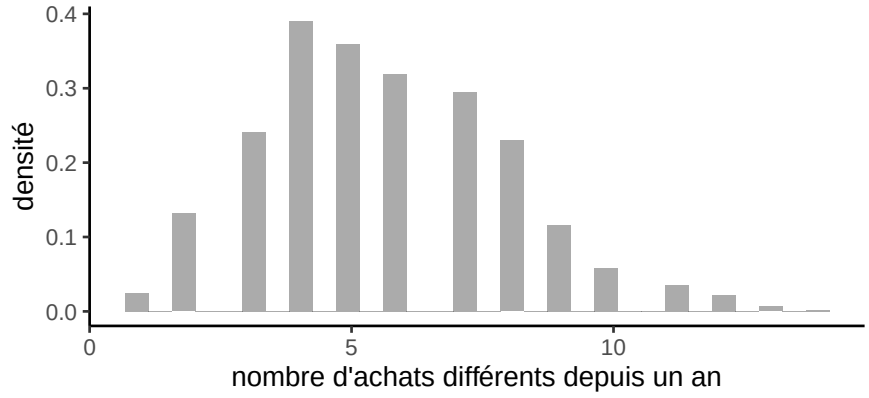

densité
0.06
0.04
0.02
0.00
20
30
40
50
60
70
âge


densité
0.03
0.02
0.01
0.00
50
100
150
200
250
montant du dernier achat (en dollar)


densité
0.004
0.003
0.002
0.001
0.000
0
500
1000
montant total dépensé depuis un an (en dollars)


densité
0.12
0.09
0.06
0.03
0.00
0
20
40
nombre de semaines depuis le dernier achat


densité
0.15
0.10
0.05
0.00
3
6
9
ancienneté comme client


densité
0.4
0.3
0.2
0.1
0.0
0
5
10
nombre d'achats différents depuis un an

variable	moyenne	écart-type	min	max
x2	37.06	9.27	20	70
x6	6.01	2.92	1	11
x7	9.97	9.34	1	52
x8	48.41	28.27	20	252
x9	229.27	173.97	22	1407
x10	5.64	2.31	1	14

variable	description
x2	âge
x6	nombre d'année comme client
x7	nombre de semaines depuis le dernier achat
x8	montant du dernier achat
x9	montant total dépensé sur un an
x10	nombre d'achats différents sur un an

Prédire le montant `ymontant`

Le montant moyen dépensé s'écrit

$$\mathsf{E}(\texttt{ymontant}) = \mathsf{E}(\texttt{ymontant} \mid \texttt{yachat} = 1)\,\mathsf{Pr}(\texttt{yachat} = 1).$$

On bâtit un modèle de régression linéaire pour le montant moyen dépensé, $\mathsf{E}(\texttt{ymontant} \mid \texttt{yachat} = 1)$.

Le modèle utilise les données des 210 personnes de l'échantillon d'apprentissage qui ont acheté suite à l'envoi du catalogue.

Séparation de l'échantillon

On conserve 100 000 observations test pour vérifier la performance

- (oracle) réponses inconnues à toutes fins pratiques

```
data(dbm, package = "hecmulti")
dbm_a <- dbm |>
  dplyr::filter(
    test == 0, #données d'entraînement
    !is.na(ymontant)) # personnes qui ont acheté
```

Recherche d'un modèle

Idée initiale:

- essayer tous les modèles possibles,
- estimer pour chacun la performance (erreur quadratique moyenne) avec validation croisée ou critères d'information,
- déterminer le meilleur modèle parmi l'ensemble de modèles.

Approximation

Soit $\mathsf{E}(Y \mid \mathbf{X}) = f(\mathsf{X}_1, \dots, \mathsf{X}_p)$ pour f une fonction inconnue supposée lisse.

On peut approximer la fonction en faisant une expansion (série de Taylor) de degré 2 (tous les termes quadratiques, et les interactions d'ordre 2):

- on n'inclut pas le carré de variables indicatrices binaires (car $0^2 = 0, 1^2 = 1$).
- idem pour les interactions entre indicateurs qui représentent des niveaux d'une même variable catégorielle.

