UNIVERSITÉ DU QUÉBEC À MONTRÉAL

ALGORITHMES DE RECHERCHE POUR SÉLECTION DE MODÈLES

MÉMOIRE

PRÉSENTÉ

COMME EXIGENCE PARTIELLE

DE LA MAÎTRISE EN MATHÉMATIQUES

PAR

CLAUDIU MIRCEA MOTOC

NOVEMBRE 2011

REMERCIEMENTS

Je voudrais remercier mes professeurs de l'UQAM pour m'avoir aidé à découvrir les mystères de la statistique. Particulièrement, je remercie Sorana Froda qui m'a convaincu de commencer le programme de maîtrise et Geneviève Lefebvre pour sa patience et pour sa grande disponibilité pendant la rédaction de mon mémoire jusqu'à la fin.

Je désire aussi remercier mes parents pour leur soutien et pour leurs encouragements durant ces dernières années.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

RÉSUMÉ

Dans ce mémoire, nous nous intéressons à des algorithmes de sélection de modèles dans un contexte de régression linéaire et logistique. Nous expliquons premièrement les notions de régression linéaire et logistique et deux critères de sélection, AIC et BIC. Ensuite, nous faisons une revue des aspects théoriques des algorithmes les plus connus en détaillant deux d'entre eux, Leaps and Bounds et Occam's Window. Pour ces deux derniers, nous présentons aussi les détails pratiques des logiciels qui font leur implantation.

La partie finale est consacrée à l'étude des trois méthodes de sélection des modèles basées sur les algorithmes Leaps and Bounds, Occam's Window et sur une combinaison entre les deux, en utilisant la technique du moyennage de modèles. Nous présentons les performances de prédiction calculées à l'aide de la technique de validation croisée et les temps d'exécution de ces trois méthodes pour plusieurs jeux de données.

Mots clés : sélection de modèles, moyennage de modèles, régression linéaire, régression logistique, AIC, BIC, algorithme Leaps and Bounds, algorithme Occam's Window, validation croisée.

INTRODUCTION

Étant donné un ensemble d'observations sur plusieurs variables, la sélection de modèles consiste à choisir, parmi l'ensemble des modèles possibles, un ou plusieurs modèles statistiques qui décrivent le mieux les observations. Dépendamment du type de modèles et du nombre de variables, la sélection peut être longue et difficile. Pour automatiser cette tâche, plusieurs algorithmes ont été développés. Même si leur utilisation sans le conseil d'experts dans le domaine d'intérêt n'est pas recommandée, ces algorithmes sont de plus en plus employés.

Dans ce mémoire, nous présentons les algorithmes les plus importants en détaillant deux d'entre eux, "Leaps and Bounds" et "Occam's Window", et évaluons leurs performances de prédiction. Les deux premières sections du Chapitre I expliquent les notions de régression linéaire et logistique. À la Section 1.3, nous introduisons deux critères de sélection bien connus, AIC et BIC. À la fin de ce chapitre, la Section 1.4 présente trois algorithmes que nous appelons approximatifs, car ils n'explorent pas de façon exhaustive l'ensemble des modèles possibles.

Le Chapitre II décrit l'algorithme "Leaps and Bounds". Cet algorithme considère que l'ensemble des modèles est organisé dans une structure logique d'arbre. Pour mieux comprendre son fonctionnement, nous décrivons aussi les notions d'arbre, arbre inverse et arbre pairé et la façon de parcourir l'arbre pairé.

Au Chapitre III, nous introduisons l'algorithme "Occam's Window". La Section 3.1 explique la méthodologie de sélection des modèles. Par la suite, la Section 1.4 fait l'introduction du contexte bayésien et de la technique de moyennage de modèles bayésien. Le reste du chapitre explique les détails de cet algorithme, son efficacité et ses difficultés d'implantation.

Le Chapitre IV fait une présentation des logiciels que nous utilisons dans ce mémoire pour la sélection des modèles. Il s'agit de la liste des fonctions contribuées pour l'algorithme "Occam's Window" écrites dans le langage C et aussi de la liste des fonctions dans le paquetage BMA du langage R. Nous considérons qu'il est important de comprendre le fonctionnement de ces fonctions et de leurs paramètres avant de commencer l'étude comparative du Chapitre V.

Le Chapitre V présente les résultats obtenus en utilisant trois méthodes de sélection de modèles. La première méthode utilise l'algorithme "Leaps and Bounds" seulement, la deuxième est une combinaison entre "Leaps and Bounds" et "Occam's Window" et la troisième emploie seulement l'algorithme "Occam's Window". La Section 5.1 décrit la technique de validation croisée utilisée pour évaluer les performances des trois méthodes. Les sections 5.2 et 5.3 présentent respectivement les résultats obtenus pour la régression linéaire et logistique, pour un jeu réel et un jeu généré dans chaque cas. La Section 5.4 concerne la discussion de ces résultats.

CHAPITRE I

ALGORITHMES ET MÉTHODES UTILISÉS POUR LA SÉLECTION DE MODÈLES

La sélection de modèles est un problème très actuel dans le domaine statistique. Il y a beaucoup de documentation sur ce sujet et plusieurs approches ont été développées en fonction du but envisagé.

Nous cherchons à modéliser une variable d'intérêt Y, appelée *variable réponse*, en fonction d'autres variables $X = (X_1, \ldots, X_p)$, $p \geq 1$ entier, appelées *covariables*. Différents modèles peuvent être considérés et ce choix dépendra entre autres de la relation, linéaire ou non, entre Y et X.

En pratique, nous avons un échantillon aléatoire de taille n de la population, $n > p$, comprenant les valeurs Y_i et $(X_{i1}, X_{i2}, \ldots, X_{ip})$, $i = 1, 2, \ldots, n$. En nous basant sur l'échantillon, il y a plusieurs méthodes pour estimer les paramètres du modèle considéré. Deux méthodes très connues sont la méthode des moindres carrés et la méthode du maximum de vraisemblance. La première estime les paramètres du modèle en minimisant la somme des carrés des résidus, ces derniers représentant les différences entre les valeurs de la variable Y de l'échantillon et les valeurs calculées à l'aide du modèle. La deuxième méthode est basée sur la distribution des données $p(Y|\theta, X)$, considérée comme une fonction des paramètres θ, est appelée *fonction de vraisemblance*, $L(\theta)$. L'estimation de θ est obtenue en maximisant la vraisemblance, c'est-à-dire, $\hat{\theta} = \max_{\theta}\{L(\theta)\}$, où θ peut être un vecteur de paramètres.

Dans le contexte de la sélection de modèles, il est pertinent de distinguer deux objectifs. Premièrement, nous pouvons être intéressés à prédire la variable Y à l'aide de variables X obtenues ou mesurées ultérieurement. Les covariables considérées sont souvent plus facilement mesurables que la variable réponse. Deux modèles très différents peuvent être également acceptables lorsque la capacité de prédiction est ce qui nous intéresse. Si nous devons choisir entre deux modèles, le choix peut se baser sur des critères de simplicité, ou sur la facilité à mesurer les variables importantes. Une deuxième situation consiste à déterminer l'effet d'une ou plusieurs variables sur la variable réponse, en conditionnant (ajustant) possiblement pour des variables de nuisance dont l'effet n'est pas principalement d'intérêt.

Dans ce qui suit, nous supposons qu'il est possible d'inclure ou d'exclure du modèle toutes les variables considérées, ce qui n'est pas toujours le cas en pratique. Même si nous sommes tentés d'utiliser le modèle complet, avec toutes les variables disponibles, ce modèle n'est pas nécessairement optimal au point de vue de la capacité de prédiction. En effet, il est bien connu que la variance des valeurs prédites augmente avec la dimension du modèle. Ce phénomène est à la base du compromis entre le biais et la variance que nous devons effectuer lors du choix d'un modèle. En général, nous cherchons un sous-modèle avec un petit nombre de variables, qui donne une bonne prédiction et qui explique bien les données. Comme nous le verrons, plusieurs critères statistiques ont été introduits pour nous aider dans cette tâche.

1.1 Régression linéaire multiple

Le modèle de régression linéaire multiple est utilisé lorsqu'il y a une relation linéaire entre la variable réponse et les covariables. Pour la population entière, un tel modèle est décrit par l'équation suivante :

$$E[Y|X] = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p,$$

où les coefficients $\beta_i, i = 0, 1, \ldots, p$ sont appelés les *paramètres* du modèle.

Nous avons un échantillon aléatoire de taille n de la population, $n > p$, comprenant les valeurs Y_i et $(1, X_{i1}, X_{i2}, \ldots, X_{ip})$, $i = 1, 2, \ldots, n$. Si nous notons le vecteur réponse par $Y = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{pmatrix}$, le vecteur des erreurs par $\epsilon = \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{pmatrix}$ et le vecteur des paramètres par $\beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix}$, alors nous pouvons écrire le modèle sous une forme matricielle :

$$Y = X\beta + \epsilon,$$

où $X = \begin{pmatrix} 1 & X_{11} & \ldots & X_{1p} \\ 1 & X_{21} & \ldots & X_{2p} \\ \vdots & \vdots & \ldots & \vdots \\ 1 & X_{n1} & \ldots & X_{np} \end{pmatrix}$ est la matrice des données.

La méthode la plus utilisée pour estimer les paramètres β du modèle est la méthode des moindres carrés dont la solution minimise la somme des carrés des résidus (RSS). En supposant que $E[\epsilon_i] = 0$, $Var[\epsilon_i] = \sigma^2$ et que les ϵ_i sont indépendants, $i = 1, 2, \ldots, n$, la méthode des moindres carrés nous donne un estimateur non biaisé pour le vecteur des paramètres β :

$$\hat{\beta} = (X'X)^{-1}X'Y.$$

Cet estimateur est aussi celui obtenu par maximisation de la vraisemblance en présumant les erreurs normalement distribuées.

Une fois que l'estimation $\hat{\beta}$ est obtenue, nous pouvons prédire la valeur de la variable réponse en calculant $\hat{y} = x\hat{\beta}$ pour n'importe quel ensemble de données $x = (x_1, x_2, \ldots, x_p)$ provenant de la même population. Évidemment, nous pouvons aussi calculer les valeurs prédites pour chaque y_i, notées par $\hat{y}_i$, $i = 1, 2, \ldots, n$. Nous supposons que les erreurs ϵ_i ne sont pas observables. Toutefois, nous pouvons les estimer en utilisant la différence suivante $\hat{\epsilon}_i = y_i - \hat{y}_i$, appelée *résidu*.

1.2 Régression logistique multiple

Dans le cas de la régression logistique multiple, nous voulons modéliser le résultat d'un évènement, représenté par une variable binaire $Y = 0, 1$, en fonction d'une collection de covariables (lesquelles peuvent être continues ou catégoriques) notée par le vecteur $X' = (X_1, X_2, \ldots, X_p)$. Désignons la probabilité conditionnelle de la réalisation de l'événement par $\pi(X) = P(Y = 1|X)$. Le modèle de régression logistique multiple (D. W. Hosmer et S. Lemeshow, 2000) est alors donné par :

$$\pi(X) = \frac{e^{g(X)}}{1 + e^{g(X)}}, \tag{1.1}$$

où $g(X) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p$.

En appliquant à ce modèle la transformation logit, nous voyons que nous retrouvons une fonction linéaire des covariables :

$$\ln\left[\frac{\pi(X)}{1 - \pi(X)}\right] = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p.$$

Étant donné un échantillon aléatoire de taille n de la population, $n > p$, comprenant les valeurs Y_i et $(X_{i1}, X_{i2}, \ldots, X_{ip})$, $i = 1, 2, \ldots, n$, alors $\pi_i = P[Y_i = 1|(X_{i1}, X_{i2}, \ldots, X_{ip}), \beta]$. Nous définissons la fonction de vraisemblance du modèle de la façon suivante :

$$L(\beta_0, \beta_1, \ldots, \beta_p) = \prod_{i=1}^{n} \pi_i^{y_i}(1 - \pi_i)^{1-y_i}. \tag{1.2}$$

L'expression (1.2) découle du fait que les Y_i sont considérés comme des variables aléatoires indépendantes distribuées selon une loi Bernoulli(π_i).

Le maximum de la fonction de vraisemblance est la solution du système des équations obtenu en calculant les dérivées partielles d'ordre un et deux en β du logarithme de $L(\beta_0, \beta_1, \ldots, \beta_p)$. Nous obtenons les $p+1$ équations suivantes :

$$\begin{cases} \sum_{i=1}^{n}(y_i - \pi_i) = 0, & \text{et} \\ \sum_{i=1}^{n} x_{ik}(y_i - \pi_i) = 0, & \text{pour} \quad k = 1, 2, \ldots, p. \end{cases} \tag{1.3}$$

La résolution du système (1.3) n'est pas facile. En effet, nous ne pouvons trouver qu'une solution approximative, à l'aide d'un algorithme itératif (IRLS - Iteratively Reweighted Least Squares) de type Newton-Raphson :

1. L'algorithme commence avec l'estimation d'une solution initiale,

$$\hat{\beta}^0 = (\hat{\beta}_0^0, \hat{\beta}_1^0, \ldots, \hat{\beta}_p^0).$$

   Nous pouvons trouver cette solution initiale par la méthode des moindres carrés, en ajustant aux données un modèle de régression linéaire non pondéré. Un tel modèle suppose que les probabilités initiales $\hat{\pi}_i^0$ sont égales, pour $i = 1, 2, \ldots, n$.

2. À partir de $\hat{\beta}^0$, il est facile de calculer les probabilités $\hat{\pi}_i^1$ et d'ajuster un modèle de régression linéaire pondéré avec ces poids afin de trouver une meilleure solution $\hat{\beta}^1$.

3. L'étape 2 se répète tant qu'une meilleure solution n'est pas trouvée. Pour évaluer une solution, nous calculons la valeur du logarithme de la fonction de vraisemblance pour voir si elle est croissante ou non. L'algorithme s'arrête quand la différence entre deux valeurs successives est suffisamment petite (proche de zéro).

Notons que l'algorithme n'est pas toujours convergent. Si la solution initiale n'est pas suffisamment proche du point de maximum global, il est possible que l'algorithme ne converge pas. Si elle est plus proche d'un point de maximum local, l'algorithme trouve celui-ci avant de s'arrêter. Même si la solution initiale est bien choisie, il est possible que la dernière solution dépasse le point de maximum global. Dans ce cas, l'algorithme fait quelques étapes supplémentaires en prenant comme solution le point milieu entre la dernière solution trouvée et la solution précédente, en essayant de se rapprocher de nouveau du point de maximum global.

1.3 Mesures d'ajustement

Après l'estimation des paramètres d'un modèle, nous voulons voir si celui-ci est bien ajusté aux données observées. De façon générale, nous pouvons répondre à cette question

en comparant les valeurs observées avec les valeurs prédites par le modèle, à l'aide d'une fonction mathématique que nous calculons. Selon le type de modèle considéré, des fonctions différentes sont utilisées.

Pour des modèles de régression linéaire, une fonction très utilisée est la somme des carrés des résidus notée et définie par $RSS = \sum_{i=1}^{n} \epsilon_i{}^2$. Parmi l'ensemble des modèles considérés pour les données, le modèle possédant le plus petit RSS offre le meilleur ajustement aux données. Toutefois, il ne possède pas nécessairement le meilleur pouvoir prédictif.

En pratique, nous utilisons souvent des critères d'information pour le choix d'un modèle. Premièrement, le critère d'information d'Akaike (AIC) (H. Akaike, 1974) effectue un compromis entre l'ajustement du modèle aux données et la complexité de celui-ci de la façon suivante : $AIC = 2(p+1) - 2\ln(L)$, où p est le nombre de paramètres du modèle (sans inclure l'ordonnée à l'origine β_0) et L la vraisemblance. Le deuxième critère est le critère d'information bayésien (BIC) (G. Schwartz, 1978), défini par $BIC = (p+1)\ln(n) - 2\ln(L)$, où n est le nombre d'observations. Le BIC effectue similairement un compromis entre l'ajustement aux données et la complexité du modèle, mais pénalise davantage cette dernière que le critère AIC (facteur 2 versus $\ln(n)$). Nous favorisons les modèles avec de petites valeurs de AIC ou de BIC.

Dans le cas de la régression linéaire, nous pouvons exprimer ces deux critères de sélection en fonction de la somme des carrés des résidus si chaque terme d'erreur est distribué indépendamment selon une loi normale $N(0, \sigma^2)$. En effet, nous obtenons :

$$AIC = 2(p+1) + n\ln(\tfrac{RSS}{n}) \qquad \text{et}$$
$$BIC = (p+1)\ln(n) + n\ln(\tfrac{RSS}{n}).$$

Notons que les critères AIC et BIC sont des fonctions croissantes en p et RSS. Comme nous le verrons au Chapitre III, le critère BIC est utilisé comme quantité fondamentale dans le moyennage de modèles bayésien. Cette technique sera utilisée pour la quantification du pouvoir de prédiction des ensembles de modèles sélectionnés par les techniques "Leaps and Bound" et "Occam's Window" qui feront l'objet d'études approfondies.

1.4 Algorithmes approximatifs

Dans ce qui suit, nous allons faire un résumé de quelques algorithmes utilisés couramment pour parcourir l'espace des modèles et effectuer une sélection des variables. Les algorithmes que nous présentons dans cette sous-section ont la particularité d'être approximatifs, dans le sens qu'ils n'offrent pas une garantie de trouver les meilleurs modèles. Nous reportons la description des algorithmes d'intérêt "Leaps and Bounds" et "Occam's Window" aux chapitres 2 et 3 puisqu'ils feront l'objet d'études plus approfondies.

1.4.1 Sélection progressive (Forward selection)

Dans la procédure de sélection progressive, nous avons un modèle complet avec p covariables $(X_1, X_2, \ldots, X_p)$ avec une somme résiduelle, $RSS_p = \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} \hat{\beta}_j x_{ij}\right)^2$ correspondante et nous voulons trouver un sous-ensemble de $(X_1, X_2, \ldots, X_p)$ dont la somme des carrés résiduelle est suffisamment petite. Nous considérons dans ce cas le vecteur X_1 comme étant le vecteur unitaire correspondant à l'ordonnée à l'origine du modèle.

Nous commençons par l'ensemble vide et nous ajoutons premièrement la variable X_j qui minimise la somme $RSS_1 = \sum_{i=1}^{n} \left(y_i - \hat{\beta}_j x_{ij}\right)^2$, j fixé. Cette variable, notée $X_{(1)}$ fera partie de tous les sous-ensembles de covariables considérés. Supposons que nous ayons déjà ajouté r variables à notre modèle, $X_{(1)}, X_{(2)}, \ldots, X_{(r)}$, $1 \leq r \leq p-1$. Nous ajoutons aux variables déjà sélectionnées la variable $X_{(r+1)}$, qui minimise la somme des carrés résiduelle partielle $RSS_{r+1} = \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{r+1} \hat{\beta}_{(j)} x_{i(j)}\right)^2$, calculée avec les variables $X_{(1)}, X_{(2)}, \ldots, X_{(r)}$ et $X_{(r+1)}$. Les sommes sont calculées à l'aide de rotations planaires et de réductions orthogonales (A. Miller, 2002). Supposant que le nombre de covariables cherchées est connu, nous pouvons vérifier à chaque étape si toutes ont été rajoutées. Sinon, l'algorithme nécessite un critère d'arrêt en fonction de RSS_r.

Les variables $X_{(1)}, X_{(2)}, \ldots, X_{(p)}$ sont donc choisies successivement, chacune étant choi-

sie parce qu'elle minimise la somme des carrés partielle quand elle est ajoutée aux autres variables déjà sélectionnées. Comme, en général, le sous-ensemble de $r+1$ covariables qui minimise RSS_{r+1} ne contient pas le sous-ensemble de r covariables qui minimise RSS_r, cette méthode ne nous donne aucune garantie de trouver les modèles les mieux ajustés aux données.

