

Science des données – 3 Apprentissage Statistique

PHILIPPE BESSE

NB. Les cours et travaux pratiques (scénarios, ateliers) des sites wikistat.fr et [github.com/wikistat.fr](https://github.com/wikistat/fr) sont dispensés en formation initiale à l'[INSA de Toulouse](#) dans la spécialité : Ingénieur en *Mathématiques Appliquées*. Ils sont également proposés dans le cadre de stages de *Formation Professionnelle Qualifiante*.

Équipe de Statistique et Probabilités
Institut de Mathématiques de Toulouse — UMR CNRS C5219

Département Génie Mathématique et Modélisation
Institut National des Sciences Appliquées de Toulouse — 31077 – Toulouse cedex 4.

Table des matières

• Statistique & Science des Données	page 1
• Statistique & Apprentissage	page 5
• Qualité de prévision et risque	page 16
• Sélection de modèle en régression multiple	page 28
• Régression logistique	page 45
• Régression PLS, ACP et PLS parcimonieuses	page 53
• Analyse discriminante décisionnelle	page 64
• Arbres binaires de décision	page 70
• Réseaux neuronaux	page 79
• Agrégation de modèles	page 88
• Machines à vecteurs supports	page 104
• Imputation de données manquantes	page 111
• Détection d'anomalies	page 121
• En guise de conclusion	page 131
• Annexes	
• Introduction au bootstrap	page 139

Attention ce cours est dense, la lecture de ce document ne suffira pas à la bonne compréhension des outils qui y sont décrits de façon synthétique. La présence aux cours et la participation active aux TDs sont indispensables à l'acquisition des compétences incontournables pour une utilisation raisonnable et raisonnée des techniques d'apprentissage statistique. Toutes les méthodes sont développées sous la forme de tutoriels disponibles dans le dépôt github.com/Wikistat.

De la Statistique à la Science des (grosses) Données

Résumé

Statistique, fouille ou Science des Données, les appellations changent le volume et la diversité des données explosent, les technologies se succèdent, les modèles et algorithmes se complexifient. L'estimation devient un apprentissage, la prévision remplace l'explication. Le parcours pour devenir data scientist est structuré en quatre parties :

Saison 1 (L3) *Statistique élémentaire, descriptive vs. inférentielle.*

Saison 2 (M1) *Statistique Exploratoire multidimensionnelle et apprentissage non supervisé.*

Saison 3 *Apprentissage Statistique / Machine supervisé.*

Saison 4 (M2) *Technologies pour la Science des (grosses) Données.*

plus des réflexions sur : *Statistique et Déontologie scientifique.*

1 Origines de la Data Science

Le terme de *data scientist* a été "inventé" par Dhanurjay "DJ" Patil (LinkedIn)¹ et Jeff Hammerbacher (Facebook) en cherchant comment caractériser les métiers des données pour afficher des offres d'emploi : *Analyste, ça fait trop Wall Street; statisticien, ça agace les économistes; chercheur scientifique, ça fait trop académique. Pourquoi pas "data scientist"?*

Une "définition" attribuée à J. Wills (Cloudera) est souvent reprise : *Data scientist (n) : Person who is better at statistics than any software engineer and better at software than any statistician*

La Science des Données n'est pas une nouvelle science créée *ex nihilo* mais l'association de compétences (informatique, mathématiques, métiers) résultant

d'une longue évolution parallèle à celle des moyens de calcul et des volumes de données concernés. Cette évolution est passée par l'*analyse des données* en France, l'*Exploratory Data Analysis* ou EDA au USA, le *data mining* ou fouille des données puis la *Bioinformatique*.

En voici un bref résumé nécessairement schématique avec une chronologie linéaire :

1930-70 – hOctets Il était une fois la *Statistique* (inférentielle) : une question, (e.g. biologique), associée à une *hypothèse expérimentalement réfutable* H_0 , une expérience planifiée avec un échantillon *représentatif* de $n \approx 30$ individus observés sur p (moins de 10) variables, un modèle *linéaire gaussien* supposé vrai, un test, une décision, donc une réponse qui peut être inférée à la population en contrôlant le risque (généralement 5%) de rejeter à tort H_0 .

1970s – kO Les premiers outils informatiques se généralisant et, pour échapper à l'impérialisme du modèle linéaire, l'*analyse des données* (Caillez et Pages, 1976)[?] se développe en France ; l'*Exploratory Data Analysis* ou EDA aux États-Unis (Tukey 1977)[?]. L'objectif est alors de décrire ou explorer, prétendument sans modèle, des données déjà plus volumineuses.

1980s – MO En Intelligence Artificielle (IA), les *systèmes experts* expirent, supplantés par l'apprentissage des *réseaux de neurones*. La Statistique développe des modèles non-paramétriques ou fonctionnels.

1990s – GO *Data Mining et Premier changement de paradigme*. Les données ne sont plus *planifiées*, elles sont préalablement acquises et basées dans des entrepôts pour les objectifs usuels (e.g. comptables) de l'entreprise. L'aide à la décision les valorise : *From Data Mining to Knowledge Discovery* (Fayyad ; 1997)[?]. Les logiciels de fouille regroupent dans un même environnement des outils de gestions de bases de données, des techniques exploratoires et de modélisation statistique. C'est l'avènement du marketing quantitatif et de la gestion de la relation client (GRC ou CRM). L'IA se développe avec l'émergence du (*Machine Learning*) dont un sous-ensemble de méthodes est mis en exergue par le livre de Vapnik (1998) : *The Nature of Statistical Learning Theory*.

1. Entretien publié dans un article de l'Obs.

2000s – TO Deuxième changement de paradigme. Le nombre p de variables explose (de l'ordre de 10^4 à 10^6), notamment avec les biotechnologies omiques où $p >> n$ et la Bioinformatique. Le FDR (*False Discovery Rate*) de Benjamini et Hochberg (1995)[?] se substitue à la p -valeur et l'Apprentissage Statistique (Hastie et al. 2009)[?] sélectionne des modèles en optimisant leur complexité par un meilleur compromis *biais vs. variance*; minimiser conjointement erreur d'*approximation* (biais) et erreur d'*estimation* (variance).

2010s – PO Troisième changement de paradigme. Dans les applications industrielles, le e-commerce, avec la géo-localisation, la *datafication* du quotidien où toutes les traces numériques sont enregistrées, c'est le nombre n d'individus qui explose ; les statistiques usuelles de test, toutes significatives, perdent leur utilité au profit des méthodes d'apprentissage non supervisées ou supervisées ; les bases de données se déstructurent et se stockent dans les nuages (*cloud computing*), les moyens de calculs se groupent (*cluster*), mais la puissance brute ne suffit plus à la voracité (*greed*) des algorithmes. Un troisième terme d'erreur est à prendre en compte : celle d'*optimisation*, induite par la limitation du temps de calcul ou celle du volume des données considéré ; leur flux nécessite la construction de décisions adaptatives ou séquentielles.

Une présentation plus détaillée de la "science des données" et ses implications notamment économiques est proposée par [Besse et Laurent \(2015\)](#).

2 Environnement logiciel

2.1 Logiciels de fouille de données

Dès les années 90, et provoquant l'avènement de la *fouille de données* (*data mining*), les éditeurs de logiciels commerciaux et les communautés de logiciels libres ont inclus dans leurs suites, en plus des modèles linéaires classiques, les différents algorithmes d'apprentissage au fur et à mesure de leur apparition. Ceux-ci ont été intégrés à un ensemble plus complet de traitement des données en connexion avec les gestionnaires de bases de données relationnelles, le tout pilotable par une interface graphique plus ou moins conviviale : *Clementine* de SPSS, *Enterprise Miner* de SAS, *Insightfull Miner* de Splus, KXEN, SPAD,



FIGURE 1 – À copier 100 fois.

Statistica Data Miner, Statsoft, WEKA... Leur apparente simplicité d'utilisation a largement contribué à la diffusion de méthodes sophistiquées dans des milieux difficilement perméables à une conceptualisation mathématique abstraite et peu armés pour des développements logiciels importants.

2.2 R vs. Python

Dans ce paysage en constante évolution ou révolution, les langages R (2015)[?] et Python (Rossum et Guido ; 1995)[?] jouent un rôle particulier. L'analyse des offres de stage et d'emploi montre de profonds changements dans les demandes. SAS, plébiscité jusqu'à la fin du siècle dernier est largement supplanté par R et maintenant Python pour des raisons d'évidente économie mais aussi de flexibilité.

R

Toute méthode d'apprentissage est implémentée en R sous la forme d'une librairie (*package*) librement accessible. C'est même le mode de diffusion privilégié de nouvelles méthodes. Pour faciliter la tâche de leurs utilisateurs et

surtout uniformiser l'intégration de méthodes développés par des auteurs différents, Kuhn (2008)[?] propose une méta-librairie (*caret*) pouvant exécuter plus de 200 méthodes ou variantes de méthodes à partir de la même syntaxe. Néanmoins et comme Matlab, R est un langage interprété ; même en utilisant des librairies spécifiques pour paralléliser certains calculs compilés en C, les temps d'exécution de R deviennent vite rédhibitoires avec des données un peu volumineuses. De plus, son utilisation est rendue impossible (ou très difficile) dès que les limites de la mémoire interne de l'ordinateur sont atteintes.

Python

Plus récent Ross (1995)[?], le langage Python s'est considérablement développé notamment pour le traitement et l'analyse de signaux, images et séries financières. Python permet de paralléliser facilement la préparation (*data munging*) de grosses données sans les charger en mémoire avant de passer à la phase d'exploration puis de modélisation qui est elle toujours traitée en chargeant les données en mémoire.

Une des librairies : *Scikit-learn* (Pedregosa et al. 2011)[?] met à disposition les principales méthodes d'apprentissage supervisées ou non. Cette librairie n'est pas ouverte au sens où le choix d'implémentation d'une méthode est décidé au sein du groupe des développeurs principaux. L'avantage est un développement intégré et homogène, l'inconvénient, qui peut être aussi un avantage, est un choix plus restreint de méthodes accessibles. Également interprété, Python s'avère beaucoup plus rapide que R en gérant par défaut les possibilités de parallélisation d'une machine, même sous Windows.

R vs. Scikit-Learn

Le choix entre ces deux environnements repose sur les quelques points suivants :

- R et ses librairies offrent beaucoup plus de possibilités pour une exploration, des sélections et comparaisons de modèles, des interprétations statistiques détaillées avec des graphes produits par défaut.
- Mise en œuvre souvent implicite des possibilités de parallélisation, même sous Windows, par les librairies de Python.
- *Scikit-Learn* ne reconnaît pas (ou pas encore ?) la classe *DataFrame* développée dans la librairie *pandas*. Cette classe est largement utilisée en R pour gérer différents types de variables. C'est un problème dans

Scikit-Learn pour la prise en compte de variables qualitatives complexes. Une variable binaire est simplement remplacée par une indicatrice (0, 1) mais, en présence de plusieurs modalités, une variable qualitative est remplacée par l'ensemble des indicatrices (*dummy variables* (0, 1)) de ses modalités. Ceci complique les stratégies de sélection de modèles et rend obscure leur interprétation.

En résumé, préférer R pour modéliser et interpréter des modèles statistiques mais préférer Python pour des modélisations efficaces à seule fin prédictive au détriment de l'interprétation. Les deux approches pouvant d'ailleurs être traitées de façon complémentaire.

Enfin, si les données sont trop volumineuses pour la mémoire interne voire pour le disque d'un ordinateur, ou encore si les données sont déjà archivées sur une architecture distribuée, d'autres approches sont à considérer et abordées en saison 4 avec Spark.

2.3 Reproductibilité des analyses

Donoho (2015)[?] insiste à juste titre sur la question importante de la reproductibilité des analyses. Les médias se font régulièrement l'écho de manquements déontologiques et plus généralement du problème récurrent du manque de reproductibilité des résultats publiés dans des journaux ou revues que ce soit par exemple en Biologie ou en Psychologie. Pour un statisticien, contribuer à la prise en compte de ces problèmes consiste à produire des chaînes de traitements ou d'analyses (*pipeline*) facilement transmissibles pour être reproductibles sur des matériels standards. Deux environnements s'y prêtent particulièrement. Le premier concerne l'automatisation de la production d'un rapport en intégrant des commandes R (librairie *sweave* ou *knitr*) ou Python (*pweave*) au sein d'un source *LATEX*. Ces commandes, automatiquement exécutées, provoquent l'insertion de tableaux ou graphiques. Le deuxième, plus en amont, consiste à enregistrer systématiquement l'enchaînement des commandes et de leurs résultats numériques ou graphique dans un calepin (*notebook IPython* ou *Jupyter*). La sauvegarde est faite sous un format ré-exécutable dans un environnement similaire ou sous forme de fichier au format *html*, *pdf*. Ce type de résultat est obtenu en exécutant le bon noyau (Python, R, Julia...) dans le même environnement *Jupyter* à partir d'un simple navigateur. C'est pour cette raison que tous les tutoriels sont exécutables sous la forme d'un calepin, notamment pour les

- Tutoriels d'[initiation à R](#).
- Tutoriels d'[initiation à Python](#).

À exécuter et approfondir parallèlement à la maîtrise des principales méthodes.

3 Méthodes de la Science des Données

3.1 Méthodes traitées

L'historique précédent illustre schématiquement une progression pédagogique car il est difficile d'analyser de grands ensembles de données sans maîtriser les outils de base développés pour des données plus modestes à condition de bien identifier et faire coïncider les objectifs d'une étude : exploratoire, explicatif ou prédictif, avec ceux des méthodes mis en œuvre. C'est aussi une progression méthodologique, des outils les plus simples aux plus sophistiqués, pour aborder un nouvel ensemble de données.

Cette présentation propose donc de découper schématiquement la progression de la formation d'un *data scientist*, du L3 au M2, en quatre étapes ou *saisons* regroupant chacune un ensemble de scénarios ou épisodes couplant présentation théoriques et tutoriels pratiques des différentes méthodes et donc compétences à acquérir.

Saison 1 (L3) [Statistique élémentaire](#), descriptive vs. inférentielle.

Saison 2 (M1) [Statistique Exploratoire multidimensionnelle](#) et apprentissage non supervisé.

Saison 3 [Apprentissage Statistique / Machine supervisé](#).

Saison 4 (M2) [Technologies pour la Science des \(grosses\) Données](#).

N.B. Cette formation s'appuie sur des compétences parallèlement acquises en Statistique mathématique, calcul des Probabilités, Optimisation, Analyse Fonctionnelle pour une compréhension approfondie des méthodes et algorithmes utilisées, de leurs limites, et en Informatique pour leur mise en exploitation.

3.2 Méthodes auxquelles vous avez échappé

Certains points n'ont pas été intégrés à ce déroulement notamment en lien avec le V de *variété* ou celui de *vélocité*. Il faut se rendre à l'évidence qu'il n'est pas possible de former à bac+5 un mouton à 7 pattes supposé maîtriser

toute la "science des données". Il a fallu faire des choix laissant de côté certains points :

- Méthodes d'apprentissage machine mais pas d'apprentissage statistique comme celles issues du domaine de la logique formelle. La recherche de règles d'associations (problème du panier de la ménagère) en est une. Elle consiste à identifier les co-occurrences les plus fréquentes ou significatives par un ensemble de règles logiques associant variables et valeurs de celles-ci. Elle n'est pas adaptée à des volumétries importantes.
- Traitement de données structurées (variété) : graphes, trajectoires, images, signaux. Ces dernières nécessitent la projection des données sur des bases fonctionnelles adaptées (Fourier, ondelettes, splines) ou l'utilisation de distances (trajectoires GPS, graphes) ou noyaux spécifiques.
- Traitement de flux de données (vélocité). L'apprentissage se fait en ligne, voire en temps réel, et sans stockage par des algorithmes d'optimisation stochastique pour produire des décisions séquentielles, des recommandations de produits par des algorithmes de bandit.

Apprentissage Machine / Statistique

Résumé

Statistique, fouille ou Science des Données, les appellations changent le volume et la diversité des données explosent, les technologies se succèdent, les modèles et algorithmes se complexifient. L'estimation devient un apprentissage, la prévision remplace l'explication. Le parcours pour devenir data scientist est structuré en quatre parties :

Retour à l'introduction générale

Saison 1 (L3) *Statistique élémentaire, descriptive vs. inférentielle.*

Saison 2 (M1) *Statistique Exploratoire multidimensionnelle et apprentissage non supervisé.*

Saison 3 *Apprentissage Statistique / Machine* supervisé.

Saison 4 (M2) *Technologies pour la Science des (grosses) Données.*

plus des réflexions sur : *Statistique et Déontologie scientifique.*

1 Introduction

1.1 Objectifs de l'apprentissage

Questions ?

Identifier les facteurs aggravants de certains types de cancer en fonction de variables cliniques et démographiques, rechercher des gènes potentiellement impliqués dans une maladie à partir de données de séquençage ou, plus généralement, des bio-marqueurs pour un diagnostic précoce, identifier des chiffres manuscrits de codes issus d'images digitalisées, prévoir un taux de pollution atmosphérique en fonction de conditions météorologiques (cf. figure 1), établir des scores d'appétence ou d'attrition en gestion de la relation client (GRC), construire des méta-modèles ou modèles de substitution à un code numérique trop complexe pour analyser la sensibilité aux paramètres, détecter ou mieux

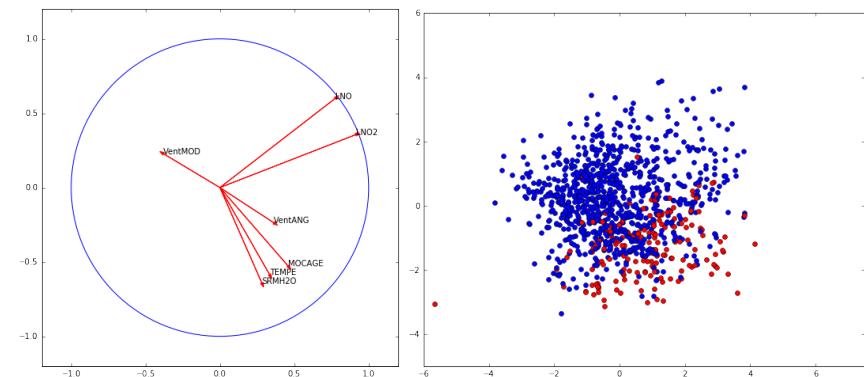


FIGURE 1 – *Ozone* : Préliminaire à la prévision par adaptation statistique d'une prévision déterministe. Premier plan de l'analyse en composantes principales (47% de variance expliquée). En rouge, les jours à prévoir de dépassement du seuil critique.

prévoir les défaillances d'un procédé... sont autant d'exemples où l'objectif est de minimiser une *erreur de prévision* ou *risque*. C'est encore la recherche d'un *modèle* plus généralement d'une *méthode optimale* au sens d'un critère à définir précisément.

Parallèlement, les méthodes et algorithmes issus de l'Intelligence Artificielle (e.g.[réseaux de neurones](#)) se focalisaient sur le même objectif pour devenir l'*Apprentissage Machine* incluant les méthodes et modèles de l'*apprentissage statistique*. La notion de d'apprentissage statistique (*statistical learning*) a été introduite par Vapnik (1998)[?] et popularisée par Hastie et al.(2001)[5].

Les choix de méthodes, de modèles, sont complexes à opérer et se déclinent en sous-objectifs qui restreignent où précisent les classes de modèles à considérer. L'objectif est-il seulement *prédictif*? Sous-entendu, un modèle *boîte noire* suffit-il à répondre aux besoins sans interprétation détaillée? En revanche, une compréhension du modèle, donc de l'impact des variables, attributs ou facteurs, est-elle recherchée voire indispensable? Ou encore, plus précisément, est-ce la détermination d'un petit sous-ensemble de ces variables

(e.g. des biomarqueurs) qui est recherchée pour rendre opérationnelle une prévision suffisamment précise et peu coûteuse ?

Historiquement, la Statistique s'est beaucoup développée autour de ce type de problèmes et a proposé des *modèles* incorporant d'une part des *variables explicatives ou prédictives* et, d'autre part, une composante aléatoire ou *bruit*. Il s'agit alors d'*estimer* les *paramètres* du modèle à partir des observations en contrôlant au mieux les propriétés et donc le comportement de la partie aléatoire. Dans la même situation, la communauté informatique parle plutôt d'*apprentissage* visant le même objectif; apprentissage machine (ou *machine learning*), reconnaissance de forme (pattern recognition) en sont les principaux mots-clés.

Objectif

L'objectif général est donc un objectif de *modélisation* qui peut se préciser en sous-objectifs à définir clairement préalablement à une étude car ceux-ci conditionnent en grande part les méthodes qui pourront être mises en œuvre :

Modéliser pour :

explorer ou vérifier, représenter, décrire, les variables, leurs liaisons et positionner les observations de l'échantillon,

expliquer ou tester l'influence d'une variable ou facteur dans un modèle supposé connu a priori,

prévoir & sélectionner un meilleur ensemble de prédicteurs comme par exemple dans la recherche de bio-marqueurs,

prévoir par une éventuelle meilleure "boîte noire" sans besoin d'interprétation explicite.

Rien n'empêche de construire et comparer tous types de modèles, qu'ils soient interprétatifs ou non, avec sélection de variables ou non ; les approches sont complémentaires. Compréhension préalables des données et connaissance des modèles, performances des prévisions, majoration ou contrôle des erreurs, efficacité algorithmique, sont autant de considérations à prendre en compte. Plus précisément les compétences issues des deux champs disciplinaires concernés sont nécessaires pour atteindre le but visé.

Des paramètres importants du problème sont les dimensions : n nombre d'observations ou taille de l'échantillon et p nombre de variables observées sur

cet échantillon. Lorsque les méthodes statistiques traditionnelles se trouvent mises en défaut pour de grandes valeurs de p , éventuellement plus grande que n , le sous-ensemble de l'*apprentissage machine* nommé *apprentissage statistique (statistical learning)* propose un ensemble de méthodes et algorithmes pertinents car efficaces. Les stratégies de choix de modèle parmi un ensemble plus ou moins complexe, de choix de méthode, sont au cœur de la thématique de ce cours. La fouille et maintenant la science des données se focalisent sur des pratiques, méthodes ou algorithmes dont Hastie et al. (2009)[\[5\]](#) proposent un tour d'horizon assez exhaustif.

Buts

L'objectif est bien de minimiser une erreur de prévision mais dans quel contexte ou pour quel but ? Schématiquement, s'agit-il de faire accepter un article dans une revue académique (Statistique, Apprentissage Machine, Bioinformatique...) ou de développer une solution "industrielle" (commerce électronique, détection de fraude ou de défaillance,...) ou encore de gagner un concours de prévision de type *Netflix* ou *Kaggle*. Le même objectif de minimisation d'une erreur de prévision peut alors conduire à des solutions radicalement différentes. La publication d'une nouvelle méthode d'apprentissage ou de nouvelles options de méthodes existantes nécessite de montrer qu'elle surpassé ses concurrentes sur une batterie d'exemples, généralement issus du site hébergé à l'Université de Californie Irvine (*UCI Repository*[\[6\]](#)). Les biais inhérents à cette démarche sont discutés dans de nombreux articles (e.g. Hand; 2006)[\[4\]](#) et conférences (e.g. Donoho; 2015)[\[3\]](#). Il est notable que la pression académique de publication a provoqué une explosion du nombre de méthodes et de leurs variantes, alors que celles-ci peuvent conduire à des différences de performances peu ou pas significatives.

L'analyse du déroulement des concours de type *Kaggle* et de leurs solutions gagnantes est très instructive. La pression, donc le biais, est tout à fait différent. Il conduit à des combinaisons, voire architecture de modèles, d'une telle complexité (cf. e.g. figure 2) que ces solutions sont concrètement inexploitables pour des différences de performances minimes (3 ou 4ème décimale).

En effet, surtout si les données sont en plus volumineuses (cf. saison 4), les solutions opérationnelles et "industrialisées", nécessairement robustes et rapides, se contentent souvent d'outils méthodologiques assez rudimentaires et peu glamour dirait Donoho (2015)[\[3\]](#).

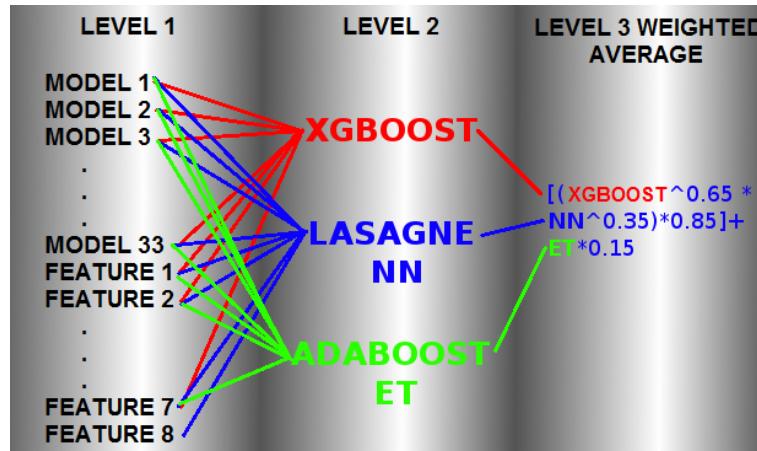


FIGURE 2 – Solution gagnante d'un concours kaggle : *Identify people who have a high degree of Psychopathy based on Twitter usage. Combinaison pondérée des combinaisons (boosting, réseaux de neurones) de trente trois modélisations (random forest, boosting, k plus proches voisins...) et 8 nouvelles variables (features) ad hoc.*

Cette saison propose d'aborder la grande variété des critères et méthodes proposés, leurs conditions de mise en œuvre, les choix à opérer, notamment pour optimiser la complexité des modèles. C'est aussi l'occasion de rappeler que des méthodes robustes et linéaires ainsi que les stratégies anciennes (descendantes, ascendantes, pas-à-pas) ou plus récentes (lasso) de sélection de modèles linéaires ou polynomiaux ne doivent pas être trop rapidement évacuées des pratiques académiques ou industrielles.

1.2 Définitions

Apprentissage Supervisé vs. non-supervisé

Distinguons deux types de problèmes : la présence ou non d'une variable à expliquer Y ou d'une forme à reconnaître qui a été, conjointement avec X , observée sur les mêmes objets. Dans le premier cas il s'agit bien d'un problème de modélisation ou *apprentissage supervisé* : trouver une fonction f susceptible, au mieux selon un critère à définir, de reproduire Y ayant observé X .

$$Y = \hat{f}(X) + \varepsilon$$

où ε représente le bruit ou erreur de mesure avec le parti pris le plus commun que cette erreur est additive. En cas d'erreur multiplicative, une transformation logarithmique ramène au problème précédent.

Dans le cas contraire, en l'absence d'une variable à expliquer, il s'agit alors d'apprentissage dit *non-supervisé*. L'objectif généralement poursuivi est la recherche d'une typologie ou taxinomie des observations : comment regrouper celles-ci en classes homogènes mais les plus dissemblables entre elles. C'est un problème de classification (*clustering*).

Attention, l'anglais *classification* se traduit plutôt en français par discrimination ou classement (apprentissage supervisé) tandis que la recherche de classes (*clustering*) (apprentissage non-supervisé) fait appel à des algorithmes de [classification ascendante hiérarchique](#), de [réallocation dynamique](#) (*kmeans*), DBSCAN ou encore des cartes auto-organisatrices (Kohonen) et bien d'autres...

Cette saison 3 est consacrée à l'apprentissage supervisé, pour lequel on dispose d'un *ensemble d'apprentissage* constitué de données d'observations de type entrée-sortie : $d_1^n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ avec $x_i \in \mathcal{X}$ quelconque (souvent égal à \mathbb{R}^p), $y_i \in \mathcal{Y}$ pour $i = 1 \dots n$.

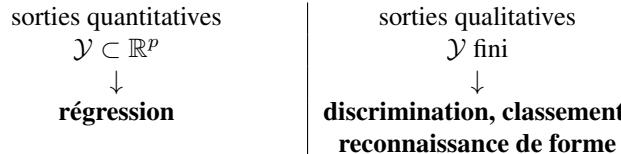
Les algorithmes d'apprentissage non supervisée sont abordés dans la [saison 2](#).

1.3 Les données

Dans tous les problèmes rencontrés, des caractéristiques, attributs (*features*), facteurs ou variables $X = (X^{(1)}, \dots, X^{(p)})$ dites explicatives ou prédictives ont été observées sur un ensemble de n objets, individus, *instances* ou unités statistiques. Contrairement à une démarche statistique traditionnelle dans laquelle l'observation des données est intégrée à la méthodologie (planification de l'expérience), les données sont généralement *préalables* à l'analyse.

1.4 Discrimination vs. régression

L'objectif est de construire, à partir de cet échantillon d'apprentissage, un modèle, qui va nous permettre de prévoir la sortie Y associée à une nouvelle entrée (ou prédicteur) X . La sortie Y peut être quantitative (prix d'un stock, consommation électrique, carte de pollution ..) ou qualitative (survenue d'un cancer, reconnaissance de chiffres...) selon l'espace dans lequel elle prend ses valeurs : ensemble de cardinal fini ou réelles voire fonctionnelles. Certaines méthodes d'apprentissage ou de modélisation s'adaptent à tout type de variables explicatives tandis que d'autres sont spécialisées. Si Y à expliquer est qualitative, on parle de discrimination, classement ou reconnaissance de forme tandis que si Y est quantitative on parle, par habitude, d'un problème de régression. Certaines méthodes sont spécifiques (régression linéaire, analyse discriminante) à un type de variable à modéliser tandis que d'autres s'adaptent sans modification profonde remettant en cause leur principe (réseaux de neurones, arbres de décision...).