Modèle complet et syntaxe **R**

```r
# (...)^2 crée toutes les interactions d'ordre deux
# I(x^2) permet de créer les termes quadratiques
formule <-
  formula(ymontant ~
            (x1 + x2 + x3 + x4 + x5 +
               x6 + x7 + x8 + x9 + x10)^2 +
              I(x2^2) + I(x6^2) + I(x7^2) +
              I(x8^2) + I(x9^2) + I(x10^2))
mod_complet <- lm(formule, data = dbm_a)
# Matrice avec toutes les variables
matmod <- model.matrix(mod_complet)
```

Le modèle est clairement **surajusté** avec 105 coefficients pour 210 variables.

Fléau de la dimensionalité

Combien de modèles incluant les combinaisons de p variables?

Dans l'exemple, $p = 14$ variables de base en incluant les indicatrices pour les variables catégorielles multiniveaux (revenu `x3` et région `x4`)

Chaque variable est incluse (ou pas): il y a $2^p = 2 \times 2 \times \cdots \times 2$ (p fois) modèles.

Table 1: Nombres de modèles en fonction du nombre de paramètres.

p	nombre de paramètres
5	32
10	1024
15	32768
20	1048576
25	33554432
30	1073741824

Recherche exhaustive

Essayer **tous** les modèles et choisir le meilleur (si p est petit).

L'algorithme par séparation et évaluation (*branch and bound*) recherche de manière efficace sans essayer tous les modèles candidats et écarte d'office les modèles sous-optimaux.

```
# Recherche exhaustive avec variables de base
rec_ex <- leaps::regsubsets(
  x = ymontant ~ x1+x2+x3+x4+x5+x6+x7+x8+x9+x10,
  nvmax = 13L,
  method = "exhaustive",
  data = dbm_a)
resume_rec_ex <- summary(rec_ex,
                         matrix.logical = TRUE)
# Trouver le modèle avec le plus petit BIC
min_BIC <- which.min(resume_rec_ex$bic)
# Nom des variables dans le modèle retenu
rec_ex$xnames[resume_rec_ex$which[min_BIC,]]
```

```
 [1] "(Intercept)" "x1"          "x32"         "x33"         "x45"
 [6] "x5"          "x6"          "x7"          "x8"          "x10"
```

Mot d'ordre

Quelques mantras pour la suite:

- préférer la flexibilité (réduire biais potentiel)
- être conscient de notre budget (surajustement)
- porter une attention particulière aux interactions entre variables catégorielles
  - estimations correspondent à des 'moyennes de groupe'
  - impact élevé potentiel des valeurs aberrantes et des extrêmes

Méthodes de sélection séquentielles

Recherche exhaustive typiquement trop coûteuse.

On peut plutôt opter pour un algorithme glouton:

- à chaque étape, on maximise l'utilité (horizon d'optimisation limité) en retirant ou en ajoutant une seule variable.
- au début, p variables à regarder, puis il y a $p - 1$ choix l'étape suivante, etc.
- moins de modèles explorés, mais utile pour faire une recherche rapide.

Digression

Le modèle à K variables qui a la plus petite erreur moyenne quadratique a aussi

- est aussi le meilleur (pour K variables) selon les critères d'information
- et selon le coefficient de régression R^2

Si on enlève ou on ajoute séquentiellement des variables, on peut traquer l'un ou l'autre avant de comparer avec le modèle précédent.

Sélections ascendante et descendante

Sélection ascendante: à partir du modèle de base (ordonnée à l'origine), ajouter à chaque étape au modèle précédent la variable qui améliore le plus l'ajustement.

Sélection descendante: éliminer du modèle complet la variable qui contribue le moins à l'ajustement.