1.4.2 Sélection pas-à-pas (Stepwise selection)

La sélection pas-à-pas, un algorithme proposé par Efroymson en 1960, est une variante de la sélection progressive. Après l'addition de chaque covariable à l'ensemble des covariables sélectionnées, l'algorithme fait un test pour déterminer s'il est possible d'écarter une des variables déjà sélectionnées sans trop augmenter la somme des carrées des résidus. L'algorithme de sélection utilise les trois critères suivants :

a) Critère d'ajout :

Si RSS_r est la somme des carrés calculée avec r variables et RSS_{r+1} est la somme des carrés qui correspond à l'ajout de la $(r+1)^e$ covariable, l'algorithme calcule le ratio $R = \frac{RSS_r - RSS_{r+1}}{\frac{RSS_{r+1}}{(n-r-2)}}$ et le compare avec une valeur prédéterminée F_a. Si $R > F_a$, la variable est sélectionnée.

b) Critère de suppression :

Notons RSS_r la plus petite valeur de RSS que nous pouvons obtenir en écartant une variable de l'ensemble sélectionné $(X_{(1)}, X_{(2)}, \ldots, X_{(r+1)})$. Le ratio $R = \frac{RSS_r - RSS_r + 1}{\frac{RSS_{r+1}}{(n-r-2)}}$ est calculé et comparé avec une autre valeur prédéterminée F_d. Si $R < F_d$ la variable est écartée de l'ensemble.

c) Critère d'arrêt :

Le critère d'ajout est satisfait quand $RSS_{r+1} \leq \frac{RSS_r}{[1+F_a/(n-r-2)]}$. Aussi, le critère de suppression est satisfait lorsque $RSS_r \leq RSS_{r+1}[1 + F_d/(n-r-2)]$. Alors, quand une addition est suivie d'une suppression, le nouveau RSS obtenu, RSS_r^*, satisfait l'inégalité : $RSS_r^* \leq RSS_r \frac{1+F_d/n-r-2}{1+F_a/n-r-2}$. L'algorithme s'arrête quand il n'est plus possible de faire des ajouts ou suppressions en s'assurant à chaque étape de

minimiser RSS_r. La convergence de l'algorithme est assurée si $F_d < F_a$, car dans ce cas $\frac{1+F_d/n-r-2}{1+F_a/n-r-2} < 1$ et RSS_r a comme borne inférieure la valeur $\min(RSS_r)$.

Le calcul des valeurs F_a et F_d nécessite l'utilisation de l'intégration numérique multidimensionnelle. Une estimation grossière peut être obtenue en regardant le ratio R comme étant le maximum d'une suite de $p - r$ variables Fisher (A. Miller, 2002).

En pratique, l'algorithme d'Efroymson performe mieux que la sélection progressive si quelques covariables sont fortement corrélées entre elles. De plus, comme la sélection progressive, il n'offre pas la garantie de trouver les meilleurs modèles.

1.4.3 Sélection régressive (Backward selection)

La sélection régressive est la méthode inverse de la sélection progressive dans le sens qu'elle débute avec l'ensemble complet de p covariables et considère l'élimination des variables une à la fois. Si RSS_p est la somme des carrés des résidus du modèle complet, la covariable à écarter est choisie telle que le RSS_{p-1} qui résulte est minimal. Par la suite, en minimisant RSS_{p-2}, une autre variable est choisie parmi les $p-1$ variables qui restent. Le processus continue jusqu'à ce qu'il ne reste qu'une seule variable ou jusqu'à ce que le critère d'arrêt soit satisfait.

Dépendamment du nombre de variables initiales et de la dimension du modèle recherché, en pratique, l'élimination régressive nécessite plus d'opérations parce qu'elle doit généralement écarter plus de variables que la sélection progressive en ajoute. Finalement, les deux méthodes ne parviennent pas aux meilleurs modèles, même dans le cas où elles trouvent les mêmes ensembles à chaque niveau.

CHAPITRE II

L'ALGORITHME "LEAPS AND BOUNDS"

L'algorithme "Leaps and Bounds" (G. M. Furnival et R. W. Wilson, Jr., 1974) est très utilisé pour la sélection d'un sous-ensemble de dimension connue satisfaisant un critère d'optimalité. Essentiellement, l'algorithme "Leaps and Bounds" est une technique "branch and bound" (A. H. Land et A. G. Doig, 1960), cette dernière étant un algorithme général pour des problèmes d'optimisation dans un cas discret. L'objectif de "branch and bound" est de trouver les limites supérieures et inférieures de la quantité qui doit être optimisée, pour écarter, en masse, de grands sous-ensembles de candidats qui ne respectent pas le critère d'optimalité. L'optimum est une valeur connue ou inconnue que nous cherchons et est défini comme étant le maximum ou le minimum de la fonction considérée, cette dernière pouvant être évaluée pour chaque candidat. Étant donné un ensemble discret d'éléments, "branch and bound" trouve l'optimum d'une façon plus rapide que l'énumération de chaque candidat.

L'algorithme "Leaps and Bounds" résout similairement un problème d'optimisation. Considérons le modèle général de régression linéaire multiple présenté à la Section 1.1. Pour sélectionner un modèle, nous désirons trouver le sous-ensemble de l'ensemble $\{X_1, X_2, \ldots, X_p\}$ de dimension k, $k = 0, ..., p$, tel que le modèle de régression basé sur celui-ci possède la "meilleure" capacité de prédiction telle que mesurée avec le critère de sélection choisi. Plus particulièrement, dans une exécution de "Leaps and Bounds", cet algorithme identifie le modèle de dimension k minimisant la somme des carrés résiduelle (RSS), pour $k = 1, 2, \ldots, p$ séparément. À partir des solutions de "Leaps and Bounds",

nous pouvons aussi trouver facilement le modèle minimisant certains critères de sélection (comme AIC ou BIC), car cet optimum global est parmi les valeurs déjà trouvées pour l'ensemble des k.

2.1 La notion d'arbre

Nous définissons maintenant les concepts liés à la théorie des graphes nécessaires à la description de l'algorithme.

Un *arbre* (fig. 2.1) est une structure de données définie récursivement (D. E. Knuth, 1997) à partir d'un ensemble fini T possédant un ou plusieurs éléments appelés *nœuds*. Plus spécifiquement,

1. il y a un nœud spécial appelé la *racine* de l'arbre ;
2. les autres nœuds (en excluant la racine) sont partitionnés en $m \geq 0$ sous-ensembles disjoints $T_1, \ldots, T_m$ de T qui sont chacun à leur tour des arbres. Les arbres $T_1, \ldots, T_m$ sont appelés les *sous-arbres* de la racine.

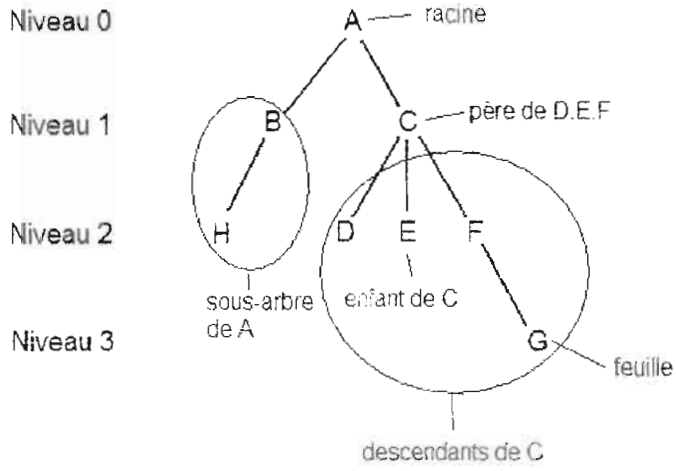

Figure 2.1: Représentation graphique d'un arbre

Chaque nœud de l'arbre est à son tour la racine d'un sous-arbre. Le nombre de sous-arbres d'un nœud est appelé le *degré* du nœud. Les nœuds de degré zéro sont appelés des *nœuds terminaux* ou *feuilles*. Les nœuds du sous-arbre (en excluant sa racine) sont appelés les *descendants* du nœud racine. Les descendants directs d'un nœud sont appelés ses *enfants* et, en retour pour chaque enfant, ce nœud est appelé nœud *père* ou *parent*. Nous pouvons regarder la structure d'un arbre par niveaux de façon récursive : la racine se situe au niveau 0, ses enfants au niveau 1, les enfants des enfants au niveau 2, et ainsi de suite.

2.2 L'algorithme "Leaps and Bounds"

L'idée de l'algorithme "Leaps and Bounds" original est basée sur la construction de deux arbres, l'arbre inverse et l'arbre pairé, et le parcours de ce dernier en ignorant les sous-ensembles de nœuds qui ne respectent pas le critère d'optimalité.

Même si l'algorithme "Leaps and Bounds" utilise l'arbre pairé, nous introduisons maintenant l'arbre inverse pour la compréhension de la structure de l'arbre pairé.

2.2.1 L'arbre inverse

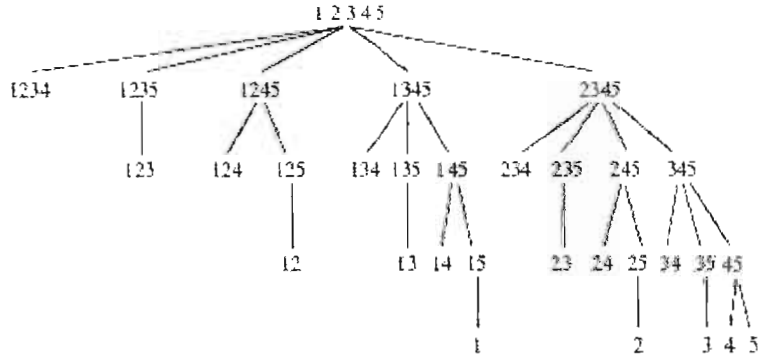

Figure 2.2: Arbre inverse avec $p = 5$ covariables

L'arbre inverse (fig. 2.2) est construit récursivement :

1. la racine est l'ensemble complet $\{1, 2, \ldots, p\}$, où les éléments de cet ensemble représentent les indices des covariables correspondantes ;
2. le premier niveau contient les p enfants de la racine en écartant un élément à la fois de l'ensemble complet, en ordre décroissant, $p, p-1, \ldots, 2, 1$;
3. considérons un nœud associé avec l'ensemble $\{i_1, i_2, \ldots, i_m\}, m \geq 1, i_1 \leq i_2 \leq \ldots \leq i_m$, qui est le j^e enfant de son parent. Au niveau suivant ce nœud aura $j-1$ enfants qui sont générés en écartant un élément à la fois de l'ensemble $\{i_1, i_2, \ldots, i_m\}$ dans l'ordre $i_m, i_{m-1}, \ldots, i_{m+2-j}$;
4. l'arbre s'arrête de croître quand il reste seulement des sous-ensembles avec un seul élément.

L'arbre inverse possède les propriétés suivantes (N. Xuelei, 2006) :

a) l'arbre contient tous les $2^p - 1$ sous-ensembles de $\{1, 2, \ldots, p\}$, différents de l'ensemble vide $\emptyset$;

b) chaque sous-ensemble est présent seulement une fois dans l'arbre ;

c) tous les sous-ensembles du niveau k ont $p-k$ éléments ;

d) chaque sous-ensemble associé avec un nœud de l'arbre est obtenu en écartant un élément du sous-ensemble associé avec le nœud parent.

2.2.2 L'arbre pairé

Nous construisons l'arbre pairé (fig. 2.3) par induction, à partir de l'arbre inverse, en considérant qu'à chaque sous-niveau l'arbre inverse a la même structure.

Pour $p = 2$, l'arbre pairé noté $AP(2)$ (fig. 2.4) contient la racine $(\emptyset, \{12\})$ et un seul enfant $(\{1\}, \{2\})$:

Pour $p = 3$, nous obtenons $AP(3)$ (fig. 2.5) en trois étapes à partir de $AP(2)$.

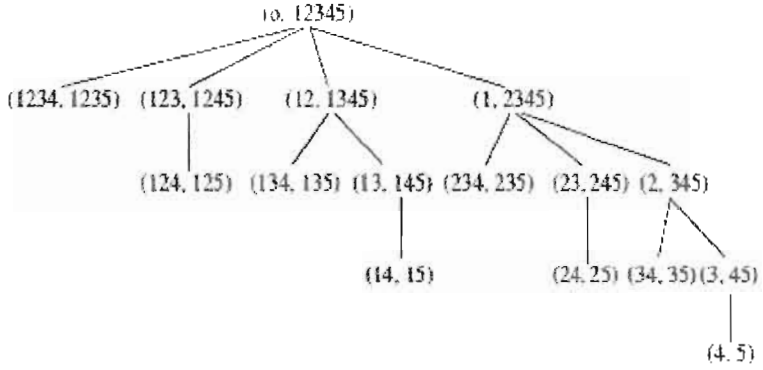

Figure 2.3: Arbre pairé avec $p = 5$ covariables

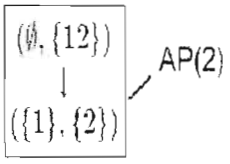

Figure 2.4: Arbre pairé $AP(2)$

En général, étant donné $AP(p)$, nous pouvons générer $AP(p+1)$ de la façon suivante :

1. nous additionnons 1 $(i \to i+1, i = 1, 2, \ldots, p)$ à chaque élément de chaque sous-ensemble dans $AP(p)$, et nous notons A_1 l'arbre obtenu ;
2. nous insérons l'élément 1 dans chaque sous-ensemble de A_1 différent de l'ensemble vide et nous obtenons A_2 ;
3. dans A_1, nous remplaçons $\emptyset$ par $\{1\}$ et nous additionnons l'arbre obtenu comme sous-arbre de la racine de A_2. Les deux arbres combinés forment $AP(p+1)$.

Les propriétés de l'arbre pairé (N. Xuelei, 2006) sont :

a) l'arbre contient chaque sous-ensemble de l'ensemble $\{1, 2, \ldots, p\}$ une seule fois, et chaque nœud a deux sous-ensembles associés ;

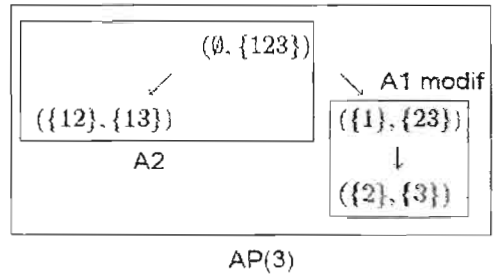

Figure 2.5: Arbre pairé $AP(3)$

b) pour un nœud (Ω_1, Ω_2), tous les sous-ensembles des nœuds enfants sont des sous-ensembles de Ω_2 ;

c) pour un ensemble E, nous notons $|E|$ le nombre d'éléments de l'ensemble E, appelé le *cardinal* de E. Pour un nœud (Ω_1, Ω_2), si $|\Omega_1| = c_1$ et $|\Omega_2| = c_2, 1 \leq c_1, c_2 \leq p$, les cardinaux des sous-ensembles associés à n'importe quel nœud enfant sont plus grands ou égaux à $\min\{c_1, c_2\}$;

d) un nœud (Ω_3', Ω_4') qui est l'enfant de (Ω_1, Ω_2) peut être obtenu de deux façons. Si nous considérons l'ordre de gauche à droite parmi les enfants de (Ω_1, Ω_2), et si dans cet ordre (Ω_3', Ω_4') est le premier enfant de (Ω_1, Ω_2), alors le sous-ensemble Ω_3' (resp. Ω_4') est obtenu en écartant le dernier (resp. l'avant-dernier) élément de Ω_2). Sinon, si (Ω_3', Ω_4') est le j^e enfant de son parent et s'il y a un nœud (Ω_1', Ω_2') directement à la gauche de (Ω_3', Ω_4'), alors Ω_3' est obtenu en écartant le dernier élément de Ω_1', et Ω_4' est obtenu en écartant le $(j+1)^e$ dernier élément de Ω_2. Cette relation assure une façon efficace de parcourir l'arbre de haut en bas et de gauche à droite.

2.2.3 L'algorithme et les tests d'optimalité

Les tests d'optimalité sont essentiels dans le but de réduire le nombre de sous-ensembles à considérer, et donc déterminent la complexité et la vitesse de l'algorithme. Ils sont basés sur le fait que, pour n'importe quel nœud (Ω_1, Ω_2) de l'arbre pairé, nous avons $|\Omega_1| \leq |\Omega_2|$. En étudiant les propriétés et la construction de l'arbre pairé, nous voyons

que tous les modèles de dimension $c+1$ sont situés au niveau $p-c+2$ ou $p-c+1$, c entier. Aussi, tous les modèles de dimension c sont situés au niveau $p-c+1$ ou $p-c$ et ils sont des enfants des modèles de dimension $c+1$. Ainsi pour chaque modèle de dimension c, il existe un modèle de dimension $c+1$ avec un RSS plus petit. Pour un entier c, si nous notons par $RSS(c)$ la somme des carrés des résidus minimale de tous les sous-ensembles de dimension c considérés en parcourant l'arbre pairé jusqu'au nœud (Ω_1, Ω_2), alors nous obtenons l'inégalité : $RSS(c+1) \leq RSS(c)$. Comme chaque sous-ensemble Ω' dans l'arbre représente un sous-modèle de régression, nous notons par $RSS(\Omega')$ la somme des carrés obtenue du sous-modèle correspondant. Parce que l'algorithme trouve $\min(RSS(c))$ pour $c = 1, 2, \ldots, p$, nous décidons de considérer, ou non, un enfant du nœud (Ω_1, Ω_2) en fonction des valeurs $RSS(c)$ et $RSS(\Omega_2)$.

Les tests d'optimalité sont les suivants :

a) nous commençons avec la racine de l'arbre pairé et nous calculons la somme des carrés de chaque sous-ensemble de la racine et des nœuds du premier niveau ;

b) supposons qu'en parcourant l'arbre de haut en bas et de gauche à droite nous avons atteint le nœud (Ω_1, Ω_2). Trois situations peuvent survenir :

   1. si $RSS(|\Omega_1|) \leq RSS(\Omega_2)$, alors nous pouvons ignorer tous les descendants du nœud (Ω_1, Ω_2), car leurs sous-ensembles associés ont une somme de carrés plus grande que $RSS(c)$, c entier, $|\Omega_1| \leq c \leq |\Omega_2|$;

   2. si $RSS(|\Omega_2| - c) \leq RSS(\Omega_2) < RSS(|\Omega_2| - c - 1)$ pour certains c, où $1 \leq c \leq |\Omega_2| - |\Omega_1| - 1$, alors nous pouvons ignorer les premiers c enfants du nœud (Ω_1, Ω_2) ;

   3. si $RSS(\Omega_2) < RSS(|\Omega_2| - 1)$, nous devons considérer tous les enfants du nœud (Ω_1, Ω_2).

D'autres tests d'optimalité qui n'étaient pas dans l'algorithme original ont été introduits par la suite. Ces tests d'optimalité ont l'avantage de réduire les calculs et d'améliorer la vitesse de l'algorithme (par exemple, N. Xuelei, 2006).