Régression *réelle* lorsque $\mathcal{Y} \subset \mathbb{R}$ et discrimination *binaire* lorsque $\mathcal{Y} = \{0, 1\}$ ou $\{-1, 1\}$.

1.5 Estimation vs. apprentissage

Tout au long de ce document, les termes d'*estimation* et d'*apprentissage* sont utilisés comme des synonymes mais ceci nécessite de préciser quelques nuances. Dans la tradition statistique, la notion de *modèle* est centrale surtout avec une finalité *explicative*. Il s'agit alors d'approcher la réalité, le *vrai* modèle, supposé exister, éventuellement basé sur une théorie physique, économique, biologique... sous-jacente et la forme du modèle est guidée par des indications théoriques et des critères d'*ajustement*; les décisions de validité, de présence d'effets sont basées sur des *tests* reposant elles-mêmes sur des hypothèses probabilistes. L'interprétation du rôle de chaque variable explicative est prépondérante dans la démarche.

En revanche, si l'objectif est essentiellement la *prévision*, il apparaît que le meilleur modèle n'est pas nécessairement celui qui ajusterait le mieux le vrai modèle. La théorie de l'*apprentissage* (Vapnik, 1998)[?] montre alors que le cadre théorique est différent et les majorations d'erreur requièrent une autre approche. Les choix sont basés sur des critères de qualité de *prévision* visant à la recherche de *modèles parcimonieux*, c'est-à-dire de complexité (nombre de paramètres ou flexibilité limitée) dont l'interprétabilité passe au deuxième plan. La deuxième devise (cf. figure 3) des Shadoks n'est pas une référence à suivre en apprentissage statistique !

1.6 Statistique, informatique et taille des données

Lorsque les dimensions du problèmes (n, p) sont raisonnables et que des hypothèses relatives au modèle (linéarité) et aux distributions sont vérifiées c'est-à-dire, le plus souvent, lorsque l'échantillon ou les résidus sont supposés suivre des lois se mettant sous la forme d'une famille exponentielle (gaussienne, binomiale, poisson...), les techniques statistiques de modélisation tirées du modèle linéaire général sont optimales (maximum de vraisemblance) et, surtout dans le cas d'échantillons de taille restreinte, il semble difficile de faire beaucoup mieux.

En revanche, dès que les hypothèses distributionnelles ne sont pas vérifiées, dès que les relations supposées entre les variables ou la variable à modéliser ne sont pas linéaires ou encore dès que le volume des données (*big data*) est important, d'autre méthodes viennent concurrencer les modèles statistiques rudimentaires.



FIGURE 3 – Deuxième devise Shadok

Prenons un exemple simple : expliquer une variable quantitative Y par un ensemble $\{X^1, \dots, X^p\}$ de variables également quantitatives :

$$Y = f(X^1, \dots, X^p) + \varepsilon.$$

observées sur un échantillon $(y_i, \mathbf{x}_i); i = 1, \dots, n$ de taille n . Si la fonction f est supposée linéaire et p petit, de l'ordre d'une dizaine ; le problème est bien connu et largement débattu dans la littérature. Dans le cas où la fonction f n'est pas franchement linéaire et n grand, il est possible d'estimer précisément un nombre plus important de paramètres et donc d'envisager des modèles plus sophistiqués. Si on s'en tient au modèle gaussien usuel, même le cas le plus simple d'un modèle polynomial devient vite problématique. En effet, lorsque la fonction f est linéaire, prenons $p = 10$, la procédure de choix de modèle est confrontée à un ensemble de 2^{10} modèles possibles et des algorithmes astucieux permettent encore de s'en sortir. En revanche, considérer, pour estimer f , un simple polynôme du deuxième voire troisième degré avec toutes ses interactions, amène à considérer un nombre considérable de paramètres et donc, par explosion combinatoire, un nombre astronomique de modèles possibles. D'autres méthodes doivent alors être considérées en prenant en compte nécessairement la complexité algorithmique des calculs. Le souci de calculabilité

l'emporte sur la complexité mathématique du modèle et conduit à l'optimisation d'un critère d'ajustement de la fonction f sur un ensemble de solutions plus ou moins riche. Les méthodes développées dans la communauté d'apprentissage machine : k plus proches voisins, réseaux de neurones, arbres de décisions, *support vector machine*... deviennent des alternatives crédibles dès lors que le nombre d'observations est suffisant ou le nombre de variables très important.

2 Stratégies de choix

2.1 Choix de méthode

Avec le développement du *data mining*, de très nombreux articles comparent et opposent les techniques sur des jeux de données publics et proposent des améliorations incrémentales de certains algorithmes. Après une période fiévreuse où chacun tentait d'afficher la suprématie de sa méthode, un consensus s'est établi autour de l'idée qu'il n'y a pas de "meilleure" méthode. Chacune est plus ou moins bien adaptée au problème posé, à la nature des données ou encore aux propriétés de la fonction f à approcher ou estimer. Sur le plan méthodologique, il est alors important de savoir comparer des méthodes afin de choisir la plus pertinente. Cette comparaison repose sur une estimation d'erreur (de régression ou de classement) qu'il est nécessaire de conduire avec soin.

2.2 Choix de modèle : équilibre biais-variance

Le point central est la construction de *modèles parcimonieux* quelque soit la méthode utilisée. Toutes les méthodes sont concernées : nombre de variables explicatives, de feuilles dans un arbre ou de neurones dans une couche cachée.... Seuls certains algorithmes de combinaison de modèles (*e.g. bagging, random forest*) contournent cette étape au prix d'un accroissement sensible du volume des calculs et surtout de l'interprétabilité des résultats obtenus.

L'alternative est claire, plus un modèle est complexe et donc plus il intègre de paramètres et plus il est flexible donc capable de s'ajuster aux données engendrant ainsi une erreur faible d'ajustement. En revanche, un tel modèle peut s'avérer défaillant lorsqu'il s'agira de prévoir ou généraliser, c'est-à-dire de s'appliquer à des données qui n'ont pas participé à son estimation.

Un exemple élémentaire illustre ce point fondamental dans le cas d'un pro-

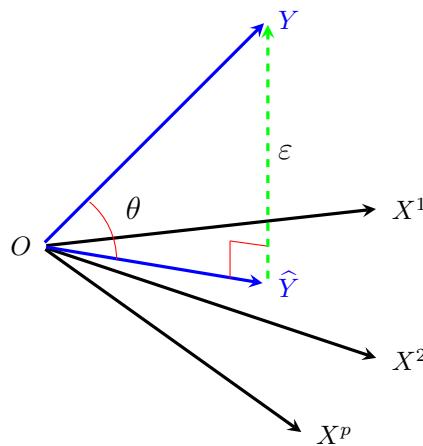


FIGURE 4 – Régression linéaire par projection \hat{Y} de Y sur l'espace vectoriel $\text{Vect}\{\mathbf{1}, X^1, \dots, X^p\}$

blème de régression (figure 4) : la qualité d'ajustement du modèle (R^2) augmente nécessairement avec le nombre de variables mais la variance des estimateurs peut exploser dès que la matrice à inverser ($\mathbf{X}'\mathbf{X}$) devient mal conditionnée. Dans un problème de discrimination dans \mathbb{R}^2 (figure 4) : une frontière dont le modèle "vrai" est quadratique est, par exemple à cause d'erreurs de mesure, sous-ajustée par une régression linéaire mais sur-ajustée par un polynôme de degré plus élevé ou l'algorithme local des k plus proches voisins avec k petit.

Ce problème s'illustre aussi facilement en régression classique. Ajouter des variables explicatives dans un modèle ne peut que réduire l'erreur d'ajustement (le R^2) et réduit le biais si le "vrai" modèle est un modèle plus complet. Mais, ajouter des variables fait rédhibitoirement croître la variance des estimateurs et donc celle des prévisions qui se dégradent, voire explosent, avec la multicollinearité des variables explicatives. Un risque pour le modèle, ou erreur quadratique de prévision, s'exprimant comme le carré du biais plus la variance, il est important d'optimiser le dosage entre biais et variance en contrôlant le nombre

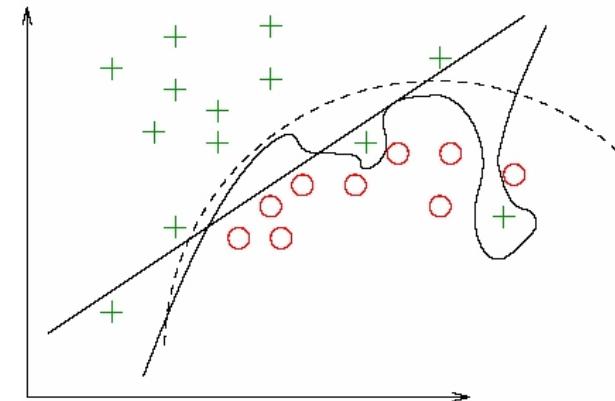


FIGURE 5 – Sous-ajustement linéaire et sur-ajustement local (proches voisins) d'un modèle quadratique.

de variables dans le modèle (sa complexité) afin de minimiser le risque. Ces remarques conduisent à la définition de critères de choix de modèle dont le C_p de Mallows fut un précurseur en régression suivi par d'autres propositions : Akaike (AIC), Schwartz (BIC)...

Parfois plus que celui de la méthode, le choix du bon modèle dans une classe ou ensemble de modèles pour une méthode donnée est primordial. En conséquence, les problèmes d'optimisation considérés doivent mettre en œuvre un critère qui prend en compte la *complexité du modèle*, c'est-à-dire la complexité de l'espace ou de la classe dans lequel la solution est recherchée.

2.3 Choix de modèle : sélection vs. régularisation

Selon la méthode considérée, la complexité du modèle s'exprime de différentes façons. Simple lors d'une sélection de variable en régression linéaire, la complexité est directement liée à la dimension de l'espace engendré et donc au nombre de variables. Les choses se compliquent pour les modèles non-linéaires lorsque, à dimension fixée, c'est la plus ou moins grande flexibilité des solutions qui doit être pénalisée.

C'est typiquement le cas en régression non-paramétrique ou fonctionnelle. Une pénalisation faisant intervenir la norme carrée de la dérivée seconde contrôle la flexibilité d'un lissage spline. La "largeur de fenêtre" du noyau contrôle également la régularité de la solution. En régression linéaire, si le nombre et les variables sont déterminés, la version "ridge" de la régression pénalise la norme carrée du vecteur des paramètres et restreint ainsi, par *régularisation*, l'espace des solutions pour limiter l'effet de la multicolinéarité. Ce principe de régularisation par pénalisation a été transposé à la plupart des méthodes connues.

3 Stratégie de l'apprentissage statistique

3.1 Les étapes d'une analyse

En situation réelle, la préparation initiale des données, simple nettoyage (*cleaning*, points 1 et 2 ci-dessous) ou trafic (*munging, wrangling*) plus complexe : extraction, nettoyage, vérification, imputation éventuelle de données manquantes, transformations... est la phase la plus ingrate, celle qui demande le plus de temps, de ressources humaines et des compétences très variées : informatique, statistique et métier. Ne nécessitant pas de développements théoriques majeurs mais plutôt beaucoup de bon sens, d'expérience et une bonne connaissance des données du métier, cette préparation est négligée dans les travaux académiques généralement illustrés, par souci de concision, sur des données déjà pré-traitées donc nettoyées. Une fois bien menée à terme, la phase de modélisation ou apprentissage en découle finalement assez automatiquement même pour des méthodes et algorithmes sophistiqués.

De façon systématique et aussi très schématique, l'analyse, maintenant la *Science, des Données* enchaînent les étapes suivantes pour la plupart des domaines d'application. On retrouve l'essentiel des items des 6 divisions de Do-noho (2015)[3] avec un découpage différent.

1. Extraction des données avec ou sans échantillonnage faisant référence à des techniques de sondage appliquées ou applicables à des bases de données structurées (SQL) ou pas (NOSQL).
2. Visualisation, exploration des données pour la détection de valeurs atypiques, incohérences, erreurs ou anomalies ; étude des distributions et des structures de corrélation et recherche de transformations des variables, construction de nouvelles variables et/ou représentation dans

des bases (Fourier, spline, ondelettes...) adaptées, recherche de typologies des observations...

3. Prise en compte, par simple suppression, par imputation ou non, des données manquantes.
4. Partition aléatoire de l'échantillon (apprentissage, validation, test) en fonction de sa taille et choix d'une fonction perte ou critère qui seront utilisées pour estimer une erreur de prévision en vue des étapes d'optimisation de modèle, puis de choix de méthode.
5. Pour chacune des méthodes considérées : modèle linéaire général (gaus-sien, binomial ou poissonien), discrimination paramétrique (linéaire ou quadratique) ou non paramétrique (k plus proches voisins), réseau de neurones (perceptron), arbre binaire de décision, machine à vecteurs supports, combinaison de modèles (*bagging, boosting, random forest*...).
 - estimer le modèle pour une valeur donnée d'un paramètre (ou plusieurs) de *complexité* : nombre de variables, de voisins, de feuilles, de neurones, pénalisation ou régularisation... ;
 - optimiser ce paramètre (ou ces paramètres) en fonction de la technique d'estimation de l'erreur retenue : critère de pénalisation, échantillon de validation, validation croisée... .
6. Comparaison des modèles optimaux obtenus (un par méthode) par estimation de l'erreur de prévision sur l'échantillon test.
7. Itération éventuelle de la démarche précédente ou validation croisée *Monte Carlo*, si l'échantillon test est trop réduit, depuis (4). Partitions aléatoires successives de l'échantillon (apprentissage et test) pour étudier la distribution des erreurs et moyenneur sur plusieurs cas l'estimation finale de l'erreur de prévision et s'assurer de la robustesse du modèle obtenu.
8. Choix de la méthode retenue en fonction de ses capacités de prévision, de sa robustesse mais aussi, éventuellement, de l'interprétabilité recherchée du modèle.
9. Ré-estimation du modèle avec la méthode, le modèle et sa complexité optimisée à l'étape précédente sur l'ensemble des données.
10. Mise en exploitation sur la base complète ou de nouvelles données.

La fin de cette démarche peut être modifiée par la construction d'un *mieux compromis* ou d'une combinaison des différentes méthodes testées plutôt que de sélectionner la meilleure. C'est souvent le cas des solutions gagnantes "usines à gaz" des concours de type *Kaggle*. Cela a aussi été théorisé en deux approches conduisant à une *collaboration* entre modèles : COBRA de Biau et al. (2016)[2] et *SuperLearner* de van der Laan et al. (2007)[7]. La première revient à exécuter une forme d'algorithme des k plus proches voisins avec une définition spécifique de la distance tandis que la deuxième cherche, par minimisation d'une estimation de l'erreur par validation croisée, une meilleure combinaison convexe des prévisions.

L'une des principales difficultés pratiques est d'arriver à déterminer où faire porter l'effort ou les efforts :

- la saisie, la gestion, la sélection des données et variables ; la préparation des données est de toute façon essentielle ;
- la sélection des méthodes à comparer ;
- l'optimisation des choix de modèles ;

Tout ceci en fonction des méthodes considérées, de la structure des données, des propriétés des variables notamment celle à modéliser.

3.2 Les méthodes

Chaque méthode ou famille de méthodes de modélisation et d'apprentissage parmi les plus répandues, est présentée de façon plus ou moins succincte dans une vignette distincte avec un objectif de prévision.

- Une première vignette incontournable est consacrée aux techniques d'estimation d'une *erreur de prévision ou d'un risque* sur lequel reposent les choix opérationnels décisifs : de modèle, de méthode mais aussi l'évaluation de la précision des résultats escomptés.
- La *régression linéaire* ou modèle gaussien, classique en statistique, donne lieu à une bibliographie abondante. Conceptuellement plus simple, elle permet d'introduire plus facilement les questions rencontrées comme celle du choix d'un modèle selon les deux approches types : *sélection de variable* ou *pénalisation* (*ridge*, *Lasso*).
- La *régression PLS* propose une autre solution de choix de modèle par projection sur une base de facteurs orthogonaux avec une version *sparse* ou parcimonieuse pour simplifier l'interprétation.
- Toujours dans le cadre du *modèle linéaire général*, la *régression logistique* ou modèle binomial reste toujours très utilisée même et surtout lorsque les données sont massives.

- La présentation de l'*analyse discriminante décisionnelle*, paramétrique ou non paramétrique (dont les k plus proches voisins), permet d'introduire également des notions de théorie bayésienne de la décision.
- La vignette suivante est consacrée aux arbres binaires de décision (*classification and regression trees* ou CART)
- puis à des algorithmes plus directement issus de la théorie de l'apprentissage machine : *réseau de neurones* (perceptron).
- Viennent ensuite les méthodes d'*agrégation de modèles* (*boosting*, *random forest*),
- de *support vector machine* (SVM).
- Enfin une vignette s'intéresse à l'*imputation de données manquantes*
- avant de *conclure* par une présentation synthétique des différentes méthodes exposées ainsi que des considérations éthiques sur la *loyauté des décisions algorithmiques*.

3.3 Tutoriels : cas d'usage et "fil rouge"

Cette saison accorde beaucoup d'importance aux exemples pour illustrer le comportement des critères généralement utilisés et analyser les performances des méthodes étudiées tout en se formant à leur usage. En plus des exemples pédagogiques illustrant simplement les différentes méthodes étudiées, d'autres exemples en vraie grandeur permettent d'en évaluer réellement l'efficacité mais aussi toute la complexité de mise en œuvre.

L'analyse de ces différents cas d'usage se présente sous la forme de tutoriels contenus dans des calepins (*jupyter notebook*) en R ou Python. Ils sont ou seront disponibles dans le dépôt github.com/wikistat

Fil Rouge

Exécuter le tutoriel (calepin) : [Prévision du pic d'ozone en R et Python](#) en se référant aux descriptifs des méthodes autant que de nécessaire. L'exécution des tutoriels est découpé en *épisodes*. Cet exemple étudié par Besse et al. (2007)[1] est une situation réelle dont l'objectif est de prévoir, pour le lendemain, les risques de dépassement du seuil légal de concentration d'ozone dans des agglomérations. Le problème peut être considéré comme un cas de

régression : la variable à prévoir est une concentration en ozone, mais également comme une discrimination binaire : dépassement ou non du seuil légal. Il n'y a que 8 variables explicatives dont une est déjà une prévision de concentration d'ozone mais obtenue par un modèle déterministe de mécanique des fluides (équations de Navier et Stockes). Il s'agit d'un exemple d'*adaptation statistique*. La prévision déterministe sur la base d'un maillage global (30 km) est améliorée localement, à l'échelle d'une ville, par un modèle statistique incluant cette prévision ainsi que des informations connues sur la base d'une grille locale, spatiale et temporelle plus fine : concentration d'oxyde et dioxyde d'azote, de vapeur d'eau, température, vitesse et direction du vent.

Cet exemple, à la fois de régression et de discrimination, présente des vertus pédagogiques certaines qui permettent de l'utiliser comme *fil rouge* de comparaison entre toutes méthodes. L'étude préliminaire rudimentaire a conduit à la transformation (log) de certaines variables de concentration. Les données sont résumées par leur représentation dans le premier plan de l'analyse en composantes principales réduite (cf. figure 1). Ce graphique résume la structure de corrélation assez intuitives des variables et met en évidence les difficultés à venir pour discriminer les deux classes : présence ou non d'un pic de concentration avec dépassement du seuil légal.

Exemples jouet

Jeux de données élémentaires dans dans \mathbb{R}^2 . Exécuter les calepins en faisant varier les paramètres de complexité des modèles. [Discrimination en R](#) de mélanges gaussiens. [Discrimination en Python](#) de mélanges de points ou *blobs*.

Diagnostic d'une maladie coronarienne

Cas d'usage : [Diagnostic d'une maladie coronarienne](#) en R avec initiation à l'utilisation du package xgboost.

Spectrographie en proche infra-rouge (NIR)

Depuis de très nombreuses années, l'industrie agroalimentaire est confrontée à des problèmes de grande dimension pour l'analyse de données de spectrométrie comme par exemple dans le proche infra-rouge (NIR). Sous l'appellation de *Chimiométrie* de très nombreuses méthodes et stratégies ont été développées ou enrichies (*i.e.* la [régression PLS](#)) afin de prendre en compte la

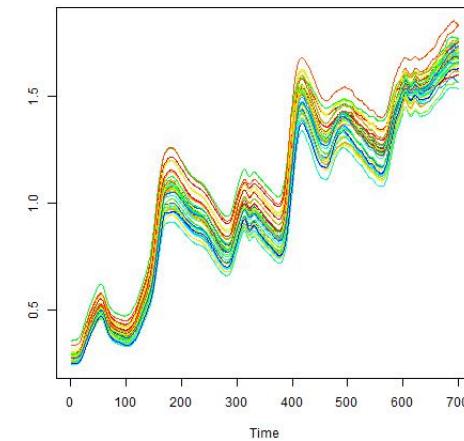


FIGURE 6 – Cookies : Spectres proche infrarouge (NIR) d'échantillons de pâtes à gâteaux. La couleur dépend du taux de sucre.

spécificité des problèmes rencontrés par la discrétisation de spectres conduisant très généralement à un nombre de variables $p > n$. Dans un premier exemple, il s'agit de modéliser, la teneur en sucre d'une pâte à gâteau ([cookies](#) où $n = 72, p = 700$) à partir des spectres (cf. figure 6) tandis que dans un deuxième ([Tecator](#) ou $n = 215, p = 100$), c'est la teneur en matière grasse qui est recherchée. Ces questions sont considérées comme des problèmes de *calibration* d'un appareil de mesure (le spectromètre) pour arriver à la quantification d'une mesure chimique dont l'évaluation classique est beaucoup plus coûteuse ou encore destructive.

QSAR : criblage virtuel de molécule

Une stratégie classique de l'industrie pharmaceutique consiste à tester *in silico* un nombre considérable de molécules avant de ne synthétiser que celles jugées intéressantes pour passer aux étapes de recherche clinique *in vitro* puis *in*

vivo. Une propriété thérapeutique d'un ensemble de molécules d'apprentissage (perméabilité de la paroi intestinale ou à la barrière sanguine du cerveau, adéquation à une cible donnée...) étant connue, un grand ensemble de caractéristiques physico-chimiques sont évaluées, calculées par un logiciel spécifique : ce sont des données dites **QSAR Quantitative structure-activity relationship**. S'il est possible de raisonnablement prévoir la propriété thérapeutique à partir des caractéristiques physico-chimiques, ce modèle est systématiquement appliqué à un grand ensemble de molécules virtuelles ; c'est le criblage ou *screening* virtuel de molécule. Deux jeux de données sont étudiés l'un illustrant un problème de régression (blood brain barrier data) avec $n = 208, p = 134$ tandis que l'autre est un problème de discrimination à deux classes (multidrug resistance reversal) avec $n = 528, p = 342$.

Biologie : sélection de gènes

Les techniques de microbiologie permettent de mesurer simultanément l'expression (la quantité d'ARN messager produite) de milliers de gènes dans des situations expérimentales différentes, par exemple entre des tissus sains et d'autres cancéreux. L'objectif est donc de déterminer quels gènes sont les plus susceptibles de participer aux réseaux de régulation mis en cause dans la pathologie ou autre phénomène étudié. Le problème s'énonce simplement mais révèle un redoutable niveau de complexité et pose de nouveaux défis au statisticien. En effet, contrairement aux cas précédents pour lesquels des centaines voire des milliers d'individus peuvent être observés et participer à l'apprentissage, dans le cas des biopuces, seuls quelques dizaines de tissus sont analysés à cause essentiellement du prix et de la complexité d'une telle expérience. Compte tenu du nombre de gènes ou variables, le problème de discrimination est sévèrement indéterminé. D'autres approches, d'autres techniques sont nécessaires pour pallier à l'insuffisance des méthodes classiques de discrimination.

L'exemple concerne les expressions de gènes dans une expérience croisant deux facteurs le **régime alimentaire** (5 niveaux) chez $n = 40$ souris de 2 génotypes. Il s'agit de mettre en évidence l'impact des facteurs sur les expressions de $p = 120$ gènes puis d'expliquer un ensemble de $q = 21$ variables phénotypiques (concentrations d'acides gras) par ces mêmes expressions.

Banque, finance, assurance : Marketing

L'objectif est une communication personnalisée et adaptée au mieux à chaque client. L'application la plus courante est la recherche d'un *score estimé* sur un échantillon de clientèle pour l'apprentissage puis extrapolé à l'ensemble en vue d'un objectif commercial :

- *Appétence* pour un nouveau produit financier : modélisation de la probabilité de posséder un bien (contrat d'assurance...) puis application à l'ensemble de la base. Les clients, pour lesquels le modèle prédit la possession de ce bien alors que ce n'est pas le cas, sont démarchés (télémarketing, publipostage ou mailing, phoning,...) prioritairement.
- *Attrition* ; même chose pour évaluer les risques de départ ou d'attrition (churn) des clients par exemple chez un opérateur de téléphonie. Les clients pour lesquels le risque prédit est le plus important reçoivent des incitations à rester.
- *Risque* pour l'attribution d'un crédit bancaire ou l'ouverture de certains contrats d'assurance ; risque de faillite d'entreprises.
- ...

Des jeux de données sont abordés, le premier construit un score d'appétence de la carte visa premier. Le deuxième analyse une enquête de l'INSEE sur les patrimoines des français pour l'estimation d'un score d'appétence des produits d'assurance vie.

L'exemple traité suit un schéma classique d'analyse de données bancaires. Après la **phase exploratoire**, il s'agit de construire un **score d'appétence** de la carte Visa Premier dans l'idée de fidéliser les meilleurs clients. La variable à prévoir est binaire : possession ou non de cette carte en fonction des avoirs et comportements bancaires décrits par $p = 32$ variables sur $n = 825$ clients.

Santé : aide au diagnostic

Les outils statistiques sont largement utilisés dans le domaine de la santé. Ils le sont systématiquement lors des essais cliniques dans un cadre législatif stricte mais aussi lors d'études épidémiologiques pour la recherche de facteurs de risques dans des grandes bases de données ou encore pour l'aide au diagnostic. L'exemple étudié illustre ce dernier point : il s'agit de prévoir un diagnostic à partir de tests biologiques et d'examens élémentaires. Bien entendu, la variable à prédire, dont l'évaluation nécessite souvent une analyse très coûteuse

voire une intervention chirurgicale, est connue sur l'échantillon nécessaire à l'estimation des modèles.

Dans l'exemple étudié ([breast cancer](#)), il s'agit de prévoir le type de la tumeur (bénigne, maligne) lors d'un cancer du sein à l'aide de $p = 9$ variables explicatives biologiques observées sur $n = 700$ patientes.

Adult census

Étude d'une enquête aux USA et prévision du dépassement d'un seuil de revenu.

Détection de pourriel

Les données sont composées d'un ensemble de messages ou courriels dont certains sont identifiés comme pourriels ou *spams*. Les données ont déjà été préparées ou simplifiées, les variables ou *features* sont les effectifs ou présence / absence de certains mots ou caractères spécifiques (\$, ! . . . L'objectif est d'apprendre à détecter un pourriel.

Références

- [1] P. Besse, H. Milhem, O. Mestre, A. Dufour et V. H. Peuch, *Comparaison de techniques de Data Mining pour l'adaptation statistique des prévisions d'ozone du modèle de chimie-transport MOCAGE*, Pollution Atmosphérique **195** (2007), 285–292.
- [2] G. Biau, A. Fischer, B. Guedj et J. D. Malley, *COBRA : A Nonlinear Aggregation Strategy*, Journal of Multivariate Analysis **146** (2016), 18–28.
- [3] David Donoho, *50 years of Data Science*, Princeton NJ, Tukey Centennial Workshop, 2015.
- [4] David J. Hand, *Classifier Technology and the Illusion of Progress*, Statist. Sci. **21** (2006), n° 1, 1–14.
- [5] T. Hastie, R. Tibshirani et J. Friedman, *The elements of statistical learning : data mining, inference, and prediction*, Springer, 2009, Second edition.
- [6] M. Lichman, *UCI Machine Learning Repository*, 2013, <http://archive.ics.uci.edu/ml>.
- [7] M. J. van der Laan, E. C. Polley et A. E. Hubbard, *Super learner*, Statistical Applications in Genetics and Molecular Biology **6 :1** (2007).

Qualité de prévision et risque

Résumé

Définition et propriétés du risque ou erreur de prévision ou erreur de généralisation dans le cas de la régression et de la classification. Décomposition biais / variance du risque. Critères de pénalisation et méthodes ou algorithmes d'estimation du risque empirique. Estimation sur échantillons de validation ou de test, par critère pénalisé, par validation croisée, par bootstrap, courbe ROC en discrimination binaire.

[Retour à l'introduction.](#)

Tous les tutoriels sont disponibles sur le dépôt :
github.com/wikistat

1 Introduction

1.1 Objectif

La performance du modèle ou algorithme issu d'une méthode d'apprentissage s'évalue par un *risque* ou *erreur de prévision*, dite encore *capacité de généralisation* dans la communauté informatique. La mesure de cette performance est très importante puisque, d'une part, elle permet d'opérer une *sélection de modèle* dans une famille associée à la méthode d'apprentissage utilisée et, d'autre part, elle guide le *choix de la méthode* en comparant chacun des modèles optimisés à l'étape précédente. Enfin, elle fournit, tous choix faits, une mesure de la qualité ou encore de la *confiance* que l'on peut accorder à la prévision.

Une fois que la notion de modèle statistique ou *règle de prévision* est précisée, le *risque* est défini à partir d'une fonction *perte* associée. En pratique, ce risque nécessite d'être estimé et différentes stratégies sont proposées.