Dans les deux cas, la procédure se termine quand on ne peut satisfaire le critère d'arrêt (par exemple, critère d'information)

Sélection séquentielle

À partir du modèle de base (d'ordinaire),

- alterner sélection séquentielle ascendante et descendante.
- on continue ainsi tant que le modèle retourné par l'algorithme n'est pas identique à celui de l'étape précédente.
- une variable peut entrer dans le modèle et sortir plus tard dans le processus.
  - préférable aux procédures ascendantes et descendantes (car plus de modèles).
  - lui préférer la recherche exhaustive quand c'est possible

Code **R** avec `leaps`

```r
# Cette procédure séquentielle retourne
# la liste de modèles de 1 variables à
# nvmax variables.
rec_seq <-
  leaps::regsubsets(
    x = formule,
    data = dbm_a,
    method = "seqrep",
    nvmax = length(coef(mod_complet)))
which.min(summary(rec_seq)$bic)
```

Sélection séquentielle avec critères d'informations en **R**

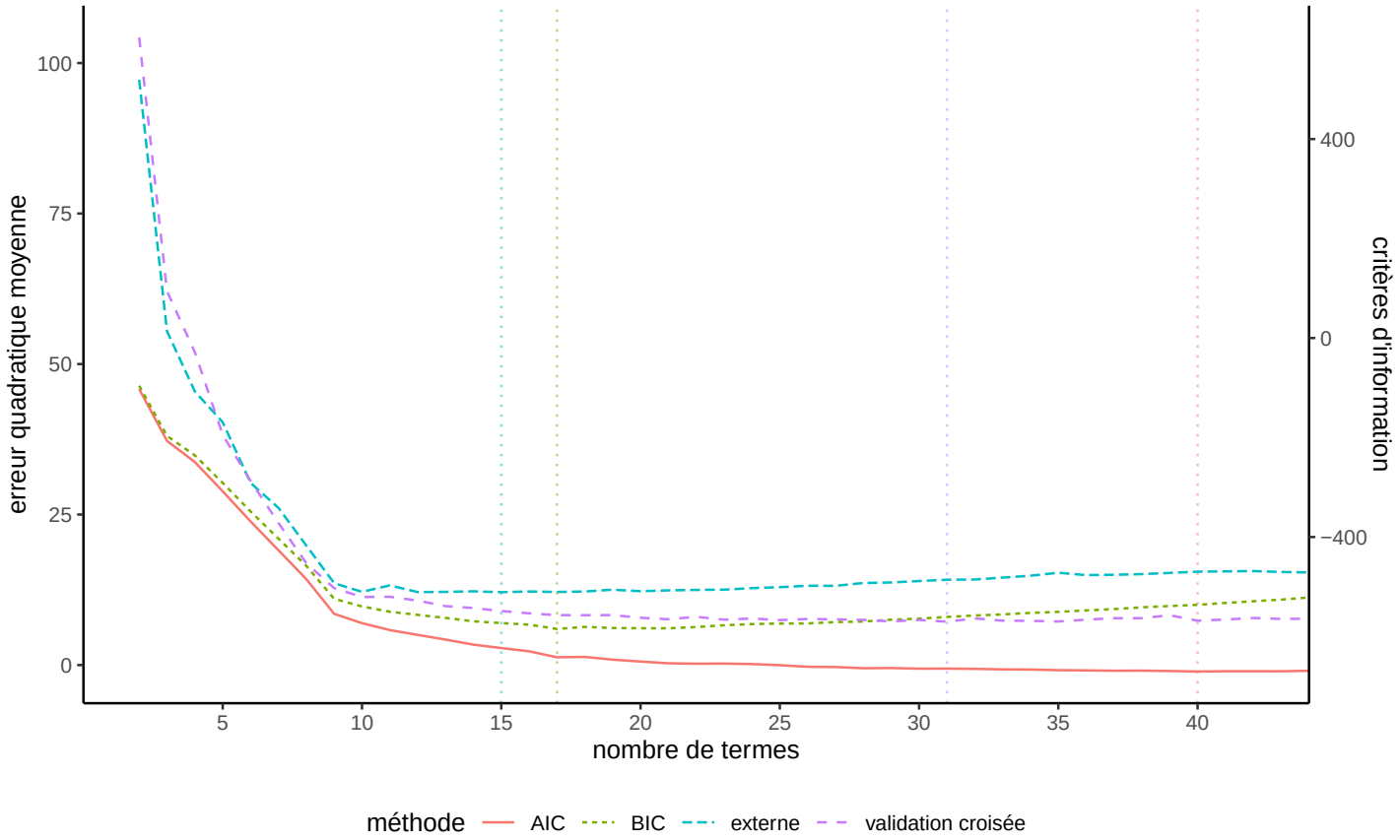
La procédure est plus longue à rouler(car les modèles linéaires sont ajustés)