2.2.4 Comment parcourir l'arbre pairé

Nous avons présenté les arbres inverse et pairé pour permettre une meilleure compréhension de la structure des données et des tests d'optimalité. En pratique, pour programmer l'algorithme "Leaps and Bounds", nous utilisons l'algorithme suivant qui possède la particularité que chaque nœud de l'arbre pairé $AP(p)$ est atteint seulement une fois. Pour un ensemble ordonné Ω et pour j entier, $1 \leq j \leq |\Omega|$, nous notons par $r(\Omega, j)$ le sous-ensemble obtenu de Ω en écartant le j^e dernier élément de Ω. Nous pouvons maintenant généraliser l'opération r, $r^s(\Omega, j) = \underbrace{r(r(\ldots r}_{s\,fois}(\Omega, j)\ldots, j), j)$. Par exemple, $r(\{12345\}, 1) = \{1234\}$ et $r^2(\{12345\}, 1) = \{123\}$.

À l'aide de l'opération r, nous construisons une liste dont chaque élément contient trois informations, c'est-à-dire les deux ensembles Ω_1 et Ω_2 de chaque nœud et un entier s représentant l'ordre du nœud parmi les enfants de son parent.

L'algorithme de parcours se décrit ainsi en trois étapes :

Nous commençons avec une liste vide. À partir de la racine $(\emptyset, \{1, 2, \ldots, p\})$ nous ajoutons les éléments suivants :

$$\begin{array}{lll} r(\{1,2,\ldots,p\},1), & r(\{1,2,\ldots,p\},2), & 1, \\ r^2(\{1,2,\ldots,p\},1), & r(\{1,2,\ldots,p\},3), & 2, \\ r^3(\{1,2,\ldots,p\},1), & r(\{1,2,\ldots,p\},4), & 3, \\ \vdots & \vdots & \vdots \\ r^{p-1}(\{1,2,\ldots,p\},1), & r(\{1,2,\ldots,p\},3), & p-1. \end{array}$$

Par exemple, pour obtenir l'arbre de la fig. 2.3, nous commençons en ajoutant à la liste les éléments suivants :

$$\begin{array}{lll} (1,2,3,4), & (1,2,3,5), & 1, \\ (1,2,3), & (1,2,4,5), & 2, \\ (1,2), & (1,3,4,5), & 3, \\ (1), & (2,3,4,5), & 4. \end{array}$$

La deuxième étape se décrit comme suit. Si la liste des nœuds n'est pas vide et que son premier élément est (Ω_1, Ω_2, s), où $\Omega_1, \Omega_2 \subset \{1, 2, \ldots, p\}$ et $s \geq 1$, alors :

- si $s = 1$, nous écartons le premier élément dans la liste ;
- si $s > 1$, nous écartons le premier élément et nous ajoutons les éléments suivants à la fin de la liste :

$$\begin{array}{lll} r(\Omega_2, 1), & r(\Omega_2, 2), & 1, \\ r^2(\Omega_2, 1), & r(\Omega_2, 3), & 2, \\ \vdots & \vdots & \vdots \\ r^{s-1}(\Omega_2, 1), & r(\Omega_2, s), & s-1. \end{array}$$

Notez que, dans cette étape, si un élément ne satisfait pas aux tests d'optimalité, alors il n'est pas inséré à la fin de la liste.

Finalement, la troisième étape constitue à répéter l'étape 2 jusqu'à ce que la liste soit vide.

En conclusion, cet algorithme fait une recherche exhaustive parmi tous les modèles avec k paramètres, $k = 1, 2, \ldots, p$, et il est garanti qu'il trouve le modèle avec la somme de carrés des résidus minimale $RSS(k)$ pour chaque k. Cette recherche nécessite un grand effort computationnel et implique que l'algorithme n'est pas efficace pour des modèles considérant plus de 30 covariables (A. Miller, 2002).

Notez qu'il est possible de modifier les tests d'optimalité et de comparer $RSS(\Omega_2)$, pour un nœud (Ω_1, Ω_2) donné, avec la j^e plus petite somme de carrés. De cette façon, il est possible de déterminer les premiers j modèles optimaux de dimension k, $k = 1, ..., p$ sans affecter sensiblement la vitesse et la complexité de l'algorithme (A. Miller, 2002). Cette approche peut être intéressante lorsque nous nous intéressons, non pas uniquement au meilleur modèle, mais à un ensemble élargi de bons modèles.

CHAPITRE III

SELECTION DE MODÈLES DANS LE CONTEXTE BAYÉSIEN

En pratique, les chercheurs font l'analyse des données d'une expérience et trouvent plusieurs modèles, chacun bien ajusté, qui donnent des estimations de l'effet des covariables ou des valeurs prédites complètement différentes. Comment procéder et s'arrêter sur un modèle dans cette situation ? Le moyennage de modèles bayésien (BMA) offre une solution à ce problème. En considérant une moyenne des quantités d'intérêt sur tous les modèles probables, nous éliminons l'incertitude liée au choix du modèle et, du coup, améliorons nos capacités de prédiction (J. A. Hoeting, D. Madigan, A. E. Raftery et C. T. Volinsky, 1999).

Nous commençons le chapitre avec la méthodologie de sélection du meilleur des deux modèles considérés, suivie par la description de la technique BMA. Les deux font partie intégrante du principe de l'algorithme "Occam's Window" présenté dans la Section 3.3.

3.1 Comparaison de deux modèles

Avant d'aborder le moyennage de modèles bayésien, nous introduisons la méthodologie nous permettant de sélectionner un des deux modèles considérés. Supposons que nous voulons comparer les deux modèles, M_0 et M_1. À cette fin, nous voulons évaluer le rapport $\frac{pr(M_0|Y)}{pr(M_1|Y)}$, où $pr(M_0|Y)$ et $pr(M_1|Y)$ sont respectivement les probabilités a posteriori de M_0 et M_1 basées sur les données observées Y. En appliquant la règle de Bayes à ces deux probabilités, nous obtenons :

$$\frac{pr(M_0|Y)}{pr(M_1|Y)} = \frac{pr(Y|M_0)pr(M_0)}{pr(Y|M_1)pr(M_1)}.$$

Si nous supposons qu'avant d'observer les données nous n'avons aucune information concernant les modèles, et donc que leurs probabilités a priori sont égales ($pr(M_0) = pr(M_1)$), alors :

$$\frac{pr(M_0|Y)}{pr(M_1|Y)} = \frac{pr(Y|M_0)}{pr(Y|M_1)} \overset{notation}{=} B_{01},$$

où $pr(Y|M_0)$ et $pr(Y|M_1)$ sont les vraisemblances marginales, définies comme :

$$\begin{array}{l} pr(Y|M_0) = \int pr(Y|\beta_0, M_0)pr(\beta_0|M_0)d\beta_0, \quad \text{et} \\ pr(Y|M_1) = \int pr(Y|\beta_1, M_1)pr(\beta_1|M_1)d\beta_1. \end{array} \tag{3.1}$$

Les variables β_0 et β_1 dans les intégrales 3.1 sont les vecteurs de paramètres des deux modèles considérés et $pr(\beta_0|M_0)$ respectivement $pr(\beta_1|M_1)$ leurs probabilités a priori.

La quantité B_{01} est appelée *facteur de Bayes* et représente le degré d'évidence pour M_0 par rapport à M_1. Si nous notons par $BIC_{01} = BIC_0 - BIC_1$, où BIC_i est le critère BIC pour le modèle $i, i = 0, 1$, alors nous pouvons montrer que (A. E. Raftery, 1991) :

$$B_{01} \approx e^{-\frac{1}{2}BIC_{01}}.$$

L'expression à droite du signe d'équivalence est souvent utilisée en pratique comme approximation au facteur de Bayes.

3.2 Moyennage de modèles bayésien - BMA

Dans toutes les expériences, il y a une quantité d'intérêt, Δ, que nous voulons estimer à partir des données observées, Y. Notons $M_1, \ldots, M_K$ l'ensemble des modèles considérés. Alors, la distribution a posteriori de Δ étant donné Y est :

$$pr(\Delta|Y) = \sum_{k=1}^{K} pr(\Delta|M_k, Y)pr(M_k|Y). \tag{3.2}$$

L'expression (3.2) représente la moyenne de la distribution a posteriori de la quantité d'intérêt en considérant chacun des modèles associés aux poids $pr(M_k|Y)$. Rappelons que la probabilité a posteriori d'un modèle M_k est donnée par :

$$pr(M_k|Y) = \frac{pr(Y|M_k)pr(M_k)}{\sum_{l=1}^{K} pr(Y|M_l)pr(M_l)}, \text{ où}$$

$$pr(Y|M_k) = \int pr(Y|\beta_k, M_k)pr(\beta_k|M_k)d\beta_k$$

est l'intégrale de la fonction de vraisemblance du modèle M_k ($pr(Y|\beta_k, M_k)$), β_k est le vecteur de paramètres du modèle M_k, $pr(\beta_k|M_k)$ est la densité a priori de β_k et $pr(M_k)$ est la probabilité a priori du modèle M_k.

Si pour le modèle M_k, nous estimons la quantité d'intérêt par $\hat{\Delta}_k = E[\Delta|Y, M_k]$, alors le moyennage de modèles bayésien nous indique de calculer l'espérance et la variance a posteriori de Δ de la façon suivante :

$$E[\Delta|Y] = \sum_{k=1}^{K} \hat{\Delta}_k pr(M_k|\Delta) \text{ et}$$

$$Var[\Delta|Y] = \sum_{k=1}^{K} (Var[\Delta|Y, M_k] + \hat{\Delta}_k^2)pr(M_k|Y) - E[\Delta|Y]^2.$$

L'utilisation du moyennage de modèles nécessite l'obtention de tous les modèles importants. L'algorithme Occam's Window que nous décrivons dans ce qui suit a été suggéré pour l'obtention de ceux-ci, bien qu'une variante hybride soit utilisée par le paquetage R BMA qui fait l'implantation du moyennage de modèles bayésien pour les modèles statistiques les plus courants (davantage de détails seront donnés au Chapitre 4).

3.3 L'Algorithme "Occam's Window"

L'algorithme "Occam's Window" utilise le principe énoncé par William of Occam, un logicien anglais du XIV^e siècle, qui affirmait que "les entités ne doivent pas être multipliées au-dessus de la nécessité". La conclusion de ce principe est que les explications et

les stratégies les plus simples ont tendance à être les meilleures. Ce principe peut être appliqué à la sélection de modèles de la façon suivante. Supposons que nous avons K modèles. Alors si $pr(M_1|Y) = pr(M_2|Y) = \ldots = pr(M_K|Y)$, nous préférons le modèle le plus simple, avec le plus petit nombre de paramètres.

L'algorithme "Occam's Window" se sert d'un intervalle (une fenêtre) pour comparer deux modèles à la fois dans l'espace des modèles afin de sélectionner un ensemble de bons modèles. Pour une expérience avec p covariables initiales, chaque modèle est une combinaison d'une ou plusieurs de ces covariables, ainsi l'espace des modèles correspondant contient 2^p éléments. Cet algorithme utilise le facteur de Bayes B_{01} ou la différence de BIC, BIC_{01}, dans le but de déterminer lequel des deux modèles est le meilleur. Les bornes de l'intervalle proposées par Raftery (1991) sont $O_L = 0$ et $O_R = 9.2$ pour BIC_{01}, respectivement $O_L = 10^{-2}$ et $O_R = 1$ pour B_{01}. La règle de base pour parcourir l'espace des modèles est que, si un modèle a été rejeté, alors tous ses sous-modèles sont aussi rejetés. De plus, nous avons les trois situations suivantes :

1) si $BIC_{01} < O_L$, alors nous acceptons M_0 et nous rejetons M_1 ;

2) si $O_L \leq BIC_{01} < O_R$, alors nous acceptons M_0 et M_1 ;

3) si $BIC_{01} \geq O_R$, alors nous acceptons M_1 et nous rejetons M_0.

Dans la première situation, il y a une forte évidence favorisant M_0, mais pas d'évidence pour M_1. La deuxième situation regroupe deux possibilités : il y a une faible évidence pour M_1 donc nous ne pouvons pas rejeter M_0 ou il y a une faible évidence pour M_0 et nous ne pouvons pas rejeter M_1. Dans ce cas, les deux modèles sont acceptés. Finalement, dans la troisième situation, les données favorisent M_1 seulement.

Si nous utilisons le facteur de Bayes au lieu de la différence de BIC, les trois situations sont inversées de la façon suivante :

1) si $B_{01} < O_L$, alors nous acceptons M_1 et nous rejetons M_0 ;

2) si $O_L \leq B_{01} < O_R$, alors nous acceptons M_0 et M_1 ;

3) si $B_{01} \geq O_R$, alors nous acceptons M_0 et nous rejetons M_1.

3.3.1 Description de l'algorithme "Occam's Window"

Notons EM l'ensemble des modèles, A l'ensemble des modèles "acceptés" et F l'ensemble des modèles "en considération". Nous utilisons dans la description de l'algorithme la notation formelle $E_1 \leftarrow E_2$, pour deux ensembles quelconques E_1 et E_2, pour décrire le fait que l'ensemble E_1 devient l'ensemble E_2. Il y a trois façons de parcourir l'espace des modèles, chacune commençant avec $A = \emptyset$ et $F =$ l'ensemble des modèles initiaux. Ces variations de l'algorithme "Occam's Window" sont appelées : "Down", "Up" et "Up - Down", respectivement. Nous décrivons dans un premier temps la variation "Down" :

1. On sélectionne un modèle $M \in F$;
2. $F \leftarrow F - M$ et $A \leftarrow A + M$;
3. On sélectionne un sous-modèle M_0 de M ;
4. On calcule $B = \frac{pr(M_0|Y)}{pr(M|Y)}$ (ou $BIC = BIC_{M_0} - BIC_M$) ;
5. Si $B > O_R$ (respectivement $BIC < O_L$), alors $A \leftarrow A - M$ et si $M_0 \notin F, F \leftarrow F + M_0$;
6. Si $O_L \leq B \leq O_R$ (respectivement $O_L \leq BIC \leq O_R$), alors si $M_0 \notin F, F \leftarrow F + M_0$;
7. S'il y a encore des sous-modèles de M, on répète à partir de l'étape 3 ;
8. Si $F \neq \emptyset$, on répète à partir de l'étape 1.

La variation "Up" de l'algorithme "Occam's Window" se décrit comme suit :

1. On sélectionne un modèle $M \in F$;
2. $F \leftarrow F - M$ et $A \leftarrow A + M$;
3. On sélectionne un super-modèle M_1 de M ;
4. On calcule $B = \frac{pr(M|Y)}{pr(M_1|Y)}$ (ou $BIC = BIC_M - BIC_{M_1}$) ;
5. Si $B < O_L$ (respectivement $BIC > O_R$), alors $A \leftarrow A - M$ et si $M_1 \notin F, F \leftarrow F + M_1$;
6. Si $O_L \leq B \leq O_R$ (respectivement $O_L \leq BIC \leq O_R$), alors si $M_1 \notin F, F \leftarrow F + M_1$;

7. S'il y a encore des super-modèles de M, on répète à partir de l'étape 3 ;

8. Si $F \neq \emptyset$, on répète à partir de l'étape 1.

La variation "Up - Down" est une combinaison des variations "Up" et "Down" et intervient dans la situation suivante. Si l'ensemble de départ F est constitué du modèle complet, qui est le super-modèle de tous les autres modèles, nous pouvons utiliser la variation "Up" et si F contient tous les modèles minimaux, alors nous employons la variation "Down". Si les bornes de la fenêtre de l'algorithme sont choisies de telle façon que celui-ci ne manque aucun modèle acceptable, et si l'ensemble de départ est constitué du modèle complet pour la variation "Up", respectivement de tous les modèles de dimension un pour la variation "Down", alors les deux variations sont équivalentes en terme d'ensemble A final. Quand F est un ensemble quelconque de départ, nous employons la variation "Up - Down" (ou "Down - Up") en deux étapes, en exécutant premièrement l'algorithme "Up" et après l'algorithme "Down", afin de sélectionner les meilleurs modèles. L'ordre dans lequel nous décidons d'exécuter les deux variations a peu d'influence sur le résultat final.

3.3.2 L'efficacité de l'algorithme et difficultés d'implantation

Il est intéressant de s'attarder sur l'algorithme "Occam's Window" tel que décrit ci-dessus. Les problématiques relevées sont résolues dans l'implantation de l'algorithme présenté dans le Chapitre 4. Premièrement, l'algorithme n'est pas efficace en ce qui concerne le temps d'exécution, car il fait des calculs inutiles. En effet, si nous générons tous les sous-modèles de chacun des modèles initiaux, il est possible de considérer le même sous-modèle plus d'une fois. Par exemple, si en appliquant le critère de sélection, l'algorithme décide d'explorer les sous-modèles des nœuds $\{1,2,3,4,5\}$ et $\{1,2,3,4,6\}$, il génère le sous-modèle $\{1,2,3,4\}$ deux fois. Récursivement, il génère aussi les sous-modèles de ce dernier et il fait les calculs pour décider également de l'accepter ou non deux fois. Deuxièmement, s'il y a des répétitions, il est nécessaire de vérifier chaque fois lorsque nous acceptons un modèle (aux étapes 5 et 6) s'il est déjà inclus dans l'ensemble

F. Cette opération peut prendre beaucoup de temps, particulièrement si l'ensemble F contient un grand nombre d'éléments.

Pour résoudre ces deux problèmes simultanément, nous regroupons logiquement les modèles dans une structure d'arbre inverse (voir la Section 2.2.1). Ce regroupement cause un nouveau problème d'implantation. En effet, l'arbre inverse contient alors des "vrais" nœuds terminaux associés aux modèles minimaux et des "faux" nœuds terminaux qui contiennent des modèles associés à des sous-modèles que nous ne voulons pas considérer, car ils se répètent dans d'autres parties de l'arbre. Par exemple, dans la figure de l'arbre inverse (fig. 2.2), les nœuds $\{1,2,3\}$ et $\{1,2,4\}$ sont de faux nœuds terminaux et les nœuds $\{1\}$, $\{2\}$, $\{3\}$, $\{4\}$, $\{5\}$ sont vrais. Quand l'algorithme atteint un faux nœud terminal, il doit générer ses sous-modèles seulement dans le but de décider s'il va accepter ou non le modèle associé, même s'il ne va pas continuer la recherche dans cette direction. Nous nous souvenons que seulement les modèles qui n'ont pas de meilleurs sous-modèles sont acceptés ; ainsi, il est nécessaire de calculer le BIC de chaque sous-modèle pour déterminer si l'un d'entre eux est meilleur ou non.

Pour obtenir une meilleure vitesse d'exécution, l'algorithme ne compare pas un modèle avec tous ses sous-modèles. Les comparaisons sont faites seulement entre un modèle et ses enfants directs présents dans l'arbre inverse. Parce qu'ils n'ont jamais été comparés, il est possible de sélectionner simultanément un modèle et un ou plusieurs de ses meilleurs sous-modèles. Après la sélection, il est ainsi nécessaire de parcourir encore une fois l'ensemble des modèles sélectionnés pour écarter les modèles en question.

3.3.3 Validation des sous-modèles

En pratique, les modèles peuvent aussi contenir des covariables catégoriques et des interactions de covariables. À une covariable avec c catégories correspond $c-1$ colonnes dans la matrice des données. Nous ne pouvons pas regarder ces colonnes comme étant associées à $c-1$ covariables "simples". Au contraire, nous devons les traiter ensemble, et les éliminer simultanément de la matrice quand nous générons un sous-modèle qui

n'inclut pas la covariable catégorique associée.

Dans le processus de sélection, il est important de considérer des modèles valides. Un modèle est dit valide s'il ne contient pas d'interaction sans aussi contenir les covariables composantes. Comme il est possible pour un modèle non valide d'avoir des sous-modèles valides, en éliminant l'interaction qui cause ce problème, nous devons valider chacun des sous-modèles à la fois. Ce mécanisme soulève une nouvelle question. Comment comparer un modèle valide avec un autre qui ne l'est pas ? Nous avons deux situations :

a) si le modèle considéré est valide alors pour décider s'il est accepté ou non, nous comparons son BIC seulement avec les BIC de ses sous-modèles valides ;

b) si le modèle n'est pas valide, nous comparons les BIC de ses sous-modèles avec le BIC de son plus proche super-modèle valide ; nous supposons que la racine de l'arbre est toujours valide.