Le **principal enjeu** est de construire une estimation *sans biais* de ce risque en notant que le *risque dit empirique*, qui est aussi l'erreur d'ajustement du modèle, est estimé sur les mêmes données d'*apprentissage* du modèle; par construction le risque empirique est *biaisé par optimisme* car toute erreur

ou risque estimé sur d'autres données (capacité de généralisation) et qui n'ont pas servi à estimer le modèle ou apprendre l'algorithme, conduit, en moyenne, à des valeurs plus élevées et sans biais si ces données sont bien représentatives.

Trois stratégies sont décrites pour construire des *estimations sans biais du risque* :

1. une *pénalisation* de l'erreur d'ajustement ou risque empirique ;
2. un partage de l'échantillon : apprentissage, (validation,) test afin de distinguer l'estimation du modèle et celle du risque ;
3. par simulation : validation croisée, *bootstrap*.

Le choix dépend de plusieurs facteurs dont l'objectif recherché, la taille de l'échantillon initial, la complexité du modèle envisagé, la variance de l'erreur, la complexité des algorithmes.

1.2 Risques et choix de modèle

Une estimation du risque ou qualité de la prévision est donc un élément central de la mise en place de la stratégie de choix de modèle, choix de méthode, en science des données, telle que cette [stratégie est décrite](#) dans l'introduction.

La recherche d'un meilleur modèle, qui sera déclinée ensuite pour chaque méthode d'apprentissage, conduit à optimiser un ou des paramètres contrôlant la flexibilité ou complexité du modèle. C'est facile à illustrer dans le cas de la régression (variable à prévoir quantitative) polynomiale (figure 1). Dans ce cas élémentaire, la *complexité du modèle* est le degré du polynôme ou encore le nombre de paramètres. En augmentant ce degré, l'erreur d'ajustement décroît jusqu'à s'annuler lorsque le degré est tel que les observations sont exactement ajustées par interpolation.

Intuitivement, si les données sont bruitées, le graphique montre que des prévisions réalisées en dehors des observations peuvent conduire à des valeurs très excentrées et donc à des erreurs quadratiques moyennes très élevées.

Le point important à souligner, et qui sera constamment rappelé, est que le "meilleur" modèle en un sens prédictif n'est pas nécessairement celui qui ajuste le mieux les données, ni même le "vrai" modèle si la variance des estimations est importante. L'objectif est la recherche d'un modèle optimal *parcimonieux* dont on verra qu'en régression il réalise un meilleur *compromis biais*

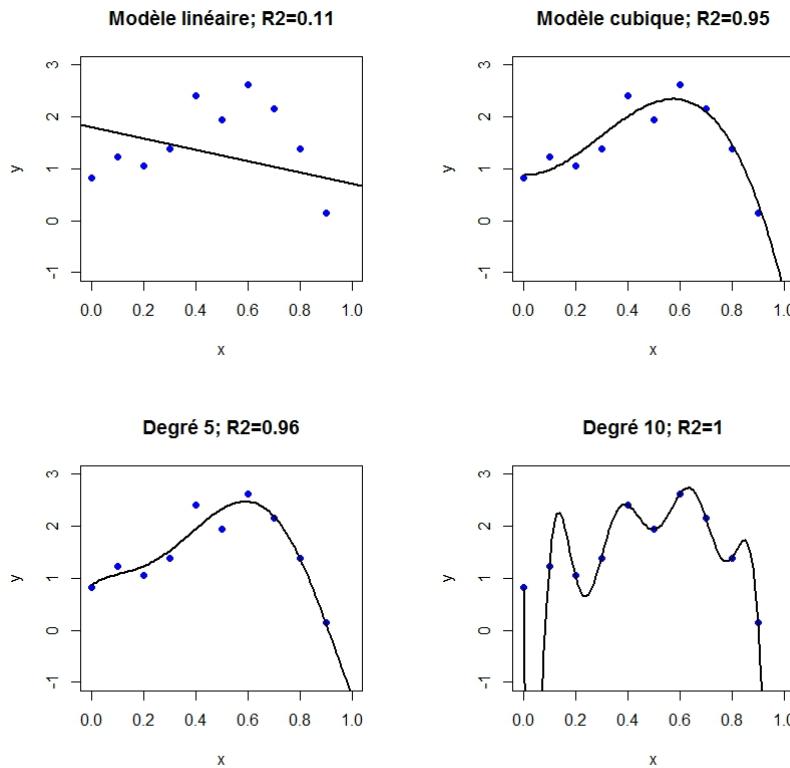


FIGURE 1 – Régression polynomiale : ajustement par successivement des polynômes de degré 1, 2, 5 et 10

/variance.

2 Risque, risque empirique

2.1 Modèle statistique

On suppose que D^n est l'observation d'un n -échantillon

$$D^n = \{(\mathbf{X}_1, Y_1), \dots, (\mathbf{X}_n, Y_n)\}$$

d'une loi conjointe P sur $\mathcal{X} \times \mathcal{Y}$, *inconnue*, et que x est une observation de la variable \mathbf{X} , (\mathbf{X}, Y) étant un couple aléatoire de loi conjointe P *indépendant* de D^n .

L'échantillon D^n est appelé *échantillon d'apprentissage*.

Une *règle de prévision, régression / discrimination* ou prédicteur est une fonction (mesurable) $f : \mathcal{X} \rightarrow \mathcal{Y}$ qui associe la sortie $f(x)$ à l'entrée $x \in \mathcal{X}$.

Pour mesurer la qualité de prévision, on introduit une fonction de perte :

DÉFINITION 1. — Une fonction (mesurable) $l : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}_+$ est une fonction de perte si $l(y, y) = 0$ et $l(y, y') > 0$ pour $y \neq y'$.

2.2 Risque

Si f est une règle de prévision, x une entrée, y la sortie qui lui est réellement associée, alors $l(y, f(x))$ mesure une perte encourue lorsque l'on associe à x la sortie $f(x)$.

En régression réelle : on définit les pertes \mathbb{L}^p ($p \geq 1$)

$$l(y, y') = |y - y'|^p.$$

Si $p = 1$ on parle de perte absolue, si $p = 2$ de perte quadratique.

En discrimination binaire : $\mathcal{Y} = \{-1, 1\}$

$$l(y, y') = \mathbb{1}_{y \neq y'} = \frac{|y - y'|}{2} = \frac{(y - y')^2}{4}.$$

On va s'intéresser au comportement moyen de cette fonction de perte, il s'agit de la notion de *risque* :

DÉFINITION 2. — Étant donnée une fonction de perte l , le risque – ou l’erreur de généralisation – d’une règle de prévision f est défini par

$$R_P(f) = \mathbb{E}_{(\mathbf{X}, Y) \sim P}[l(Y, f(\mathbf{X}))].$$

Une estimation \hat{f} de f sur un échantillon d’apprentissage \mathbf{D}^n est obtenue par un *algorithme de prévision*

DÉFINITION 3. — Un algorithme de prévision est représenté par une application (mesurable) $\hat{f} : (\mathcal{X} \times \mathcal{Y})^n \rightarrow \mathcal{F}$ qui à un ensemble d’apprentissage $\mathbf{d}^n = \{(\mathbf{x}_i, y_i), 1 \leq i \leq n\}$ associe une règle de prévision $\hat{f}(\mathbf{d}^n)$, ou par une suite $(\hat{f}_n)_{n \geq 1}$ d’applications (mesurables) telles que pour $n \geq 1$, $\hat{f}_n : (\mathcal{X} \times \mathcal{Y})^n \rightarrow \mathcal{F}$

DÉFINITION 4. — Le risque moyen d’un algorithme de prévision \hat{f} est défini par

$$\mathbb{E}_{\mathbf{D}^n \sim P^{\otimes n}}[R_P(\hat{f}(\mathbf{D}^n))].$$

Les travaux de Vapnik en théorie de l’apprentissage ont conduit à focaliser l’attention sur la présence ou l’absence de propriétés théoriques basiques d’une technique d’apprentissage ou de modélisation :

consistance qui garantit la capacité de généralisation. Un processus d’apprentissage est dit *consistant* si le risque empirique et un risque estimé sur un échantillon test indépendant convergent en probabilité vers la même limite lorsque les tailles de l’échantillon augmentent.

vitesse de convergence. Une évaluation, quand elle est possible, de la vitesse de convergence de l’estimation du risque lorsque la taille augmente, est une indication sur la façon dont la généralisation s’améliore et informe sur la nature des paramètres, comme le nombre de variables explicatives, dont elle dépend.

contrôle Est-il possible, à partir d’un échantillon d’apprentissage de taille fini donc sans considération asymptotique, de contrôler la capacité de généralisation et donc de majorer le terme de risque ?

Plus précisément :

DÉFINITION 5. — Un algorithme de prévision est dit universellement consistant si

$$\forall P \lim_{n \rightarrow +\infty} \left\{ \mathbb{E}_{\mathbf{D}^n \sim P^{\otimes n}}[R_P(\hat{f}_n(\mathbf{D}^n))] \right\} = \inf_{f \in \mathcal{F}} R_P(f).$$

On montre ainsi, par le théorème de Stone (1977)[9], que l’algorithme dit des *k plus proches voisins*, qui opère par moyennage locale, est universellement consistant. Il en est de même pour les *arbres de décision* (CART) (Breiman et al. 1984)[2]

2.3 Minimisation du risque empirique

Définitions

Le risque d’une règle de prévision f est défini par

$$R_P(f) = \mathbb{E}_{(\mathbf{X}, Y) \sim P}[l(Y, f(\mathbf{X}))]$$

qui dépend de P inconnue.

En l’absence de toute information ou hypothèse sur la loi P (cadre non paramétrique), il est naturel de remplacer P par P_n , mesure empirique associée à l’échantillon \mathbf{D}^n , et de minimiser le risque empirique.

DÉFINITION 6. — Le risque empirique (associé à $\mathbf{D}^n = \{(\mathbf{X}_i, Y_i), 1 \leq i \leq n\}$) d’une règle de prévision $f \in \mathcal{F}$ est défini par

$$\widehat{R}_n(f, \mathbf{D}^n) = \frac{1}{n} \sum_{i=1}^n l(Y_i, f(\mathbf{X}_i)).$$

La minimisation du risque empirique, qui est une extension de la procédure d’estimation d’un modèle, par exemple par les moindres carrés, a été développée par Vapnik (1999)[11].

DÉFINITION 7. — Étant donné un sous-ensemble F de \mathcal{F} (un modèle), l’algorithme de minimisation du risque empirique sur F est défini par :

$$\hat{f}_F(\mathbf{D}^n) \in \operatorname{argmin}_{f \in F} \widehat{R}_n(f, \mathbf{D}^n).$$

Le choix d’un modèle $f \in F$ adéquat est crucial et relève des méthodes de sélection de modèles.

Décomposition approximation/estimation (ou biais/variance)

Soit f^* telle que $R_P(f^*) = \inf_{f \in \mathcal{F}} R_P(f)$, f^* est appelé "oracle". L'objectif est de déterminer un modèle F pour lequel le risque de l'estimateur $\hat{f}_F(\mathbf{D}^n)$ est proche de celui de l'oracle.

$$R_P(\hat{f}_F(\mathbf{D}^n)) - R_P(f^*) = \underbrace{\left\{ R_P(\hat{f}_F(\mathbf{D}^n)) - \inf_{f \in F} R_P(f) \right\}}_{\text{Erreur d'estimation (Variance)}} + \underbrace{\left\{ \inf_{f \in F} R_P(f) - R_P(f^*) \right\}}_{\text{Erreur d'approximation (Biais)}} \nearrow \searrow (\text{taille de } F)$$

Ces deux termes sont de natures différentes. Pour les évaluer, nous aurons recours à des considérations issues respectivement de la statistique et de la théorie de l'approximation.

La sélection d'un modèle \hat{F} parmi une collection de modèles \mathcal{C} pour lequel le risque de $\hat{f}_{\hat{F}}(\mathbf{D}^n)$ est proche de celui de l'oracle va s'obtenir par la minimisation d'un critère pénalisé du type :

$$\hat{F} = \operatorname{argmin}_{F \in \mathcal{C}} \{ \hat{R}_n(\hat{f}_F(\mathbf{D}^n), \mathbf{D}^n) + \operatorname{pen}(F) \}.$$

La pénalité permet de pénaliser les modèles de "grande" taille, afin d'éviter le sur-ajustement. Le choix optimal de la pénalité (selon les modèles statistiques considérés) est un sujet de recherche très actif en statistique.

Très généralement, plus un modèle (la famille des fonctions admissibles) est complexe, plus il est flexible et peut s'ajuster aux données observées et donc plus le biais est réduit. En revanche, la partie variance augmente avec le nombre de paramètres à estimer et donc avec cette complexité. L'enjeu, pour minimiser le risque quadratique ainsi défini, est donc de rechercher un meilleur compromis entre biais et variance : accepter de biaiser l'estimation comme par exemple en régression ridge pour réduire plus favorablement la variance.

3 Estimations d'un risque

Le *risque empirique* défini ci-dessus exprime la qualité d'ajustement du modèle sur l'échantillon observé. C'est ce critère, moindres carrés en régression, taux d'erreur en discrimination, qui est généralement minimisé lors de l'estimation des paramètres d'un modèle. Il constitue également une mesure de la qualité, mais biaisée, car associée à une estimation par principe trop *optimiste*, de l'erreur de prévision. Celle-ci est liée aux données qui ont servi à l'ajustement du modèle et est d'autant plus faible que le modèle est complexe ; sélectionner la complexité d'un modèle en minimisant le risque empirique conduit au sur-apprentissage ou sur-ajustement. Cette estimation ne dépend que de la partie "biais" de l'erreur de prévision et ne prend pas en compte la partie "variance" de la décomposition.

L'estimation du risque empirique est obtenue par :

$$\widehat{R}_n(\hat{f}(D_n), D_n) = \frac{1}{n} \sum_{i=1}^n l(y_i, \hat{f}(D_n)(x_i)).$$

où l désigne une fonction perte adaptée au problème et D_n un échantillon de taille n .

Comme le risque empirique, le R^2 (coefficient de détermination de la régression) ne peut-être un "bon" critère de sélection de modèles ; il ne peut servir qu'à comparer des modèles de même dimension ou complexité car sinon conduit à sélectionner le modèle le plus complexe, c'est-à-dire celui correspond au plus grand espace de projection, ce qui entraîne un sur-ajustement. C'est simplement illustré dans le cas de la régression polynomiale. La figure 1 représente un jeu de données $Y_i = f(x_i) + \varepsilon_i$, $i = 1, \dots, n$ et $x_i \in [0, 1]$ ajusté par des polynômes de degrés croissants. Le critère R^2 augmente pour atteindre la valeur 1 pour le polynôme qui interpole toutes les observations. L'ajustement du modèle mesuré par la R^2 croît logiquement avec le nombre de paramètres.

L'estimation d'une erreur de prévision fait appel à différentes stratégies pour contrôler l'optimisme ou réduire le biais.

3.1 Estimation avec échantillons indépendants

La façon la plus simple d'estimer *sans biais* l'erreur de prévision ou un risque consiste à utiliser un échantillon indépendant n'ayant pas participé à

l'estimation du modèle. Ceci nécessite donc d'éclater aléatoirement l'échantillon en trois parties respectivement appelées *apprentissage*, *validation* et *test* :

$$D_n = D_{n_1}^{\text{Appr}} \cup D_{n_2}^{\text{Valid}} \cup D_{n_3}^{\text{Test}} \quad \text{avec} \quad \text{avec } n_1 + n_2 + n_3 = n.$$

1. $\widehat{R}_n(\widehat{f}(D_{n_1}^{\text{Appr}}), D_{n_1}^{\text{Appr}})$ est minimisé pour déterminer l'estimateur ou l'algorithme de prévision : $\widehat{f}(D_{n_1}^{\text{Appr}})$, pour modèle fixé; par exemple un modèle de régression polynomiale de degré fixé;
2. $\widehat{R}_n(\widehat{f}(D_{n_1}^{\text{Appr}}), D_{n_2}^{\text{Valid}})$ sert à la comparaison des modèles au sein d'une même famille afin de sélectionner celui qui minimise cette erreur; par exemple une famille de modèles polynomiaux de degrés variés.
3. $\widehat{R}_n(\widehat{f}, D_{n_3}^{\text{Test}})$ est utilisée pour comparer entre eux les meilleurs modèles de chacune des méthodes considérées; par exemple le meilleur modèle polynomial au meilleur réseau de neurones.

Cette solution n'est acceptable que si la taille de l'échantillon initiale est importante sinon la qualité d'ajustement est dégradée car n_1 est trop faible et la variance de l'estimation de l'erreur peut être importante (n_2, n_3 petits).

3.2 Estimation par pénalisation

La notion de sélection de modèle avec l'objectif d'une prévision est ancienne en Statistique et a donné lieu à de nombreux développements et critères bien adaptés à de petits échantillons. Sommairement, ceux-ci se présentent comme l'ajout d'une pénalisation de la fonction objectif du problème d'optimisation : moindres carrés ou risque empirique, vraisemblance.

C_p de Mallows

Le C_p de Mallows (1973)[6] fut, historiquement, le premier critère visant à une meilleure estimation de l'erreur de prévision que la seule considération de l'erreur d'ajustement (ou le R^2) dans le modèle linéaire. Il repose sur une mesure de la qualité sur la base d'un risque quadratique. L'erreur de prévision se décompose en :

$$\widehat{R}_P(\widehat{f}(D_n)) = \widehat{R}_n(\widehat{f}(D_n), D_n) + \text{Optim}$$

qui est le risque empirique plus une estimation du biais par abus d'optimisme. Il s'agit donc d'estimer cet optimisme pour apporter une correction et ainsi

une meilleure estimation de l'erreur recherchée. Cette correction peut prendre plusieurs formes. Elle est liée à l'estimation de la variance dans la décomposition en biais et variance du risque ou c'est encore une pénalisation associée à la complexité du modèle.

Son expression est détaillée dans le cas de la régression linéaire avec une fonction perte quadratique. On montre (cf. Hastie et al. 2009)[5], à des fins de comparaison qu'il peut aussi se mettre sous une forme équivalente :

$$C_p = \widehat{R}_n(\widehat{f}(D_n), D_n) + 2\frac{d}{n}\widehat{\sigma}^2$$

où d est le nombre de paramètres du modèles (nombre de variables plus un), n le nombre d'observations, $\widehat{\sigma}^2$ une estimation de la variance de l'erreur par un modèle de faible biais. Ce dernier point est fondamental pour la qualité du critère, il revient à supposer que le modèle complet (avec toutes les variables) est le "vrai" modèle ou tout du moins un modèle peu biaisé afin de conduire à une bonne estimation de σ^2 .

La figure 2 montre le comportement du C_p dans l'exemple trivial de la régression polynomiale. Ce critère décroît avec le biais jusqu'à un choix optimal de dimension 3 (figure 2) avant d'augmenter à nouveau approximativement linéairement avec la variance.

AIC, AIC_c, BIC

Contrairement au C_p associé à une fonction perte quadratique, le critère d'information d'Akaike (1974)[1] (AIC) découle d'une expression de la qualité du modèle basée sur la dissemblance de Kullback. Il se présente sous une forme similaire mais plus générale que le C_p de Mallows. Il s'applique en effet à tout modèle estimé par maximisation d'une log-vraisemblance L et suppose que la famille de densités considérée pour modéliser la loi de Y contient la "vraie" densité de Y .

Après quelques développements incluant de nombreuses approximations (estimation de paramètres par maximum de vraisemblance, propriétés asymptotiques, formule de Taylor), le critère d'Akaike se met sous la forme :

$$\text{AIC} = -2L + 2\frac{d}{n}$$

Dans le cas gaussien en supposant la variance connue, moindres carrés et dé-

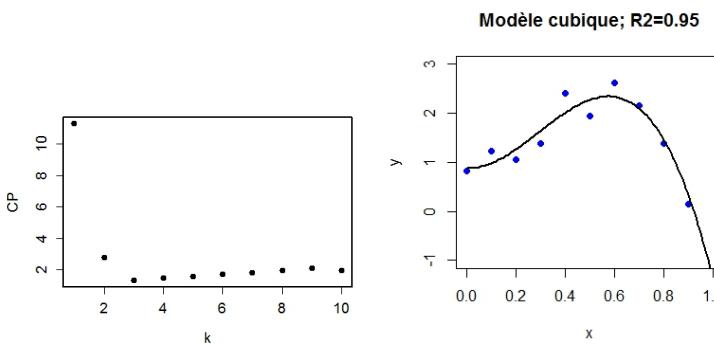


FIGURE 2 – Régression polynomiale : C_p de Mallows en fonction du degré du polynôme et modèle de degré 3 sélectionné.

viance coïncident, AIC est équivalent au C_p . Ce critère possède une version plus raffinée (AIC_c) dans le cas gaussien et plus particulièrement adaptée aux petits échantillons et asymptotiquement équivalente lorsque n est grand.

$$AIC = -2\mathcal{L} + \frac{n+d}{n-d-2}.$$

Une argumentation de type bayésien conduit à un autre critère BIC (*Bayesian Information Criterion* (Schwartz ; 1978)[8] qui cherche, approximativement (asymptotiquement), le modèle associé à la plus grande probabilité *a posteriori*. Dans le cas d'un modèle issu de la maximisation d'une log-vraisemblance, il se met sous la forme :

$$BIC = -2L + \log(n) \frac{d}{n}.$$

On montre, dans le cas gaussien et en supposant la variance connue que BIC est proportionnel à AIC avec le facteur 2 remplacé par $\log n$. Ainsi, dès que $n > e^2 \approx 7.4$, BIC tend à pénaliser plus lourdement les modèles complexes. Asymptotiquement, on montre que la probabilité pour BIC de choisir le bon modèle tend vers 1 lorsque n tend vers l'infini. Ce n'est pas le cas d'AIC ni du

C_p qui tendent alors à choisir des modèles trop complexes. Néanmoins à taille finie, petite, BIC risque de se limiter à des modèles trop simples.

Quelque soit le critère adopté, il suffit de choisir le modèle présentant le plus faible AIC, AIC_c ou BIC parmi ceux considérés.

3.3 Estimation par simulation

Plusieurs stratégies sont proposées pour estimer des erreurs de prévision sans biais ou de biais réduit en évitant d'utiliser les mêmes données pour estimer un modèle et une erreur. Les plus utilisées sont deux versions de validation croisée et le *bootstrap*. Ces stratégies diffèrent essentiellement par le choix de procédure permettant de séparer itérativement l'échantillon initial en parties *apprentissage* et *validation*. L'estimation du risque ou de l'erreur de prévision est itérée puis toutes les estimations moyennées avant d'en calculer la moyenne pour réduire la variance et améliorer la précision lorsque la taille de l'échantillon initial est trop réduite pour en extraire des échantillons de validation et test indépendants de taille suffisante.

Validation croisée en V segments

La *V -fold cross validation* ou validation croisée en V segments est décrite dans l'algorithme ci-dessous. Il consiste à partager aléatoirement l'échantillon en V segments puis, itérativement à faire jouer à chacun de ces segments le rôle d'échantillon de validation tandis que les $V-1$ autres constituent l'échantillon d'apprentissage servant à estimer le modèle.

Algorithm 1 Validation croisée en V segments

Découper aléatoirement l'échantillon en V parts (V -fold) de tailles approximativement égales selon une loi uniforme ;

for $k=1$ à V **do**

Mettre de côté l'une des partie,

Estimer le modèle sur les $V-1$ parties restantes,

Calculer l'erreur sur chacune des observations qui n'ont pas participé à l'estimation

end for

Moyenner toutes ces erreurs pour aboutir à l'estimation par validation croisée.

Plus précisément, soit $\tau : \{1, \dots, n\} \mapsto \{1, \dots, V\}$ la fonction d'indexation qui, pour chaque observation, donne l'attribution uniformément aléatoire de sa classe. L'estimation par *validation croisée* de l'erreur de prévision est :

$$\widehat{R}_{CV} = \frac{1}{n} \sum_{i=1}^n l(y_i, \widehat{f}^{(-\tau(i))}(\mathbf{x}_i))$$

où $\widehat{f}^{(-v)}$ désigne l'estimation de f sans prendre en compte la k -ième partie de l'échantillon.

Le choix de V entre 5 et 15, est couramment $V = 10$ par défaut dans les logiciels. Historiquement, la validation croisée a été introduite avec $V = n$ (*leave-one-out or "loo" cross validation* ou PRESS de Allen (1974)) en régression linéaire. Ce dernier choix n'est possible que pour n relativement petit à cause du volume des calculs nécessaires. D'autre part, l'estimation de l'erreur présente alors une variance importante car comme chaque couple de modèle partagent $(n - 2)$ observations, ceux-ci peuvent être très similaires donc très dépendants ; cette dépendance limite la réduction de variance issue du calcul de la moyenne. En revanche, si V est petit (*i.e.* $V = 5$), la variance sera plus faible mais le biais (pessimiste) devient un problème dépendant de la façon dont la qualité de l'estimation se dégrade avec la taille de l'échantillon. L'optimisation de V qui correspond donc encore à un meilleur équilibre entre biais et variance, nécessite généralement trop d'observations pour être pratiquée ; d'où le choix par défaut.

Minimiser l'erreur estimée par validation croisée est une approche largement utilisée pour optimiser le choix d'un modèle au sein d'une famille paramétrée. \widehat{f} est défini par

$$\widehat{\theta} = \arg \min_{\theta} \widehat{R}_{CV}(\theta)$$

Validation croisée Monte Carlo

Le deuxième principe de validation croisée consiste à itérer plusieurs fois la division de l'échantillon initial en une partie validation ou plutôt *test* et une partie *apprentissage*.

Cette stratégie de validation croisée est généralement couplée avec celle en V segments dans l'algorithme ci-dessous.

Algorithm 2 Validation croisée Monte Carlo

for $k=1$ à B **do**

 Séparer aléatoirement l'échantillon en deux parties : *test* et *apprentissage* selon une proportion à déterminer,

for méthode *in* liste de méthodes **do**

 Estimer le modèle de la méthode en cours sur l'échantillon d'apprentissage

 Optimiser les paramètres du modèle (complexité) par *Validation Croisée* à V segments.

 Calculer l'erreur sur la partie test de l'échantillon pour la méthode courante

end for

end for

Calculer pour chaque méthode la moyenne des B erreurs et tracer les graphes des distributions de ces erreurs (diagramme boîte).

La proportion entre échantillons : apprentissage et test ; dépend de la taille initial de l'échantillon afin de préserver une part suffisante à la qualité de l'apprentissage ; le nombre d'itérations dépend des moyens de calcul. Plus l'échantillon initial est réduit et moins les évaluations des erreurs sont "indépendantes" et donc la réduction de variance obtenue à l'issue de la moyenne.

Bootstrap

L'idée, d'approcher par simulation (*Monte Carlo*) la distribution d'un estimateur lorsque l'on ne connaît pas la loi de l'échantillon ou, plus souvent, lorsque l'on ne peut pas supposer qu'elle est gaussienne, est l'objectif même du *bootstrap* (Efron, 1982)[3].

Le principe fondamental de cette technique de ré-échantillonnage est de substituer, à la distribution de probabilité inconnue F , dont est issu l'échantillon d'apprentissage, la distribution empirique F_n qui donne un poids $1/n$ à chaque réalisation. Ainsi on obtient un échantillon de taille n dit *échantillon bootstrap* selon la distribution empirique F_n par n tirages aléatoires avec remise parmi les n observations initiales.

Il est facile de construire un grand nombre d'échantillons bootstrap (*e.g.*

$B = 100$) sur lesquels calculer l'estimateur concerné. La loi simulée de cet estimateur est une approximation asymptotiquement convergente sous des hypothèses raisonnables¹ de la loi de l'estimateur. Cette approximation fournit ainsi des estimations du biais, de la variance, donc d'un risque quadratique, et même des intervalles de confiance (avec B beaucoup plus grand) de l'estimateur sans hypothèse (normalité) sur la vraie loi. Les grands principes de cette approche sont rappelés dans l'annexe sur le [bootstrap](#).

Estimateur naïf

Soit \mathbf{z}^* un échantillon bootstrap (n tirages avec remise) des données tiré selon la loi empirique \hat{F} associée à l'échantillon d'apprentissage \mathbf{D}^n

$$\mathbf{z}^* = \{(\mathbf{x}_1^*, y_1^*), \dots, (\mathbf{x}_n^*, y_n^*)\}.$$

L'estimateur *plug-in* de l'erreur de prévision $R_P(\hat{f}(\mathbf{D}^n))$ est donné par :

$$\hat{R}_n(\hat{f}_{\mathbf{z}^*}, \mathbf{D}^n) = \frac{1}{n} \sum_{i=1}^n l(y_i, \hat{f}_{\mathbf{z}^*}(\mathbf{x}_i))$$

où $\hat{f}_{\mathbf{z}^*}$ désigne l'estimation de f à partir de l'échantillon bootstrap. Il conduit à l'estimation bootstrap de l'erreur moyenne de prévision $\mathbb{E}_{\mathbf{D}^n \sim P^{\otimes n}}[R_P(\hat{f}(\mathbf{D}^n))]$ par

$$R_{\text{Boot}} = E_{\mathbf{Z}^* \sim \hat{F}}[\hat{R}_n(\hat{f}_{\mathbf{z}^*}, \mathbf{D}^n)] = E_{\mathbf{Z}^* \sim \hat{F}} \left[\frac{1}{n} \sum_{i=1}^n l(y_i, f_{\mathbf{Z}^*}(\mathbf{x}_i)) \right].$$

Cette estimation est approchée par simulation :

$$\widehat{R}_{\text{Boot}} = \frac{1}{B} \sum_{b=1}^B \frac{1}{n} \sum_{i=1}^n l(y_i, f_{\mathbf{z}^{*b}}(\mathbf{x}_i)).$$

L'estimation ainsi construite de l'erreur de prévision est généralement biaisée par optimisme car, au gré des simulations, les mêmes observations (\mathbf{x}_i, y_i) apparaissent à la fois dans l'estimation du modèle et dans celle de l'erreur. D'autres approches visent à corriger ce biais.