On ajoute ou retire la variable qui améliore le plus le critère de sélection à chaque étape (différent de **SAS**).

```
seq_AIC <- MASS::stepAIC(
  lm(ymontant ~ 1, data = dbm_a),
  # modèle initial sans variables explicative
    scope = formule, # modèle maximal possible
    direction = "both", #séquentielle
    trace = FALSE, # ne pas imprimer le suivi
    keep = function(mod, AIC, ...){
      # autres sorties des modèles à conserver
      list(bic = BIC(mod),
           coef = coef(mod))},
    k = 2) #
# Remplacer k=2 par k = log(nrow(dbm_a)) pour BIC
```

L'historique des étapes est disponible via `seq_AIC$anova`

Performance en fonction de la complexité



40 premiers modèles de la procédure séquentielle en fonction du nombre de termes inclus. Oracle: 100K données de validation (libellé externe)

Méthodes de régularisation

Objectif: prévenir le surajustement.

L'erreur moyenne quadratique se décompose comme

$$\text{biais carré} + \text{variabilité}$$

Les méthodes de régularisation introduisent du biais dans l'estimation des coefficients en pénalisant leur norme.

Préalable à la régression avec régularisation

Pénalisation la norme de $\beta_1, \dots, \beta_p$

- Modèles avec pénalités pas les mêmes selon l'échelle des données
  - pas invariant aux transformations affines (par ex., conversion de Celcius en Farenheit)

Solution: standardiser variables explicatives $X_1, \dots, X_p$ **et** variable réponse y (moyenne zéro, écart-type unitaire).