CHAPITRE IV

DESCRIPTION DES LOGICIELS

Nous avons présenté dans les Chapitres 2 et 3 les détails théoriques des algorithmes "Leaps and Bounds" et "Occam's Window". Dans ce chapitre, nous expliquons, dans des termes informatiques, comment utiliser de façon pratique les ensembles de fonctions qui implémentent ces deux algorithmes, soient le paquetage BMA de R (R Development Core Team, 2010) et l'implantation contribuée de "Occam's Window". Nous présentons dans ce qui suit la liste des principales fonctions et nous expliquons leur structure interne. Nous discutons aussi des paramètres les plus importants, à savoir ceux qui déterminent leur résultat final, c'est-à-dire l'ensemble des modèles sélectionnés. Ces informations nous aident à effectuer l'étude comparative entre les deux méthodes, dans le Chapitre 5.

4.1 Ensemble de fonctions contribuées pour l'algorithme "Occam's Window"

4.1.1 Liste de fonctions

L'implantation maison de l'algorithme "Occam's Window" contient les fonctions suivantes écrites dans le langage "C"

checkFile - vérifie si le fichier de données existe et qu'il contient toutes les informations qui décrivent le jeu de données ; un tel fichier contient quatre sections : DATA, CATE, COLN et INTE ; la section DATA est un tableau qui contient la matrice de données, la variable réponse et les poids des observations ; les autres sections spécifient les noms

de covariables (COLN), les covariables catégoriques (CATE) et s'il y a des interactions entre les covariables (INTE) ;

checkData - vérifie si le fichier de données a suffisamment de lignes et que ces dernières ont la même longueur ;

createData - lit le fichier de données et fait l'initialisation des structures vectorielles utilisées pour emmagasiner les données dans la mémoire interne. Ces vecteurs sont utilisés plus tard pour calculer les coefficients des modèles et diverses statistiques comme le BIC ou l'erreur quadratique moyenne ;

fit_linreg - calcule les coefficients d'un modèle de régression linéaire en utilisant la fonction **gsl_multifit_linear** de la librairie GSL (GNU Scientific Library) ; GSL est une librairie scientifique développée pour le langage C, qui contient fonctions pour des calcules numériques et statistiques ;

linreg - initialise la matrice de données, appelle la fonction **fit_linreg** et affiche les coefficients du modèle de régression linéaire ;

fit_linreg_sweep - calcule les coefficients d'un modèle de régression linéaire en utilisant l'opérateur de rotation (sweep) ; pour plus d'information sur l'opérateur "sweep" voir l'Annexe B ;

linreg_sweep - initialise la matrice de données, appelle la fonction **fit_linreg_sweep** et affiche les coefficients du modèle de régression linéaire ;

fit_logreg - calcule les coefficients d'un modèle de régression logistique en utilisant la fonction **fit_linreg_sweep** dans un algorithme de type IRLS (Iteratively Reweighted Least Square) ;

logreg - initialise la matrice de données, appelle la fonction **fit_logreg** et affiche les coefficients du modèle de régression logistique ;

occamswin_linreg_with_validation - sélectionne les meilleurs modèles de régression linéaire valides de l'espace des modèles en utilisant l'algorithme "Occam's Window" ;

occamswin_linreg_no_validation - sélectionne les meilleurs modèles de régression linéaire sans validation en utilisant l'algorithme "Occam's Window" ;

occamswin_logreg_with_validation - sélectionne les meilleurs modèles de régression logistique valides de l'espace des modèles en utilisant l'algorithme "Occam's Window" ;

occamswin_logreg_no_validation - sélectionne les meilleurs modèles de régression logistique sans validation en utilisant l'algorithme "Occam's Window" ;
printGslMatrix - affiche les modèles trouvés sur l'écran ;
printGslMatrixToFile - écrit les modèles trouvés dans un fichier texte ;

4.1.2 Détails d'implantation

Pour mieux comprendre leur fonctionnement, les paramètres qui déterminent leur résultat et même le résultat qu'elles trouvent, nous présentons maintenant quelques détails pour les fonctions les plus compliquées.

La fonction **fit_logreg** est une implantation de l'algorithme IRLS (voir section 1.2). Elle utilise la fonction interne **eval_f** pour recalculer les probabilités des observations et pour évaluer à chaque étape la valeur du logarithme de la fonction de vraisemblance. Notons qu'il est possible, si la solution initiale n'est pas bien choisie, que l'algorithme ne converge pas. Si l'algorithme est convergent, la valeur du logarithme de la fonction de vraisemblance est croissante à chaque étape. Si nous obtenons une valeur plus petite que dans l'étape précédente, il est possible que l'algorithme ait dépassé le point de maximum. Dans ce cas, il exécute quelques étapes de plus en prenant comme point de départ le point milieu entre le point courant et le point trouvé à l'étape précédente, jusqu'à ce qu'il trouve une meilleure solution ou qu'il décide de s'arrêter sans converger. Notons qu'il est aussi possible que certaines des probabilités des observations recalculées à chaque étape soient *0* ou *1* à cause des arrondissements de l'ordinateur et que, dans ce cas aussi, l'algorithme ne soit pas convergent. Si l'algorithme ne converge pas, la fonction affiche un message d'erreur.

Les deux fonctions qui traitent de la sélection des modèles logistiques, **occamswin_logreg_with_validation** et **occamswin_logreg_no_validation** appellent la fonction **fit_logreg** qui, pour certains jeux de données et pour certains modèles, peut obtenir des probabilités pour les observations égales à *0* ou *1*. Dans ce cas, les valeurs de *BIC* pour

certains modèles peuvent être égales à plus infini. Si un modèle a un *BIC* infini, il va être automatiquement rejeté, et l'algorithme va afficher un message d'erreur.

Toutes les fonctions qui implémentent l'algorithme "Occam's Window" ont deux paramètres qui déterminent la largeur de la fenêtre d'Occam. Les valeurs proposées pour ces bornes sont *0* pour la borne inférieure et *9.2* pour la borne supérieure. Si nous utilisons une plus grande valeur pour la borne inférieure, l'algorithme trouve des modèles de plus petite dimension. Par contre, si nous employons une valeur plus petite pour la borne supérieure, l'algorithme trouve des modèles avec un plus grand nombre de paramètres.

4.2 Le paquetage Leaps de R

Ce paquetage a été développé par T. Lumley (basé sur le code Fortran de A. Miller, 2009) et contient des fonctions qui implémentent plusieurs algorithmes pour la sélection de modèles, comme Leaps and Bounds et les algorithmes de sélection progressive, pas-à-pas, et régressive. Dans ce qui suit, nous décrivons la seule fonction de ce paquetage que nous avons utilisée, **regsubsets**.

4.2.1 La fonction regsubsets

La fonction **regsubsets** implémente plusieurs algorithmes pour la sélection de modèles et a trois formes qui diffèrent seulement dans leur façon de spécifier le jeu de données :

regsubsets (x=, data=, weights=NULL, nbest=1, nvmax=8, force.in=NULL, force.out=NULL, intercept=TRUE, method=c("exhaustive", "backward", "forward", "seqrep"), really.big=FALSE,...)

regsubsets (x=, y=, weights=rep(1,length(y)), nbest=1, nvmax=8, force.in=NULL, force.out=NULL, intercept=TRUE, method=c("exhaustive", "backward", "forward", "seqrep"), really.big=FALSE,...)

regsubsets (x, nbest=1, nvmax=8, force.in=NULL, method=c("exhaustive", "backward", "forward", "seqrep"), really.big=FALSE,...).

Les arguments de ces trois formes sont :

x - la matrice de données, un objet R de type formule ou un objet R de type biglm ;

data - un objet de type data.frame contenant les variables du modèle ;

y - le vecteur de réponses ;

weights - le vecteur de poids ;

nbest - une valeur numérique entière positive spécifiant le nombre de modèles de chaque dimension retourné par la fonction regsubsets ;

nvmax - la dimension maximale des modèles sélectionnés ;

force.in - index de variables (colonnes dans la matrice de données) forcées d'apparaître dans tous les modèles sélectionnés ;

force.out - index de variables (colonnes dans la matrice de données) forcées de ne pas apparaître dans les modèles sélectionnés ;

intercept - si TRUE, rajoute l'intercept au modèle ;

method - spécifie la méthode utilisée, leaps and bounds, sélection progressive, pas-à-pas ou régressive ;

really.big - doit être TRUE pour la recherche exhaustive si le nombre de variables du modèle complet est plus grand que 50 ;

object - un objet R de type regsubsets ;

all.best - si TRUE, affiche tous les meilleurs modèles ; sinon seulement un modèle de chaque dimension ;

matrix - si TRUE, affiche la matrice des variables pour chacun des modèles sélectionnés ; sinon, affiche autres statistiques ;

matrix.logical - quand matrix = TRUE, si matrix.logical = TRUE les variables dans la matrice réponse sont spécifiées en utilisant les mots "TRUE" et "FALSE" ; si matrix.logical = FALSE, les caractères utilisés sont "*" et l'espace ;

df - spécifie le degré de liberté pour les statistiques affichées ; la valeur par défaut est $n - 1$, où n est le nombre d'observations ;

id - l'index de modèles pour lesquels la fonction doit retourner les coefficients et la matrice de variance-covariance ;

vcov - si TRUE, retourne la matrice de variance-covariance.

4.3 Le paquetage BMA de R

Le paquetage BMA est une implantation de la technique BMA dans les langages R et Fortran (A. Raftery, J. Hoeting, C. Volinsky, I. Painter et K. Y. Yeung, 2009). Il contient des procédures qui font le moyennage de modèles pour des modèles de régression linéaire et généralisée et aussi pour des modèles de survie. Ce paquetage dépend de deux autres modules, R-leaps et R-survival. La sélection des modèles est faite à partir de la probabilité a posteriori relative de chaque modèle, calculée à l'aide du critère BIC. Notons par ailleurs que ce paquetage est couramment utilisé pour effectuer du moyennage de modèles dans un cadre d'inférence purement fréquentiste. Dans ce qui suit, nous allons énumérer la liste des fonctions en détaillant deux d'entre elles qui nous intéressent plus, **bicreg** et **bic.glm**. La liste complète des fonctions du paquetage BMA est présentée dans l'Annexe A.

4.3.1 Les fonctions bicreg et bic.glm

La fonction **bicreg** implémente la technique BMA pour des modèles de régression linéaire et a la forme :

bicreg (x, y, wt = rep(1, length(y)), strict = FALSE, OR = 20,
maxCol = 31, drop.factor.levels = TRUE, nbest = 10)

Ses arguments sont :

x - la matrice de données ;

y - le vecteur de réponses ;

wt - le vecteur de poids ;

strict - variable logique ; si FALSE, la fonction retourne tous les modèles ayant une probabilité a posteriori qui n'est pas plus petite que 1/OR fois la probabilité du meilleur modèle ; si TRUE, la fonction élimine les modèles qui ont des sous-modèles avec de plus grandes probabilités a posteriori ;

OR - le ratio maximal pour exclure un modèle dans l'algorithme "Occam's Window" ;

maxCol - le nombre maximal de colonnes (variables) dans la matrice des données ; les

variables supplémentaires sont éliminées avec un algorithme de régression pas-à-pas ;

drop.factor.levels - variable logique qui indique si les catégories d'une covariable catégorique sont traitées ensemble ou séparément pendant l'élimination des colonnes supplémentaires ;

nbest - une valeur numérique positive entière spécifiant le nombre de modèles de chaque dimension et retourné à la fonction **bicreg** par le module leaps ;

La fonction **bic.glm** est similaire à **bicreg**, mais pour le traitement des modèles généralisés. En particulier, nous pouvons l'utiliser pour des modèles logistiques. Cette fonction a deux formes :

bic.glm (x, y, glm.family, wt = rep(1, nrow(x)), strict = FALSE,
prior.param = c(rep(0.5, ncol(x))), OR = 20, maxCol = 30, OR.fix = 2,
nbest = 150, dispersion = , factor.type = TRUE,
factor.prior.adjust = FALSE, occam.window = TRUE, ...)

bic.glm (f, data, glm.family, wt = rep(1, nrow(data)), strict = FALSE,
prior.param = c(rep(0.5, ncol(x))), OR = 20, maxCol = 30, OR.fix = 2,
nbest = 150, dispersion = , factor.type = TRUE,
factor.prior.adjust = FALSE, occam.window = TRUE, ...).

Les deux formes diffèrent seulement par leur façon de spécifier le jeu de données. La première forme prend comme paramètres directement la matrice de données et le vecteur de réponses. La deuxième est appelée avec un objet R de type formule. Les arguments de ces deux formes sont :

x - la matrice de données ;

y - le vecteur de réponses ;

f - un objet R de type formule ;

data - un objet de type data.frame contenant les variables du modèle ;

glm.family - description de la distribution des erreurs et de la fonction de lien du modèle ; cela peut être une chaîne de caractères, un objet R de type "family" ou le résultat d'une fonction qui retourne un objet R de type "family" ; par exemple, nous pouvons utiliser glm.family = "binomial" pour des modèles de régression logistique ;

wt - le vecteur de poids ;

strict - variable logique ; si FALSE, la fonction retourne tous les modèles ayant une probabilité a posteriori plus grande que 1/OR fois la probabilité du meilleur modèle ; si TRUE, la fonction élimine les modèles qui ont des sous-modèles avec une probabilité a posteriori plus grande ;

prior.param - un vecteur spécifiant les poids a priori de chaque covariable ;

OR - le ratio maximal pour exclure un modèle dans l'algorithme "Occam's Window" ;

maxCol - le nombre maximal de colonnes dans la matrice des données ; les colonnes supplémentaires sont éliminées avec un algorithme de sélection pas-à-pas ;

OR.fix - la largeur de la fenêtre de l'algorithme "Occam's Window" après l'approximation du BIC effectuée par "Leaps and Bounds" ; puisque leaps donne seulement une approximation du BIC, il est nécessaire d'augmenter la largeur de cette fenêtre pour s'assurer qu'aucun des bons modèles n'est rejeté ; le niveau de ce ratio est $\frac{1}{OR^{OR.fix}}$ et la valeur par défaut pour OR.fix est 2 ;

nbest - une valeur numérique entière positive spécifiant le nombre de modèles de chaque dimension retourné à la fonction bic.glm par l'algorithme leaps ;

dispersion - une valeur logique spécifiant si la dispersion est calculée ou non ; la valeur de défaut est TRUE lorsque glm.family est "poisson" ou "binomial" ;

factor.type - variable logique qui indique si les catégories d'une covariable catégorique sont traitées ensemble ou séparément ; si FALSE, les catégories sont traitées séparément ;

factor.prior.adjust - variable logique qui indique dans le cas factor.type = F si les distributions a priori correspondantes à chaque catégorie sont ajustées ou non ; si factor.type = F, toutes les probabilités a priori des catégories de la covariable i sont égales à prior.param[i] ; si factor.type = T, les probabilités a priori des catégories de la covariable i sont ajustées telles que leur somme est égale à prior.param[i] ;

occam.window - une valeur logique spécifiant si l'algorithme "Occam's Window" est utilisé ou non ; si FALSE, tous les modèles sélectionnés par l'algorithme "Leaps and Bounds" sont retournés ;

4.3.2 Détails d'implantation

L'étude du code des fonctions **bicreg et bic.glm** nous aide à mieux comprendre leur utilisation, leur structure interne et les autres algorithmes et fonctions qu'elles emploient. Il est intéressant d'observer que, même si elles traitent de types de modèles différents, les deux fonctions ont la même structure. Premièrement, s'il y a plus de *maxCols* covariables, celles-ci sont éliminées à l'aide de la fonction interne *dropcols*, qui est une implantation de la technique de sélection régressive (voir la section 1.4.3 pour plus de détails). La valeur de défaut pour le paramètre *maxCols* est 31. Deuxièmement, une des fonctions R **leaps** ou **regsubsets** est appelée pour éliminer la plupart des modèles. Ces deux fonctions appellent des routines FORTRAN qui implémentent l'algorithme "leaps and bounds". Seulement *nbest* modèles de chaque dimension sont retenus. Finalement, à partir des modèles retenus, l'algorithme "Occam's Window" réduit encore plus le nombre de modèles. Cette dernière réduction est toujours faite à l'intérieur de la fonction **bicreg**. Dans le cas de la fonction **bic.glm**, la réduction est faite seulement si le paramètre *occam.window* a la valeur TRUE.

En conclusion, les fonctions bicreg et bic.glm implémentent un algorithme hybride qui est en fait une combinaison entre les algorithmes "Leaps and Bounds" et "Occam's Window". Le résultat est un objet R de type **bicreg** (respectivement **bic.glm**) qui contient un vecteur des modèles sélectionnés et des informations supplémentaires sur ceux-ci.

CHAPITRE V

ÉTUDE COMPARATIVE DES MÉTHODES

Le but de ce mémoire est de faire une étude comparative entre les deux algorithmes de sélection de modèles présentés dans les chapitres antérieurs, "Leaps and Bounds" (LB) et "Occam's Window" (OW). En effet, nous allons comparer trois façons d'appliquer ces deux algorithmes dans la sélection des modèles de régression linéaire et logistique :

1) Leaps and Bound seulement (LB) ;

2) Occam's Window seulement (OW) ;

3) Combinaison entre Leaps and Bound et Occam's Window (LB-OW).

L'objectif de ces comparaisons est d'évaluer le pouvoir prédictif des ensembles de modèles sélectionnés dans chaque cas et de déterminer s'il y a une méthode qui se démarque. Pour ce faire, nous utilisons le moyennage de modèles bayésien (BMA) et une autre technique nommée "validation croisée" que nous expliquons dans la Section 5.1.

Même s'il n'est pas pertinent de comparer les temps d'exécution des algorithmes, parce qu'ils sont implémentés dans des langages de programmation différents, nous allons les mentionner uniquement dans un but informatif.

5.1 Validation croisée

La validation croisée est une technique très simple utilisée pour l'évaluation du pouvoir de prédiction d'un modèle ou du moyennage des plusieurs modèles. Cette technique a

plusieurs variantes. Dans sa forme la plus simple, nous partitionnons aléatoirement les observations en deux catégories, les données de construction, Y^C, à l'aide desquelles nous ajustons un modèle M, et les données test, Y^T, pour évaluer son pouvoir de prédiction.

Une façon simple de mesurer le pouvoir prédictif d'un modèle est de calculer l'erreur quadratique moyenne (EQM) des données test Y^T. Si pour une observation quelconque, i, nous notons y_i la valeur observée et $\hat{y}_i$ sa valeur prédite obtenue à l'aide du modèle M, alors l'EQM est définie comme :

$$EQM = \frac{1}{|Y^T|} \sum_{i \in Y^T} (y_i - \hat{y}_i)^2, \tag{5.1}$$

où $|Y^T|$ est le cardinal de l'ensemble Y^T. La formule (5.1) s'applique également dans le cas où nous voulons évaluer le pouvoir prédictif d'un ensemble de K modèles. Dans ce cas, la valeur prédite $\hat{y}_i$ est calculée en faisant la moyenne pondérée des valeurs prédites par chacun des modèles, en prenant comme poids leurs probabilités a posteriori :

$$\hat{y}_i = \sum_{k=1}^{K} \hat{y}_{i(k)} pr(M_k|Y^C),$$

où $\hat{y}_{i(k)}$ est la valeur prédite de l'observation i par le modèle M_k.