1. Échantillon indépendant de même loi et estimateur indépendant de l'ordre des observations.

Estimateur *out-of-bag*

La première s'inspire simplement de la validation croisée. Elle considère d'une part les observations tirées dans l'échantillon bootstrap et, d'autre part, celles qui sont laissées de côté pour l'estimation du modèle mais retenue pour l'estimation de l'erreur.

$$\widehat{R}_{\text{oob}} = \frac{1}{n} \sum_{i=1}^n \frac{1}{B_i} \sum_{b \in K_i} l(y_i, f_{\mathbf{z}^{*b}}(\mathbf{x}_i))$$

où K_i est l'ensemble des indices b des échantillons bootstrap ne contenant pas la i ème observation à l'issue des B simulations et $B_i = |K_i|$ le nombre de ces échantillons ; B doit être suffisamment grand pour que toute observation n'ait pas été tirée au moins une fois ou bien les termes avec $K_i = 0$ sont supprimés.

L'estimation \widehat{R}_{oob} résout le problème d'un biais optimiste auquel est confrontée $\widehat{R}_{\text{Boot}}$ mais n'échappe pas au biais introduit pas la réduction tel qu'il est signalé pour l'estimation pas validation croisée \widehat{R}_{CV} . C'est ce qui a conduit Efron et Tibshirani (1997)[4] à proposer des correctifs.

Estimateur .632-bootstrap

La probabilité qu'une observation soit tirée dans un échantillon bootstrap est

$$P[\mathbf{x}_i \in \mathbf{x}^{*b}] = 1 - (1 - \frac{1}{n})^n \approx 1 - \frac{1}{e} \approx 0,632.$$

Très approximativement, la dégradation de l'estimation provoquée par le bootstrap et donc la surévaluation de l'erreur sont analogues à celle de la validation croisée avec $K = 2$. À la suite d'un raisonnement trop long pour être reproduit ici, Efron et Tibshirani (1997) proposent de compenser : excès d'optimisme du taux apparent d'erreur et excès de pessimisme du bootstrap *out-of-bag*, par une combinaison :

$$\widehat{R}_{.632} = 0,368 \times \widehat{R}_n(\hat{f}(\mathbf{D}^n), \mathbf{D}^n) + 0,632 \times \widehat{R}_{\text{oob}}.$$

Remarques

- Toutes les estimations du risque empirique considérées (pénalisation, validation croisée, bootstrap) sont asymptotiquement équivalentes et il

n'est pas possible de savoir laquelle concrètement sera, à n fini, la plus précise. Une large part d'arbitraire ou d'"expérience" préside donc le choix d'une estimation plutôt qu'une autre.

- Conceptuellement, le bootstrap est plus compliqué et pratiquement encore peu utilisé. Néanmoins, cet outil joue un rôle central dans les algorithmes récents de [combinaison de modèles](#) en association avec une estimation *out-of-bag* de l'erreur. Il ne peut être négligé.
- L'estimateur .632-bootstrap pose des problèmes en situation de surajustement aussi les mêmes auteurs ont proposé un rectificatif complémentaire noté .632+bootstrap.

En conclusion, l'estimation d'une erreur de prévision est une opération délicate aux conséquences importantes. Il est donc nécessaire

- d'utiliser le *même estimateur* pour comparer l'efficacité de deux méthodes,
- de se montrer très prudent, en dehors de tout système d'hypothèses probabilistes, sur le caractère absolu d'une estimation des erreurs.

3.4 Discrimination et courbe ROC

Dans une situation de discrimination le seul critère de risque ,comme le taux d'erreur de classement, n'est pas toujours bien adapté surtout, par exemple, dans le cadre de classes déséquilibrées : un modèle trivial qui ne prédit jamais une classe peu représentée ne commet pas un taux d'erreur supérieur au pourcentage de cette classe. Cette situation est souvent délicate à gérer et nécessite une pondérations des observations ou encore l'introduction de coûts de mauvais classement dissymétriques afin de forcer le modèle à prendre en compte une petite classe.

Discrimination à deux classes

Dans le cas fréquent de la discrimination de deux classes, plusieurs critères sont proposés afin d'évaluer précisément une qualité de discrimination. La plupart des méthodes (*e.g* régression logistique) estiment, pour chaque individu i , un *score* ou une probabilité $\hat{\pi}_i$ que cette individu prenne la modalité $Y = 1$ (ou succès, ou possession d'un actif, ou présence d'une maladie...). Cette probabilité ou ce score compris entre 0 et 1 est comparé avec une valeur seuil s fixée

a priori (en général 0,5) :

$$\text{Si } \hat{\pi}_i > s, \hat{y}_i = 1 \text{ sinon } \hat{y}_i = 0.$$

Matrice de confusion

Pour un échantillon de taille n dont l'observation de Y est connue ainsi que les scores $\hat{\pi}_i$ fournis par un modèle, il est alors facile de construire la matrice dite de *confusion* croisant les modalités de la variable prédictive pour une valeur de seuil s avec celles de la variable observée dans une table de contingence :

Prévision	Observation		Total
	$Y = 1$	$Y = 0$	
$\hat{y}_i = 1$	$n_{11}(s)$	$n_{10}(s)$	$n_{1+}(s)$
$\hat{y}_i = 0$	$n_{01}(s)$	$n_{00}(s)$	$n_{0+}(s)$
Total	n_{+1}	n_{+0}	n

Dans une situation classique de diagnostic médical ou en marketing les quantités suivantes sont considérées :

- Vrais positifs les $n_{11}(s)$ observations bien classées ($\hat{y}_i = 1$ et $Y = 1$),
- Vrais négatifs les $n_{00}(s)$ observations bien classées ($\hat{y}_i = 0$ et $Y = 0$),
- Faux négatifs les $n_{01}(s)$ observations mal classées ($\hat{y}_i = 0$ et $Y = 1$),
- Faux positifs les $n_{10}(s)$ observations mal classées ($\hat{y}_i = 1$ et $Y = 0$),
- Le taux d'erreur : $t(s) = \frac{n_{01}(s)+n_{10}(s)}{n}$,
- Le taux de vrais positifs ou *sensibilité* = $\frac{n_{11}(s)}{n_{+1}}$ ou taux de positifs pour les individus qui le sont effectivement,
- Le taux de vrais négatifs ou *spécificité* = $\frac{n_{00}(s)}{n_{+0}}$ ou taux de négatifs pour les individus qui le sont effectivement,
- Le taux de faux positifs = $1 - \text{Spécificité} = 1 - \frac{n_{00}(s)}{n_{+0}} = \frac{n_{10}(s)}{n_{+0}}$.

Courbe ROC et AUC

Les notions de *spécificité* et de *sensibilité* proviennent de la théorie du signal ; leurs valeurs dépendent directement de celle du seuil s . En augmentant s , la sensibilité diminue tandis que la spécificité augmente car la règle de décision devient plus exigeante ; un bon modèle associe grande sensibilité et grande spécificité pour la détection d'un signal. Ce lien est représenté graphiquement par

la courbe ROC (Receiver Operating Characteristic) de la sensibilité (probabilité de détecter un vrai signal) en fonction de un moins la spécificité (probabilité de détecter un signal à tort) pour chaque valeur s du seuil. Notons que la courbe ROC est une fonction monotone croissante :

$$1 - \frac{n_{00}(s)}{n_{+0}} < 1 - \frac{n_{00}(s')}{n_{+0}} \Rightarrow s < s' \Rightarrow \frac{n_{11}(s)}{n_{+1}} < \frac{n_{11}(s')}{n_{+1}}.$$

Plus la courbe (figure 3) se rapproche du carré, meilleure est la discrimination, correspondant à la fois à une forte sensibilité et une grande spécificité. L'aire sous la courbe : AUC (*area under curve*) mesure la qualité de discrimination du modèle tandis qu'une analyse de la courbe aide au choix du seuil.

L'aire sous la courbe est calculée en considérant toutes les paires (i, i') formées d'un premier individu avec $y_i = 1$ et d'un second avec $y_{i'} = 0$. Une paire est dite concordante si $\hat{\pi}_i > \hat{\pi}_{i'}$; discordante sinon. Le nombre d'*ex aequo* est $n_{+0}n_{+1} - n_c - n_d$ où n_c est le nombre de paires concordantes et n_d le nombre de paires discordantes. Alors,

$$\text{AUC} = \frac{n_c + 0,5(n_{+0}n_{+1} - n_c - n_d)}{n_{+0}n_{+1}}.$$

On montre par ailleurs (voir par exemple Tennenhaus (2007)[10]) que le numérateur de cette expression est encore la Statistique de test de Mann-Whitney tandis que le coefficient de Gini, qui est le double de la surface entre la diagonale et la courbe, vaut $2 \times \text{AUC} - 1$.

Pour comparer des modèles ou méthodes de complexités différentes, ces courbes doivent être estimées sur un échantillon test. Elles sont bien évidemment optimistes sur l'échantillon d'apprentissage. De plus, l'AUC ne définit pas un ordre total entre modèles car les courbes ROC peuvent se croiser.

Détection du dépassement de seuil d'ozone, optimisation par sélection

Deux modèles de régression logistique sont optimisés pour estimer la probabilité de dépasser le seuil. Le premier est linéaire tandis que le 2ème, faisant intervenir les interactions, comme dans le cas de l'ANCOVA, est quadratique. Le même procédure ascendante est utilisée par minimisation de l'AIC. Sur un échantillon test de 209 observations, avec un seuil fixé à 0.5, les taux d'erreur sont respectivement de 12.4, 8.6 et 25% pour les régressions logistiques

linéaire et quadratique et enfin pour le modèle MOCAGE. Les résultats sont précisés avec les courbes ROC de la figure 3 également calculées sur l'échantillon test.

Ces résultats montrent encore plus clairement l'intérêt de l'adaptation statistique de la prévision MOCAGE mais aussi la difficulté de la décision qui découle de la courbe ROC. Le choix du seuil, et donc de la méthode à utiliser si les courbes se croisent, dépend d'un choix dans ce cas politique : quel est le taux de faux positifs acceptable d'un point de vue économique ou le taux de vrais positifs à atteindre pour des raisons de santé publique ? Le problème majeur est de quantifier les coûts afférents, par la définition d'une matrice dissymétrique de ces coûts de mauvais classement.

Autre critère pour la discrimination à deux classes

Cette situation illustre bien que le taux d'erreur de classement n'est pas toujours adapté à une situation surtout dans le cas de classes très déséquilibrées. Un modèle trivial qui ne prédit jamais une classe peu représentée ne commet pas un taux d'erreur supérieur au pourcentage de cette classe.

D'autres critères ont été proposés pour intégrer cette difficulté dont le *Score de Pierce* basés sur le taux de bonnes prévisions : $H = \frac{n_{11}(s)}{n_{+1}(s)}$ et le taux de fausses alertes : $F = \frac{n_{10}(s)}{n_{+0}}$. Le score de Pierce est alors défini par $\text{PSS} = H - F$ et est compris entre -1 et 1 . Il évalue la qualité de la prévision. Si ce score est supérieur à 0 , le taux de bonnes prévisions est supérieur à celui des fausses alertes et plus il est proche de 1 , meilleur est le modèle.

Le score de Pierce a été conçu pour la prévision d'événements climatiques rares afin de pénaliser les modèles ne prévoyant jamais ces événements ($H = 0$) ou encore générant trop de fausses alertes ($F = 1$). Le modèle idéal prévoyant tous les événements critiques ($H = 1$) sans fausse alerte ($F = 0$). Des coûts de mauvais classement peuvent être introduits pour pondérer ce score.

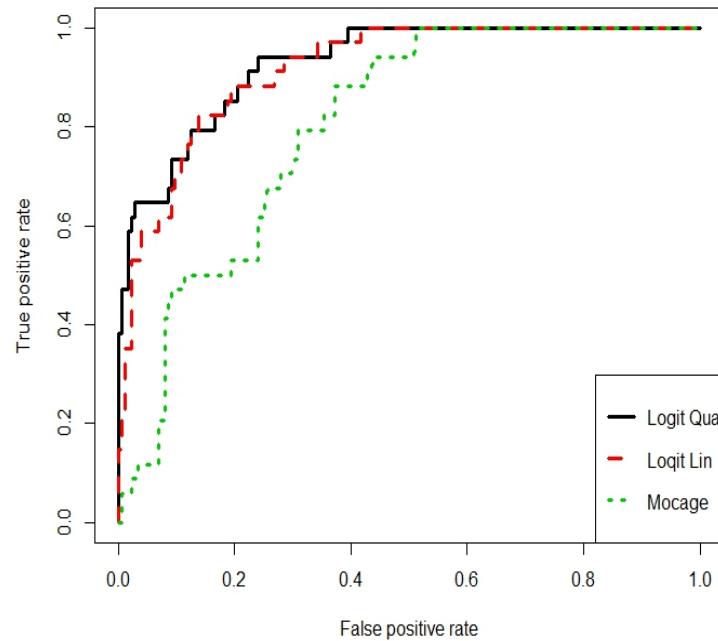


FIGURE 3 – Ozone : Courbe ROC dans le cas de l’ANCOVA quadratique, de la régression logistique quadratique et du modèle MOCAGE.

Log loss

Ce critère est mentionné ici car souvent utilisé comme score de référence dans les concours de prévision pour départager les concurrents. Il est défini par

$$L1 = -\frac{1}{n} \sum_{i=1}^n \sum_{k=1}^L y_{ik} \log(\hat{\pi}_{ik})$$

où y_{ik} est l’indicatrice qui vaut 1 si le i ème individu prend la modalité k , 0 sinon, et $\hat{\pi}_{ik}$ est la probabilité estimée par le modèle que l’individu i prenne la k ème modalité de Y . Comme pour la définition de l’entropie, la convention $0 \log(0) = 0$ est adoptée. Pour chaque observation, seule le terme de la bonne classe de Y est comptabilisé dans la somme. Ce critère est minoré par 0 si toutes les observations sont bien classées avec une probabilité 1 tandis qu’un avis neutre conduit à une valeur de $\log(L)$ alors qu’une confiance exagérée dans une mauvaise classe à pour conséquence de faire exploser ce critère qui n’est pas borné. Cette propriété incite les participants d’un concours à la prudence en pondérant les estimations des probabilités par lissage dit de Laplace ou additif (cf. Manning et al. 2008)[7] p60).

Contrairement à la courbe ROC, le *Log loss* est un critère qui s’adapte à la mesure d’un risque lorsque la variables à modéliser est qualitative avec plus de deux modalités. Un autre critère utilisable dans ce cas est le risque bayésien ; il est présenté dans la vignette sur l’[analyse discriminante](#).

Références

- [1] H. Akaike, *A new look at the statistical model identification*, IEEE Transactions on Automatic Control **19** (1974).
- [2] L. Breiman, J. Friedman, R. Olshen et C. Stone, *Classification and regression trees*, Wadsworth & Brooks, 1984.
- [3] B. Efron, *The Jackknife, the Bootstrap and other Resampling Methods*, SIAM, 1982.
- [4] B. Efron et R. Tibshirani, *Improvements on Cross-Validation : The .632+ Bootstrap Method*, Journal of the American Statistical Association **92** (1997), n° 438, 548–560.

- [5] T. Hastie, R. Tibshirani et J Friedman, *The elements of statistical learning : data mining, inference, and prediction*, Springer, 2009, Second edition.
- [6] C.L. Mallows, *Some Comments on Cp*, Technometrics **15** (1973), 661–675.
- [7] C. Manning, P. Raghavan et H. Schütze, *Introduction to information retrieval*, Cambridge University Press, 2008.
- [8] G. Schwarz, *Estimating the dimension of a model*, Annals of Statistics **6** (1978), 461–464.
- [9] M. Stone, *An Asymptotic Equivalence of Choice of Model by Cross-Validation and Akaike's Criterion*, Journal of The Royal Statistical Society B **39** (1977), 44–47.
- [10] M. Tenenhaus, *Statistique : méthodes pour décrire, expliquer et prévoir*, Dunod, 2007.
- [11] V.N. Vapnik, *Statistical learning theory*, Wiley Inter science, 1999.

Sélection de modèle en régression linéaire

Résumé

Le modèle linéaire gaussien ou régression multiple est considéré avec pour objectif la prévision d'une variable quantitative par un ensemble de variables quantitatives ou un mélange de quantitatives et qualitatives (analyse de covariance). Recherche d'un modèle parcimonieux assurant un bon équilibre entre la qualité de l'ajustement et la variance des paramètres afin de minimiser le risque empirique. Algorithmes (backward, forward, stepwise...) de sélection de modèle par sélection de variables et minimisation de critères pénalisés (C_p , AIC, BIC). Algorithmes de sélection de modèle par pénalisation ridge, Lasso, elastic net.

[Retour à l'introduction.](#)

Tous les tutoriels sont disponibles sur le dépôt :

github.com/wikistat

1 Régression multiple

Les modèles classiques de régression (linéaire, logistique) sont anciens et moins l'occasion de battage médiatique que ceux récents issus de l'apprentissage machine. Néanmoins, compte tenu de leur robustesse, de leur stabilité face à des fluctuations des échantillons, de leur capacité à passer à l'échelle des données massives... tout ceci fait qu'ils restent toujours très utilisés en production notamment lorsque la fonction à modéliser est bien linéaire et qu'il serait contre productif de chercher plus compliqué.

1.1 Modèle

Une variable quantitative \mathbf{Y} dite à expliquer (ou encore, réponse, exogène, dépendante) est mise en relation avec p variables quantitatives $\mathbf{X}^1, \dots, \mathbf{X}^p$ dites explicatives (ou encore de contrôle, endogènes, indépendantes, régresseurs, prédicteurs).

Les données sont supposées provenir de l'observation d'un échantillon statistique de taille n ($n > p + 1$) de $\mathbb{R}^{(p+1)}$:

$$(x_i^1, \dots, x_i^j, \dots, x_i^p, y_i) \quad i = 1, \dots, n.$$

L'écriture du *modèle linéaire* dans cette situation conduit à supposer que l'espérance de \mathbf{Y} appartient au sous-espace de \mathbb{R}^n engendré par $\{\mathbf{1}, \mathbf{X}^1, \dots, \mathbf{X}^p\}$ où $\mathbf{1}$ désigne le vecteur de \mathbb{R}^n constitué de 1s. C'est-à-dire que les $(p + 1)$ variables aléatoires vérifient :

$$Y_i = \beta_0 + \beta_1 X_i^1 + \beta_2 X_i^2 + \dots + \beta_p X_i^p + \varepsilon_i \quad i = 1, 2, \dots, n$$

avec les hypothèses suivantes :

1. Les ε_i sont des termes d'erreur indépendants et identiquement distribués ; $E(\varepsilon_i) = 0$, $Var(\varepsilon) = \sigma^2 \mathbf{I}$.
2. Les termes \mathbf{X}^j sont supposés déterministes (facteurs contrôlés) **ou bien** l'erreur ε est indépendante de la distribution conjointe de $\mathbf{X}^1, \dots, \mathbf{X}^p$. On écrit dans ce dernier cas que :
 $E(\mathbf{Y}|\mathbf{X}^1, \dots, \mathbf{X}^p) = \beta_0 + \beta_1 \mathbf{X}^1 + \beta_2 \mathbf{X}^2 + \dots + \beta_p \mathbf{X}^p$ et $Var(\mathbf{Y}|\mathbf{X}^1, \dots, \mathbf{X}^p) = \sigma^2$.
3. Les paramètres inconnus β_0, \dots, β_p sont supposés constants.
4. En option, pour l'étude spécifique des lois des estimateurs, une quatrième hypothèse considère la normalité de la variable d'erreur ε ($\mathcal{N}(0, \sigma^2 \mathbf{I})$). Les ε_i sont alors i.i.d. de loi $\mathcal{N}(0, \sigma^2)$.

Les données sont rangées dans une matrice $\mathbf{X}(n \times (p + 1))$ de terme général X_i^j , dont la première colonne contient le vecteur $\mathbf{1}$ ($X_0^i = 1$), et dans un vecteur \mathbf{Y} de terme général Y_i . En notant les vecteurs $\varepsilon = [\varepsilon_1 \dots \varepsilon_p]'$ et $\beta = [\beta_0 \beta_1 \dots \beta_p]'$, le modèle s'écrit matriciellement :

$$\mathbf{Y} = \mathbf{X}\beta + \varepsilon.$$

1.2 Estimation

Conditionnellement à la connaissance des valeurs des \mathbf{X}^j , les paramètres inconnus du modèle : le vecteur β et σ^2 (paramètre de nuisance), sont estimés par minimisation des carrés des écarts (M.C.) ou encore, en supposant (4.), par maximisation de la vraisemblance (M.V.). Les estimateurs ont alors les mêmes expressions, l'hypothèse de normalité et l'utilisation de la vraisemblance conférant à ces derniers des propriétés complémentaires.

1.3 Estimation par moindres carrés

L'expression à minimiser sur $\beta \in \mathbb{R}^{p+1}$ s'écrit :

$$\begin{aligned}\sum_{i=1}^n (Y_i - \beta_0 - \beta_1 X_i^1 - \cdots - \beta_p X_i^p)^2 &= \|\mathbf{Y} - \mathbf{X}\beta\|^2 \\ &= \mathbf{Y}'\mathbf{Y} - 2\beta'\mathbf{X}'\mathbf{Y} + \beta'\mathbf{X}'\mathbf{X}\beta.\end{aligned}$$

Par dérivation matricielle de la dernière équation on obtient les *équations normales* :

$$\mathbf{X}'\mathbf{Y} - \mathbf{X}'\mathbf{X}\beta = 0$$

dont la solution correspond bien à un minimum car la matrice hessienne $2\mathbf{X}'\mathbf{X}$ est semi définie-positive.

Nous faisons l'hypothèse supplémentaire que la matrice $\mathbf{X}'\mathbf{X}$ est inversible, c'est-à-dire que la matrice \mathbf{X} est de rang $(p+1)$ et donc qu'il n'existe pas de colinéarité entre ses colonnes. Si cette hypothèse n'est pas vérifiée, il suffit en principe de supprimer des colonnes de \mathbf{X} et donc des variables du modèle. Une approche de réduction de dimension (régression *ridge*, Lasso, PLS ...) est à mettre en œuvre.

Alors, l'estimation des paramètres β_j est donnée par :

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

et les valeurs ajustées (ou estimées, prédictes) de \mathbf{Y} ont pour expression :

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

où $\mathbf{H} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$ (*hat matrix*). Géométriquement, c'est la matrice de projection orthogonale dans \mathbb{R}^n sur le sous-espace $\text{Vect}(\mathbf{X})$ engendré par les vecteurs colonnes de \mathbf{X} .

On note

$$\mathbf{e} = \mathbf{Y} - \hat{\mathbf{Y}} = \mathbf{Y} - \mathbf{X}\hat{\beta} = (\mathbf{I} - \mathbf{H})\mathbf{Y}$$

le vecteur des résidus ; c'est la projection de \mathbf{Y} sur le sous-espace orthogonal de $\text{Vect}(\mathbf{X})$ dans \mathbb{R}^n .

1.4 Propriétés

Les estimateurs des M.C. $\hat{\beta}_0, \hat{\beta}_1, \dots, \hat{\beta}_p$ sont des estimateurs sans biais : $E(\hat{\beta}) = \beta$, et, parmi les estimateurs sans biais fonctions linéaires des Y_i , ils sont de variance minimum (théorème de Gauss-Markov) ; ils sont donc BLUE : *best linear unbiased estimators*. Sous hypothèse de normalité, les estimateurs du M.V. sont uniformément meilleurs (efficaces) et coïncident avec ceux des moindres carrés.

On montre que la matrice de covariance des estimateurs se met sous la forme

$$E[(\hat{\beta} - \beta)(\hat{\beta} - \beta)'] = \sigma^2(\mathbf{X}'\mathbf{X})^{-1},$$

celle des prédicteurs est

$$E[(\hat{\mathbf{Y}} - \mathbf{X}\beta)(\hat{\mathbf{Y}} - \mathbf{X}\beta)'] = \sigma^2\mathbf{H}$$

et celle des estimateurs des résidus est

$$E[\mathbf{e}\mathbf{e}'] = \sigma^2(\mathbf{I} - \mathbf{H})$$

tandis qu'un estimateur sans biais de σ^2 est fourni par :

$$\hat{\sigma}^2 = \frac{\|\mathbf{e}\|^2}{n-p-1} = \frac{\|\mathbf{Y} - \mathbf{X}\beta\|^2}{n-p-1} = \frac{\text{SSE}}{n-p-1}.$$

Ainsi, les termes $\hat{\sigma}^2 h_i^2$ sont des estimations des variances des prédicteurs \hat{Y}_i .

Conséquence importante : si la matrice $\mathbf{X}'\mathbf{X}$ est mal conditionnée (déterminant proche de 0), son inversion fait apparaître des termes très élevés sur la diagonale et conduit donc à des variances très importantes des estimations des paramètres.

1.5 Sommes des carrés

SSE est la somme des carrés des résidus (*sum of squared errors*),

$$\text{SSE} = \|\mathbf{Y} - \hat{\mathbf{Y}}\|^2 = \|\mathbf{e}\|^2.$$

On définit également la somme totale des carrés (*total sum of squares*) par

$$\text{SST} = \|\mathbf{Y} - \bar{\mathbf{Y}}\mathbf{1}\|^2 = \mathbf{Y}'\mathbf{Y} - n\bar{\mathbf{Y}}^2$$

et la somme des carrés de la régression (*regression sum of squares*) par

$$\text{SSR} = \|\widehat{\mathbf{Y}} - \bar{\mathbf{Y}}\mathbf{1}\|^2 = \widehat{\mathbf{Y}}'\widehat{\mathbf{Y}} - n\bar{\mathbf{Y}}^2 = \mathbf{Y}'\mathbf{H}\mathbf{Y} - n\bar{\mathbf{Y}}^2 = \widehat{\boldsymbol{\beta}}'\mathbf{X}'\mathbf{Y} - n\bar{\mathbf{Y}}^2.$$

On vérifie alors : $\text{SST} = \text{SSR} + \text{SSE}$.

1.6 Coefficient de détermination

On appelle *coefficient de détermination* le rapport

$$R^2 = \frac{\text{SSR}}{\text{SST}}$$

qui est donc la part de variation de \mathbf{Y} expliquée par le modèle de régression. Géométriquement, c'est un rapport de carrés de longueur de deux vecteurs. C'est donc le cosinus carré de l'angle entre ces vecteurs : \mathbf{Y} et sa projection $\widehat{\mathbf{Y}}$ sur $\text{Vect}(\mathbf{X})$.

La quantité R est appelée *coefficient de corrélation multiple* entre \mathbf{Y} et les variables explicatives, c'est le coefficient de corrélation usuel entre \mathbf{Y} et sa prévision $\widehat{\mathbf{Y}}$.

Par construction, le coefficient de détermination croît avec le nombre p de variables.

1.7 Inférence dans le cas gaussien

En principe, l'hypothèse optionnelle (4.) de normalité des erreurs est nécessaire pour cette section. En pratique, des résultats asymptotiques, donc valides pour de grands échantillons, ainsi que des études de simulation, montrent que cette hypothèse n'est pas celle dont la violation est la plus pénalisante pour la fiabilité des modèles.

Inférence sur les coefficients

Pour chaque coefficient β_j on note $\widehat{\sigma}_j^2$ l'estimateur de la variance de β_j obtenu en prenant j -ème terme diagonal de la matrice $\widehat{\sigma}^2(\mathbf{X}'\mathbf{X})^{-1}$. On montre que la statistique

$$\frac{\widehat{\beta}_j - \beta_j}{\widehat{\sigma}_j}$$

suit une loi de Student à $(n - p - 1)$ degrés de liberté. Cette statistique est donc utilisée pour tester une hypothèse $H_0 : \beta_j = a$ ou pour construire un intervalle de confiance de niveau $100(1 - \alpha)\%$:

$$\widehat{\beta}_j \pm t_{\alpha/2; (n-p-1)} \widehat{\sigma}_j^2.$$

Attention, cette statistique concerne un coefficient et ne permet pas d'inférer conjointement sur d'autres coefficients car leurs estimateurs sont corrélés. De plus elle dépend des absences ou présences des autres variables \mathbf{X}^k dans le modèle. Par exemple, dans le cas particulier de deux variables \mathbf{X}^1 et \mathbf{X}^2 très corrélées, chaque variable, en l'absence de l'autre, peut apparaître avec un coefficient significativement différent de 0 ; mais, si les deux sont présentes dans le modèle, l'une peut apparaître avec un coefficient insignifiant.

De façon plus générale, si \mathbf{c} désigne un vecteur non nul de $(p+1)$ constantes réelles, il est possible de tester la valeur d'une combinaison linéaire $\mathbf{c}'\boldsymbol{\beta}$ des paramètres en considérant l'hypothèse nulle $H_0 : \mathbf{c}'\boldsymbol{\beta} = a$; a connu. Sous H_0 , la statistique

$$\frac{\mathbf{c}'\widehat{\boldsymbol{\beta}} - a}{(\widehat{\sigma}^2 \mathbf{c}'(\mathbf{X}'\mathbf{X})^{-1} \mathbf{c})^{1/2}}$$

suit une loi de Student à $(n - p - 1)$ degrés de liberté.