- vérifier selon le logiciel, cette étape peut-être effectuée implicitement

LASSO

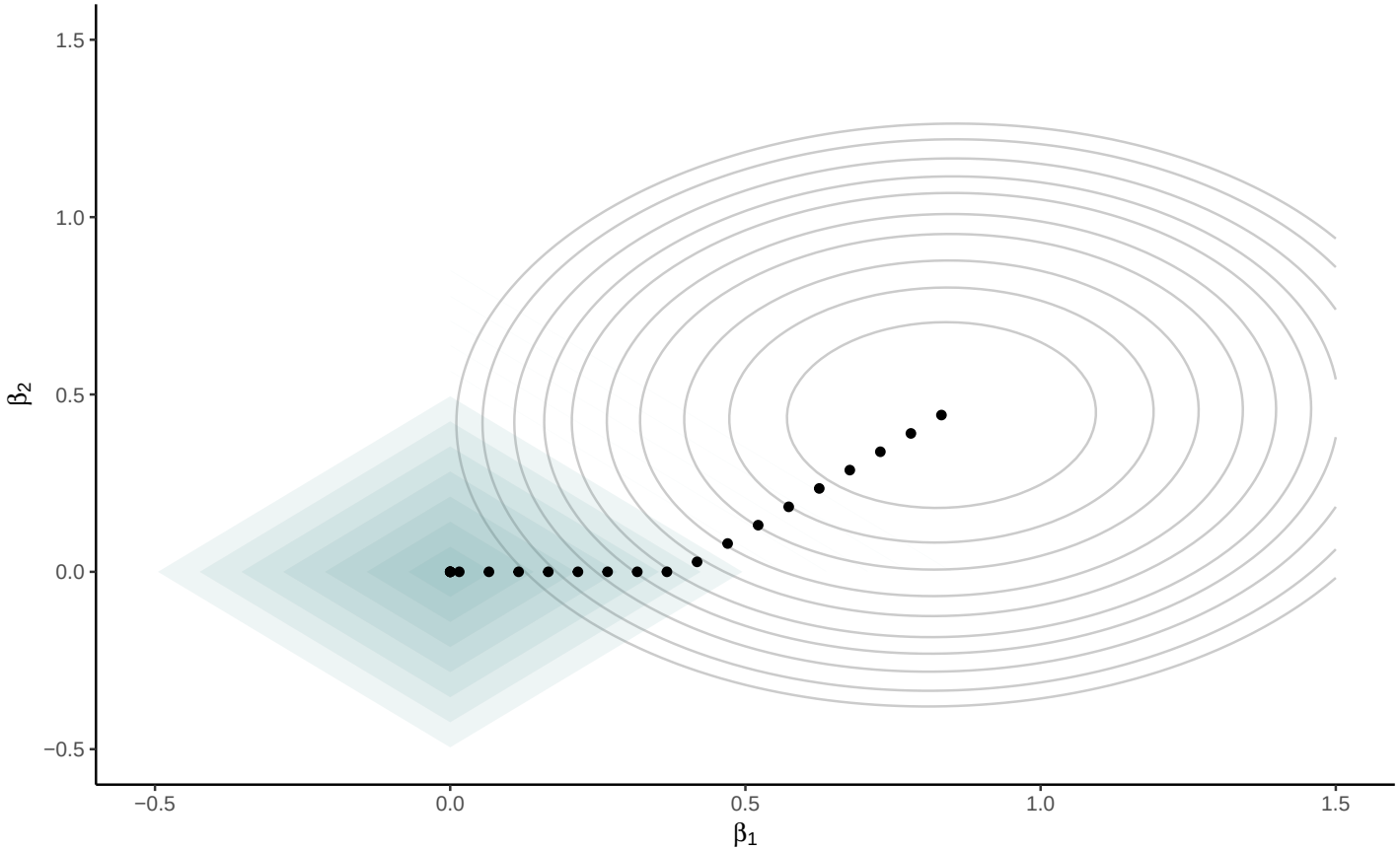
Pénalité avec norme l_1 pour la valeur absolue des coefficients,

$$\min_{\beta} \left\{ n\text{EQM}(\beta) + \lambda(|\beta_1| + \cdots + |\beta_p|) \right\}.$$

Hyperparamètre $\lambda > 0$ qui détermine la force de la pénalisation.

Rétrécissement de certains coefficients **exactement** à zéro: sélection implicite de variable.