Une autre variante de la technique de validation croisée est de diviser les données en m parties et d'appliquer le même principe en m étapes. À chaque étape, la partie considérée constitue l'ensemble de données de test, à l'aide duquel nous testons le pouvoir de prédiction. Nous ajustons le modèle d'intérêt avec les données qui restent après avoir écarté de l'ensemble de données complet les observations appartenant à l'ensemble de données de test. Finalement, nous pouvons faire une moyenne des résultats obtenus à chaque étape pour un modèle donné.

La technique de validation croisée fonctionne aussi pour les modèles de régression logistique, sauf qu'il n'est plus adéquat d'utiliser l'erreur quadratique moyenne pour mesurer le pouvoir de prédiction de chaque méthode. Dans ce cas, la réponse est une variable dichotomique et la valeur prédite correspondante à une observation i ne représente plus l'estimation de la réponse, mais l'estimation de la probabilité que la réponse prenne la valeur *1* :

$$\hat{\pi}_i = P(Y_i = 1|(X_{i1}, X_{i1}, \ldots, X_{ip})).$$

Une façon simple de prédire les réponses est de considérer une probabilité de contrôle (par exemple $pr = 0.50$) et de calculer :

$$\hat{y}_i = \begin{cases} 1, & \hat{\pi}_i \geq pr\,; \\ 0, & \text{sinon.} \end{cases}$$

Pour mesurer le pouvoir de prédiction d'un modèle de régression logistique, nous pouvons calculer le taux de mauvaise classification (TMC),

$$TMC = \frac{Nb.\ d'observations\ pour\ lesquelles\ y_i \neq \hat{y}_i}{Nb.\ total\ d'observations}. \tag{5.2}$$

Toutes proportions gardées, une petite valeur de l'EQM ou de TMC indique un grand pouvoir prédictif.

5.2 Comparaison des méthodes dans le cas de la régression linéaire

5.2.1 Données Longley

Nous commençons les comparaisons avec le jeu de données Longley, présenté à l'Annexe C.1. Nous voulons modéliser la variable Employés en fonction des covariables EI, PNB, SE, AF, Population et Année. En ajustant un modèle de régression linéaire aux données complètes, nous obtenons les résultats présentés au Tableau 5.1.

Covariable	Estimation	Err type	Valeur-p
(Intercept)	-3.482e+03	8.904e+02	0.004
EI	1.506e-02	8.492e-02	0.863
PNB	-3.582e-02	3.349e-02	0.313
SE	-2.020e-02	4.884e-03	0.003
AF	-1.033e-02	2.143e-03	0.001
Population	-5.110e-02	2.261e-01	0.826
Année	1.829e+00	4.555e-01	0.003

Tableau 5.1: Résultats de la régression linéaire pour le jeu de données Longley

Le Tableau 5.1 indique que les variables EI, PNB et Population ne sont pas significatives à un seuil de 95%. En ce qui concerne l'ajustement du modèle, nous voyons dans la Figure 5.1 que les suppositions de la régression linéaire ne sont pas en totalité respectées. Dans le premier graphique, les couples des valeurs prédites et observées sont alignés sur une droite, mais les résidus obtenus ne semblent pas suivre une loi normale. Dans le graphique quantiles-quantiles (Q-Q), nous pouvons comparer la distribution des résidus standardisés arrangés en ordre croissant avec la distribution théorique de la loi normale N(0,1). Comme les points ne sont pas alignés sur une droite, les résidus ne suivent pas une loi normale. Les deux derniers graphiques affichent les couples des valeurs prédites (respectivement des valeurs observées) et les résidus. Nous constatons que les points dans les deux derniers graphiques sont aléatoirement distribués, ce qui suggère que les résidus sont indépendants et homoscédastiques.

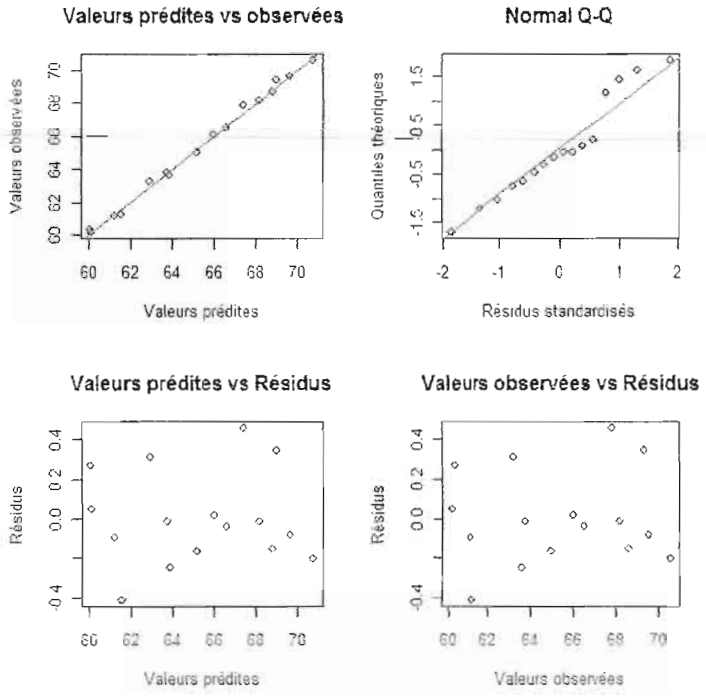

Figure 5.1: Évaluation des suppositions du modèle de régression linéaire pour le jeu de données Longley

Pour un modèle avec *6+1* paramètres comme le modèle complet du jeu de données

Longley, le nombre de sous-modèles possibles de chaque dimension est présenté dans le Tableau 5.2. Les algorithmes LB et OW ne font pas de sélection de variables sur l'intercept, qui est toujours considéré présent.

Nb covariables	1	2	3	4	5	6	Total
Nb modèles	6	15	20	15	6	1	63

Tableau 5.2: Nombre de modèles possibles de chaque dimension pour le jeu de données Longley

La technique BMA est utile quand nous analysons des jeux de données avec un petit nombre d'observations. En effet, dans cette situation, l'incertitude liée au choix du modèle est grande et BMA nous permet de prendre en compte directement cette incertitude dans notre inférence. Nous sélectionnons aléatoirement deux observations pour former l'ensemble des données test et considérons les 14 autres observations comme l'ensemble des données de construction. Nous appliquons par la suite les trois méthodes de sélection sur l'ensemble des données test. Nous rappelons que le paramètre *nbest* spécifie le nombre de meilleurs modèles de chaque dimension à sélectionner dans la méthode LB et que les paramètres *inf* et *sup* sont les bornes inférieure et supérieure de la fenêtre d'Occam. Rappelons également que la combinaison LB-OW est la méthode qui utilise l'algorithme "Leaps and Bounds" pour faire une sélection initiale des modèles suivie par un tri supplémentaire effectué par l'algorithme Occam's Windows. Les résultats sont présentés dans le Tableau 5.3.

Méthode	Paramètres	EQM	Nb modèles	Temps
LB	nbest = 05	0.215	26	0.020s
LB	nbest = 10	0.215	43	0.030s
LB	nbest = 15	0.215	58	0.020s
LB-OW	nbest = 05 ; inf = 0 ; sup = 6	0.210	1	0.060s
LB-OW	nbest = 10 ; inf = 0 ; sup = 6	0.210	1	0.060s
LB-OW	nbest = 15 ; inf = 0 ; sup = 6	0.210	1	0.070s
OW	inf = 2 ; sup = 6	0.202	2	0.008s
OW	inf = 0 ; sup = 6	0.188	3	0.008s
OW	inf = 0 ; sup = 10	0.188	3	0.008s

Tableau 5.3: Résultats de la sélection de modèles pour les données Longley pour nbest = 5, 10, 15 et les bornes (2 ; 6), (0 : 6) et (0 ; 10)

En analysant le Tableau 5.3, nous constatons, pour ce jeu de données et la partition aléatoire des données considérée, que les deux premières méthodes ont obtenu des EQM égales (à l'intérieur de chacune d'entre elles), indépendamment des paramètres utilisés. Par contre, la méthode OW est plus sensible au changement des bornes de la fenêtre d'Occam. Nous voyons aussi que les pouvoirs prédictifs des trois méthodes sont proches.

Nous présentons dans les Tableaux 5.4 - 5.7 les modèles sélectionnés par chacune des trois méthodes. Les modèles sont spécifiés à l'aide de variables indicatrices binaires indiquant la présence (1) ou l'absence (0) des covariables.

Int	EI	PNB	SE	AF	Pop	Année	BIC	Taille	Pr. Rel.
1	0	1	1	1	0	1	-66.05	5	0.549
1	0	1	1	1	1	1	-63.97	6	0.194
1	1	1	1	1	0	1	-63.41	6	0.147
1	1	1	1	1	1	1	-61.64	7	0.061
1	0	0	1	1	1	1	-59.80	5	0.024
1	0	0	1	1	0	1	-58.21	4	0.011
1	1	0	1	1	1	1	-58.03	6	0.010
1	1	0	1	1	0	1	-55.67	5	0.003
1	0	1	0	1	1	0	-49.55	4	< 0.001
1	0	1	1	1	1	0	-49.20	5	< 0.001

Tableau 5.4: Meilleurs modèles sélectionnés par la méthode LB pour nbest = 5, 10 et 15. Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Int	EI	PNB	SE	AF	Pop	Année	BIC	Taille	Pr. Rel.
1	0	1	1	1	0	1	-68.69	5	1

Tableau 5.5: Meilleurs modèles sélectionnés par la méthode LB-OW pour nbest = 5, 10, 15 et les bornes (0 ; 6). Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Int	EI	PNB	SE	AF	Pop	Année	BIC	Taille	Pr. Rel.
1	0	1	1	1	0	1	-30.36	5	0.980
1	0	0	1	1	0	1	-22.53	4	0.020

Tableau 5.6: Meilleurs modèles sélectionnés par la méthode OW pour les bornes (2 ; 6). Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Int	EI	PNB	SE	AF	Pop	Année	BIC	Taille	Pr. Rel.
1	0	1	1	1	0	1	-30.36	5	0.940
1	0	0	1	1	1	1	-24.12	5	0.041
1	0	0	1	1	0	1	-22.53	4	0.019

Tableau 5.7: Meilleurs modèles sélectionnés par la méthode OW pour les bornes (0 ; 6) et (0 ; 10). Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Pour ce jeu de données, la méthode OW avec fenêtres de 0 à 6 et 0 à 10 possède le meilleur pouvoir prédictif, en faisant la moyenne de 3 modèles. La méthode OW performe mieux que la méthode LB-OW parce que cette dernière a manqué deux modèles importants, (1001111) et (1001101). Les deux modèles ont été trouvés par la méthode LB mais ils ont été écartés dans l'étape OW de la méthode LB-OW. L'explication du fait que les deux modèles ont été retenus par OW et écartés par LB-OW est la façon différente des deux méthodes de faire la sélection dans l'étape OW. Contrairement à la méthode OW, qui compare les modèles deux par deux, la méthode LB-OW les compare toujours avec le modèle qui a le BIC minimum parmi les modèles sélectionnés dans l'étape LB.

Nous voyons que choisir un nombre trop grand ou trop petit de modèles a un faible impact négatif sur le pouvoir de prédiction dans ce cas-ci. C'est plutôt la qualité des modèles choisis (leur BIC et leur probabilité relative) qui compte pour obtenir un bon pouvoir prédictif. Pour exemplifier, la méthode LB a trouvé un grand nombre de modèles, mais son pouvoir de prédiction n'est pas meilleur que la méthode OW. La raison de ce comportement est expliquée en détail dans la démarche qui suit.

Un grand nombre de modèles sélectionnés n'implique pas nécessairement que tous les modèles soient importants, particulièrement pour LB qui retourne les meilleurs modèles pour chacune des dimensions séparément. Pour ce faire, nous retenons les meilleurs modèles afin de représenter environ 95% des probabilités totales attribuées aux différents modèles. Après avoir sélectionné les meilleurs modèles, nous recalculons leurs probabilités relatives et leur pouvoir de prédiction. Les résultats sont présentés dans les Tableaux 5.8, 5.9, 5.10 et 5.11.

Int	EI	PNB	SE	AF	Pop	Année	BIC	Taille	Pr. Rel.	Pr. Rel. Rec.
1	0	1	1	1	0	1	-66.05	5	0.549	0.578
1	0	1	1	1	1	1	-63.97	6	0.194	0.204
1	1	1	1	1	0	1	-63.41	6	0.147	0.154
1	1	1	1	1	1	1	-61.64	7	0.061	0.064

Tableau 5.8: Meilleurs modèles sélectionnés (correspondant à 95% des probabilités relatives initiales) par la méthode LB pour nbest = 5, 10 et 15. Pr. Rel. et Pr. Rel. Rec. sont respectivement les probabilités relatives initiales et recalculées. Une valeur 1 indique que la covariable a été sélectionnée.

Int	EI	PNB	SE	AF	Pop	Année	BIC	Taille	Pr. Rel.	Pr. Rel. Rec.
1	0	1	1	1	0	1	-68.69	5	1	1

Tableau 5.9: Meilleurs modèles sélectionnés (correspondant à 95% des probabilités relatives initiales) par la méthode LB-OW pour nbest = 5, 10, 15 et les bornes (0 ; 6). Pr. Rel. et Pr. Rel. Rec. sont respectivement les probabilités relatives initiales et recalculées. Une valeur 1 indique que la covariable a été sélectionnée.

Int	EI	PNB	SE	AF	Pop	Année	BIC	Taille	Pr. Rel.	Pr. Rel. Rec.
1	0	1	1	1	0	1	-30.36	5	0.980	1

Tableau 5.10: Meilleurs modèles sélectionnés (correspondant à 95% des probabilités relatives initiales) par la méthode OW pour les bornes (2 ; 6). Pr. Rel. et Pr. Rel. Rec. sont respectivement les probabilités relatives initiales et recalculées. Une valeur 1 indique que la covariable a été sélectionnée.

Int	EI	PNB	SE	AF	Pop	Année	BIC	Taille	Pr. Rel.	Pr. Rel. Rec.
1	0	1	1	1	0	1	-30.36	5	0.940	0.958
1	0	0	1	1	1	1	-24.12	5	0.041	0.042

Tableau 5.11: Meilleurs modèles sélectionnés (correspondant à 95% des probabilités relatives initiales) par la méthode OW pour les bornes (0 ; 6) et (0 ; 10). Pr. Rel. et Pr. Rel. Rec. sont respectivement les probabilités relatives initiales et recalculées. Une valeur 1 indique que la covariable a été sélectionnée.

Les méthodes LB et LB-OW ont retenu les mêmes 4, respectivement 1 modèle(s), indépendamment de la valeur du paramètre nbest. Dans le cas de LB, même si le nombre initial de modèles trouvés était grand, seulement 4 d'entre eux avaient des grandes probabilités, les autres étant peu probables. La méthode OW a retenu 1 modèle (le

même que LB-OW) pour les bornes (2;6) et 2 modèles pour les bornes (0;6) et (0;10).

Après avoir retenu les meilleurs modèles, nous recalculons les pouvoirs prédictifs des trois méthodes, et nous obtenons les résultats du Tableau 5.12.

Méthode	Paramètres	EQM	Nb modèles
LB	nbest = 05	0.235	4
LB	nbest = 10	0.235	4
LB	nbest = 15	0.235	4
LB-OW	nbest = 05 ; inf = 0 ; sup = 6	0.210	1
LB-OW	nbest = 10 ; inf = 0 ; sup = 6	0.210	1
LB-OW	nbest = 15 ; inf = 0 ; sup = 6	0.210	1
OW	inf = 2 ; sup = 6	0.210	1
OW	inf = 0 ; sup = 6	0.195	2
OW	inf = 0 ; sup = 10	0.195	2

Tableau 5.12: EQM basées sur les meilleurs modèles seulement pour les données Longley pour nbest = 5, 10, 15 et les bornes (2 ; 6), (0 ; 6) et (0 ; 10)

Retenir les meilleurs modèles correspondant à 95% des probabilités relatives n'a pas aidé au pouvoir de prédiction des trois méthodes. En effet, les EQM obtenues sont plus grandes, mais elles ne sont pas très différentes de celles obtenues précédemment. Toutefois, il est nécessaire de souligner l'effet de l'étape OW dans LB, qui élimine les modèles que nous pouvons considérer comme peu importants pour l'estimation.

Jusqu'à maintenant, nous avons fait l'analyse des données Longley pour une seule partition des données en jeu test et construction. Pour assurer que les résultats ne sont pas attribuables à une partition particulière, il est nécessaire de considérer plusieurs partitions et de voir en moyenne quelle méthode performe le mieux. Nous générons des partitions de données en divisant de façon aléatoire cinq fois le jeu de départ en 15 observations de construction et 1 observation de test. Nous continuons l'analyse de ces partitions de données obtenues avec une valeur du paramètre nbest = 15 et les bornes de la fenêtre d'Occam de (0 ; 10). Les résultats sont présentés dans le Tableau 5.13.

Partition	Méthode	EQM	Nb Modèles	Temps
1	LB	0.436	58	0.010s
	LB-OW	0.381	3	0.060s
	OW	0.381	3	0.008s
2	LB	< 0.001	58	0.030s
	LB-OW	< 0.001	3	0.060s
	OW	< 0.001	3	0.003s
3	LB	0.178	58	0.030s
	LB-OW	0.180	3	0.050s
	OW	0.180	3	0.007s
4	LB	0.252	58	0.030s
	LB-OW	0.232	3	0.060s
	OW	0.232	3	0.007s
5	LB	0.054	58	0.020s
	LB-OW	0.050	3	0.060s
	OW	0.050	3	0.008s
Moyenne	LB	0.184	58	0.024s
	LB-OW	0.169	3	0.058s
	OW	0.169	3	0.007s

Tableau 5.13: Résultats pour les données Longley pour cinq partitions différentes du jeu original (15 observations pour la construction et 1 observation pour le test) pour $nbest = 15$ et les bornes $(0 ; 10)$

En moyenne, les méthodes LB-OW et OW ont le même pouvoir prédictif, suivies de près par la méthode LB, bien qu'il n'y ait pas de grandes différences entre les trois méthodes pour ce jeu de données. En ce qui concerne la vitesse d'exécution, la méthode OW est la plus rapide, suivie par les méthodes LB et LB-OW. Nous voyons dans le tableau 5.14 que, après la rétention des meilleurs modèles, les pouvoirs de prédiction des trois méthodes sont plus petits, mais que l'ordre de performance n'est pas changé.

Partition	Méthode	EQM	Nb modèles
1	LB	0.466	4
	LB-OW	0.414	1
	OW	0.414	1
2	LB	< 0.001	7
	LB-OW	< 0.001	3
	OW	< 0.001	3

3	LB	0.174	7
	LB-OW	0.180	3
	OW	0.180	3
4	LB	0.259	6
	LB-OW	0.241	2
	OW	0.241	2
5	LB	0.054	6
	LB-OW	0.050	3
	OW	0.050	3
Moyenne	LB	0.191	6
	LB-OW	0.177	2.4
	OW	0.177	2.4

Tableau 5.14: Résultats pour les données Longley (basées sur les meilleurs modèles seulement) pour cinq partitions différentes du jeu original (15 observations pour la construction et 1 observation pour le test) pour nbest = 15 et les bornes (0 ; 10)

5.2.2 Données générées

Nous voulons maintenant étudier le comportement des trois méthodes à l'aide d'un jeu de données de taille modérée avec plusieurs paramètres. À cette fin, nous simulons avec le générateur de nombres aléatoires du langage R un vecteur de 15 coefficients, $\beta = (\beta_1, \beta_2, \ldots, \beta_{15})$, une matrice $X = (x_{ij})$ de dimension 200×15 $(i = 1, \ldots, 200, j = 1, \ldots, 15)$ et un vecteur d'erreurs normales $e = (e_1, e_2, \ldots, e_{200})$. La variable réponse Y est calculée avec l'équation du modèle, $Y = \beta X + e$ (voir l'Annexe C.2 pour plus de détails sur la génération du jeu de données).