Inférence sur le modèle

Le modèle peut être testé globalement. Sous l'hypothèse nulle $H_0 : \beta_1 = \beta_2 = \dots = \beta_p = 0$, la statistique

$$\frac{\text{SSR}/p}{\text{SSE}/(n - p - 1)} = \frac{\text{MSR}}{\text{MSE}}$$

suit une loi de Fisher avec p et $(n - p - 1)$ degrés de liberté. Les résultats sont habituellement présentés dans un tableau *d'analyse de la variance* sous la forme suivante :

Source de variation	d.d.l.	Somme des carrés	Variance	F
Régression	p	SSR	$\text{MSR} = \text{SSR}/p$	MSR/MSE
Erreur	$n - p - 1$	SSE	$\text{MSE} = \text{SSE}/(n - p - 1)$	
Total	$n - 1$	SST		

Inférence sur un modèle réduit

Le test précédent amène à rejeter H_0 dès que l'une des variables \mathbf{X}^j est liée à \mathbf{Y} . Il est donc d'un intérêt limité. Il est souvent plus utile de tester un modèle réduit c'est-à-dire dans lequel certains coefficients, à l'exception de la constante, sont nuls contre le modèle complet avec toutes les variables. En ayant éventuellement réordonné les variables, on considère l'hypothèse nulle $H_0 : \beta_1 = \beta_2 = \dots = \beta_q = 0, q < p$.

Notons respectivement SSR_q , SSE_q , R_q^2 les sommes de carrés et le coefficient de détermination du modèle réduit à $(p - q)$ variables. Sous H_0 , la statistique

$$\frac{(\text{SSR} - \text{SSR}_q)/q}{\text{SSE}/(n - p - 1)} = \frac{(R^2 - R_q^2)/q}{(1 - R^2)/(n - p - 1)}$$

suit une loi de Fisher à q et $(n - p - 1)$ degrés de liberté.

Dans le cas particulier où $q = 1 (\beta_j = 0)$, la F -statistique est alors le carré de la t -statistique de l'inférence sur un paramètre et conduit donc au même test.

1.8 Prévision

Connaissant les valeurs des variables \mathbf{X}^j pour une nouvelle observation : $\mathbf{x}'_0 = [x_0^1, x_0^2, \dots, x_0^p]$ appartenant au domaine dans lequel l'hypothèse de linéarité reste valide, une prévision, notée \hat{y}_0 de \mathbf{Y} ou $E(\mathbf{Y})$ est donnée par :

$$\hat{y}_0 = \hat{\beta}_0 + \hat{\beta}_1 x_0^1 + \dots + \hat{\beta}_p x_0^p.$$

Les intervalles de confiance des prévisions de \mathbf{Y} et $E(\mathbf{Y})$, pour une valeur $\mathbf{x}_0 \in \mathbb{R}^p$ et en posant $\mathbf{v}_0 = (1|\mathbf{x}'_0)' \in \mathbb{R}^{p+1}$, sont respectivement

$$\begin{aligned}\hat{y}_0 &\pm t_{\alpha/2;(n-p-1)} \hat{\sigma} (1 + \mathbf{v}'_0 (\mathbf{X}' \mathbf{X})^{-1} \mathbf{v}_0)^{1/2}, \\ \hat{y}_0 &\pm t_{\alpha/2;(n-p-1)} \hat{\sigma} (\mathbf{v}'_0 (\mathbf{X}' \mathbf{X})^{-1} \mathbf{v}_0)^{1/2}.\end{aligned}$$

Les variances de ces prévisions, comme celles des estimations des paramètres, dépendent directement du conditionnement de la matrice $\mathbf{X}' \mathbf{X}$.

1.9 Diagnostics

La validité d'un modèle de régression multiple et donc la fiabilité des prévisions, dépendent de la bonne vérification des hypothèses :

- homoscédasticité : variance σ^2 des résidus constante,
- linéarité du modèle : paramètres β_j constant,
- absence de points influents par la distance de Cook :

$$D_i = \frac{1}{s^2(p+1)} (\hat{\mathbf{y}} - \hat{\mathbf{y}}_{(i)})' (\hat{\mathbf{y}} - \hat{\mathbf{y}}_{(i)}),$$

- éventuellement la normalité des résidus,
- le conditionnement de la matrice $\mathbf{X}' \mathbf{X}$.

Tracer le graphe des résidus standardisés en fonction des valeurs ajustées montre leur plus ou moins bonne répartition autour de l'axe $y = 0$. La forme de ce nuage est susceptible de dénoncer une absence de linéarité ou une hétérosécédasticité.

Le conditionnement de la matrice $\mathbf{X}' \mathbf{X}$ est indiqué par le rapport $\kappa = \lambda_1/\lambda_p$ où $\lambda_1, \dots, \lambda_p$ sont les valeurs propres de la matrice des corrélations \mathbf{R} rangées par ordre décroissant. Ainsi, des problèmes de variances excessives voire même de précision numérique apparaissent dès que les dernières valeurs propres sont relativement trop petites.

1.10 Exemple

Les données sont extraites de Jobson (1991)[3] et décrivent les résultats comptables de 40 entreprises du Royaume Uni.

RETCAP	Return on capital employed
WCFTDT	Ratio of working capital flow to total debt
LOGSALE	Log to base 10 of total sales
LOGASST	Log to base 10 of total assets
CURRAT	Current ratio
QUIKRAT	Quick ratio
NFATAST	Ratio of net fixed assets to total assets
FATTOT	Gross fixed assets to total assets
PAYOUT	Payout ratio
WCFTCL	Ratio of working capital flow to total current liabilities
GEARRAT	Gearing ratio (debt-equity ratio)
CAPINT	Capital intensity (ratio of total sales to total assets)
INVTAST	Ratio of total inventories to total assets

Modèle complet

La procédure SAS/REG fournit les résultats classiques de la régression multiple.

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Prob>F
Model	12	0.55868 (2)	0.04656 (5)	8.408 (7)	0.0001 (8)
Error	27	0.14951 (3)	0.00554 (6)		
C Total	39	0.70820 (4)			
Root MSE	0.07441 (9)	R-square	0.7889 (12)		
Dep Mean	0.14275 (10)	Adj R-sq	0.6951 (13)		
C.V.	52.12940 (11)				

- (1) degrés de liberté de la loi de Fisher du test global
(2) SSR
(3) SSE ou déviance
(4) SST=SSE+SSR
(5) SSR/DF
(6) MSE=SSE/DF est l'estimation de σ^2
(7) Statistique F du test de Fisher du modèle global
(8) $P(f_{p,n-p-1} > F)$; H_0 est rejetée au niveau α si $P < \alpha$
(9) s =racine de MSE
(10) moyenne empirique de la variable à expliquée
(11) Coefficient de variation $100 \times (9)/(10)$
(12) Coefficient de détermination R^2
(13) Coefficient de détermination ajusté R'^2

Parameter Estimates						
Variable	DF	Parameter Estimate	Standard Error	T for H0:		Variance Inflation
				Parameter=0	Prob> T	
INTERCEP	1	0.188072	0.13391661	1.404	0.1716	.
WCFTCL	1	0.215130	0.19788455	1.087	0.2866	0.03734409
WCFTDT	1	0.305557	0.29736579	1.028	0.3133	0.02187972
GEARRAT	1	-0.040436	0.07677092	-0.527	0.6027	0.45778579
LOGSALE	1	0.118440	0.03611612	3.279	0.0029	0.10629382
LOGASST	1	-0.076960	0.04517414	-1.704	0.0999	0.21200778
...						

- (1) estimations des paramètres ($\hat{\beta}_j$)
(2) écarts-types de ces estimations $\hat{\sigma}_j$
(3) statistique T du test de Student de $H_0 : \beta_j = 0$
(4) $P(t_{n-p-1} > T)$; H_0 est rejetée au niveau α si $P < \alpha$
(5) $1 - R^2_{(j)}$
(6) VIF= $1/(1 - R^2_{(j)})$

Ces résultats soulignent les problèmes de colinéarités. De grands VIF (facteurs d'inflation de la variance) sont associés à de grands écarts-types des estimations des paramètres. D'autre part les nombreux tests de Student non significatifs montrent que trop de variables sont présentes dans le modèle. Cette idée est renforcée par le calcul de l'indice de conditionnement : 8.76623/0.00125.

2 Analyse de covariance (AnCoVa)

L'analyse de covariance se situe encore dans le cadre général du modèle linéaire et où une variable quantitative est expliquée par plusieurs variables à la fois quantitatives et qualitatives. Les cas les plus complexes associent plusieurs facteurs (variables qualitatives) avec une structure croisée ou hiérarchique ainsi que plusieurs variables quantitatives intervenant de manière linéaire ou polynomiale. Le principe général, dans un but explicatif ou décisionnel, est toujours d'estimer des modèles *intra-groupes* et de faire apparaître (tester) des effets différentiels *inter-groupes* des paramètres des régressions. Ainsi, dans le cas plus simple où seulement une variable parmi les explicatives est quantitative, des tests interrogent l'hétérogénéité des constantes et celle des pentes (interaction) entre différents modèles de régression linéaire.

Ce type de modèle permet également, avec un objectif prédictif, de s'intéresser à la modélisation d'une variable quantitative par un ensemble de variables explicatives à la fois quantitatives et qualitatives.

La possible prise en compte d'*interactions* entre les variables complique la procédure de sélection de variables.

2.1 Modèle

Le modèle est explicité dans le cas élémentaire où une variable quantitative \mathbf{Y} est expliquée par une variable qualitative \mathbf{T} à J niveaux et une variable quantitative, appelée encore covariable, \mathbf{X} . Pour chaque niveau j de \mathbf{T} , on observe n_j valeurs X_{1j}, \dots, X_{nj} de \mathbf{X} et n_j valeurs Y_{1j}, \dots, Y_{nj} de \mathbf{Y} ; $n = \sum_{j=1}^J n_j$ est la taille de l'échantillon.

En pratique, avant de lancer une procédure de modélisation et tests, une démarche exploratoire s'appuyant sur une représentation en couleur (une par modalité j de \mathbf{T}) du nuage de points croisant \mathbf{Y} et \mathbf{X} et associant les droites de régression permet de se faire une idée sur les effets respectifs des variables : parallélisme des droites, étirement, imbrication des sous-nuages.

On suppose que les moyennes conditionnelles $E[\mathbf{Y}|\mathbf{T}]$, c'est-à-dire calculées à l'intérieur de chaque cellule, sont dans le sous-espace vectoriel engendré par les variables explicatives quantitatives, ici \mathbf{X} . Ceci s'écrit :

$$Y_{ij} = \beta_{0j} + \beta_{1j}X_{ij} + \varepsilon_{ij}; \quad j = 1, \dots, J; \quad i = 1, \dots, n_j$$

où les ε_{ij} sont i.i.d. suivant une loi centrée de variance σ^2 qui sera supposée $\mathcal{N}(0, \sigma^2)$ pour la construction des tests.

Notons \mathbf{Y} le vecteur des observations $[Y_{ij}|i = 1, n_j; j = 1, J]'$ mis en colonne, \mathbf{x} le vecteur $[X_{ij}|i = 1, n_j; j = 1, J]'$, $\boldsymbol{\varepsilon} = [\varepsilon_{ij}|i = 1, n_j; j = 1, J]'$ le vecteur des erreurs, $\mathbf{1}_j$ les variables indicatrices des niveaux et $\mathbf{1}$ la colonne de 1s. On note encore $\mathbf{x} \cdot \mathbf{1}_j$ le produit terme à terme des deux vecteurs, c'est-à-dire le vecteur contenant les observations de \mathbf{x} sur les individus prenant le niveau j de \mathbf{T} et des zéros ailleurs.

La résolution simultanée des J modèles de régression est simplement obtenue en considérant globalement le modèle :

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

dans lequel \mathbf{X} est la matrice $n \times 2J$ constituée des blocs $[\mathbf{1}_j | \mathbf{X} \cdot \mathbf{1}_j]; j = 1, \dots, J$. L'estimation de ce modèle global conduit, par bloc, à estimer les modèles de régression dans chacune des cellules.

Comme pour l'analyse de variance (AnOVA), les logiciels opèrent une reparamétrisation faisant apparaître des effets différentiels par rapport au dernier niveau ou par rapport à un effet moyen, afin d'obtenir directement les bonnes hypothèses dans les tests. Ainsi, dans le premier cas, on considère la matrice de même rang (sans la J ème indicatrice)

$$\mathbf{X} = [1 | \mathbf{X} | \mathbf{1}_1 | \dots | \mathbf{1}_{J-1} | \mathbf{x} \cdot \mathbf{1}_1 | \dots | \mathbf{x} \cdot \mathbf{1}_{J-1}]$$

associée aux modèles :

$$\begin{aligned} Y_{ij} &= \beta_{0J} + (\beta_{0j} - \beta_{0J}) + \beta_{1J}X_{ij} + (\beta_{1j} - \beta_{1J})X_{ij} + \varepsilon_{ij}; \\ j &= 1, \dots, J-1; i = 1, \dots, n_j. \end{aligned}$$

2.2 Tests

Différentes hypothèses sont alors testées en comparant le modèle complet

$$\begin{aligned} \mathbf{Y} = \beta_{0J}\mathbf{1} &+ (\beta_{01} - \beta_{0J})\mathbf{1}_1 + \dots + (\beta_{0J-1} - \beta_{0J})\mathbf{1}_{J-1} + \beta_{1J}\mathbf{x} + \\ &+ (\beta_{11} - \beta_{1J})\mathbf{x} \cdot \mathbf{1}_1 + \dots + (\beta_{1J-1} - \beta_{1J})\mathbf{x} \cdot \mathbf{1}_{J-1} + \boldsymbol{\varepsilon} \end{aligned}$$

à chacun des modèles réduits :

- (i) $\mathbf{Y} = \beta_{0J}\mathbf{1} + (\beta_{01} - \beta_{0J})\mathbf{1}_1 + \dots + (\beta_{0J-1} - \beta_{0J})\mathbf{1}_{J-1} + \beta_{1J}\mathbf{x} + \boldsymbol{\varepsilon}$
- (ii) $\mathbf{Y} = \beta_{0J}\mathbf{1} + (\beta_{01} - \beta_{0J})\mathbf{1}_1 + \dots + (\beta_{0J-1} - \beta_{0J})\mathbf{1}_{J-1} + \boldsymbol{\varepsilon}$
- (iii) $\mathbf{Y} = \beta_{0J}\mathbf{1} + \beta_{1J}\mathbf{x} + (\beta_{1j} - \beta_{1J})\mathbf{x} \cdot \mathbf{1}_1 + \dots + (\beta_{1J-1} - \beta_{1J})\mathbf{x} \cdot \mathbf{1}_{J-1} + \boldsymbol{\varepsilon}$
- (iv) $\mathbf{Y} = \beta_{0J}\mathbf{1} + \boldsymbol{\varepsilon}$

par un test de Fisher. Ceci revient à considérer les hypothèses suivantes :

- H_0^i : pas d'interaction entre variables \mathbf{X} et \mathbf{T} , $\beta_{11} = \dots = \beta_{1J}$, les droites partagent la même pente β_{1J} .
- H_0^{ii} : $\beta_{11} = \dots = \beta_{1J} = 0$ (pas d'effet de \mathbf{x})
- H_0^{iii} : $\beta_{01} = \dots = \beta_{0J}$, les droites partagent la même constante à l'origine β_{0J} .
- H_0^{iv} les variables \mathbf{X} et \mathbf{T} n'ont aucun effet sur \mathbf{Y} .

Commencer par évaluer i ; si le test n'est pas significatif, regarder ii qui, s'il n'est pas non plus significatif, conduit à l'absence d'effet de la variable X . De même, toujours si i n'est pas significatif, s'intéresser à iii pour juger de l'effet du facteur T .

3 Choix de modèle par sélection de variables

3.1 Introduction

De façon schématique, la pratique de la modélisation statistique vise trois objectifs éventuellement complémentaires.

Descriptif : rechercher de façon exploratoire les liaisons entre \mathbf{Y} et d'autres variables, potentiellement explicatives, \mathbf{X}^j qui peuvent être nombreuses afin, par exemple d'en sélectionner un sous-ensemble. À cette stratégie, à laquelle peuvent contribuer des Analyses en Composantes Principales, correspond des algorithmes de recherche (pas à pas) moins performants mais économiques en temps de calcul si p est grand.

Attention, si n est petit, et la recherche suffisamment longue avec beaucoup de variables explicatives, il sera toujours possible de trouver un modèle expliquant y ; c'est l'effet *data mining* dans les modèles économétriques appelé maintenant *data snooping*.

Explicatif : Le deuxième objectif est sous-tendu par une connaissance *a priori* du domaine concerné et dont des résultats théoriques peuvent vouloir être confirmés, infirmés ou précisés par l'estimation des paramètres. Dans ce cas, les résultats inférentiels permettent de construire le bon test conduisant à la prise de décision recherchée. Utilisées hors de ce contexte, les statistiques de test n'ont qu'une valeur indicative au même titre que d'autres critères plus empiriques.

Prédicatif : Dans le troisième cas, l'accent est mis sur la qualité des prévisions. C'est la situation rencontrée en *apprentissage*. Ceci conduit à rechercher des modèles *parcimonieux* c'est-à-dire avec un nombre volontairement restreint de variables explicatives pour réduire la variance. Le modèle ainsi obtenu peut favoriser des estimateurs biaisés au profit d'une variance plus faible même si le théorème de Gauss-Markov indique que, parmi les estimateurs sans biais, celui des moindres carrés est de variance minimum. Un bon modèle n'est donc plus celui qui explique le mieux les données au sens d'un R^2 maximum mais celui conduisant aux prévisions les plus fiables.

Ceci est illustré ceci par un exemple simple (mais pédagogique) en régression polynomiale : Les figures 1 et 2) représentent un jeu de données simulées : $Y_i = f(x_i) + \epsilon_i$, $i = 1, \dots, n$ et $x_i \in [0, 1]$ sur lesquelles des polynômes de degrés croissants sont ajustés. L'ajustement du modèle mesuré par le R^2 croît logiquement avec le nombre de paramètres et atteint la valeur 1 lorsque le polynôme interpolate les observations.

Le R^2 ne peut-être un bon critère de sélection de modèles ; il ne peut servir qu'à comparer des modèles de même dimension car sinon conduit à sélectionner le modèle le plus complexe, c'est-à-dire celui correspond au plus grand espace de projection, et conduit donc au sur-ajustement.

Il y a principalement deux façons de biaiser un modèle linéaire dans le but de restreindre la variance :

- en réduisant le nombre de variables explicatives et donc en simplifiant le modèle (sélection ou pénalisation Lasso l_1),
- en contrignant les paramètres du modèle, en les rétrécissant (*schrinkage*), par une régression *ridge* qui opère une régularisation par pénalisation l_2 .

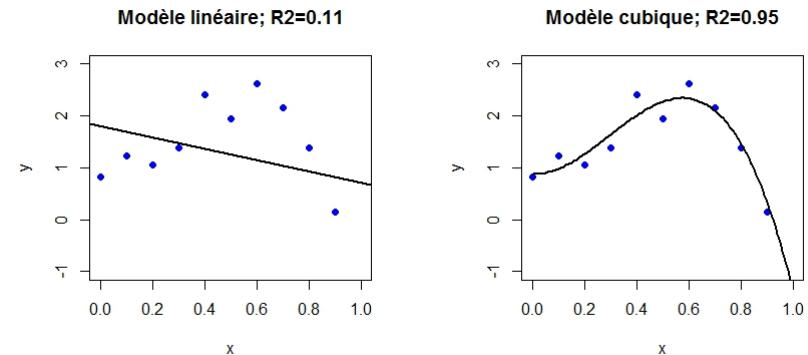


FIGURE 1 – Régression polynomiale : ajustement par, à gauche, $y = \beta_0 + \beta_1 x + \epsilon$, et à droite, $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \epsilon$

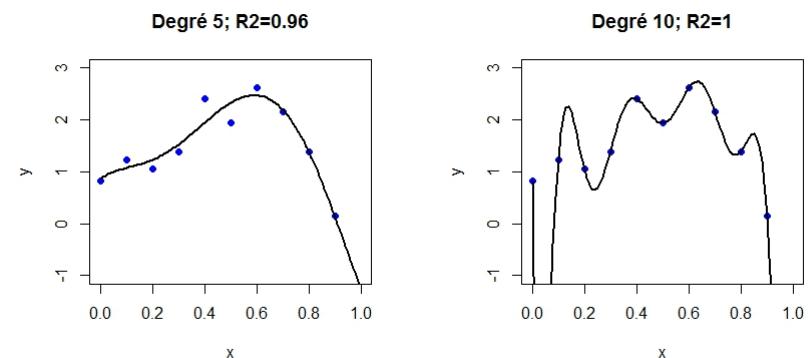


FIGURE 2 – Régression polynomiale : ajustement par, à gauche : $y = \beta_0 + \beta_1 x + \dots + \beta_5 x^5 + \epsilon$, et à droite, $y = \beta_0 + \beta_1 x + \dots + \beta_{10} x^{10} + \epsilon$.

3.2 Critères de sélection de variables

De nombreux critères de choix de modèle sont présentés dans la littérature sur la régression linéaire multiple.

Une tradition ancienne, encore présente dans certains livres ou fonction de logiciels, propose d'utiliser la statistique du test de Fisher de comparaison d'un modèle avec un sous-modèle comme critère de sélection de variables. *Attention*, la significativité de la présence d'une variable basée sur la p -valeur du test de nullité de son coefficient n'est du tout une indication sur l'importance de cette variable pour la qualité de la prévision. Explicatif ou prédictif sont deux objectifs différents de la modélisation ; ne pas les confondre. D'autre part, pour éviter le caractère croissant du coefficient de détermination R^2 en fonction du nombre de variables, une version pénalisée a été proposée : le R^2 ajusté mais qui conduit très généralement à des modèles trop complexes. Ces deux approches : statistique du test de Fisher et R^2 ajusté sont à oublier.

D'autres critères sont eux basés sur une qualité de prévision. Le C_p de Mallows, le critère d'information d'Akaike (AIC), celui bayésien de Sawa (BIC)... Ils sont équivalents, également avec le R^2 , lorsque le nombre de variables à sélectionner, ou complexité du modèle, est fixé. Le choix du critère est déterminant lorsqu'il s'agit de comparer des modèles de complexité différentes. Certains critères se ramènent, dans le cas gaussien, à l'utilisation d'une expression pénalisée de la fonction de vraisemblance afin de favoriser des modèles parcimonieux. En pratique, les plus utilisés ou ceux généralement fournis par les logiciels sont les suivants.

C_p de Mallows

L'indicateur proposé par Mallows (1973)[5] est une estimation de l'erreur quadratique moyenne de prévision qui s'écrit aussi comme la somme d'une variance et du carré d'un biais. L'erreur quadratique moyenne de prévision s'écrit ainsi :

$$\text{MSE}(\hat{Y}_i) = \text{Var}(\hat{Y}_i) + [\text{Biais}(\hat{Y}_i)]^2$$

puis après sommation et réduction :

$$\frac{1}{\sigma^2} \sum_{i=1}^n \text{MSE}(\hat{Y}_i) = \frac{1}{\sigma^2} \sum_{i=1}^n \text{Var}(\hat{Y}_i) + \frac{1}{\sigma^2} \sum_{i=1}^n [\text{Biais}(\hat{Y}_i)]^2.$$

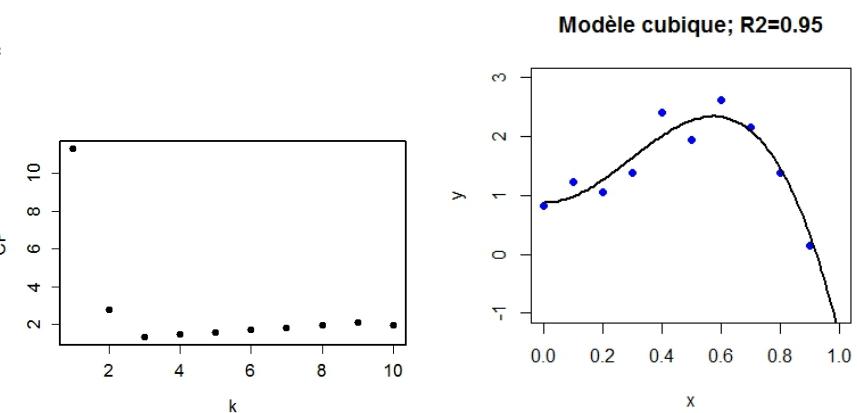


FIGURE 3 – C_p de Mallows en fonction du degré du polynôme et modèle sélectionné de degré 3.

En supposant que les *estimations du modèle complet sont sans biais* et en utilisant des estimateurs de $\text{Var}(\hat{Y}_i)$ et σ^2 , l'expression de l'erreur quadratique moyenne totale standardisée (ou réduite) pour un modèle à j variables explicatives s'écrit :

$$C_p = (n - q - 1) \frac{\text{MSE}_j}{\text{MSE}} - [n - 2(q + 1)]$$

et définit la valeur du C_p de Mallows pour les q variables considérées. Il est alors d'usage de rechercher un modèle qui minimise le C_p généralement proche de $(q + 1)$. Ceci revient à considérer que le "vrai" modèle complet est moins fiable qu'un modèle réduit donc biaisé mais d'estimation plus précise.

La figure 3 montre le comportement du C_p dans l'exemple de la régression polynomial. Ce critère décroît avec le biais jusqu'à un choix optimal de dimension 3 avant de ré-augmenter avec la variance.

AIC, BIC et PRESS

Dans le cas du modèle linéaire, et si la variance des observations est supposée connue, le critère AIC (*Akaike's Information criterion*) est équivalent au

critère C_p de Mallows.

Le PRESS de Allen (1974)[?] est l'introduction historique de la validation croisée ou *leave one out (loo)*. On désigne par $\widehat{Y}_{(i)}$ la prévision de Y_i calculée sans tenir compte de la i ème observation $(Y_i, X_i^1, \dots, X_i^p)$, la somme des erreurs quadratiques de prévision (PRESS) est définie par

$$\frac{1}{n} \sum_{i=1}^n \left[y_i - \widehat{f}^{(-i)}(\mathbf{x}_i) \right]^2 = \frac{1}{n} \sum_{i=1}^n \left[\frac{y_i - \widehat{f}(\mathbf{x}_i)}{1 - h_{ii}} \right]^2.$$

et permet de comparer les capacités prédictives de deux modèles.

La vignette sur [Qualité de prévision et risque](#) donne plus de détails sur ces derniers critères.

3.3 Algorithmes de sélection de variables

Dans le cas général et évidemment le plus courant en pratique, les variables ne sont pas pré-ordonnées par importance. Lorsque p est grand, il n'est pas raisonnable de penser explorer les 2^p modèles possibles afin de sélectionner le meilleur au sens de l'un des critères ci-dessus. Différentes stratégies sont donc proposées qui doivent être choisies en fonction de l'objectif recherché, de la valeur de p et des moyens de calcul disponibles. Deux types d'algorithmes sont résumés ci-dessous par ordre croissant de temps de calcul nécessaire c'est-à-dire par nombre croissant de modèles considérés parmi les 2^p et donc par capacité croissante d'optimalité.

Pas à pas

Stratégie correspondant à la fonction `StepAIC` de R. Comme écrit ci-dessus, oublier les sélections basées sur la statistique de Fisher.

Sélection (forward) À chaque pas, une variable est ajoutée au modèle.

C'est celle qui permet de réduire au mieux le critère AIC du modèle obtenu. La procédure s'arrête lorsque toutes les variables sont introduites ou lorsque AIC ne décroît plus.

Élimination (backward) L'algorithme démarre cette fois du modèle complet. À chaque étape, la variable dont l'élimination conduit à l'AIC le plus faible est supprimée. La procédure s'arrête lorsque AIC ne décroît plus.

Mixte (stepwise) Cet algorithme introduit une étape d'élimination de variable après chaque étape de sélection afin de retirer du modèle d'éventuels variables qui seraient devenues moins indispensables du fait de la présence de celles nouvellement introduites.

Global

L'algorithme de Furnival et Wilson (1974)[2] (librairie `leaps` de R) est utilisé pour comparer tous les modèles possibles en cherchant à optimiser l'un des critères : C_p , AIC, BIC choisi par l'utilisateur. Par souci d'économie, cet algorithme évite de considérer des modèles de certaines sous-branches de l'arborescence dont on peut savoir *a priori* qu'ils ne sont pas compétitifs. Cet algorithme affiche le ou les meilleurs modèles de chaque niveau q . Rappel : à q fixé tous les critères sont équivalents mais les choix de q optimal peut différer d'un critère à l'autre. Il n'est pas raisonnable de considérer plus d'une quinzaine de variables avec cet algorithme.

3.4 Sélection en analyse de covariance

Un modèle d'analyse de covariance pose des problèmes spécifiques de sélection notamment par la prise en compte possible d'interactions entre variables dans la définition du modèle. La recherche d'un modèle efficace, donc parcimonieux, peut conduire à négliger des interactions ou effets principaux lorsqu'une faible amélioration du R^2 le justifie et même si le test correspondant apparaît comme significatif. L'utilisation du C_p est théoriquement possible mais en général ce critère n'est pas calculé car d'utilisation délicate. En effet, il nécessite la considération d'un modèle de référence sans biais ou tout du moins d'un modèle de faible biais pour obtenir une estimation raisonnable de la variance de l'erreur. En régression multiple (toutes les variables explicatives quantitatives), le modèle complet est considéré comme étant celui de faible biais mais en analyse de covariance quels niveaux de complexité des interactions faut-il considérer pour construire le modèle complet jugé de faible biais ? Il est alors plus simple et plus efficace d'utiliser le critère AIC, choix par défaut dans plusieurs logiciels comme R.

L'algorithme de recherche descendant est le plus couramment utilisé avec la contrainte suivante :

un effet principal n'est supprimé qu'à la condition qu'il n'apparaisse plus

dans une interaction.