Contrainte budgétaire et moindres carrés



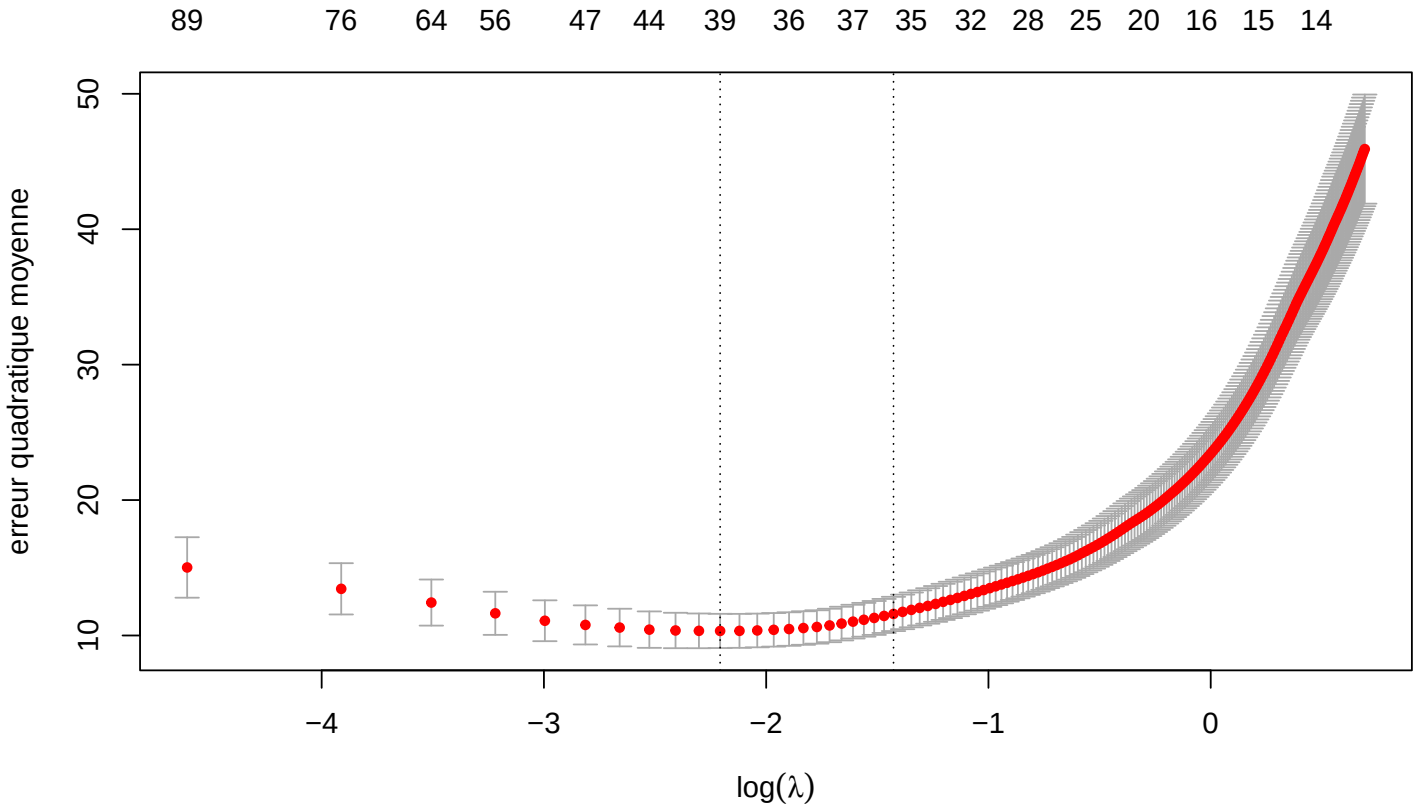
Code **R** pour le LASSO

Paramètre de pénalité déterminé par validation croisée à partir d'un vecteur de valeurs candidates.

```r
library(glmnet)
lambda_seq <- seq(from = 0.01, to = 2, by = 0.01)
cv_output <-
  glmnet::cv.glmnet(x = as.matrix(matmod),
                    y = dbm_a$ymontant,
                    alpha = 1,
                    lambda = lambda_seq)
plot(cv_output)
```

Trajectoire LASSO

On choisit typiquement λ par validation croisée en appliquant la règle du un écart-type (le première modèle à un erreur-type de la valeur minimale).



Paramètres

```
lasso_path <-
  glmnet::glmnet(
    x = as.matrix(as.matrix(matmod)),
    y = dbm_a$ymontant,
    alpha = 1,
    lambda = seq(from = 0.01, to = 10, by = 0.01))
plot(lasso_path)
```

Évaluation et prédiction

Une fois la valeur de λ choisie, on réestime le modèle avec la pénalité.

```
lambopt <- cv_output$lambda.min #ou cv_output$lambda.1se
lasso_best <-
  glmnet::glmnet(
    x = as.matrix(as.matrix(matmod)),
    y = dbm_a$ymontant,
    alpha = 1,
    lambda = lambopt)
```

Prédictions

On crée une matrice avec les données de validation et on calcule l'erreur quadratique moyenne.

```
# Prédictions et calcul de l'EQM
# Données externes
dbm_v <- dbm |>
  dplyr::filter(
    test == 1,
    !is.na(ymontant))
pred <- predict(lasso_best,
                s = lambopt,
                newx = as.matrix(
                  model.matrix(formule,
                               data = dbm_v)))
eqm_lasso <- mean((pred - dbm_v$ymontant)^2)
```

Motivation pour la moyenne de modèles

Une seule base de données (un seul échantillon) à disposition!