Les résultats de la régression linéaire pour le jeu de données généré sont présentés dans le Tableau 5.15.

Covariable	Estimation	Err type	Valeur-p
(Intercept)	-0.032	0.067	0.636
X_1	-0.411	0.012	< 0.001
X_2	0.568	0.012	< 0.001
X_3	-0.183	0.012	< 0.001
X_4	0.776	0.012	< 0.001
X_5	0.883	0.012	< 0.001

X_6	-0.061	0.012	< 0.001
X_7	0.005	0.012	0.657
X_8	0.033	0.012	0.006
X_9	-0.026	0.012	0.024
X_{10}	-0.007	0.012	0.572
X_{11}	0.068	0.012	< 0.001
X_{12}	-0.011	0.012	0.358
X_{13}	0.028	0.013	0.032
X_{14}	0.012	0.012	0.307
X_{15}	-0.030	0.012	0.012

Tableau 5.15: Résultats de la régression linéaire pour le jeu de données généré

Nous observons dans le tableau 5.15 que les coefficients $\beta_0, \beta_7, \beta_{10}, \beta_{12}$ et β_{14} ne sont pas significatifs à un seuil $\alpha = 0.05$. La conclusion est que le modèle est suprasaturé, donc nous pouvons réduire la dimensionnalité du modèle sans perte significative.

Même si le modèle a été construit en respectant toutes les suppositions de la régression linéaire, nous pouvons vérifier que ces suppositions sont respectées (voir Figure 5.2).

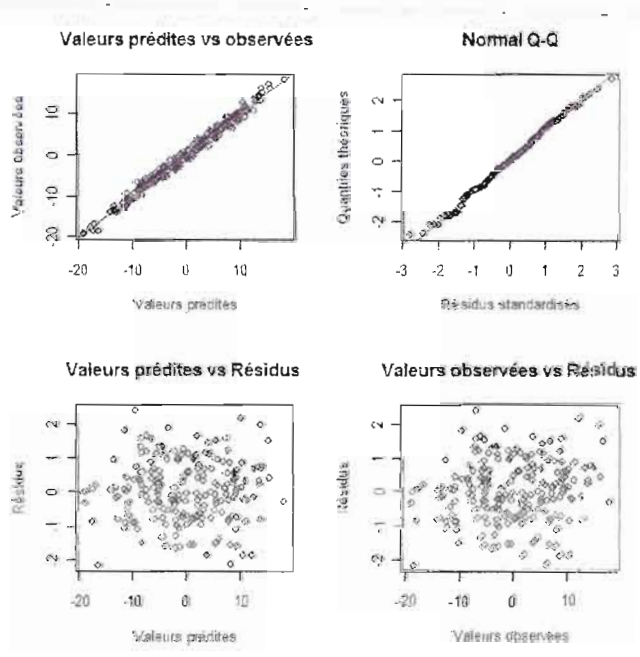

Figure 5.2: Évaluation des suppositions du modèle de régression linéaire pour le jeu de données générées

Le nombre de modèles possibles de chaque dimension pour ce jeu de données généré est présenté dans le Tableau 5.16.

Nb covariables	1	2	3	4	5	6	7	8
Nb modèles	15	105	455	1365	3003	5005	6435	6435
Nb covariables	9	10	11	12	13	14	15	Total
Nb modèles	5005	3003	1365	455	105	15	1	32767

Tableau 5.16: Nombre de modèles de régression linéaire possibles de chaque dimension pour le jeu de données généré

Nous sélectionnons aléatoirement 20 observations pour former l'ensemble des données test et considérons les 180 autres observations comme l'ensemble des données de construction. Nous appliquons les trois méthodes de sélection sur l'ensemble des données construction et examinons le pouvoir prédictif des ensembles de modèles sélectionnés avec les données test. Les résultats sont présentés dans le Tableau 5.17.

Méthode	Paramètres	EQM	Nb modèles	Temps
LB	nbest = 05	1.147	71	0.030s
LB	nbest = 15	1.146	211	0.090s
LB	nbest = 30	1.146	391	0.220s
LB-OW	nbest = 05 ; inf = 0 ; sup = 6	1.227	2	0.070s
LB-OW	nbest = 15 ; inf = 0 ; sup = 6	1.227	2	0.160s
LB-OW	nbest = 30 ; inf = 0 ; sup = 6	1.227	2	0.260s
OW	inf = 2 ; sup = 6	1.227	2	0.829s
OW	inf = 0 ; sup = 6	1.221	4	0.808s
OW	inf = 0 ; sup = 10	1.221	4	0.775s

Tableau 5.17: Résultats de la sélection de modèles de régression linéaire pour les données générées pour nbest = 5, 15, 30 et les bornes (2 ; 6), (0 ; 6) et (0 ; 10)

En analysant le Tableau 5.17, nous constatons aussi pour ce jeu de données qu'il n'y a pas de grandes différences entre les EQM des trois méthodes. Nous observons par ailleurs que faire la moyenne sur 200 ou 400 modèles donne une EQM proche de celle obtenue en faisant la moyenne sur 2 ou 4 modèles. L'ordre des trois méthodes en ce qui concerne leur pouvoir de prédiction est LB suivie par OW et finalement par LB-OW. La méthode OW est dans ce cas la plus lente et la méthode LB la plus rapide.

Les modèles sélectionnés par la méthode LB sont trop nombreux pour les énumérer ici. Nous présentons les modèles sélectionnés par les méthodes LB-OW et OW dans les Tableaux 5.18, 5.19 et 5.20. Nous remarquons que les variables non significatives n'ont pas été sélectionnées et que le même meilleur modèle a été trouvé par les méthodes LB-OW et OW indépendamment des paramètres utilisés.

Int	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	BIC	Taille	Pr. Rel.
1	1	1	1	1	1	1	0	1	0	0	1	0	0	0	0	-724.65	9	0.940
1	1	1	1	1	1	1	0	0	0	0	1	0	1	0	0	-719.15	9	0.060

Tableau 5.18: Meilleurs modèles sélectionnés par la méthode LB-OW pour nbest = 5, 10, 15 et les bornes (0 ; 6). Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Int	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	BIC	Taille	Pr. Rel.
1	1	1	1	1	1	1	0	1	0	0	1	0	0	0	0	18.04	9	0.970
1	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	24.72	8	0.030

Tableau 5.19: Meilleurs modèles sélectionnés par la méthode OW pour les bornes (2 ; 6). Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Int	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	BIC	Taille	Pr. Rel.
1	1	1	1	1	1	1	0	1	0	0	1	0	0	0	0	18.04	9	0.880
1	1	1	1	1	1	1	0	0	0	0	1	0	1	0	0	23.60	9	0.050
1	1	1	1	1	1	1	0	0	0	0	1	0	0	0	1	24.41	9	0.040
1	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	24.72	8	0.030

Tableau 5.20: Meilleurs modèles sélectionnés par la méthode OW pour les bornes (0 ; 6) et (0 ; 10). Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Dans ce cas également, nous retenons les meilleurs modèles (correspondant à 95% des probabilités relatives) et nous constatons que les performances des méthodes n'ont pas beaucoup changé (voir Tableau 5.21).

Méthode	Paramètres	EQM	Nb modèles
LB	nbest = 05	1.152	15
LB	nbest = 15	1.149	38
LB	nbest = 30	1.146	47
LB-OW	nbest = 05 ; inf = 0 ; sup = 6	1.227	2
LB-OW	nbest = 15 ; inf = 0 ; sup = 6	1.227	2
LB-OW	nbest = 30 ; inf = 0 ; sup = 6	1.227	2
OW	inf = 2 ; sup = 6	1.228	1
OW	inf = 0 ; sup = 6	1.221	3
OW	inf = 0 ; sup = 10	1.221	3

Tableau 5.21: EQM recalculées (basées sur les meilleurs modèles de régression linéaire seulement) pour les données générées pour nbest = 5, 15, 30 et les bornes (2 ; 6), (0 ; 6) et (0 ; 10)

Après la rétention des meilleurs modèles, le nombre de modèles sélectionnés par la méthode LB est toujours grand, ce qui est une indication qu'il y a beaucoup de modèles avec des probabilités relatives proches, donc les différences entre leurs BICs sont petites. C'est la raison pour laquelle la méthode LB a un meilleur pouvoir prédictif que les deux autres méthodes. Pour la même raison, la méthode OW est la plus lente, car elle a dû effectuer un grand nombre de comparaisons. Pour ce jeu de données, le modèle complet a 16 covariables et le meilleur modèle, avec le plus petit BIC, a seulement 9 covariables, incluant l'intercept.

Finalement, nous divisons les données générées aléatoirement en 5 parties et nous appliquons de nouveau les trois méthodes, pour une valeur de paramètre nbest = 30 et les bornes de la fenêtre d'Occam de (0 ; 10). Les résultats sont présentés dans le Tableau 5.22.

Partition	Méthode	EQM	Nb modèles	Temps
1	LB	0.973	391	0.190s
	LB-OW	1.006	6	0.340s
	OW	1.006	6	0.817s
2	LB	1.329	391	0.180s
	LB-OW	1.371	3	0.280s
	OW	1.371	3	0.765s

3	LB	1.170	391	0.190s
	LB-OW	1.173	8	0.320s
	OW	1.173	8	0.748s
4	LB	0.836	391	0.190s
	LB-OW	0.844	6	0.320s
	OW	0.855	7	0.836s
5	LB	0.656	391	0.200s
	LB-OW	0.662	10	0.360s
	OW	0.662	9	0.811s
Moyenne	LB	0.993	391	0.190s
	LB-OW	1.011	6.6	0.324s
	OW	1.013	6.6	0.795s

Tableau 5.22: Résultats pour les données générées dans le cas de la régression linéaire pour cinq partitions différentes du jeu original (160 observations pour la construction et 40 observations pour le test) pour nbest = 15 et les bornes (0 ; 10)

Quand nous divisons les données en cinq parties, les résultats obtenus sont comparables à ceux obtenus pour une seule partition. Nous voyons dans le Tableau 5.22 qu'en moyenne les méthodes ont des résultats similaires. La méthode OW est toujours la plus lente et son pouvoir prédictif est le moins grand, mais les différences entre les pouvoirs prédictifs des trois méthodes sont presque inexistantes. Après la rétention des 95% meilleurs modèles, la situation ne change pas.

Méthode	EQM	Err type	Nb modèles
LB	0.994	0.266	58.8
LB-OW	1.014	0.278	4.6
OW	1.016	0.278	4.8

Tableau 5.23: EQM moyennes (basées sur les meilleurs modèles de régression linéaire seulement) pour les données générées pour cinq partitions différentes du jeu original (160 observations pour la construction et 40 observations pour le test) pour nbest = 15 et les bornes (0 ; 10)

5.3 Comparaison des méthodes dans le cas de la régression logistique

5.3.1 Données Mélanome

Le jeu de données Mélanome modifié (voir l'Annexe C.3) contient 6 variables et 191 observations. Nous voulons modéliser la réponse Statut, une variable dichotomique, en fonction des covariables Sexe, Age, Année, Grandeur et Ulcère à l'aide d'un modèle de régression logistique. Les résultats de la régression logistique pour le jeu de données Mélanome sont présentés dans le Tableau 5.24.

Covariable	Estimation	Err type	Valeur-p
(Intercept)	-460.624	150.469	0.002
Sexe	-0.503	0.369	0.173
Age	-0.022	0.012	0.063
Année	0.236	0.077	0.002
Grandeur	-0.114	0.067	0.088
Ulcère	-1.446	0.394	< 0.001

Tableau 5.24: Résultats de la régression logistique pour le jeu de données Mélanome

Nous pouvons visualiser graphiquement le modèle de régression logistique complet pour les données Mélanome dans la Figure 5.3 qui représente graphiquement les probabilités de chaque observation (valeurs prédites) par rapport aux valeurs de la fonction logit arrangées en ordre croissant (prédicteurs linéaires). Les points représentent les valeurs de la variable réponse. Nous observons qu'il y a plus de points correspondant à la valeur 0 à la gauche de la courbe, où les probabilités sont entre 0 et 0.5.

Pour ce jeu de données, les modèles possibles de chaque dimension sont présentés dans le Tableau 5.25.

Nb covariables	1	2	3	4	5	Total
Nb modèles	5	10	10	5	1	31

Tableau 5.25: Nombre de modèles possibles de chaque dimension pour le jeu de données Mélanome

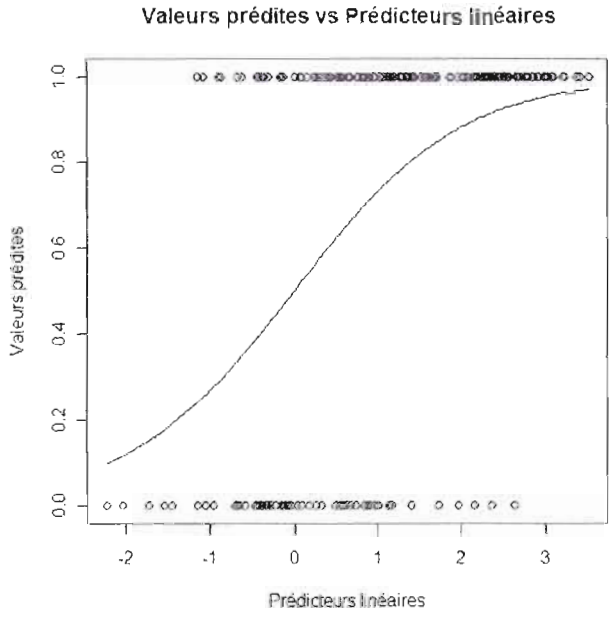

Figure 5.3: *Représentation graphique du modèle de régression logistique pour les données Mélanome*

Comme dans le cas de la régression linéaire, nous commençons l'analyse avec les données divisées en deux ensembles, 176 observations pour la construction du modèle et 15 observations pour l'évaluation de son pouvoir prédictif. Le Tableau 5.26 présente les performances des trois méthodes pour la régression logistique.

Méthode	Paramètres	TMC	Nb modèles	Temps
LB	nbest = 05	0.2	22	0.360s
LB	nbest = 10	0.2	32	0.480s
LB	nbest = 15	0.2	32	0.500s
LB-OW	nbest = 05 ; inf = 0 ; sup = 10	0.2	4	0.280s
LB-OW	nbest = 10 ; inf = 0 ; sup = 10	0.2	4	0.310s
LB-OW	nbest = 15 ; inf = 0 ; sup = 10	0.2	4	0.310s
OW	inf = 1 ; sup = 6	0.2	2	0.100s
OW	inf = 0 ; sup = 6	0.2	4	0.160s
OW	inf = 0 ; sup = 10	0.2	4	0.140s

Tableau 5.26: Résultats de la sélection de modèles pour les données Mélanome pour nbest = 5, 10, 15 et les bornes (1 ; 6), (0 ; 6) et (0 ; 10)

Les taux de mauvaise classification sont égaux pour toutes les méthodes, indépendamment des paramètres utilisés. La méthode OW est dans ce cas la plus rapide. Il est surprenant que la méthode LB-OW soit plus rapide que LB. L'explication de ce comportement est une particularité d'implantation de la fonction bic.glm qui utilise un algorithme Leaps and Bounds modifié. Avant l'étape OW, cet algorithme modifié écarte lui-même les pires modèles en fonction du paramètre OR. Nous avons utilisé une valeur différente de ce paramètre pour les deux méthodes. Pour la méthode LB, la valeur du paramètre OR choisie est grande ($OR = 10^6$) pour s'assurer que tous les nbest modèles sont sélectionnés. Dans le cas de la méthode LB-OW les valeurs des paramètres $OR = 20$ et $OR.fix = 1.7$ ont été choisies pour obtenir la valeur 10 pour la borne supérieure de l'étape OW. Donc, après l'étape LB, la méthode LB sélectionne plus de modèles que la méthode LB-OW et a, par conséquent, un temps d'exécution plus grand. Notons de plus que la fonction bic.glm trouve aussi le modèle nul, incluant l'intercept seulement, ainsi elle a trouvé 32 modèles pour les valeurs du paramètre $nbest = 10$ et $nbest = 15$, même si le nombre total de modèles présenté dans le Tableau 5.25 est 31.

Dans le cas de la régression logistique, il est également intéressant de voir si le fait de prendre seulement les meilleurs modèles avec la somme des probabilités relatives totalisant 95% donne des résultats différents en ce qui concerne les pouvoirs de prédiction des trois méthodes. En regardant le Tableau 5.27, nous constatons que les taux de mauvaise classification ne changent pas. En effet, le nombre de modèles sélectionnés par chaque méthode en utilisant différents paramètres est le même, sauf pour la méthode OW.

Méthode	Paramètres	TMC	Nb modèles
LB	nbest = 05	0.2	11
LB	nbest = 10	0.2	11
LB	nbest = 15	0.2	11
LB-OW	nbest = 05 ; inf = 0 ; sup = 10	0.2	4
LB-OW	nbest = 10 ; inf = 0 ; sup = 10	0.2	4
LB-OW	nbest = 15 ; inf = 0 ; sup = 10	0.2	4

OW	inf = 1 ; sup = 6	0.2	2
OW	inf = 0 ; sup = 6	0.2	4
OW	inf = 0 ; sup = 10	0.2	4

Tableau 5.27: TMC recalculés (basés sur les meilleurs modèles seulement) pour les données Mélanome pour nbest = 5, 10, 15 et les bornes (1 ; 6), (0 ; 6) et (0 ; 10)

Dans les Tableaux 5.28, 5.29, 5.30 et 5.31, nous présentons les modèles sélectionnés par chaque méthode.

Int	Sexe	Age	Année	Grandeur	Ulcère	BIC	Taille	Pr. Rel.
1	0	0	1	1	1	-713.78	4	0.228
1	0	0	0	1	1	-713.54	3	0.202
1	0	0	1	0	1	-713.12	3	0.164
1	0	0	0	0	1	-712.24	2	0.106
1	0	1	1	0	1	-712.00	4	0.093
1	0	1	1	1	1	-710.93	5	0.055
1	1	0	1	0	1	-709.48	4	0.027
1	1	0	1	1	1	-709.43	5	0.026
1	0	1	0	1	1	-709.07	4	0.022
1	1	0	0	1	1	-708.90	4	0.020
1	0	1	0	0	1	-708.88	3	0.020

Tableau 5.28: Meilleurs 11 modèles sélectionnés par la méthode LB pour nbest = 5, 10 et 15. Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Int	Sexe	Age	Année	Grandeur	Ulcère	BIC	Taille	Pr. Rel.
1	0	0	1	1	1	-713.78	4	0.326
1	0	0	0	1	1	-713.54	3	0.288
1	0	0	1	0	1	-713.12	3	0.234
1	0	0	0	0	1	-712.24	2	0.151

Tableau 5.29: Meilleurs modèles sélectionnés par la méthode LB-OW pour nbest = 5, 10, 15 et les bornes (0 ; 10). Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Int	Sexe	Age	Année	Grandeur	Ulcère	BIC	Taille	Pr. Rel.
1	0	0	0	1	1	196.47	3	0.660
1	0	0	0	0	1	197.76	2	0.340

Tableau 5.30: Meilleurs modèles sélectionnés par la méthode OW pour les bornes (1 ; 6). Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Int	Sexe	Age	Année	Grandeur	Ulcère	BIC	Taille	Pr. Rel.
1	0	0	1	1	1	196.22	4	0.326
1	0	0	0	1	1	196.47	3	0.288
1	0	0	1	0	1	196.88	3	0.234
1	0	0	0	0	1	197.76	2	0.151

Tableau 5.31: Meilleurs modèles sélectionnés par la méthode OW pour les bornes (0 ; 6) et (0 ; 10). Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

En conclusion, pour ce jeu de données et cette partition, les méthodes ont le même pouvoir de prédiction. Nous remarquons que les méthodes LB avec les paramètres nbest = 5, 10 et 15 et OW avec les bornes (0 ; 6) et (0 ; 10) ont sélectionné les mêmes 4 modèles.