Voici, à titre d'exemple, une étape intermédiaire d'une sélection de variables pas à pas *stepwise* avec l'option *both* de la fonction *StepAIC* de R. A chaque étape, le critère AIC est évalué par suppression ou rajout de chacune des variables. L'option minimisant le critère AIC est retenue avant de passer à l'étape suivante. Le modèle ne comprend pas d'interactions.

```
Step: AIC=-60.79
lpsa ~ lcavol + lweight + age + lbph + svi + pgg45

Df Sum of Sq RSS AIC
- pgg45 1 0.6590 45.526 -61.374
<none> 44.867 -60.788
+ lcp 1 0.6623 44.204 -60.231
- age 1 1.2649 46.132 -60.092
- lbph 1 1.6465 46.513 -59.293
+ gleason 3 1.2918 43.575 -57.622
- lweight 1 3.5646 48.431 -55.373
- svi 1 4.2503 49.117 -54.009
- lcavol 1 25.4190 70.286 -19.248

Step: AIC=-61.37
lpsa ~ lcavol + lweight + age + lbph + svi
```

En effet, supprimer un effet principal qualitatif alors que la variable est présente dans une interaction ne change en rien le modèle car l'espace engendré par l'ensemble des indicatrices sélectionnées reste le même ; la matrice \mathbf{X} est construite sous contrainte de rang et retirer une colonne (effet principal) fait automatiquement entrer une indicatrice d'interaction supplémentaire. Le modèle est inchangé mais l'interprétation plus compliquée car le modèle ne se décompose plus en un effet principal puis ses interactions.

3.5 Exemple de sélection

Tous les modèles (parmi les plus intéressants selon l'algorithme de Furnival et Wilson) sont considérés. Seul le meilleur pour chaque niveau, c'est-à-dire pour chaque valeur p du nombre de variables explicatives sont donnés. Il est alors facile de choisir celui minimisant l'un des critères globaux (C_p ou BIC). Cet exemple calculé avec SAS est choisi pour comparer différents critères.

```
options linesize=110 pagesize=30 nodate nonumber;
title;
proc reg data=sasuser.ukcomp2 ;
model RETCAP = WCFTCL WCFTDT GEARRAT LOGSALE LOGASST
NFATAST CAPINT FATTOT INVTASt PAYOUT QUIKRAT CURRAT
```

```
/ selection=rsquare cp rsquare bic best=1;
run;

N = 40      Regression Models for Dependent Variable: RETCAP
R-sq. Adjust. C(p)   BIC   Variables in Model
In   R-sq
1 0.105 0.081 78.393 -163.2 WCFTCL
2 0.340 0.305 50.323 -173.7 WCFTDT QUIKRAT
3 0.615 0.583 17.181 -191.1 WCFTCL NFATAST CURRAT
4 0.720 0.688 5.714 -199.2 WCFTDT LOGSALE NFATAST CURRAT
5 0.731 0.692 6.304 -198.0 WCFTDT LOGSALE NFATAST QUIKRAT CURRAT
6 0.748 0.702 6.187 -197.2 WCFTDT LOGSALE NFATAST INVTASt QUIKRAT CURRAT
7 0.760 0.707 6.691 -195.7 WCFTDT LOGSALE LOGASST NFATAST FATTOT QUIKRAT CURRAT
8 0.769 0.709 7.507 -193.8 WCFTDT LOGSALE LOGASST NFATAST FATTOT INVTASt QUIKRAT CURRAT
9 0.776 0.708 8.641 -191.5 WCFTCL WCFTDT LOGSALE LOGASST NFATAST FATTOT INVTASt QUIKRAT
CURRAT
10 0.783 0.708 9.744 -189.1 WCFTCL WCFTDT LOGSALE LOGASST NFATAST FATTOT INVTASt PAYOUT
QUIKRAT CURRAT
11 0.786 0.702 11.277 -186.4 WCFTCL WCFTDT LOGSALE LOGASST NFATAST CAPINT FATTOT INVTASt
PAYOUT QUIKRAT CURRAT
12 0.788 0.695 13.000 -183.5 WCFTCL WCFTDT GEARRAT LOGSALE LOGASST NFATAST CAPINT FATTOT
INVTASt PAYOUT QUIKRAT CURRAT
```

Dans cet exemple, C_p et BIC se comportent de la même façon. Avec peu de variables, le modèle est trop biaisé. Ils atteignent un minimum pour un modèle à 4 variables explicatives puis croissent de nouveau selon la première bissectrice. La maximisation du R^2 ajusté conduirait à une solution beaucoup moins parcimonieuse. On note par ailleurs que l'algorithme remplace WCFTCL par WCFTDT. Un algorithme par sélection ne peut pas aboutir à la solution optimale retenue.

4 Régression régularisée ou pénalisée

4.1 Régression ridge

Modèle et estimation

Ayant diagnostiqué un problème mal conditionné mais désirant conserver toutes les variables explicatives pour des raisons d'interprétation, il est possible d'améliorer les propriétés numériques et la variance des estimations en considérant un estimateur biaisé des paramètres par une procédure de régularisation.

Soit le modèle linéaire :

$$\mathbf{Y} = \tilde{\mathbf{X}}\tilde{\boldsymbol{\beta}} + \epsilon,$$

où

$$\tilde{\mathbf{X}} = \begin{pmatrix} 1 & X_1^1 & X_1^2 & \dots & X_1^p \\ 1 & X_2^1 & X_2^2 & \dots & X_2^p \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & X_n^1 & X_n^2 & \dots & X_n^p \end{pmatrix},$$

$$\tilde{\boldsymbol{\beta}} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix}, \quad \boldsymbol{\beta} = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix}$$

où $\mathbf{X}^0 = (1, 1, \dots, 1)'$, et \mathbf{X} désigne la matrice $\tilde{\mathbf{X}}$ privée de sa première colonne. L'estimateur *ridge* est défini par un critère des moindres carrés, avec une pénalité de type \mathbb{L}^2 :

DÉFINITION 1. — *L'estimateur ridge de $\tilde{\boldsymbol{\beta}}$ dans le modèle*

$$\mathbf{Y} = \tilde{\mathbf{X}}\tilde{\boldsymbol{\beta}} + \epsilon,$$

est défini par :

$$\hat{\boldsymbol{\beta}}_{\text{ridge}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^{p+1}}{\operatorname{argmin}} \left(\sum_{i=1}^n (Y_i - \sum_{j=0}^p X_i^{(j)} \beta_j)^2 + \lambda \sum_{j=1}^p \beta_j^2 \right),$$

où λ est un paramètre positif, à choisir.

À noter que le paramètre β_0 n'est pas pénalisé.

PROPOSITION 2. — *L'estimateur ridge s'exprime aussi sous la forme :*

$$\hat{\beta}_0_{\text{ridge}} = \bar{Y}, \quad \begin{pmatrix} \hat{\beta}_1 \\ \hat{\beta}_2 \\ \vdots \\ \hat{\beta}_p \end{pmatrix}_{\text{ridge}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \left(\|\mathbf{Y}^{(c)} - \mathbf{X}^{(c)}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|^2 \right).$$

où $\mathbf{X}^{(c)}$ désigne la matrice \mathbf{X} recentrée (par colonnes) et $\mathbf{Y}^{(c)}$ désigne le vecteur \mathbf{Y} recentré.

Supposant désormais que \mathbf{X} et \mathbf{Y} sont centrés, l'estimateur *ridge* est obtenue en résolvant les équations normales qui s'expriment sous la forme :

$$\mathbf{X}'\mathbf{Y} = (\mathbf{X}'\mathbf{X} + \lambda \mathbf{I}_p)\boldsymbol{\beta}.$$

Conduisant à :

$$\hat{\boldsymbol{\beta}}_{\text{ridge}} = (\mathbf{X}'\mathbf{X} + \lambda \mathbf{I}_p)^{-1} \mathbf{X}'\mathbf{Y}.$$

La solution est donc explicite et linéaire en \mathbf{Y} .

Remarques :

1. $\mathbf{X}'\mathbf{X}$ est une matrice symétrique positive (pour tout vecteur \mathbf{u} de \mathbb{R}^p , $\mathbf{u}'(\mathbf{X}'\mathbf{X})\mathbf{u} = \|\mathbf{X}\mathbf{u}\|^2 \geq 0$). Il en résulte que pour tout $\lambda > 0$, $\mathbf{X}'\mathbf{X} + \lambda \mathbf{I}_p$ est nécessaire inversible.
2. La constante β_0 n'intervient pas dans la pénalité, sinon, le choix de l'origine pour \mathbf{Y} aurait une influence sur l'estimation de l'ensemble des paramètres. Alors : $\hat{\beta}_0 = \bar{Y}$; ajouter une constante à \mathbf{Y} ne modifie pas les $\hat{\beta}_j$ pour $j \geq 1$.
3. L'estimateur *ridge* n'est pas invariant par renormalisation des vecteurs $X^{(j)}$, il est préférable de normaliser (réduire les variables) les vecteurs avant de minimiser le critère.
4. La régression *ridge* revient encore à estimer le modèle par les moindres carrés sous la contrainte que la norme du vecteur $\boldsymbol{\beta}$ des paramètres ne soit pas trop grande :

$$\hat{\boldsymbol{\beta}}_{\text{ridge}} = \arg \min_{\boldsymbol{\beta}} \left\{ \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 ; \|\boldsymbol{\beta}\|^2 < c \right\}.$$

La régression *ridge* conserve toutes les variables mais, contraignant la norme des paramètres β_j , elle les empêche de prendre de trop grandes valeurs et limite ainsi la variance des prévisions.

Optimisation de la pénalisation

La figure 4 montre quelques résultats obtenus par la méthode *ridge* en fonction de la valeur de la pénalité $\lambda = l$ sur l'exemple de la régression polynomiale. Plus la pénalité augmente et plus la solution obtenue est régulière ou encore, plus le biais augmente et la variance diminue. Il y a sur-ajustement

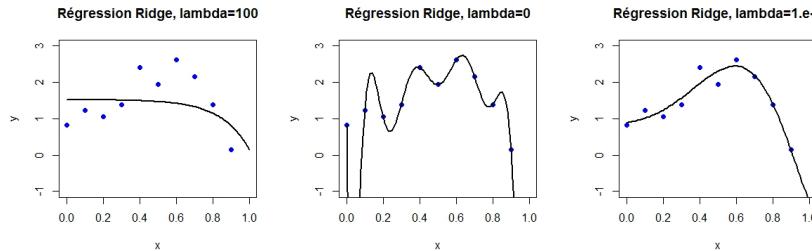


FIGURE 4 – Pénalisation *ridge* du modèle polynomial

avec une pénalité nulle : le modèle passe par tous les points mais oscille dangereusement ; il y a sous-ajustement avec une pénalité trop grande.

Comme dans tout problème de régularisation, le choix de la valeur du paramètre λ est crucial et déterminera le choix de modèle. La validation croisée est généralement utilisée pour optimiser le choix car la lecture du graphique (cf. figure 5) montrant l'évolution des paramètres en fonction du coefficient ou *chemins de régularisation ridge* n'est pas suffisante pour déterminer une valeur optimale.

Le principe de la validation croisée qui permet d'estimer sans biais une erreur de prévision est [détaillé par ailleurs](#).

4.2 Régression LASSO

La régression *ridge* permet donc de contourner les problèmes de colinéarité même en présence d'un nombre important de variables explicatives ou prédicteurs ($p > n$). La principale faiblesse de cette méthode est liée aux difficultés d'interprétation car, sans sélection, toutes les variables sont concernées dans le modèle. D'autres approches par pénalisation permettent également une sélection, c'est le cas de la régression Lasso.

Modèle et estimation

La méthode Lasso (Tibshirani, 1996)[8] correspond à la minimisation d'un critère des moindres carrés avec une pénalité de type l_1 (et non plus l_2 comme

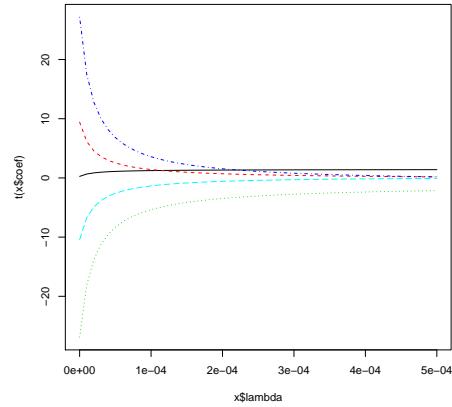


FIGURE 5 – Modèle polynomial : Chemin de régularisation en régression ridge en fonction du paramètre de la pénalisation.

dans la régression *ridge*). Soit $\|\beta\|_1 = \sum_{j=1}^p |\beta_j|$.

DÉFINITION 3. — *L'estimateur Lasso de β dans le modèle*

$$\mathbf{Y} = \mathbf{X}\beta + \epsilon,$$

est défini par :

$$\widehat{\beta}_{\text{Lasso}} = \underset{\beta \in \mathbb{R}^p}{\operatorname{argmin}} \left(\sum_{i=1}^n (Y_i - \sum_{j=0}^p X_i^{(j)} \beta_j)^2 + \lambda \sum_{j=1}^p |\beta_j| \right),$$

où λ est un paramètre positif, à choisir.

On peut montrer que ceci équivaut au problème de minimisation suivant :

$$\widehat{\beta}_{\text{Lasso}} = \underset{\beta, \|\beta\|_1 \leq t}{\operatorname{argmin}} (\|\mathbf{Y} - \mathbf{X}\beta\|^2),$$

pour un t convenablement choisi.

Comme dans le cas de la régression *ridge*, le paramètre λ est un paramètre de régularisation :

- Si $\lambda = 0$, on retrouve l'estimateur des moindres carrés.
- Si λ tend vers l'infini, on annule tous les $\hat{\beta}_j$, $j = 1, \dots, p$.

La solution obtenue est dite parcimonieuse (*sparse* en anglais), car elle comporte des coefficients nuls.

Autre pénalisation

La méthode Lasso équivaut à minimiser le critère

$$\text{Crit}(\beta) = \sum_{i=1}^n (Y_i - \beta_0 - \beta_1 X_i^{(1)} - \beta_2 X_i^{(2)} - \dots - \beta_p X_i^{(p)})^2$$

sous la contrainte $\sum_{j=1}^p |\beta_j| \leq t$, pour un $t > 0$.

Le logiciel R introduit une contrainte sous forme d'une borne relative pour $\sum_{j=1}^p |\beta_j|$: la contrainte s'exprime sous la forme

$$\sum_{j=1}^p |\beta_j| \leq \kappa \sum_{j=1}^p |\hat{\beta}_j^{(0)}|,$$

où $\hat{\beta}^{(0)}$ est l'estimateur des moindres carrés et $\kappa \in [0, 1]$.

Avec $\kappa = 1$ c'est l'estimateur des moindres carrés (pas de contrainte) et pour $\kappa = 0$, tous les $\hat{\beta}_j$, $j \geq 1$, sont nuls (contrainte maximale).

Utilisation de la régression Lasso

La pénalisation est optimisée comme en régression *ridge* par validation croisée.

Grâce à ses solutions parcimonieuses, cette méthode est surtout utilisée pour sélectionner des variables dans des modèles de grande dimension ; on peut l'utiliser si $p > n$ c'est-à-dire s'il y a plus de variables que d'observations. Bien entendu, dans ce cas, les colonnes de la matrice \mathbf{X} ne sont pas linéairement indépendantes. Il n'y a donc pas de solution explicite, on utilise des procédures d'optimisation pour trouver la solution. Il faut néanmoins utiliser la méthode avec précaution lorsque les variables explicatives sont corrélées. Pour que la méthode fonctionne, il faut que le nombre de variables influentes

(correspondant à des β_j différents de 0) ne dépasse pas n et que les variables non influentes ne soient pas trop corrélées avec celles qui le sont.

4.3 Elastic Net

La méthode *Elastic Net* permet de combiner la régression *ridge* et la régression Lasso, en introduisant les deux types de pénalités simultanément.

Le critère à minimiser est :

$$\begin{aligned} & \sum_{i=1}^n (Y_i - \beta_0 - \beta_1 X_i^{(1)} - \beta_2 X_i^{(2)} - \dots - \beta_p X_i^{(p)})^2 \\ & + \lambda \left(\alpha \sum_{j=1}^p |\beta_j| + (1 - \alpha) \sum_{j=1}^p \beta_j^2 \right) \end{aligned}$$

- Pour $\alpha = 1$, on retrouve la méthode LASSO.
- Pour $\alpha = 0$, on retrouve la régression *ridge*.

Il y a dans ce dernier cas deux paramètres à optimiser par validation croisée.

4.4 Sélection par réduction de dimension

Le principe de ces approches consiste à calculer la régression sur un ensemble de variables orthogonales deux à deux. Celles-ci peuvent être obtenues à la suite d'une analyse en composantes principales ou par décomposition en valeur singulière de la matrice \mathbf{X} : c'est la régression sur les composantes principales associées aux plus grandes valeurs propres.

L'autre approche ou régression PLS (*partial least square*) consiste à rechercher itérativement une composante linéaire des variables de plus forte covariance avec la variable à expliquer sous une contrainte d'orthogonalité avec les composantes précédentes.

Ce deux méthodes sont développées dans une [vignette](#) spécifique.

5 Exemples

5.1 Prévision de la concentration d'ozone

Les données

Les données proviennent des services de Météo-France et s'intéresse à la prévision de la concentration en Ozone dans 5 stations de mesure; ces sites ont été retenus pour le nombre important de pics de pollution qui ont été détectés dans les périodes considérées (étés 2002, 2003, 2005). Un pic de pollution est défini ici par une concentration dépassant le seuil de $150\mu\text{g}/\text{m}^3$. Météo-France dispose déjà d'une prévision (MOCAGE), à partir d'un modèle physique basé sur les équations du comportement dynamique de l'atmosphère (Navier et Stockes). Cette prévision fait partie du dispositif d'alerte des pouvoirs publics et prévoit donc une concentration de pollution à 17h locale pour le lendemain. L'objet du travail est d'en faire une évaluation statistique puis de l'améliorer en tenant compte d'autres variables ou plutôt d'autres prévisions faites par Météo-France. Il s'agit donc d'intégrer ces informations dans un modèle statistique global.

Les variables

Certaines variables de concentration ont été transformées afin de rendre symétrique (plus gaussienne) leur distribution.

O3-o Concentration d'ozone effectivement observée ou variable à prédire,

O3-pr prévision "mocage" qui sert de variable explicative ;

Tempe Température prévue pour le lendemain,

vmodule Force du vent prévue pour le lendemain,

lno Logarithme de la concentration observée en monoxyde d'azote,

lno2 Logarithme de la concentration observée en dioxyde d'azote,

rmh2o Racine de la concentration en vapeur d'eau,

Jour Variable à deux modalités pour distinguer les jours "ouvriables" (0) des jours "fériés-WE" (1).

Station Une variable qualitative indique la station concernée : Aix-en-Provence, Rambouillet, Munchhausen, Cadarache, et Plan de Cuques.

Modèle physique

Les graphiques de la figure 6 représentent la première prévision de la concentration d'ozone observée, ainsi que ses résidus, c'est-à-dire celle obtenue par

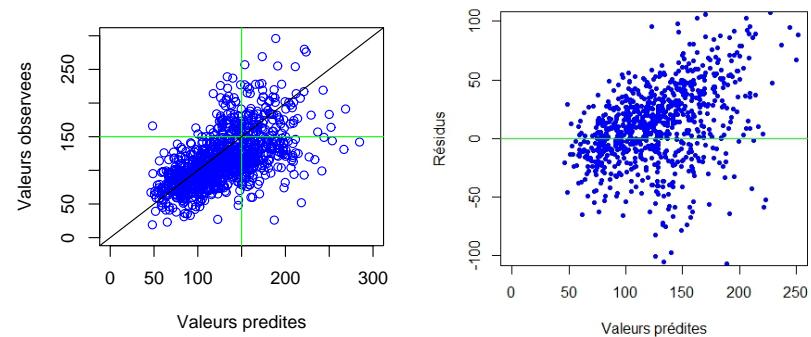


FIGURE 6 – Ozone : prévision et résidus du modèle MOCAGE de Météo-France pour 5 stations.

le modèle physique MOCAGE. Ces graphes témoignent de la mauvaise qualité de ce modèle : les résidus ne sont pas répartis de façon symétrique et les deux nuages présentent une légère forme de "banane" signifiant que des composantes non linéaires du modèle n'ont pas été prises en compte. D'autre part, la forme d'entonnoir des résidus montrent une forte hétéroscédaicité. Cela signifie que la variance des résidus et donc des prévisions croît avec la valeur. En d'autre terme, la qualité de la prévision se dégrade pour les concentrations élevées justement dans la zone sensible.

Modèle sans interaction

Un premier modèle est estimé avec R :

```
fit.lm=lm(O3-o~O3-pr+vmodule+lno2+lno+s-rmh2o+
          jour+station+TEMPE,data=donne)
```

Il introduit l'ensemble des variables explicatives mais sans interaction. Les résultats numériques sont fournis ci-dessous.

Coefficients:

Estimate	Std. Error	t value	Pr(> t)
----------	------------	---------	----------

(Intercept)	-4.99738	7.87028	-0.635	0.52559
O3_pr	0.62039	0.05255	11.805	< 2e-16 ***
vmodule	-1.73179	0.35411	-4.891	1.17e-06 ***
lno2	-48.17248	6.19632	-7.774	1.83e-14 ***
lno	50.95171	5.98541	8.513	< 2e-16 ***
s_rmh2o	135.88280	50.69567	2.680	0.00747 **
jour1	-0.34561	1.85389	-0.186	0.85215
stationAls	9.06874	3.37517	2.687	0.00733 **
stationCad	14.31603	3.07893	4.650	3.76e-06 ***
stationPla	21.54765	3.74155	5.759	1.12e-08 ***
stationRam	6.86130	3.05338	2.247	0.02484 *
TEMPE	4.65120	0.23170	20.074	< 2e-16 ***

Residual standard error: 27.29 on 1028 degrees of freedom
 Multiple R-Squared: 0.5616, Adjusted R-squared: 0.5569
 F-statistic: 119.7 on 11 and 1028 DF, p-value: < 2.2e-16

A l'exception de la variable indiquant la nature du jour, l'ensemble des coefficients sont jugés significativement différent de zéro mais la qualité de l'ajustement est faible (R^2).

Modèle avec interaction

La qualité d'ajustement du modèle précédent n'étant pas très bonne, un autre modèle est considéré en prenant en compte les interactions d'ordre 2 entre les variables. Compte tenu de la complexité du modèle qui un découle, un choix automatique est lancé par élimination successive des termes non significatifs (algorithme backward). Le critère optimisé est celui (AIC) d'Akaike. Plusieurs interactions ont été éliminées au cours de la procédure mais beaucoup subsistent dans le modèle. Attention, les effets principaux lno2, vmodule ne peuvent être retirés car ces variables apparaissent dans une interaction. En revanche on peut s'interroger sur l'opportunité de conserver celle entre la force du vent et la concentration de dioxyde d'azote.

	Df	Deviance	Resid.	Df	Resid.	Dev	F	Pr (>F)
NULL				1039	1745605			
O3_pr	1	611680	1038	1133925	969.9171	< 2.2e-16 ***		
station	4	39250	1034	1094674	15.5594	2.339e-12 ***		
vmodule	1	1151	1033	1093523	1.8252	0.176957		
lno2	1	945	1032	1092578	1.4992	0.2210886		
s_rmh2o	1	24248	1031	1068330	38.4485	8.200e-10 ***		
TEMPE	1	248891	1030	819439	394.6568	< 2.2e-16 ***		
O3_pr:station	4	16911	1026	802528	6.7038	2.520e-05 ***		
O3_pr:vmodule	1	8554	1025	793974	13.5642	0.0002428 ***		
O3_pr:TEMPE	1	41129	1024	752845	65.2160	1.912e-15 ***		
station:vmodule	4	7693	1020	745152	3.0497	0.0163595 *		

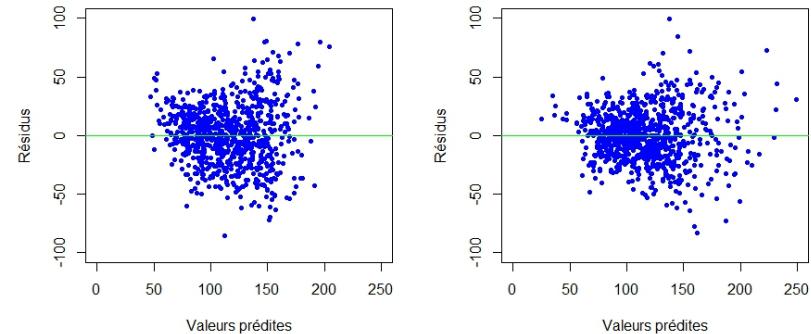


FIGURE 7 – Ozone : Résidus des modèles linéaire et quadratique.

station:lno2	4	12780	1016	732372	5.0660	0.0004811 ***
station:s_rmh2o	4	19865	1012	712508	7.8746	2.997e-06 ***
station:TEMPE	4	27612	1008	684896	10.9458	1.086e-08 ***
vmodule:lno2	1	1615	1007	683280	2.5616	0.1098033
vmodule:s_rmh2o	1	2407	1006	680873	3.8163	0.0510351 .
lno2:TEMPE	1	4717	1005	676156	7.4794	0.0063507 **
s_rmh2o:TEMPE	1	42982	1004	633175	68.1543	4.725e-16 ***

Ce sont surtout les graphes de la figure 7 qui renseignent sur l'adéquation des modèles. Le modèle quadratique fournit une forme plus "linéaire" des résidus et un meilleur ajustement avec un R^2 de 0,64 mais l'hétéroscésticité reste présente, d'autres approches s'avèrent nécessaires afin de réduire la variance liée à la prévision des concentrations élevées.

Sélection Lasso

Les résultats précédents ont été obtenus avec R qui propose des algorithmes de sélection de variables classique. Ce n'est pas le cas de la librairie scikit-learn qui se limite à des sélections par pénalisation Lasso mais sans pouvoir intégrer facilement les interactions alors que celle-ci sont justement importantes pour ces données. L'optimisation du paramètre de pénalisation et les chemins de régularisation sont obtenus par validation croisée (figur 8).

Encore un peu de travail est nécessaire pour obtenir les coefficients finale-

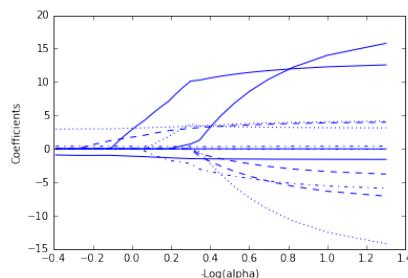
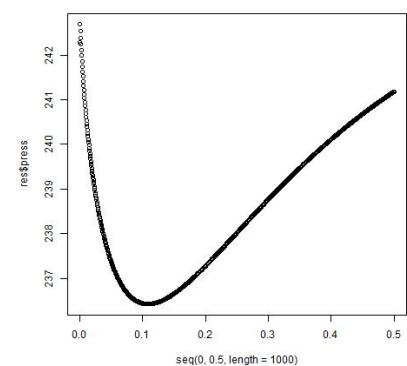
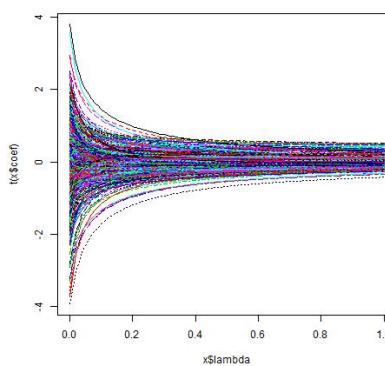


FIGURE 8 – Ozone : Régression lasso; chemin de régularisation des paramètres et optimisation de la pénalisation.



ment sélectionnés par la procédure.

5.2 Données de spectrométrie NIR

Objectif

Ce type de problème se rencontre en contrôle de qualité sur une chaîne de fabrication agroalimentaire, ici des biscuits (*cookies*). Il est nécessaire de contrôler le mélange des ingrédients avant cuisson afin de s'assurer que les proportions en lipides, sucre, farine, eau, sont bien respectées. Il s'agit de savoir s'il est possible de dépister au plus tôt une dérive afin d'intervenir sur les équipements concernés. Les mesures et analyses, faites dans un laboratoire classique de chimie, sont relativement longues et coûteuses ; elles ne peuvent être entreprises pour un suivi régulier ou même en continu de la production. Dans ce contexte, un spectromètre en proche infrarouge (NIR) mesure l'absorbance c'est-à-dire les spectres dans les longueurs d'ondes afin de construire un modèle de prévision de la concentration en sucre.

Les données

Les données originales sont dues à Osbone et al. (1984) [6] et ont été souvent utilisées pour la comparaison de méthodes (Stone et al. 1990 [7], Brown et al. 2001 [1], Krämer et al. 2008 [4]). Elles sont accessibles dans R au sein de la librairie *ppls*. Les mesures ont été faites sur deux échantillons, l'un de taille

FIGURE 9 – Cookies : Régression *ridge*; chemin de régularisation des paramètres et optimisation de la pénalisation avec *scikit-learn* en Python.

40 prévu pour l'apprentissage, l'autre de taille 32 pour les tests. Pour chacun de ces 72 biscuits, les compositions en lipides, sucre, farine, eau, sont mesurées par une approche classique tandis que le spectre est observé sur toutes les longueurs d'ondes entre 1100 et 2498 nanomètres, régulièrement espacés de 2 nanomètres. Nous avons donc 700 valeurs observées, ou variables potentiellement explicatives, par échantillon de pâte à biscuit.

Résultats par régression pénalisée

Typiquement, cette étude se déroule dans un contexte de très grande dimension avec $p \gg n$. L'étude détaillée de ces données fait l'objet d'un **scénario** avec le logiciel R.