Peu d'hétérogénéité, hors cette dernière est présente

- (rappelez-vous l'incertitude de la validation croisée)

Moyenne de modèles

Solution: générer nous-mêmes B échantillons différents à partir de l'échantillon original.

Autoamorçage nonparamétrique: échantillon choisi au hasard et avec remise dans l'échantillon original.

- Une même observation peut être sélectionnée plus d'une fois tandis qu'une autre peut ne pas être sélectionnée du tout.

Défaut: très coûteux en calcul

Moyennes de modèles en pseudocode

1. Échantillonnage aléatoire simple avec remise de B jeux de données

Pour chaque jeu de données étiqueté $b = 1, \dots, B$:

2. Effectuer la sélection de variables
3. Sauvegarder les coefficients (0 si la variable est absente)

Mettre en commun les résultats:

4. Calculer la moyenne des coefficients
5. Obtenir les prédictions

Voir code en ligne

Évaluation de la performance -

En incluant uniquement les variables de base

nombre de variables	EQM	méthode
15	25,69	toutes les variables
12	25,53	exhaustive - AIC
10	25,04	exhaustive - BIC

En incluant les termes de base, les carrés et les interactions d'ordre 2.

nombre de variables	EQM	méthode
104	19,63	toutes les variables
21	12	séquentielle, choix selon AIC
15	12,31	séquentielle, choix selon BIC
30	12	LASSO, validation croisée avec 10 groupes

Récapitulatif

- Le nombre de modèles possibles augmente rapidement avec le nombre de prédicteurs.
- Si un modèle est mal spécifié (variables importantes manquantes), alors les estimations sont biaisées.
- Si le modèle est surspécifié, les coefficients correspondants aux variables superflues incluses sont en moyenne nuls, mais contribuent à l'augmentation de la variance.
- **Compromis biais/variance**.

Récapitulatif

- La taille du modèle (le nombre de variables explicatives) est restreinte par le nombre d'observations disponibles.
- En général, il faut s'assurer d'avoir suffisamment d'observations pour estimer de manière fiable les coefficients
- Porter une attention particulière aux variables binaires et aux interactions avec ces dernières: si les effectifs de certaines modalités sont faibles, il y a danger surajustement.

Récapitulatif

- En pratique, on cherche à essayer plusieurs modèle pour trouver un choix optimal de variables.
- Une recherche exhaustive garantie le survol du plus grand nombre de modèles possibles, mais est coûteuse
- On peut effectuer une recherche exhaustive à l'aide d'algorithmes d'optimisation pour un nombre réduit de variables (max 50)
- Sinon, on a l'option d'utiliser un algorithme glouton qui ne couvre qu'un sous-ensemble de tous les modèles
- Compromis coût de calcul vs nombre de modèles explorés
- Possibilité de combiner des méthodes!

Récapitulatif

Certaines méthodes de pénalisation directe changent la fonction objective:

- introduction de biais pour les coefficients.
- idée globale: échanger biais contre variabilité moindre.

Une pénalité particulière (LASSO) contraint certains paramètres à être exactement nuls,

- correspond implicitement à une sélection de variables.

Récapitulatif

- On applique le critère de sélection sur la liste de modèles candidats pour retenir celui qui donne la meilleure performance.
- Pour éviter une sélection rigide, on peut perturber les données et répéter la procédure pour calculer une moyenne de modèles.
- Cette approche est très coûteuse en calcul.