Les résultats de l'analyse pour le jeu de données Mélanome divisé en 5 parties sont présentés dans les Tableaux 5.32 et 5.33. Nous voyons que les pouvoirs de prédiction des trois méthodes sont proches, avant et après la rétention de 95% des modèles. Les taux de mauvaise classification sont d'environ 30%, ce qui est une indication que nous omettons des covariables importantes pour la capacité de prédiction. La méthode OW est toujours la plus rapide, suivie par LB-OW et LB.

Partition	Méthode	TMC	Nb Modèles	Temps
1	LB	0.179	32	0.480
	LB-OW	0.179	5	0.360
	OW	0.179	5	0.188
2	LB	0.289	32	0.480
	LB-OW	0.289	3	0.340
	OW	0.289	4	0.188

3	LB	0.368	32	0.480
	LB-OW	0.395	1	0.280
	OW	0.395	6	0.171
4	LB	0.211	32	0.470
	LB-OW	0.211	3	0.310
	OW	0.211	4	0.177
5	LB	0.263	32	0.480
	LB-OW	0.289	2	0.390
	OW	0.289	4	0.177
Moyenne	LB	0.262	32	0.478
	LB-OW	0.283	2.8	0.336
	OW	0.278	4.6	0.180

Tableau 5.32: Résultats pour les données Mélanome pour cinq partitions différentes du jeu original (152 observations pour la construction et 39 observations pour le test) pour nbest = 15 et les bornes (0 ; 10)

Partition	Méthode	TMC	Nb modèles
1	LB	0.179	13
	LB-OW	0.179	5
	OW	0.179	5
2	LB	0.263	12
	LB-OW	0.342	3
	OW	0.342	3
3	LB	0.368	8
	LB-OW	0.395	1
	OW	0.395	2
4	LB	0.211	11
	LB-OW	0.211	3
	OW	0.211	3
5	LB	0.289	13
	LB-OW	0.289	2
	OW	0.289	2
Moyenne	LB	0.262	11.4
	LB-OW	0.283	2.8
	OW	0.283	3

Tableau 5.33: Résultats pour les données Mélanome (basés sur les meilleurs modèles seulement) pour cinq partitions différentes du jeu original (152 observations pour la construction et 39 observations pour le test) pour nbest = 15 et les bornes (0 ; 10)

5.3.2 Données générées

Ce jeu de données contient 15 covariables et 300 observations générées avec le générateur de nombres aléatoires du langage R (voir l'Annexe C.4). Nous employons un modèle logistique pour modéliser la variable dichotomique, Y, en fonction de covariables $X_1, \ldots, X_{15}$. Les détails de la régression logistique pour ce jeu de données sont présentés dans le Tableau 5.34 et la représentation graphique du modèle dans la Figure 5.4.

Covariable	Estimation	Err type	Valeur-p
(Intercept)	-0.438	1.328	0.741
X_1	-1.847	0.284	< 0.001
X_2	0.691	0.162	< 0.001
X_3	0.651	0.166	< 0.001
X_4	0.736	0.154	< 0.001
X_5	2.104	0.312	< 0.001
X_6	-0.035	0.575	0.952
X_7	-2.862	0.681	< 0.001
X_8	-0.975	0.592	0.099
X_9	0.818	0.541	0.131
X_{10}	0.493	0.556	0.376
X_{11}	1.618	0.595	0.007
X_{12}	0.235	0.591	0.691
X_{13}	-0.563	0.556	0.311
X_{14}	2.627	0.694	0.000
X_{15}	-1.718	0.591	0.004

Tableau 5.34: Résultats de la régression logistique pour le jeu de données généré

Nous voyons dans le Tableau 5.34 que l'intercept et les coefficients $\beta_6, \beta_8, \beta_9, \beta_{10}, \beta_{12}, \beta_{13}$ ne sont pas significatifs à un seuil $\alpha = 0.05$.

Le Tableau 5.35 présente le nombre de modèles de régression logistique possible de chaque dimension.

Nb covariables	1	2	3	4	5	6	7	8
Nb modèles	15	105	455	1365	3003	5005	6435	6435

Nb covariables	9	10	11	12	13	14	15	Total
Nb modèles	5005	3003	1365	455	105	15	1	32767

Tableau 5.35: Nombre de modèles de régression logistique possibles de chaque dimension pour le jeu de données généré

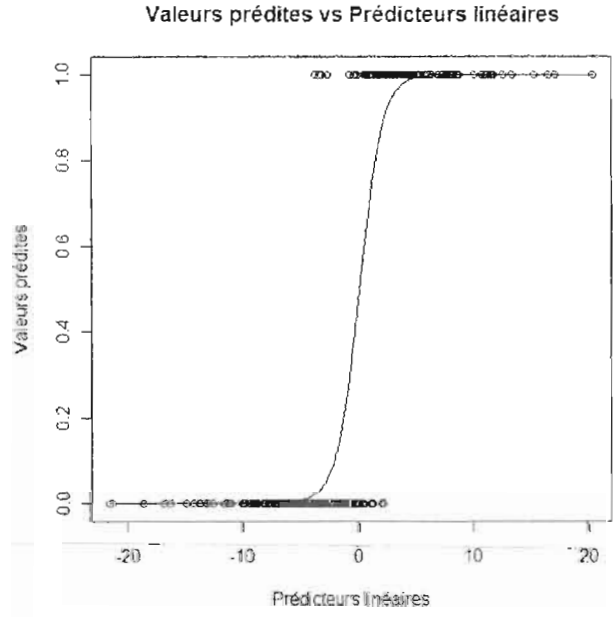

Figure 5.4: Représentation graphique du modèle de régression logistique pour les données générées

Les résultats de l'analyse pour les données divisées aléatoirement en 280 observations de construction et 20 observations de test sont présentés dans le Tableau 5.36. Nous voyons que les taux de mauvaise classification pour les trois méthodes sont égaux. La méthode LB-OW est la plus rapide, suivie par LB et par OW qui est beaucoup plus lente que les deux premières méthodes.

Méthode	Paramètres	TMC	Nb modèles	Temps
LB	nbest = 05	0.1	72	2.170s
LB	nbest = 15	0.1	212	6.010s
LB	nbest = 30	0.1	392	11.080s

LB-OW	nbest = 05 ; inf = 0 ; sup = 10	0.1	3	0.870s
LB-OW	nbest = 15 ; inf = 0 ; sup = 10	0.1	3	1.810s
LB-OW	nbest = 30 ; inf = 0 ; sup = 10	0.1	3	2.410s
OW	inf = 1 ; sup = 6	0.1	1	43.799s
OW	inf = 0 ; sup = 6	0.1	3	42.578s
OW	inf = 0 ; sup = 10	0.1	6	50.145s

Tableau 5.36: Résultats de la sélection de modèles de régression logistique pour les données générées pour nbest = 5, 15, 30 et les bornes (1 ; 6), (0 ; 6) et (0 ; 10)

Nous présentons dans ce qui suit les modèles sélectionnés par les méthodes LB-OW et OW pour diverses valeurs de leurs paramètres. La méthode LB-OW a sélectionné les mêmes 3 modèles indépendamment de la valeur du paramètre nbest (voir Tableau 5.37).

Int	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	BIC	Taille	Pr. Rel.
1	1	1	1	1	1	0	1	0	0	0	0	0	0	1	1	-1404.06	9	0.450
1	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	-1403.16	9	0.280
1	1	1	1	1	1	0	1	0	0	0	0	0	0	1	0	-1403.08	8	0.270

Tableau 5.37: Meilleurs modèles sélectionnés par la méthode LB-OW pour nbest = 5, 15, 30 et les bornes (0 ; 10). Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Par contre, la méthode OW a sélectionné un nombre différent de modèles pour des bornes différentes (voir Tableaux 5.38, 5.39 et 5.40).

Int	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	BIC	Taille	Pr. Rel.
1	1	1	1	1	1	0	1	0	0	0	0	0	0	1	0	174.66	8	1

Tableau 5.38: Meilleurs modèles sélectionnés par la méthode OW pour les bornes (1 ; 6). Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Int	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	BIC	Taille	Pr. Rel.
1	1	1	1	1	1	0	1	0	0	0	0	0	0	1	1	173.68	9	0.450
1	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	174.59	9	0.280
1	1	1	1	1	1	0	1	0	0	0	0	0	0	1	0	174.66	8	0.270

Tableau 5.39: Meilleurs modèles sélectionnés par la méthode OW pour les bornes (0 ; 6). Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Int	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	BIC	Taille	Pr. Rel.
1	1	1	1	1	1	0	1	0	0	0	0	0	0	1	1	173.68	9	0.440
1	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	174.59	9	0.280
1	1	1	1	1	1	0	1	0	0	0	0	0	0	1	0	174.66	8	0.270
1	1	1	1	1	1	0	1	1	0	1	1	0	0	0	1	191.16	11	< 0.001
1	1	1	1	1	1	0	1	1	1	1	0	0	0	0	1	194.34	11	< 0.001
1	1	1	1	1	1	0	1	0	1	1	1	1	0	0	1	196.24	12	< 0.001

Tableau 5.40: Meilleurs modèles sélectionnés par la méthode OW pour les bornes (0 ; 10). Pr. Rel. est la probabilité relative du modèle. Une valeur 1 indique que la covariable a été sélectionnée.

Nous constatons qu'en général les modèles sélectionnés sont consistants avec le modèle pour les données générées, en ce sens qu'ils ne contiennent pas les covariables non significatives pour le modèle complet. Ceci est spécialement vrai pour les modèles qui ont une grande probabilité relative.

Dans le Tableau 5.41, nous voyons que retenir les meilleurs modèles correspondant à 95% des probabilités relatives initiales n'a pas changé les pouvoirs de prédiction, mais a diminué le nombre de modèles sélectionnés par les méthodes LB et OW avec les bornes (0 ; 10).

Méthode	Paramètres	TMC	Nb modèles
LB	nbest = 05	0.1	15
LB	nbest = 15	0.1	36
LB	nbest = 30	0.1	54
LB-OW	nbest = 05 ; inf = 0 ; sup = 10	0.1	3
LB-OW	nbest = 15 ; inf = 0 ; sup = 10	0.1	3
LB-OW	nbest = 30 ; inf = 0 ; sup = 10	0.1	3
OW	inf = 1 ; sup = 6	0.1	1
OW	inf = 0 ; sup = 6	0.1	3
OW	inf = 0 ; sup = 10	0.1	3

Tableau 5.41: TMC recalculés (basés sur les meilleurs modèles de régression logistique seulement) pour les données générées pour nbest = 5, 15, 30 et les bornes (1 ; 6), (0 ; 6) et (0 ; 10)

Nous continuons l'analyse en divisant les données en 5 parties, de façon aléatoire. Comme dans le cas de la régression linéaire, chaque partie (60 observations) constitue les données de test. Les 240 observations qui restent sont utilisées comme données de construction

pour les modèles de régression logistique. Le Tableau 5.42 résume les résultats obtenus pour les 5 jeux de données.

Partition	Méthode	TMC	Nb Modèles	Temps
1	LB	0.067	392	10.170s
	LB-OW	0.133	3	1.150s
	OW	0.133	2	51.761s
2	LB	0.117	392	10.360s
	LB-OW	0.183	3	1.060s
	OW	0.083	5	55.285s
3	LB	0.083	392	10.220s
	LB-OW	0.083	3	1.480s
	OW	0.083	7	59.558s
4	LB	0.167	392	10.470s
	LB-OW	0.267	1	0.900s
	OW	0.133	5	1min 14s 672ms
5	LB	0.150	392	10.400s
	LB-OW	0.100	2	1.290s
	OW	0.150	21	1min 32s 533ms
Moyenne	LB	0.117	392	10.324s
	LB-OW	0.153	2.4	1.176s
	OW	0.117	8	1min 06s 762ms

Tableau 5.42: Résultats pour les données générées dans le cas de la régression logistique pour cinq partitions différentes du jeu original (240 observations pour la construction et 60 observations pour le test) pour nbest = 30 et les bornes (0 ; 10)

Les trois méthodes ont des taux de mauvaise classification proches, avant et après la rétention de 95% des meilleurs modèles (voir Tableaux 5.42 et 5.43). La méthode OW est beaucoup plus lente que les deux autres. Dans ce cas, la méthode la plus rapide est LB-OW suivie par LB.

Méthode	TMC	Err type	Nb modèles
LB	0.117	0.043	55
LB-OW	0.157	0.076	2.2
OW	0.113	0.030	2.6

Tableau 5.43: TMC moyens (basés sur les meilleurs modèles de régression logistique seulement) pour les données générées pour cinq partitions différentes du jeu original (240 observations pour la construction et 60 observations pour le test) pour nbest = 30 et les bornes (0 ; 10)

5.4 Discussion

Nous avons présenté dans le Chapitre 5 les méthodes d'évaluation et l'analyse des performances des trois méthodes LB, LB-OW et OW dans la sélection des modèles de régression linéaire et logistique. Pour faire cette analyse, nous avons employé des fonctions R (qui incluent des routines Fortran) et des fonctions écrites spécialement dans ce but en langage C. Ces fonctions ont des paramètres différents. Il y a aussi des différences significatives dans leurs approches. Il n'est pas possible de tirer une conclusion définitive sur les performances des trois méthodes avant de parler de ces différences. Le Tableau 5.44 récapitule les différentes fonctions appelées pour l'analyse de chaque méthode.

Méthode	Régression	Fonction	Paramètres d'intérêt	Langage	Paquetage
LB	linéaire	regsubsets	nbest	R	Leaps
LB-OW	linéaire	bicreg	nbest	R	BMA
			strict = TRUE	R	BMA
			OR	R	BMA
OW	linéaire	occamswin_linreg	Inf, Sup	C	-
LB	logistique	bic.glm	nbest	R	BMA
			strict = FALSE	R	BMA
			occam.window = FALSE	R	BMA
			OR, OR.fix	R	BMA
LB-OW	logistique	bic.glm	nbest	R	BMA
			strict = TRUE	R	BMA
			occam.window = TRUE	R	BMA
			OR, OR.fix	R	BMA
OW	logistique	occamswin_logreg	Inf, Sup	C	-

Tableau 5.44: Fonctions utilisées pour l'analyse des méthodes

Dans le cas de la régression linéaire, nous ne pouvons pas spécifier directement les valeurs des paramètres Inf et Sup pour la fonction bicreg. Utiliser bicreg avec le paramètre strict = TRUE est équivalent à spécifier une valeur 0 pour la borne inférieure de la fenêtre d'Occam. La borne supérieure dépend du paramètre OR et a en effet la valeur $2\log(OR)$. La même règle s'applique pour la fonction bic.glm dans le cas de la régression logistique, sauf qu'ici la valeur de la borne supérieure de la fenêtre d'Occam est $2OR.fix\log(OR)$. Par exemple, pour obtenir la valeur 10, nous utilisons les valeurs OR.fix = 1.7 et OR =

20.

Les fonctions bicreg et bic.glm implémentent une variante modifiée de l'algorithme Leaps and Bounds et ne retournent pas tous les nbest modèles de chaque dimension, même si la valeur du paramètre strict est FALSE. Même dans ce cas, elles retournent seulement les modèles qui ont une probabilité a posteriori plus grande que $1/OR$ fois la probabilité a posteriori du meilleur modèle. Pour nous assurer qu'elles retournent tous les nbest modèles, nous devons choisir une grande valeur pour le paramètre OR pour la méthode LB. Pour la méthode LB-OW nous avons employé la fonction bic.glm avec OR = 20, pour obtenir la valeur 10 pour la borne supérieure de la fenêtre d'Occam. La méthode LB a donc davantage exploré l'espace de modèles que la méthode LB-OW. C'est la raison pour laquelle la méthode LB est plus lente que la méthode LB-OW dans le cas de la régression logistique. Normalement, nous nous attendions qu'elle soit plus rapide, parce que LB-OW contient une étape supplémentaire.

Les trois méthodes sélectionnent souvent le même modèle, mais le BIC calculé par chacune d'entre elles est différent. La raison est que les fonctions regsubsets, bicreg et bic.glm trouvent des valeurs approximatives pour les BIC des modèles. Les fonctions C utilisées par la méthode OW calculent les valeurs exactes des BIC avec la formule présentée à la Section 1.3. Même si le BIC d'un modèle a une valeur différente en fonction de la méthode qui l'a calculé, la valeur de sa probabilité relative est la même si les méthodes sélectionnent exactement les mêmes modèles.

La dernière différence entre les fonctions est que, dans le cas de la méthode LB-OW, les fonctions bicreg et bic.glm comparent toujours les BIC des modèles avec le BIC du meilleur modèle, qui est directement trouvé après l'étape LB. Les fonctions C employées pour la méthode OW comparent un modèle avec tous ses sous-modèles, deux par deux. Nous rappelons que le plus petit BIC est celui du meilleur modèle. Par conséquent, pour un nœud qui a une valeur de BIC suffisamment proche de celle de ses sous-modèles, la méthode OW sélectionne les sous-modèles, contrairement à LB-OW qui les compare avec le meilleur modèle. Il peut donc exister des situations où OW trouve des modèles

de plus petites dimensions que LB-OW.

En regardant les résultats présentés dans le Chapitre 5, nous constatons que l'ordre des méthodes, en ce qui concerne leur pouvoir de prédiction, change en fonction du jeu de données analysé. Les différences entre les erreurs quadratiques moyennes (respectivement entre les taux de mauvaise classification) sont tellement petites que nous pouvons conclure que les trois méthodes sont équivalentes. En ce qui concerne leur vitesse d'exécution, l'ordre est aussi influencé par le jeu analysé. En général, pour des modèles de grande dimension avec beaucoup des coefficients non significatifs, la méthode OW est visiblement plus lente que les deux autres.

CONCLUSION

L'objectif de ce projet était de comparer les performances des trois méthodes de sélection de modèles de régression linéaire et logistique basées sur les algorithmes "Leaps and Bounds", "Occam's Window" et une combinaison des deux. À cette fin, nous avons employé les fonctions du paquetage BMA et Leaps du progiciel et nous avons développé de nouvelles fonctions en langage C.

Pour être en mesure de comparer les trois méthodes, nous avons exploré premièrement les aspects théoriques des algorithmes employés par les trois méthodes, particulièrement "Leaps and Bounds" et "Occam's Window". Deuxièmement, il était nécessaire d'étudier le code des fonctions R qui implémentent ces algorithmes et de bien comprendre leur fonctionnement. Voir l'effet d'utiliser différentes valeurs des paramètres dans l'appel de ces fonctions sur les résultats retournés était également important.

Afin d'évaluer les performances de prédiction et les temps d'exécution de chaque méthode, nous avons développé des programmes R qui font la moyenne des modèles sélectionnés par ces fonctions et qui calculent leur temps d'exécution.

L'étape suivante était de rouler ces programmes sur différents jeux de données, réels et générés, et d'analyser les résultats. Ensuite, nous avons présenté les modèles sélectionnés par chacune des trois méthodes et leurs pouvoirs de prédiction calculés à l'aide de la technique de validation croisée.