Voici quelques résultats partiels concernant les méthodes de régression par régression *ridge* et régression LASSO. La comparaison globale des résultats des différentes approches de modélisation est reportée en conclusion.

Références

- [1] P.J. Brown, T. Fearn et M. Vannucci, *Bayesian Wavelet Regression on*

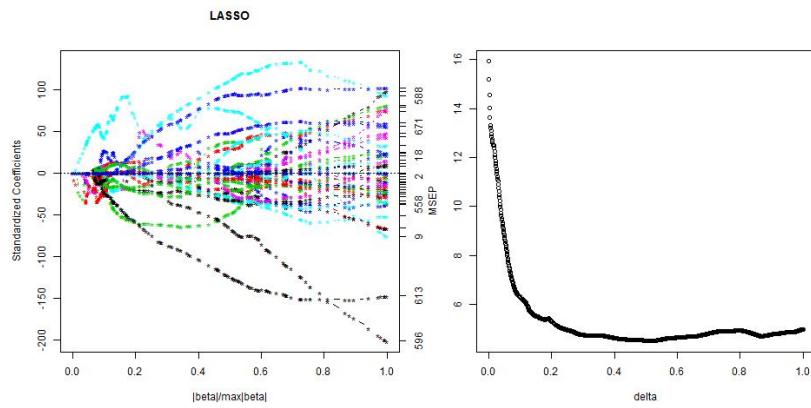


FIGURE 10 – Cookies : Régression lasso ; chemin de régularisation des paramètres et optimisation de la pénalisation.

Curves with Applications to a Spectroscopic Calibration Problem, Journal of the American Statistical Society **96** (2001), 398–408.

- [2] G. M. Furnival et R. W. Wilson, *Regression by leaps and bounds*, Technometrics **16** (1974), 499–511.
- [3] J.D. Jobson, *Applied Multivariate Data Analysis*, t. I : Regression and experimental design, Springer-Verlag, 1991.
- [4] Nicole Krämer, Anne Laure Boulesteix et Gerhard Tutz, *Penalized Partial Least Squares with applications to B-spline transformations and functional data*, Chemometrics and Intelligent Laboratory Systems **94** (2008), 60–69.
- [5] C.L. Mallows, *Some Comments on Cp*, Technometrics **15** (1973), 661–675.
- [6] B. G. Osborne, T. Fearn, A. R. Miller et S. Douglas, *Application of Near Infrared Reflectance spectroscopy to the compositional analysis of biscuits and biscuit doughs*, J. Sci. Food Agric. **35** (1984), 99–105.
- [7] M. Stone et R. J. Brooks, *Continuum regression : cross-validated sequentially constructed prediction embracing ordinary least squares, partial*

least squares and principal components regression, Journal of The Royal Statistical Society B **52** (1990), 237–269.

- [8] R. Tibshirani, *Regression shrinkage and selection via the lasso*, J. Royal Statist. Soc B **58** (1996), 267–288.

Régression logistique ou modèle binomial

Résumé

Rappels sur la régression logistique ou modèle binomial du modèle linéaire général, Les lois des observations sont discrètes et associées à des dénominations : binomiale, multinomiale. Définition de la notion de rapport de cote utile dans l'interprétation du rôle des paramètres ; modèle pour une variable binomiale ou une variable binaire (0, 1) de Bernoulli ; estimation, propriétés et difficultés spécifiques à ce modèle ; extension à la modélisation d'une variable polytomique ou ordinaire. Choix de modèle en régression logistique et exemples.

Retour à [l'introduction](#).

Tous les tutoriels sont disponibles sur le dépôt :
github.com/wikistat

1 Introduction

Historiquement, la régression logistique ou régression binomiale fut la première méthode utilisée, notamment en épidémiologie et en marketing (*scoring*), pour aborder la modélisation d'une variable binaire binomiale (nombre de succès pour n_i essais) ou de Bernoulli (avec $n_i = 1$) : décès ou survie d'un patient, absence ou présence d'une pathologie, possession ou non d'un produit, bon ou mauvais client...

Bien connue dans ces types d'application et largement répandue, la régression logistique conduit à des interprétations pouvant être complexes mais rentrées dans les usages pour quantifier, par exemple, des facteurs de risque liés à une pathologie, une faillite... Cette méthode reste donc celle la plus utilisée car interprétable même si, en terme de qualité prévisionnelle, d'autres approches sont susceptibles, en fonction des données étudiées, de conduire à de meilleures prévisions. Enfin, robuste, cette méthode passe à l'échelle des données massives. Il est donc important de bien maîtriser les différents aspects

de la régression logistiques dont l'interprétation des paramètres, la sélection de modèle par sélection de variables ou par régularisation (Lasso).

Cas particulier de modèle linéaire général, la régression logistique reprend la plupart des usages des méthodes de cette famille : estimation par maximisation de la vraisemblance, statistiques de test suivant asymptotiquement des lois du chi-deux, calcul des résidus, observations influentes, critère pénalisé (AIC) d'Akaike[1] pour la sélection de modèle.

2 Cotes et rapports de cote

Une première section définit quelques notions relatives à l'étude de la liaison entre variables qualitatives. Elles sont couramment utilisées dans l'interprétation des modèles de régression logistique.

Une variable

Soit Y une variable qualitative à J modalités. On désigne la chance, cote ou *odds*¹ de voir se réaliser la j -ème modalité plutôt que la k ème par le rapport

$$\Omega_{jk} = \frac{\pi_j}{\pi_k}$$

où π_j est la probabilité d'apparition de la j -ème modalité. Cette quantité est estimée par le rapport n_j/n_k des effectifs observés sur un échantillon. Lorsque la variable est binaire et suit une loi de Bernoulli de paramètre π , l'odds est le rapport $\pi/(1 - \pi)$ qui exprime une cote ou chance de gain.

Par exemple, si la probabilité d'un succès est 0.8, celle d'un échec est 0.2. L'odds du succès est $0.8/0.2=4$ tandis que l'odds de l'échec est $0.2/0.8=0.25$. On dit encore que la chance de succès est de 4 contre 1 tandis que celle d'échec est de 1 contre 4.

2.1 Table de contingence

On considère maintenant une table de contingence 2×2 croisant deux variables qualitatives binaires X^1 et X^2 . les paramètres de la loi conjointe se

1. Il n'existe pas, même en Québécois, de traduction consensuelle de *odds*.

mettent dans une matrice :

$$\begin{bmatrix} \pi_{11} & \pi_{12} \\ \pi_{21} & \pi_{22} \end{bmatrix}$$

où $\pi_{ij} = P[\{X^1 = i\} \text{ et } \{X^2 = j\}]$ est la probabilité d'occurrence de chaque combinaison.

- Dans la ligne 1, l'odds que la colonne 1 soit prise plutôt que la colonne 2 est :

$$\Omega_1 = \frac{\pi_{11}}{\pi_{12}}.$$

- Dans la ligne 2, l'odds que la colonne 1 soit prise plutôt que la colonne 2 est :

$$\Omega_2 = \frac{\pi_{21}}{\pi_{22}}.$$

On appelle *odds ratio* (rapport de cote) le rapport

$$\Theta = \frac{\Omega_1}{\Omega_2} = \frac{\pi_{11}\pi_{22}}{\pi_{12}\pi_{21}}.$$

Ce rapport prend la valeur 1 si les variables sont indépendantes, il est supérieur à 1 si les sujets de la ligne 1 ont plus de chances de prendre la première colonne que les sujets de la ligne 2 et inférieur à 1 sinon.

Exemple : supposons qu'à l'entrée dans une école d'ingénieurs, 7 garçons sur 10 sont reçus tandis que seulement 4 filles sur 10 le sont. La cote des garçons est alors de $0.7/0.3=2.33$ tandis que celle des filles est de $0.4/0.6=0.67$. Les rapport de cote est de $2.33/0.67=3.5$. La chance d'être reçu est 3.5 plus grande pour les garçons que pour les filles.

L'*odds ratio* est également défini pour deux lignes (a, b) et deux colonnes (c, d) quelconques d'une table de contingence croisant deux variables à J et K modalités. C'est le rapport :

$$\Theta_{abcd} = \frac{\Omega_a}{\Omega_b} = \frac{\pi_{ac}\pi_{bd}}{\pi_{ad}\pi_{bc}} \quad \text{estimé par l'odds ratio empirique} \quad \widehat{\Theta}_{abcd} = \frac{n_{ac}n_{bd}}{n_{ad}n_{bc}}.$$

3 Régression logistique

3.1 Type de données

Cette section décrit la modélisation d'une variable qualitative Z à 2 modalités : 1 ou 0, succès ou échec, présence ou absence de maladie, panne d'un

équipement, faillite d'une entreprise, bon ou mauvais client... Les modèles de régression précédents adaptés à l'explication d'une variable quantitative ne s'appliquent plus directement car le régresseur linéaire usuel $\mathbf{X}\beta$ ne prend pas des valeurs simplement binaires. L'objectif est adapté à cette situation en cherchant à expliquer les probabilités

$$\pi = P(Z = 1) \quad \text{ou} \quad 1 - \pi = P(Z = 0),$$

ou plutôt une transformation de celles-ci, par l'observation conjointe des variables explicatives. L'idée est en effet de faire intervenir une fonction réelle monotone g opérant de $[0, 1]$ dans \mathbb{R} et donc de chercher un modèle linéaire de la forme :

$$g(\pi_i) = \mathbf{x}'_i \beta.$$

Il existe de nombreuses fonctions, dont le graphe présente une forme sigmoïdale et qui sont candidates pour remplir ce rôle, trois sont pratiquement disponibles dans les logiciels :

probit : g est alors la fonction inverse de la fonction de répartition d'une loi normale, mais son expression n'est pas explicite.

log-log avec g définie par

$$g(\pi) = \ln[-\ln(1 - \pi)]$$

mais cette fonction est dissymétrique.

logit est définie par

$$g(\pi) = \text{logit}(\pi) = \ln \frac{\pi}{1 - \pi} \quad \text{avec} \quad g^{-1}(x) = \frac{e^x}{1 + e^x}.$$

Plusieurs raisons, tant théoriques que pratiques, font préférer cette dernière solution. Le rapport $\pi/(1 - \pi)$, qui exprime une cote, est l'*odds*. La *régression logistique* s'interprète donc comme la recherche d'une modélisation linéaire du *log odds* tandis que les coefficients de certains modèles expriment des *odds ratio* c'est-à-dire l'influence d'un facteur qualitatif sur le risque (ou la chance) d'un échec (d'un succès) de Z .

Cette section se limite à la description de l'usage élémentaire de la régression logistique. Des compléments concernant l'intervention de variables explicatives avec effet aléatoire, l'utilisation de mesures répétées donc dépendantes, sont à rechercher dans la bibliographie.

3.2 Modèle binomial

Pour $i = 1, \dots, I$, différentes valeurs fixées x_i^1, \dots, x_i^q des variables explicatives X^1, \dots, X^q sont observées. Ces dernières pouvant être des variables quantitatives ou qualitatives.

Pour chaque groupe, c'est-à-dire pour chacune des combinaisons de valeurs ou facteurs, sont réalisées n_i observations ($n = \sum_{i=1}^I n_i$) de la variable Z qui se mettent sous la forme $y_1/n_1, \dots, y_I/n_I$ où y_i désigne le nombre de "succès" observés lors des n_i essais. Toutes les observations sont supposées indépendantes et, à l'intérieur d'un même groupe, la probabilité π_i de succès est supposée constante. Alors, la variable Y_i sachant n_i et d'espérance $E(Y_i) = n_i\pi_i$ suit une loi binomiale $B(n_i, \pi_i)$ dont la fonction de densité s'écrit :

$$P(Y = y_i) = \binom{n_i}{y_i} \pi_i^{y_i} (1 - \pi_i)^{(n_i - y_i)}.$$

Le modèle suppose que le vecteur des fonctions *logit* des probabilités π_i appartient au sous-espace $\text{vect}\{X^1, \dots, X^q\}$ engendré par les variables explicatives :

$$\text{logit}(\pi_i) = \mathbf{x}'_i \boldsymbol{\beta} \quad i = 1, \dots, I$$

ce qui s'écrit encore

$$\pi_i = \frac{e^{\mathbf{x}'_i \boldsymbol{\beta}}}{1 + e^{\mathbf{x}'_i \boldsymbol{\beta}}} \quad i = 1, \dots, I.$$

Le vecteur des paramètres est estimé par maximisation de la log-vraisemblance. Il n'y a pas de solution analytique, celle-ci est obtenue par des méthodes numériques itératives (par exemple Newton Raphson) dont certaines (Fisher) reviennent à itérer des estimations de modèles de régression par moindres carrés généralisés avec des poids et des métriques adaptés à chaque itération.

L'optimisation fournit une estimation \mathbf{b} de $\boldsymbol{\beta}$, il est alors facile d'en déduire les estimations ou prévisions des probabilités π_i :

$$\hat{\pi}_i = \frac{e^{\mathbf{x}'_i \mathbf{b}}}{1 + e^{\mathbf{x}'_i \mathbf{b}}}$$

et ainsi celles des effectifs

$$\hat{y}_i = n_i \hat{\pi}_i.$$

Remarques

- La matrice \mathbf{X} issue de la planification expérimentale est construite avec les mêmes règles que celles utilisées dans le cadre de l'analyse de covariance mixant variables explicatives quantitatives et qualitatives. Ainsi, les logiciels gèrent avec plus ou moins de clarté le choix des variables indicatrices et donc des paramètres estimables ou contrastes associés.
- Attention**, La situation décrite précédemment correspond à l'observation de données *groupées*. Dans de nombreuses situations concrètes et souvent dès qu'il y a des variables explicatives quantitatives, les observations \mathbf{x}_i sont toutes distinctes. Ceci revient donc à fixer $n_i = 1; i = 1, \dots, I$ dans les expressions précédentes et la loi de Bernoulli remplace la loi binomiale. Certaines méthodes ne sont alors plus applicables et les comportements asymptotiques des distributions des statistiques de test ne sont plus valides car le nombre de paramètres tend vers l'infini avec n .
- Dans le cas d'une variable explicative X dichotomique, un logiciel comme SAS fournit, en plus de l'estimation d'un paramètre b , celle des odds ratios ; b est alors le log odds ratio ou encore, e^b est l'odds ratio (le rapport de cote). Ceci s'interprète en disant que Y a e^b fois plus de chance de succès (ou de maladie comme par exemple un cancer du poumon) quand $X = 1$ (par exemple pour un fumeur).
- Attention**, les différences de paramétrisation $(-1, 1)$ ou $(0, 1)$ des indicatrices explique les différences observées dans l'estimation des paramètre d'un logiciel à l'autre mais les modèles sont identiques. Mêmes exprimés dans des bases différentes, les espaces engendrés par les vecteurs des indicatrices sélectionnées sont les mêmes.

3.3 Régressions logistiques polytomique et ordinale

3.3.1 Généralisation

La régression logistique adaptée à la modélisation d'une variable dichotomique se généralise au cas d'une variable Y à plusieurs modalités. La généralisation la plus rudimentaire adoptée dans la librairie *Scikit-learn* de Python consiste à considérer autant de modèles dichotomiques que de modalités : une contre les autres.

Si ces modalités sont ordonnées, la variable est dite qualitative ordinale et la régression polythomique. Ce type de modélisation est très souvent utilisé en épidémiologie et permettent d'évaluer ou comparer des risques par exemples sanitaires. Des estimations d'odds ratio ou rapports de cotes sont ainsi utilisés pour évaluer et interpréter les facteurs de risques associés à différents types ou seuils de gravité d'une maladie ou, en marketing, cela s'applique à l'explication, par exemple, d'un niveau de satisfaction d'un client. Il s'agit de comparer entre elles des estimations de fonctions logit.

Dans une situation de *data mining* ou fouille de données, ce type d'approche se trouve lourdement pénalisé lorsque, à l'intérieur d'un même modèle polytomique ou ordinal, plusieurs types de modèles sont en concurrence pour chaque fonction logit associée à différentes modalités. Différents choix de variables, différents niveaux d'interaction rendent trop complexe et inefficace cette approche. Elle est à privilégier uniquement dans le cas d'un nombre restreint de variables explicatives.

Logits cumulatifs

À titre illustratif, explicitons le cas simple d'une variable Y à k modalités ordonnées expliquée par une seule variable dichotomique X . Notons $\pi_j(X) = P(Y = j|X)$ avec $\sum_{j=1}^k \pi_j(X) = 1$. Pour une variable Y à k modalités, il faut, en toute rigueur, estimer $k - 1$ prédicteurs linéaires :

$$g_j(X) = \alpha_j + \beta_j X \quad \text{pour } j = 1, \dots, k - 1$$

et, dans le cas d'une variable ordinale, la fonction lien logit utilisée doit tenir compte de cette situation particulière.

Dans la littérature, trois types de fonction sont considérées dépendant de l'échelle des rapports de cote adoptée :

- échelle basée sur la comparaison des catégories adjacentes deux à deux,
- sur la comparaison des catégories adjacentes supérieures cumulées,
- et enfin sur la comparaison des catégories adjacentes cumulées.

Pour $k = 2$, les trois situations sont identiques. C'est le dernier cas qui est le plus souvent adopté ; il conduit à définir les fonctions des logits cumulatifs de la forme :

$$\log \frac{\pi_{j+1} + \dots + \pi_k}{\pi_1 + \dots + \pi_j} \quad \text{pour } j = 1, \dots, k - 1.$$

Pour un seuil donné sur Y , les catégories inférieures à ce seuil, cumulées, sont comparées aux catégories supérieures cumulées. Les fonctions logit définies sur cette échelle dépendent chacune de tous les effectifs, ce qui peut conduire à une plus grande stabilité des mesures qui en découlent.

Proportionnalité des rapports de cote

Si les variables indépendantes sont nombreuses dans le modèle ou si la variable réponse Y comporte un nombre élevé de niveaux, la description des fonctions logit devient fastidieuse. La pratique consiste plutôt à déterminer un coefficient global b (mesure d'effet) qui soit la somme pondérée des coefficients b_j . Ceci revient à faire l'hypothèse que les coefficients sont homogènes (idéalement tous égaux), c'est-à-dire à supposer que les rapports de cotes sont proportionnels. C'est ce que calcule implicitement la procédure LOGISTIC de SAS appliquée à une variable réponse Y ordinaire en estimant un seul paramètre b mais $k - 1$ termes constants correspondant à des translations de la fonctions logit.

La procédure LOGISTIC fournit le résultat du test du score sur l'hypothèse H_0 de l'homogénéité des coefficients β_j .

Le coefficient b mesure donc l'association du facteur X avec la gravité de la maladie et peut s'interpréter comme suit : pour tout seuil de gravité choisi sur Y , la cote des risques d'avoir une gravité supérieure à ce seuil est e^b fois plus grande chez les exposés ($X = 1$) que chez les non exposés ($X = 0$).

3.4 Choix de modèle

Les algorithmes sont identiques à ceux décrits pour l'analyse de covariance mais, *attention*, du fait de l'utilisation d'une transformation non linéaire (logit), même si des facteurs sont orthogonaux, aucune propriété d'orthogonalité ne peut être prise en compte pour l'étude des hypothèses. Ceci impose l'élimination des termes un par un et la ré-estimation du modèle. D'autre part, un terme principal ne peut être supprimé que s'il n'intervient plus dans des termes d'interaction.

Les critères à optimiser sont les mêmes avec pour choix par défaut l'AIC en R. Dans Scikit-learn de Python, les mêmes versions de sélection par pénalisation : ridge, Lasso, elastic-net, sont proposées.

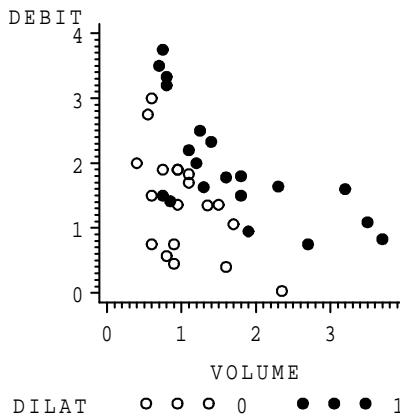


FIGURE 1 – Dilatation : Nuage des modalités de Y dans les coordonnées des variables explicatives.

4 Exemples

4.1 Exemple élémentaire avec SAS

Les données

L'influence du débit et du volume d'air inspiré impacte-t-elle l'occurrence (codée 1) de la dilatation des vaisseaux sanguins superficiels des membres inférieurs ? Un graphique élémentaire représentant les modalités de Y dans les coordonnées de $X^1 \times X^2$ est toujours instructif. Il montre une séparation raisonnable et de bon augure des deux nuages de points. Dans le cas de nombreuses variables explicatives quantitatives, une analyse en composantes principales s'impose. Les formes des nuages représentés, ainsi que l'allure des distributions (étudiées préalablement), incitent dans ce cas à considérer par la suite les logarithmes des variables. Une variable `un` ne contenant que des 1s dénombrant le nombre d'essais est nécessaire dans la syntaxe de `genmod`. Les données sont en effet non groupées.

```
proc logistic data=sasuser.debvol;
model dilat=l_debit l_volume;
run;
```

```
proc genmod data=sasuser.debvol;
model dilat/un=l_debit l_volume/d=bin;
run;
```

Criterion	Intercept		Chi-Square for Covariates
	Only	Covariates	
AIC	56.040	35.216	.
SC	57.703	40.206	.
-2 LOG L	54.040	29.216(1)	24.824 with 2 DF ($p=0.0001$)
Score	.	.	16.635 with 2 DF ($p=0.0002$)

Variable	Parameter(2)		Pr > Chi-Square	Standardized Estimate	Odds Ratio
	DF	Estimate			
INTERCPT	1	2.8782	1.3214	4.7443	0.0294
L_DEBIT	1	-4.5649	1.8384	6.1653	0.0130
L_VOLUME	1	-5.1796	1.8653	7.7105	0.0055

Cette procédure fournit des critères de choix de modèle dont la déviance (1), le vecteur b des paramètres (2) et les statistiques des tests (3) comparant le modèle excluant un terme par rapport au modèle complet tel qu'il est décrit dans la commande.

Criteria For Assessing Goodness Of Fit			
Criterion	DF	Value	Value/DF
Deviance	36	29.2156	0.8115 (1)
Scaled Deviance	36	29.2156	0.8115 (2)
Pearson Chi-Square	36	34.2516	0.9514 (3)
Scaled Pearson X2	36	34.2516	0.9514
Log Likelihood	.	-14.6078	.

Analysis Of Parameter Estimates				
Parameter	DF	Estimate (4)	Std Err	ChiSquare (5) Pr>Chi
INTERCPT	1	-2.8782	1.3214	4.7443 0.0294
L_DEBIT	1	4.5649	1.8384	6.1653 0.0130
L_VOLUME	1	5.1796	1.8653	7.7105 0.0055
SCALE (6)	0	1.0000	0.0000	.

-
- (1) Déviance du modèle par rapport au modèle saturé.
 - (2) Déviance pondérée si le paramètre d'échelle est différent de 1 en cas de sur-dispersion.
 - (3) Statistique de Pearson, voisine de la déviance, comparant le modèle au modèle saturé.
 - (4) Paramètres du modèle.
 - (5) Statistique des tests comparant le modèle excluant un terme par rapport au modèle complet.
 - (6) Estimation du paramètre d'échelle si la quasi-vraisemblance est utilisée.

4.2 Régression logistique ordinale

Les résultats d'une étude préalable à la législation sur le port de la ceinture de sécurité a été conduite dans la province de l'Alberta à Edmonton au Canada (Jobson, 1991). Un échantillon de 86 769 rapports d'accidents de voitures ont été compulsés afin d'extraire une table croisant :

1. Etat du conducteur : Normal ou Alcoolisé

2. Sexe du conducteur
3. Port de la ceinture : Oui Non
4. Gravité des blessures : 0 : rien à 3 : fatales

Les modalités de la variable à expliquer concernant la gravité de l'accident sont ordonnées. Mais dans cet exemple, l'hypothèse H_0 de proportionnalité des rapports de cote est rejetée. Le problème est alors simplifié en regroupant les conséquences de l'accident en seulement 2 modalités avec ou sans séquelles.

Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept Gr0	1	1.8699	0.0236	6264.9373	<.0001
Intercept Gr1	1	2.8080	0.0269	10914.3437	<.0001
Intercept Gr2	1	5.1222	0.0576	7917.0908	<.0001
sexe Sfem	1	-0.3118	0.0121	664.3353	<.0001
alcool A_bu	1	-0.5017	0.0190	697.0173	<.0001
ceinture Cnon	1	-0.1110	0.0174	40.6681	<.0001

Test de score pour l'hypothèse des cotes proportionnelles

Khi-2 DDL Pr > Khi-2
33.3161 6 <.0001

Modèle élémentaire Gr0 vs. GrN

Estimations des rapports de cotes		IC de Wald à 95 %
Effet	Valeur estimée	
sexe Sfem vs Shom	1.873	1.786 1.964
alcool A_bu vs Ajeu	2.707	2.512 2.918
ceinture Cnon vs Coui	1.244	1.162 1.332

4.3 Cancer du sein

Les données (Wisconsin BreastCancer Database) sont disponibles dans la librairie `mlbench` du logiciel R. Elles servent très souvent de base de référence à des comparaisons de techniques d'apprentissage. Les variables considérées sont :

- Cl.thickness** Clump Thickness
- Cell.size** Uniformity of Cell Size
- Cell.shape** Uniformity of Cell Shape
- Marg.adhesion** Marginal Adhesion
- Epith.c.size** Single Epithelial Cell Size
- Bare.nuclei** Bare Nuclei
- Bl.cromatin** Bland Chromatin
- Normal.nucleoli** Normal Nucleoli
- Mitoses** Mitoses
- Class** "benign" et "malignant".

La dernière variable est celle à prédire, les variables explicatives sont ordinaires ou nominales à 10 classes. Il reste 683 observations après la suppression de 16 présentant des valeurs manquantes.

Ce jeu de données est assez particulier car plutôt facile à ajuster. Une estimation utilisant toutes les variables conduit à des messages critiques indiquant un défaut de convergence et des probabilités exactement ajustées. En fait le modèle s'ajuste exactement aux données en utilisant toutes les variables aussi l'erreur de prévision nécessite une estimation plus soignée. Une séparation entre un échantillon d'apprentissage et un échantillon test ou une validation croisée permet une telle estimation.

Un modèle parcimonieux et obtenu par une démarche descendante minimisant le critère AIC. Ce modèle conduit à des erreurs de prévision plus faibles sur un échantillon test indépendant qu'un modèle ajustant exactement les données. La qualité de l'ajustement du modèle se résume sous la forme d'une matrice de *confusion* évaluant les taux de bien et mal classés sur l'échantillon d'apprentissage tandis que l'erreur de prévision est estimée à partir de l'échantillon test.

# erreur d'ajustement	benign	malignant
FALSE	345	6
TRUE	13	182
# erreur de prévision		
	benign	malignant
FALSE	84	5
TRUE	2	46

Le taux d'erreur apparent estimé sur l'échantillon d'apprentissage est de 3,5% (0% avec le modèle complet) tandis que le taux d'erreur estimé sans biais sur l'échantillon test est de 5,1% (5,8 avec le modèle complet).

4.4 Pic d'ozone

Plutôt que de prévoir la concentration de l'ozone puis un dépassement éventuel d'un seuil, il pourrait être plus efficace de prévoir directement ce dépassement en modélisant la variable binaire associée. Attention toutefois, ces dépassements étant relativement peu nombreux (17%), il serait nécessaire d'en

accentuer l'importance par l'introduction d'une fonction coût ou une pondération spécifique. Ceci est un problème général lorsqu'il s'agit de prévoir des phénomènes rares : un modèle trivial ne les prévoit jamais ne commettrait finalement qu'une erreur relative faible. Ceci revient à demander au spécialiste de quantifier le risque de prévoir un dépassement du seuil à tort par rapport à celui de ne pas prévoir ce dépassement à tort. Le premier a des conséquences économiques et sur le confort des usagers par des limitations de trafic tandis que le 2ème a des conséquences sur l'environnement et la santé de certaines populations. Ce n'est plus un problème statistique mais politique.

La recherche descendante d'un meilleur modèle au sens du critère d'Akaike conduit au résultat ci-dessous.

	Df	Deviance	Resid.	Df	Resid.	Dev	P(> Chi)
NULL				831	744.34		
O3_pr	1	132.89		830	611.46	9.576e-31	
vmodule	1	2.42		829	609.04	0.12	
s_rmh2o	1	33.71		828	575.33	6.386e-09	
station	4	16.59		824	558.74	2.324e-03	
TEMPE	1	129.39		823	429.35	5.580e-30	

Un modèle de régression logistique faisant intervenir les interactions d'ordre 2 et optimisé par algorithme descendant aboutit à une erreur de 10,6% tandis que le modèle quantitatif de régression quadratique du chapitre précédent conduit à une erreur de 10,1% avec le même protocole et les mêmes échantillons d'apprentissage et de test.

Matrices de confusion de l'échantillon test pour différents modèles :								
0		1		0		1		
FALSE	163	19	TRUE	162	18	FALSE	163	17
TRUE	5	21	TRUE	6	22	TRUE	5	23
logistique sans vmodule			avec vmodule			avec interactions		quantitatif

Les matrices de confusion ne sont pas symétriques et affectées du même biais : tous ces modèles oublient systématiquement plus de dépassements de seuils qu'ils n'en prévoient à tort. Une analyse plus poussée de l'estimation de l'erreur de prévision est évidemment nécessaire. À ce niveau de l'étude, ce qui est le plus utile au météorologue, c'est l'analyse des coefficients les plus significativement présents dans la régression quadratique, c'est-à-dire avec les interactions. Ils fournissent des indications précieuses sur les faiblesses ou insuffisances du modèle physique.