Finalement, nous avons conclu qu'en général toutes les méthodes ont sélectionné les meilleurs modèles et que les petites différences observées dans leurs pouvoirs de prédiction sont attribuables aux modèles possédant des petites probabilités relatives. En regardant leur temps d'exécution, nous avons observé que les méthodes basées sur l'algorithme "Leaps and Bounds" sont les plus rapides.

APPENDICE A

LISTE DE FONCTIONS DU PAQUETAGE BMA

Le paquetage BMA contient les fonctions suivantes :

bicreg - fait le moyennage de modèles bayésien pour des modèles de régression linéaire ;
bic.glm - fait le moyennage de modèles bayésien pour des modèles de régression linéaire généralisée ;
bic.surv - fait le moyennage de modèles bayésien pour des modèles de survie (modèles de Cox) ;
For.MC3.REG - utilisée par la fonction **MC3.REG** pour l'implantation de chaque étape de l'algorithme de Metropolis-Hastings ;
glib - calcule les facteurs de Bayes et les probabilités a posteriori des modèles ;
iBMA - implémente la méthode du moyennage de modèles bayésien en appelant d'une façon itérative une des procédures qui fait le BMA ; seulement les variables ayant une probabilité plus grande qu'une valeur préspécifiée sont retenues après chaque appel ;
imageplot.bma - affiche le graphique des modèles sélectionnés par **bicreg**, **bic.glm** ou **bic.surv** ;
MC3.REG - effectue une sélection simultanée des variables en utilisant la méthode Monte-Carlo par chaînes de Markov ;
MC3.REG.choose - fonction utilisée par **MC3.REG** pour choisir le modèle proposé pour une étape de l'algorithme de Metropolis-Hastings ;
MC3.REG.logpost - utilisée par **MC3.REG** pour calculer les probabilités a posteriori relatives de chaque modèle ;

orderplot - affiche le graphique des variables acceptées ou rejetées, considérées dans une procédure **iBMA** ;

out.ltsreg - identifie les valeurs extrêmes pour un certain modèle ;

plot.bicreg - affiche les probabilités a posteriori pour les covariables générées par BMA ;

summary.bic - procédure pour l'affichage d'un objet généré par l'une des fonctions **bicreg**, **bic.glm** ou **bic.surv** ;

summary.iBMA - procédure pour l'affichage d'un objet généré par l'une des fonctions **iBMA** : **iBMA.bicreg**, **iBMA.glm** ou **iBMA.surv**.

APPENDICE B

L'OPÉRATEUR DE ROTATION (SWEEP)

Une méthode très efficace pour inverser une matrice et résoudre des systèmes d'équations linéaires, et donc de calculer les coefficients d'un modèle de régression linéaire, utilise l'opérateur de rotation (sweep) (R. Hocking, 2003).

Nous appliquons l'opérateur *sweep(k)* à une matrice rectangulaire A et nous obtenons une autre matrice rectangulaire B, en effectuant les opérations suivantes :

1. $b_{kk} = \frac{1}{a_{kk}}$;
2. $b_{ik} = -\frac{a_{ik}}{a_{kk}}, \forall i \neq k$;
3. $b_{kj} = \frac{a_{kj}}{a_{kk}}, \forall j \neq k$;
4. $b_{ij} = a_{ij} - \frac{a_{ik} \cdot a_{kj}}{a_{kk}}, \forall i \neq k \; et \; j \neq k.$

Nous disons que la matrice A a été "rôtie" dans le k^e élément de sa diagonale et nous écrivons $B = sweep(k)A$.

Par exemple, si la matrice A est partitionnée de la façon suivante : $A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$, où A_{11} est de dimension $r \times r$, et nous appliquons l'opérateur sweep à tous les éléments de la diagonale de A_{11}, nous obtenons la matrice B suivante :

$$B = \prod_{i=1}^{r} sweep(i)A = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix},$$

où

$$B_{11} = A_{11}^{-1}$$
$$B_{12} = A_{11}^{-1} A_{12}$$
$$B_{21} = -A_{21} A_{11}^{-1}$$
$$B_{22} = A_{22} - A_{21} A_{11}^{-1} A_{12}.$$

L'opérateur sweep a deux propriétés. Il est indépendant de l'ordre et, en l'appliquant deux fois à la même matrice, nous obtenons la matrice initiale :

$$sweep(k)sweep(r)A = sweep(r)sweep(k)A, \forall r \neq k$$

et

$$sweep(k)sweep(k)A = A.$$

La matrice inverse A^{-1} s'obtient en appliquant l'opérateur sweep à tous les éléments de la diagonale de la matrice A.

B.1 Le calcul des coefficients de régression linéaire avec l'opérateur sweep

Supposons que nous voulons trouver les coefficients pour un modèle de régression linéaire, comme vu dans la Section 1.1. Nous pouvons utiliser l'opérateur sweep pour calculer simultanément les coefficients $\hat{\beta}$ du modèle et la somme des carrés des résidus (RSS) en deux étapes :

1. Nous construisons la matrice $A = \begin{pmatrix} X'X & X'Y \\ Y'X & Y'Y \end{pmatrix}$, qui a la dimension $(p+1) \times (p+1)$;
2. La matrice $X'X$ est de dimension $p \times p$ et a un rang p. Nous pouvons appliquer l'opérateur sweep aux premiers p éléments de la diagonale de la matrice A pour obtenir une autre matrice :

$$B = \begin{pmatrix} (X'X)^{-1} & \hat{\beta} \\ -\hat{\beta}' & RSS \end{pmatrix},$$

où

$$\hat{\beta} = (X'X)^{-1}X'Y,$$

et

$$RSS = Y'(I - X(X'X)^{-1}X')Y.$$

L'opérateur sweep calcule d'une façon très rapide les coefficients d'un modèle de régression linéaire et sa somme des carrés des résidus. De plus, il évite les calculs inutiles quand nous ajoutons ou écartons des covariables. À partir d'un modèle quelconque, pour ajouter des covariables, nous appliquons l'opérateur sweep aux éléments correspondants de la diagonale de la matrice A. De la même façon, pour écarter des covariables, nous l'appliquons aux éléments correspondants de la diagonale de la matrice B.

APPENDICE C

JEUX DE DONNÉES

C.1 Longley

Le premier jeu de données s'intitule "Longley" et peut être visualisé dans R à l'aide de la commande **longley** (voir Tableau C.1). Il contient 16 observations sur 7 indicateurs macroéconomiques annuels, observés de 1947 à 1962 :

PNB : produit national brut ;

EI : l'effet de l'inflation, un indicateur économique calculé avec la formule :

$$EI = \frac{PNB\ Nominal}{PNB\ Réel} \times 100;$$

SE : nombre de personnes sans emploi ;

AF : nombre de personnes dans les forces armées ;

Population : population non institutionnalisée âgée de 14 ans ou plus ;

Année : année courante ;

Employés : Nombre de personnes employées.

N. Obs	EI	PNB	SE	AF	Population	Année	Employés
1	83.0	234.289	235.6	159.0	107.608	1947	60.323
2	88.5	259.426	232.5	145.6	108.632	1948	61.122
3	88.2	258.054	368.2	161.6	109.773	1949	60.171
4	89.5	284.599	335.1	165.0	110.929	1950	61.187
5	96.2	328.975	209.9	309.9	112.075	1951	63.221
6	98.1	346.999	193.2	359.4	113.270	1952	63.639
7	99.0	365.385	187.0	354.7	115.094	1953	64.989

8	100.0	363.112	357.8	335.0	116.219	1954	63.761
9	101.2	397.469	290.4	304.8	117.388	1955	66.019
10	104.6	419.180	282.2	285.7	118.734	1956	67.857
11	108.4	442.769	293.6	279.8	120.445	1957	68.169
12	110.8	444.546	468.1	263.7	121.950	1958	66.513
13	112.6	482.704	381.3	255.2	123.366	1959	68.655
14	114.2	502.601	393.1	251.4	125.368	1960	69.564
15	115.7	518.173	480.6	257.2	127.852	1961	69.331
16	116.9	554.894	400.7	282.7	130.081	1962	70.551

Tableau C.1: Le jeu de données Longley

Pour mieux comprendre la structure de ce jeu de données, nous présentons dans le Tableau C.2 la matrice de corrélation des covariables. Nous remarquons une très forte corrélation entre les covariables suivantes : EI et PNB, EI et Population, EI et Année, EI et Employés, PNB et Population, PNB et Année, PNB et Employés, Population et Année, Population et Employés, Année et Employés.

Corrélation	EI	PNB	SE	AF	Population	Année	Employés (Y)
EI	1.00	0.99	0.62	0.46	0.98	0.99	0.97
PNB	0.99	1.00	0.60	0.45	0.99	1.00	0.98
SE	0.62	0.60	1.00	-0.18	0.69	0.67	0.50
AF	0.46	0.45	-0.18	1.00	0.36	0.42	0.46
Population	0.98	0.99	0.69	0.36	1.00	0.99	0.96
Année	0.99	1.00	0.67	0.42	0.99	1.00	0.97
Employés	0.97	0.98	0.50	0.46	0.96	0.97	1.00

Tableau C.2: Corrélation entre les covariables du jeu de données Longley

C.2 Régression linéaire - Données générées

Le jeu de données généré utilisé dans le contexte de la régression linéaire est simulé à l'aide du générateur de nombres aléatoires de R, avec l'option *setseed(123)*. Il possède un nombre de 200 observations et 15 coefficients $\beta_1, \ldots, \beta_{15}$ et un intercept $\beta_0 = 0$. Les 5 premiers coefficients sont générés selon une loi uniforme dans l'intervalle $[-1; 1]$ et les 10 derniers selon une loi uniforme avec des valeurs comprises entre -0.5 et 0.5. Les valeurs dans la matrice X suivent indépendamment une loi uniforme avec des valeurs

dans l'intervalle $[-10; 10]$. Correspondant à chaque observation (ligne dans la matrice X), nous avons généré une erreur selon une loi $N(0, 1)$ et nous l'avons ajoutée au résultat final, Y. L'équation réelle du modèle obtenu est :

$$\begin{aligned} Y = 0 \quad - \quad & 0.424844960X_1 + 0.576610271X_2 - 0.182046156X_3 + 0.766034808X_4 + \\ + \quad & 0.880934569X_5 - 0.045444350X_6 + 0.002810549X_7 + 0.039241904X_8 + \\ + \quad & 0.005143501X_9 - 0.004338526X_{10} + 0.045683335X_{11} - 0.004666584X_{12} + \\ + \quad & 0.017757064X_{13} + 0.007263340X_{14} - 0.039707532X_{15} + \epsilon. \end{aligned}$$

Les corrélations observées entre les variables du jeu de données généré sont présentées dans le Tableaux C.3 et C.4. La plus grande corrélation se trouve entre X_5 et Y avec une valeur de *0.61*.

Corrélation	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
X_1	1.00	-0.06	0.00	0.00	0.04	-0.05	0.06	0.00
X_2	-0.06	1.00	-0.10	-0.02	0.00	0.00	0.01	0.07
X_3	0.00	-0.10	1.00	0.04	0.05	0.02	0.06	-0.08
X_4	0.00	-0.02	0.04	1.00	0.00	-0.01	-0.01	0.08
X_5	0.04	0.00	0.05	0.00	1.00	0.08	0.01	-0.06
X_6	-0.05	0.00	0.02	-0.01	0.08	1.00	-0.07	0.01
X_7	0.06	0.01	0.06	-0.01	0.01	-0.07	1.00	0.06
X_8	0.00	0.07	-0.08	0.08	-0.06	0.01	0.06	1.00
X_9	0.01	0.01	-0.04	-0.16	0.02	-0.01	-0.13	-0.01
X_{10}	-0.01	-0.03	0.06	0.14	-0.02	-0.02	0.09	-0.05
X_{11}	0.02	0.02	-0.01	0.10	-0.06	-0.01	0.10	-0.08
X_{12}	-0.07	-0.01	-0.02	-0.01	0.10	0.01	0.05	-0.09
X_{13}	-0.09	0.01	0.25	-0.07	0.11	0.05	0.01	0.13
X_{14}	0.14	-0.05	0.01	0.08	0.12	-0.09	0.07	0.10
X_{15}	0.01	0.03	-0.15	-0.01	0.03	-0.07	0.09	-0.03
Y	-0.28	0.44	-0.12	0.56	0.61	0.02	0.00	0.07

Tableau C.3: Corrélation entre les covariables du jeu de données généré

Corrélation	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	Y
X_1	0.01	-0.01	0.02	-0.07	-0.09	0.14	0.01	-0.28
X_2	0.01	-0.03	0.02	-0.01	0.01	-0.05	0.03	0.44
X_3	-0.04	0.06	-0.01	-0.02	0.25	0.01	-0.15	-0.12
X_4	-0.16	0.14	0.10	-0.01	-0.07	0.08	-0.01	0.56
X_5	0.02	-0.02	-0.06	0.10	0.11	0.12	0.03	0.61
X_6	-0.01	-0.02	-0.01	0.01	0.05	-0.09	-0.07	0.02
X_7	-0.13	0.09	0.10	0.05	0.01	0.07	0.09	0.00
X_8	-0.01	-0.05	-0.08	-0.09	0.13	0.10	-0.03	0.07
X_9	1.00	0.00	0.00	0.06	0.08	0.01	-0.02	-0.10
X_{10}	0.00	1.00	0.11	0.03	0.06	0.05	0.05	0.05
X_{11}	0.00	0.11	1.00	0.09	0.00	-0.03	0.04	0.07
X_{12}	0.06	0.03	0.09	1.00	0.13	0.08	-0.01	0.07
X_{13}	0.08	0.06	0.00	0.13	1.00	-0.07	-0.13	0.05
X_{14}	0.01	0.05	-0.03	0.08	-0.07	1.00	0.07	0.07
X_{15}	-0.02	0.05	0.04	-0.01	-0.13	0.07	1.00	0.03
Y	-0.10	0.05	0.07	0.07	0.05	0.07	0.03	1.00

Tableau C.4: Corrélation entre les covariables du jeu de données généré

C.3 Mélanome

Pour l'analyse de la sélection des modèles de régression logistique, nous employons comme premier jeu de données le jeu "Mélanome" qui se trouve dans le paquetage "MASS" du logiciel R et qui peut être initialisé avec la commande R **library(MASS)**. Le jeu peut être visualisé avec la commande R **Melanoma**. Il contient 205 observations et les 7 covariables suivantes :

Temps : le temps de survie d'un patient (en jours) ;

Statut : 1 = décès à cause du mélanome ; 2 = en vie ; 3 = décès d'une autre cause ;

Sexe : 1 = homme ; 0 = femme ;

Age : l'âge du patient ;

Année : l'année de l'opération ;

Grandeur : la grandeur de la tumeur (en mm) ;

Ulcère : 1 = présence ; 0 = absence.

Nous modifions le jeu de données original en écartant la variable Temps, qui ne présente pas d'intérêt pour le problème. Du nombre total de 205 observations, nous retenons seulement les 191 qui correspondent à une valeur de la variable Statut égale à 0 ou à 1. Nous modifions les valeurs de la variable Statut comme suit : 0 = décès à cause du mélanome et 1 = en vie. Nous employons la nouvelle variable statut comme variable réponse pour notre jeu de régression logistique qui contient maintenant les 5 covariables qui restent : Sexe, Age, Année, Grandeur et Ulcère.

C.4 Régression logistique - Données générées

Pour générer ce jeu de données, nous employons aussi le générateur de nombres aléatoires de R avec l'option *setseed(1234)*. Nous générons le vecteur $\beta = (\beta_0, \beta_1, \ldots, \beta_{15})$ de coefficients, où $\beta_0 = 0$ et $\beta_1, \ldots, \beta_{15}$ sont des variables uniformes dans l'intervalle $[-2; 2]$. La fonction logit réelle obtenue est :

$$\begin{aligned} g(X) = 0 \quad &- \quad 1.54518635X_1 + 0.48919762X_2 + 0.43709893X_3 + 0.49351777X_4 + \\ &+ \quad 1.44366153X_5 + 0.56124242X_6 - 1.96201697X_7 - 1.06979798X_8 + \\ &+ \quad 0.66433503X_9 + 0.05700457X_{10} + 0.77436517X_{11} + 0.17989934X_{12} - \\ &- \quad 0.86906567X_{13} + 1.69373394X_{14} - 0.83073664X_{15}. \end{aligned}$$

Pour la matrice des observations, X, nous générons indépendamment 1500 valeurs selon une loi $N(0, 2)$ arrangées dans 5 colonnes et 300 lignes et 3000 valeurs Bernoulli(0.7) arrangées dans 10 colonnes et 300 lignes.

Pour obtenir la valeur de la variable réponse Y, nous calculons les probabilités de succès avec la formule (1.1) et nous générons 300 valeurs selon une loi uniforme dans l'intervalle $[0; 1]$. La variable Y prend la valeur 1 si la valeur générée est plus petite que la probabilité calculée ou la valeur 0, sinon.

BIBLIOGRAPHIE

Akaike, H. « A new look at the statistical model identification », *IEEE Transactions on Automatic Control*, vol. 19, no. 6, 1974, p. 716 – 723.

Furnival, G. M. et Wilson Jr., R. W. « Regressions by Leaps and Bounds », *Technometrics*, vol. 16, no. 4, 1974, p. 499 - 511.

Hocking, R. R. « Methods and Applications of Linear Models : Regression and the Analysis of Variance », *New York : John Wiley & Sons*, 2003.

Hoeting, J. A., Madigan, D., Raftery, A. E. et Volinsky, C. T. « Bayesian Model Averaging : A Tutorial », *Statistical Science*, vol. 14, no. 4, 1999, p. 382 – 417.

Hosmer, D. W. et Lemeshow, S. « Applied Logistic Regression, Second Edition », *New York : John Wiley & Sons*, 2000.

Knuth, D. E. « The Art of Computer Programming : Fundamental Algorithms (3rd edition) », *Addison-Wesley Longman*, vol. 1, 1997.

Land, A. H. et Doig, A. G. « An automatic method of solving discrete programming problems », *Econometrica*, vol. 28, no. 3, 1960, p. 497 - 520.

Lumley T. basé sur le code Fortran de Miller A. « Leaps : regression subset selection, paquetage R version 2.9 », en ligne : http ://CRAN.R-project.org/package=leaps, 2009.

Madigan, D. et Raftery, A. E. « Model selection and accounting for model uncertainty in graphical models using Occam's Window », *Journal of the American Statistical Association*, vol. 89, no. 428, 1994, p. 1535 - 1546.

Miller, A. « Subset Selection in Regression Second Edition », *New York : Chapman & Hall/CRC*, 2002.

R Development Core Team « R : A Language and Environment for Statistical Computing », *R Foundation for Statistical Computing, Vienna, Austria*, en ligne : http ://www.R-project.org, 2010.

Raftery A., Hoeting J., Volinsky C., Painter I. et Yeung K. Y. « BMA : Bayesian Model Averaging, paquetage R version 3.12 », en ligne : http ://CRAN.R-project.org/package=BMA, 2009.

Raftery, A. E. « Bayesian Model Selection in Structural Equation Models » In Testing Stuctural Equation Models (K.A. Bollen and J.S. Long, eds.), *Beverly Hills : Sage*, 1993, p. 163 - 180.

Schwartz, G. « Estimating the dimension of a model », *Annals of Statistics*, vol. 6, no. 2, 1978, p. 461 – 464.

Xuelei (Sherry), N. « Regressions by Enhanced Leaps-And-Bounds via Additional Optimality Tests (LBOT) » In New Results in Detection, Estimation, and Model Selection, *Thèse de doctorat, Georgia Institute of Technology*, 2006.