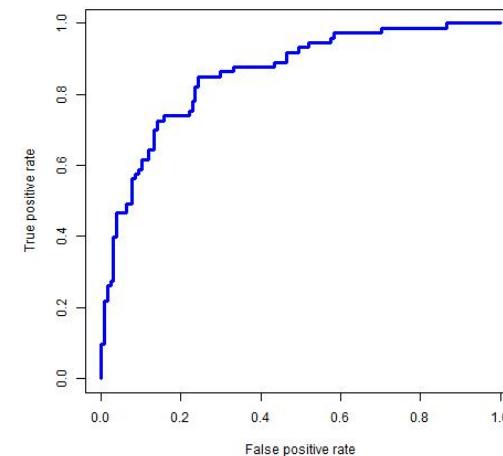


FIGURE 2 – Données bancaires : estimation sur l'échantillon test de la courbe ROC associée à la régression logistique.

4.5 Données bancaires

Il s'agit de modéliser une variable binaire représentant la possession ou non de la carte visa premier en fonction du comportement bancaire d'un client. Cet exemple est typique de la construction d'un score d'appétence en marketing quantitatif. Comme dans l'exemple précédent, la possession de ce type de produit est rare aussi, un échantillon spécifique, non représentatif, a été construit en sur-représentant cette possession.

Plusieurs stratégies peuvent être mises en œuvre sur ces données selon les transformations et codages réalisés sur les variables qualitatives. Elles sont explorées en R et en Python dans deux calepins ; avec minimisation de l'AIC en R, pénalisation Lasso en Python.

Dans ce type d'application, il est très classique d'estimer la courbe ROC sur l'échantillon test afin de calibrer le seuil en fonction des objectifs du service marketing plutôt que de le laisser par défaut à 0,5.

Références

- [1] H. Akaïke, *A new look at the statistical model identification*, IEEE Transactions on Automatic Control **19** (1974).

Composantes principales et régressions PLS parcimonieuses

Résumé

L'introduction de pénalisations en norme L_1 induit une sélection de variables optimale en régression. Même si, numériquement, ce n'est pas indispensable pour les méthodes de régression ou projection sur composantes orthogonales avec réduction de dimension, le même type de pénalisation est introduit afin de simplifier la construction des composantes et donc leur interprétation lorsque le nombre de variables est important. Cette démarche conduit à la définition de versions parcimonieuses de l'Analyse en Composantes Principales et de la régression PLS pour différents objectifs : exploration, comparaison ou intégration de deux jeux de données en version régression ou canonique, analyse discriminante PLS.

[Retour au plan du cours](#)

1 Introduction

1.1 Objectif

L'intérêt principal des méthodes de cette vignette réside dans leur capacité à prendre en compte des données de grande dimension et même de très grande dimension lorsque le nombre de variables p est largement plus grand que le nombre d'individus n : $p \gg n$. La sélection de variables devient inefficace et même ingérable par les algorithmes usuels. La construction d'un modèle de régression requiert alors une pénalisation (*ridge*, *Lasso*, *elastic net*) ou une réduction de dimension : régression sur composantes principales ou régression PLS.

1.2 Régression sur composantes principales

La régression sur composantes principales ou PCR est simple par son principe et sa mise en œuvre. L'objectif est de résumer l'ensemble des variables X^1, \dots, X^p par un sous-ensemble de variables Z^1, \dots, Z^r deux à deux orthogonales et combinaisons linéaires des variables X^1, \dots, X^p . Avec $r = p$ il n'y a pas de réduction de dimension et le même ajustement qu'en régression classique est obtenu : même espace de projection engendré. Les variables Z^1, \dots, Z^p sont simplement les composantes principales associées des variables X^1, \dots, X^p obtenues par l'[analyse en composantes principales](#) ou encore la décomposition en valeurs singulières de la matrice \mathbf{X} . Pour éviter les problèmes d'unité et l'influence d'une hétérogénéité des variances, les variables sont centrées et réduites ; c'est donc l'ACP réduite qui est calculée.

La première composante $Z^1 = \sum_{j=1}^p \alpha_j X^j$ est de variance maximale la première valeur propre λ_1 de la matrice des corrélations avec $\sum \alpha_j^2 = 1$. Tandis que Z^m est combinaison linéaire de variance maximale λ_j et orthogonale à Z^1, \dots, Z^{m-1} .

La PCR considère un prédicteur de la forme :

$$\hat{Y}^{PCR} = \sum_{m=1}^r \hat{\theta}_m Z^m$$

avec

$$\hat{\theta}_m = \frac{\langle Z^m, Y \rangle}{\|Z^m\|^2}$$

obtenu par une procédure classique de régression.

Le choix $r = p$ redonne l'estimateur des moindres carrés car le même espace est engendré tandis que $r < p$ élimine les composantes de variances nulles ou très faibles et donc résout par là les problèmes de colinéarité même dans les cas extrêmes où ($p > n$). Le choix de r est optimisé de par validation croisée.

Bien évidemment, l'interprétation des composantes est rendu difficile si p est grand. La PCR est à rapprocher de la régression *ridge* qui seuille les coefficients des composantes principales tandis que la PCR annule ceux d'ordre supérieur à r .

Le principal *Problème* posée par la PCR est que les premières composantes, associées aux plus grandes valeurs propres, ne sont pas nécessairement corrélées avec Y et ne sont donc pas nécessairement les meilleures candidates pour résumer ou modéliser Y .

Cette remarque justifie les développements de la régression PLS ou *partial least square*.

1.3 Régression PLS

La régression PLS (*partial least square*) est une méthode ancienne (Wold, 1966)[10] largement utilisée, notamment en chimiométrie dans l'agro-alimentaire lors de l'analyse de données spectrales (Near Infra-Red ou HPLC) discrétisées et donc toujours de grande dimension. La régression PLS s'avère concrètement une méthode efficace qui justifie son emploi très répandu mais présente le défaut de ne pas se prêter à une analyse statistique traditionnelle qui exhiberait les lois de ses estimateurs. Elle est ainsi restée un marge des approches traditionnelles de la Statistique mathématique.

Différentes version de régression PLS sont proposées en fonction de l'objectif poursuivi ; consulter Tenenhaus (1998)[8] pour une présentation détaillée :

PLS1 Une variable cible Y quantitative est à expliquer, modéliser, prévoir par p variables explicatives quantitatives X^j .

PLS2 Version canonique. Mettre en relation un ensemble de q variables quantitatives Y^k et un ensemble de p variables quantitatives X^j .

PLS2 Version régression. Chercher à expliquer, modéliser un ensemble de q variables Y^k par un ensemble de p variables explicatives quantitatives X^j .

PLS-DA Version discriminante. Cas particulier du cas précédent. La variable Y qualitative à q classes est remplacée par q variables indicatrices (*dummy variables*) de ces classes.

Une application utile de la PLS2 en version canonique s'opère, par exemple en Biologie à haut débit, dans la comparaison de deux plates-formes ou deux technologies de mesures sur le même échantillon : Affymetrix vs. Agilent ou encore entre les résultats obtenus par séquençage (RNA Seq) et biopuces. Toujours en Biologie, la PLS2 en version régression permet d'intégrer des jeux de données observées à des niveaux différents sur le même échantillon : expliquer

par exemple un ensemble de métabolites ou de phénotypes par des transcrits. Des applications industrielles se trouvent, par exemple, en suivi de la qualité : plusieurs variables de mesure de la qualité ou de défaillances expliquées par un ensemble de mesures du procédé de fabrication.

Dans un objectif seulement prévisionnel, l'approche PLS s'avère plutôt efficace mais, si l'objectif est aussi la recherche d'une interprétation, c'est-à-dire nécessairement la recherche des variables les plus pertinentes parmi un très grand nombre, les composantes obtenues sont difficilement exploitables. C'est pourquoi il a été proposé (Lê Cao et al. 2008[5], 2009[4], 2011[3]) de coupler les deux approches : pénalisation L_1 de type Lasso pour une sélection des variables utilisées dans la construction des composantes orthogonales. Cette démarche passe par l'utilisation d'un algorithme parcimonieux (Shen et Huang, 2008)[7] de SVD (décomposition en valeur singulière). Celui-ci permet, à la fois, de définir des versions parcimonieuses de l'ACP et aussi de la PLS en remarquant que l'algorithme de la PLS peut être défini comme une succession de premières étapes de SVD.

L'objectif principal est donc la construction de versions parcimonieuses (en anglais *sparse*) des différentes méthodes de régression PLS. Aux résultats numériques, éventuellement de prévision, s'ajoutent des *représentations graphiques* en petite dimension très utiles pour aider à l'interprétation.

2 Régression PLS

Quelques rappels pour introduire cette méthode largement employée pour traiter les situations présentant une forte multicolinéarité et même lorsque le nombre d'observations est inférieur au nombre de variables explicatives.

2.1 Régression PLS1

Une variable cible Y quantitative est à expliquer, modéliser, prévoir par p variables explicatives quantitatives X^j . Comme pour la régression sur composantes principales, le principe est de rechercher un modèle de régression linéaire sur un ensemble de composantes orthogonales construites à partir de combinaisons linéaires des p variables explicatives centrées X^j . Dans le cas de la PLS, la construction des composantes est optimisée pour que celles-ci soient les plus liées à la variable Y à prédire au sens de la covariance empi-

rique, alors que les composantes principales ne visent qu'à extraire une part de variance maximale sans tenir compte d'une variable cible.

Soit $\mathbf{X}(n \times p)$ la matrice des variables explicatives centrées avec n pouvant être inférieur à p . On cherche une matrice \mathbf{U} de coefficients ou pondérations (*loading vectors*) définissant les r composantes Ξ_h (ou variables latentes) par combinaisons linéaires des variables X_j :

$$\Xi = \mathbf{X}\mathbf{U}.$$

La matrice \mathbf{U} est solution du problème suivant :

$$\begin{aligned} \text{Pour } h = 1, \dots, r, \quad \mathbf{u}_h &= \arg \max_{\mathbf{u}} \text{Cov}(Y, \Xi_h)^2 \\ &= \arg \max_{\mathbf{u}} \mathbf{u}' \mathbf{X}' \mathbf{Y} \mathbf{Y}' \mathbf{X} \mathbf{u} \\ \text{Avec } \mathbf{u}_h' \mathbf{u}_h &= 1 \\ \text{et } \xi_h' \xi_h &= \mathbf{u}' \mathbf{X}' \mathbf{Y} \mathbf{Y}' \mathbf{X} \mathbf{u} = 0, \quad \text{pour } \ell = 1, \dots, h-1. \end{aligned}$$

La matrice \mathbf{U} est obtenue par la démarche itérative de l'algorithme 0 ; il suffit ensuite de calculer la régression de Y sur les r variables ξ_h centrées, appelées *variables latentes* ainsi construites. Le choix du nombre de composantes r est optimisé par validation croisée.

Algorithm 1 Régression PLS1

\mathbf{X} matrice des variables explicatives centrées,
Calcul de la matrice \mathbf{U} des coefficients.

for $h = 1$ à r **do**

$$\mathbf{u}_h = \frac{\mathbf{X}' \mathbf{Y}}{\|\mathbf{X}' \mathbf{Y}\|},$$

$$\xi_h = \mathbf{X} \mathbf{u}_h$$

Déflation de \mathbf{X} : $\mathbf{X} = \mathbf{X} - \xi_h \xi_h' \mathbf{X}$

end for

Exemple de PLS1 sur les données de cancer de la prostate

La figure 1 donne l'estimation par validation croisée (10-fold) de l'erreur de prévision en fonction de la dimension tandis que la figure 2 (gauche) est une aide à l'interprétation. Les *loadings* sont les coefficients ou importance

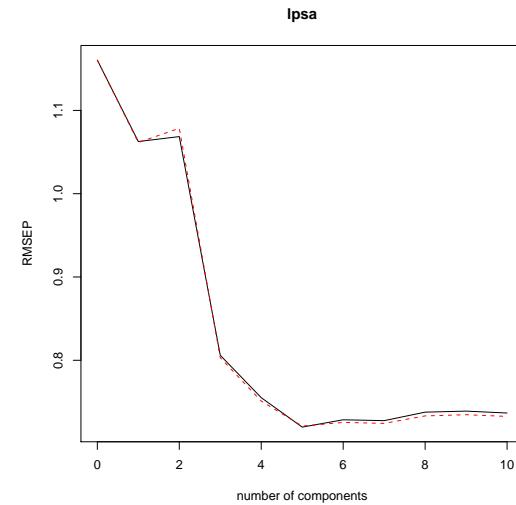


FIGURE 1 – Données cancer : optimisation du nombre de composantes en PLS1

des variables sur la première composante PLS. Le graphe de droite de la figure 2 indique simplement la plus ou moins bonne qualité de l'ajustement avec un choix de 6 composantes PLS.

2.2 Régression PLS2

Définition

L'algorithme précédent de PLS1 se généralise à une variable à expliquer Y multidimensionnelle (PLS2) : Mettre en relation ou chercher à expliquer, modéliser un ensemble de q variables Y^k par un ensemble de p variables explicatives X^j . Le critère à optimiser devient une somme des carrés des covariances entre une composante et chacune des variables réponses. Plusieurs variantes de la régression PLS multidimensionnelle ont été proposées ; le même critère est optimisé mais sous des contraintes différentes. La version *canonique* (par référence à l'analyse canonique de deux ensembles de variables), où les

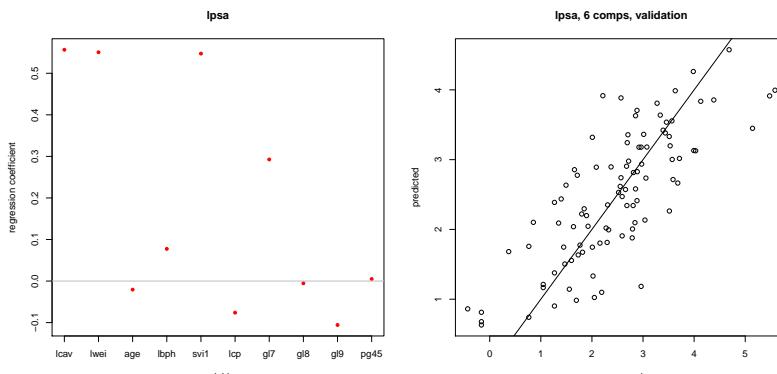


FIGURE 2 – Données cancer : Coefficient (loadings) des variables sur la première composante et qualité de l’ajustement avec 6 composantes.

deux ensembles de données jouent des rôles symétriques, diffère de la version *régression* (un paquet de variable expliqué par un autre) par l’étape dite de déflation de l’algorithme général de PLS.

Dans les deux cas, la PLS se définit par la recherche (cf. 3) de :

- variables latentes ξ_h et ω_h , ($h = 1, \dots, r$)

$$\xi_1 = \mathbf{X}\mathbf{u}_1 \text{ et } \omega_1 = \mathbf{Y}\mathbf{v}_1$$

solutions de

$$\max_{\|\mathbf{u}\|=\|\mathbf{v}\|=1} \text{cov}(\mathbf{X}\mathbf{u}, \mathbf{Y}\mathbf{v}),$$

- puis itérations sous contraintes d’orthogonalité par déflations de \mathbf{X} et \mathbf{Y} .
- Les vecteurs de coefficients $(\mathbf{u}_h, \mathbf{v}_h)_{h=1, \dots, r}$ sont appelés vecteurs *loadings*.

Tous ces vecteurs sont schématisés dans la figure 3.

Deux types de déflation sont considérés, l’un faisant jouer un rôle symétrique entre les variables (mode canonique), tandis que l’autre suppose que les variables X sont expliquées par celles Y . La régression PLS est à rapprocher

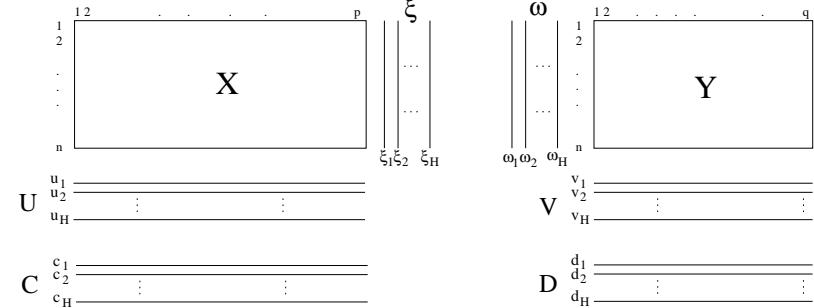


FIGURE 3 – PLS2 : les matrices \mathbf{X} and \mathbf{Y} sont successivement décomposées en ensembles de coefficients (loading vectors) $(\mathbf{u}_1, \dots, \mathbf{u}_r)$, $(\mathbf{v}_1, \dots, \mathbf{v}_r)$ et ensembles de variables latentes (ξ_1, \dots, ξ_r) , $(\omega_1, \dots, \omega_r)$, où r est la dimension recherchée ou nombre de composantes.

de l’analyse canonique des corrélations qui s’utilise dans le même contexte de deux variables multidimensionnelles X et Y à mettre en relation. La différence vient du critère optimisé en analyse canonique qui est la corrélation entre les variables latentes plutôt que la covariance :

$$\max_{\|\mathbf{u}\|=\|\mathbf{v}\|=1} \text{cor}(\mathbf{X}\mathbf{u}, \mathbf{Y}\mathbf{v}).$$

Cette optimisation requiert l’inversion des matrices $\mathbf{X}'\mathbf{X}$ et $\mathbf{Y}'\mathbf{Y}$. Ces inversions sont impossibles en cas de colinéarité des variables et donc évidemment si $n < p$ ou $n < q$. Une version régularisée ou *ridge* de l’analyse canonique rend les calculs possibles (Gonzales et al. 2008) mais les interprétations restent difficiles pour des grandes valeurs de p ou q .

Algorithme

Historiquement, la régression PLS est construite par l’algorithme NIPALS (Non linear Iterative PArtial Least Square algorithm) (cf. 2) dans lequel chaque itération h , $h = 1, \dots, r$ de l’algorithme décompose \mathbf{X} et \mathbf{Y} en faisant intervenir une étape de déflation spécifique à l’objectif.

Cet algorithme, en itérant des régressions partielles, présente de nombreux avantages. Il n’est pas nécessaire d’inverser une matrice comme en analyse

canonique ; de plus il accepte des données manquantes et même propose, par la PLS, une méthode d'imputation de celles-ci.

Algorithm 2 NIPALS

X et **Y** matrices des données centrées

Initialiser ω_1 par la première colonne de **Y**

for $h = 1$ à r **do**

while Convergence pas atteinte **do**

$$\mathbf{u}_h = \mathbf{X}'\boldsymbol{\omega}_h / \boldsymbol{\omega}_h'\boldsymbol{\omega}_h$$

$\mathbf{u}_h = \mathbf{u}_h / \mathbf{u}_h'$ \mathbf{u}_h est le vecteur *loading* associé à **X**

$\boldsymbol{\xi}_h = \mathbf{X}\mathbf{u}_h$ est la *variable latente* associée à **X**

$$\mathbf{v}_h = \mathbf{Y}'\boldsymbol{\xi}_h / (\boldsymbol{\xi}_h'\boldsymbol{\xi}_h)$$

$\mathbf{v}_h = \mathbf{v}_h / \mathbf{v}_h'$ \mathbf{v}_h est le vecteur *loading* associé à **Y**

$\boldsymbol{\omega}_h = \mathbf{Y}'\mathbf{v}_h$ est la variable latente associée à **Y**

end while

$\mathbf{c}_h = \mathbf{X}'\boldsymbol{\xi} / \boldsymbol{\xi}'\boldsymbol{\xi}$ régression partielle de **X** sur $\boldsymbol{\xi}$

$\mathbf{d}_h = \mathbf{Y}'\boldsymbol{\omega} / \boldsymbol{\omega}'\boldsymbol{\omega}$ régression partielle de **Y** sur $\boldsymbol{\omega}$

 Résidus $\mathbf{X} \leftarrow \mathbf{X} - \boldsymbol{\xi}\mathbf{c}'$ ou *déflation*

 Résidus $\mathbf{Y} \leftarrow \mathbf{Y} - \boldsymbol{\omega}\mathbf{d}'$ ou *déflation*

end for

Le nombre r d'itérations est à fixer ou optimiser par l'utilisateur tandis que la convergence de chaque étape h est analogue à celle, relativement rapide (moins d'une dizaine d'itérations), d'un algorithme de puissance itérée. En effet, à la convergence, les vecteurs vérifient :

$$\mathbf{Y}\mathbf{Y}'\mathbf{X}\mathbf{X}'\mathbf{u} = \lambda\mathbf{u}$$

$$\mathbf{Y}'\mathbf{X}\mathbf{X}'\mathbf{Y}\boldsymbol{\omega} = \lambda\boldsymbol{\omega}$$

$$\mathbf{X}\mathbf{X}'\mathbf{Y}\mathbf{Y}'\mathbf{v} = \lambda\mathbf{v}$$

$$\mathbf{X}'\mathbf{Y}\mathbf{Y}'\mathbf{X}\boldsymbol{\xi} = \lambda\boldsymbol{\xi}$$

où \mathbf{u} , $\boldsymbol{\omega}$, \mathbf{v} et $\boldsymbol{\xi}$ sont donc les vecteurs propres respectifs des matrices $\mathbf{Y}\mathbf{Y}'\mathbf{X}\mathbf{X}'$, associés à la même plus grande valeur propre λ . L'étape de déflation permet donc de calculer successivement les vecteurs propres associés aux valeurs propres décroissantes.

En résumé,

- La régression PLS2 gère des données incomplètes, bruitées, colinéaires ou de très grande dimension
- calcule les variables latentes $\boldsymbol{\xi}_h$ et $\boldsymbol{\omega}_h$ qui renseignent (graphes) sur les similarités et/ou dissimilarités des observations,
- et les vecteurs *loading* \mathbf{u}_h et \mathbf{v}_h qui renseignent sur l'importance des variables X_j et Y_k ,
- trace les Graphes illustrant les covariations des variables.

Variante de l'algorithme

Une autre approche consiste à calculer directement les vecteurs propres de la matrice $\mathbf{X}'\mathbf{Y}\mathbf{Y}'\mathbf{X}$ ou encore et c'est équivalent, les valeurs et vecteurs singuliers de la décomposition en valeurs singulières (SVD) de la matrice $\mathbf{X}'\mathbf{Y}$. Néanmoins, la perspective de gérer les données manquantes ou encore celle de réaliser les calculs sans avoir à stocker des matrices $p \times p$ pour p très grand, rend l'algorithme NIPALS tout à fait pertinent même s'il est numériquement moins performant.

PLS mode Régression vs. canonique

Deux modes de déflation sont proposés selon que les variables jouent un rôle symétrique ou que les variables X sont supposées expliquées par celles Y .

- Mode “canonique” : $\mathbf{X}_h = \mathbf{X}_{h-1} - \boldsymbol{\xi}_h\mathbf{c}'_h$ et $\mathbf{Y}_h = \mathbf{Y}_{h-1} - \boldsymbol{\omega}_h\mathbf{d}'_h$
- Mode “régression” : $\mathbf{X}_h = \mathbf{X}_{h-1} - \boldsymbol{\xi}_h\mathbf{c}'_h$ et $\mathbf{Y}_h = \mathbf{Y}_{h-1} - \boldsymbol{\xi}_h\mathbf{v}'_h$

La PLS en mode canonique poursuit donc le même objectif que l'analyse canonique des corrélations en rendant les calculs possibles même si $p > n$ car la PLS ne nécessite pas l'inversion des matrices de corrélation. Toujours avec le même objectif de rendre possible les calculs, des versions régularisées (norme L_2) de l'analyse canonique ont été proposées de façon analogue à la régression ridge. Néanmoins, cette approche conduit à des graphiques et interprétations difficiles lorsque p est grand.

PLS-DA ou discrimination PLS

La régression PLS peut facilement s'adapter au cas de la classification supervisée, ou analyse discriminante décisionnelle (*PLS-Discriminant Analysis*), dans lequel p variables quantitatives X^j expliquent une variable qualitative Y à m modalités. Il suffit de générer le paquet des m variables indicatrices

ou *dummy variables* Y^k et d'exécuter l'algorithme PLS2 (mode régression) en considérant comme quantitatives ces variables indicatrices. Le choix du nombre de dimensions peut être optimisé en minimisant l'erreur de prévision des classes par validation croisée.

2.3 Représentations graphiques

Les représentations graphiques des individus, comme celles des variables initiales, sont analogues à celles obtenues en [analyse canonique](#).

- Les variables initiales sont représentées par leurs coefficients sur les variables latentes ;
- les individus par leurs valeurs sur les composantes de X (ou de Y) comme en ACP.

3 Méthodes parcimonieuses

3.1 Objectif

La régression PLS est une régression sur composantes orthogonales qui résout efficacement les problèmes de multicolinéarité ou de trop grand nombre de variables en régression comme en analyse canonique. La contre partie, ou prix à payer, est l'accroissement souvent rédhibitoire de la complexité de l'interprétation des résultats. En effet, chaque composante est obtenue par combinaison linéaire d'un nombre pouvant être très important de l'ensemble des p variables.

Pour aider à l'interprétation, l'objectif est donc de limiter, ou contraindre, le nombre de variables participant à chaque combinaison linéaire. La façon simple de procéder est d'intégrer une contrainte de type Lasso dans l'algorithme PLS2. Plusieurs approches ont été proposées, celle décrite ci-après s'avère rapide et efficace.

3.2 Sparse SVD

La démarche adoptée est issue d'une construction d'une version parcimonieuse de l'ACP proposée par Shen et Huang (2008)[7]. Considérant que l'ACP admet pour solution la décomposition en valeurs singulières (SVD) de la matrice centrée $\bar{\mathbf{X}}$, la *sparse PCA* (s-PCA) est basée sur un algorithme qui

résout le problème :

$$\min_{\mathbf{u}, \mathbf{v}} \|\mathbf{M} - \mathbf{u}\mathbf{v}'\|_F^2 + P_\lambda(\mathbf{v})$$

où le vecteur \mathbf{v} contient les paramètres des combinaisons linéaires des variables initiales. Une pénalisation de type L_1 ($\lambda\|\mathbf{v}\|_1$) conduit à l'annulation des paramètres les plus petits pour ne laisser qu'un ensemble restreint de paramètres non-nuls dont l'effectif dépend directement de la valeur λ de la pénalisation.

Algorithm 3 sparse SVD

```
Décomposer  $\mathbf{M} = \mathbf{U}\Delta\mathbf{V}'$ 
 $\mathbf{M}_0 = \mathbf{M}$ 
for  $h$  de 1 à  $r$  do
    Fixer  $v_{old} = \delta_h v_h^*$ 
     $u_{old} = u_h^*$  avec  $v_h^*$  et  $v_h^*$  de norme 1
    while Pas de convergence de  $u_{new}$  et  $v_{new}$  do
         $v_{new} = g_\lambda(\mathbf{M}'_{h-1} u_{old})$ 
         $u_{new} = \mathbf{M}'_{h-1} v_{new} / \|\mathbf{M}'_{h-1} v_{new}\|$ 
         $u_{old} = u_{new}$ ,  $v_{old} = v_{new}$ 
    end while
     $v_{new} = v_{new} / \|v_{new}\|$ 
     $\mathbf{M}_h = \mathbf{M}_{h-1} - \delta_h u_{new} v_{new}'$ 
end for
```

L'algorithme peut adopter différents types de fonction de pénalisation, celle retenue est une fonction de seuillage "doux" avec

$$g_\lambda(y) = sign(y)(|y| - \lambda)_+.$$

3.3 Sparse PLS

Ayant remarqué qu'un étape h de PLS2 est la première étape de la décomposition en valeur singulière de la matrice $\mathbf{M}_h = \mathbf{X}'_h \mathbf{Y}_h$, la version parcimonieuse de la PLS2 est simplement construite en itérant r fois l'algorithme de *sparse SVD* (s-SVD) qui cherche à résoudre :

$$\min_{\mathbf{u}_h, \mathbf{v}_h} \|\mathbf{M}_h - \mathbf{u}_h \mathbf{v}_h'\|_F^2 + P_{\lambda_1}(\mathbf{u}_h) + P_{\lambda_2}(\mathbf{v}_h).$$

Comme pour l'algorithme de *sparse-SVD*, une pénalisation de type L_1 ($\lambda \|\mathbf{v}\|_1$) conduit à l'annulation des paramètres les plus petits pour ne laisser qu'un ensemble restreint de paramètres non-nuls dont l'effectif dépend directement des valeurs λ_1 et λ_2 de pénalisation.

Plus précisément, l'algorithme adopte pour pénalisation des fonctions de seuillage “doux” composante par composante avec

$$\begin{aligned} P_{\lambda_1}(\mathbf{u}_h) &= \sum_{j=1}^p sign(\mathbf{u}_{hj})(|\mathbf{u}_{hj}| - \lambda_1)_+ \\ P_{\lambda_2}(\mathbf{v}_h) &= \sum_{j=1}^q sign(\mathbf{v}_{hj})(|\mathbf{v}_{hj}| - \lambda_2)_+. \end{aligned}$$

Entre deux étapes de s-SVD, les matrices \mathbf{X}_h et \mathbf{Y}_h subissent une déflation (mode régression ou canonique) avant de passer à l'étape suivante.

Cette démarche soulève des questions délicates d'optimisation du nombre r de dimensions et celle des valeurs des paramètres de la fonction de pénalisation. En mode régression (PLS2 ou PLS-DA) il est possible d'optimiser ces choix en minimisant des erreurs de prévision estimées par validation croisée. En mode canonique, le “degré” de parcimonie comme le nombre de dimensions doivent être fixés *a priori* par l'utilisateur. Plus concrètement, ce sont souvent des choix *a priori* qui sont opérés en fonction de l'objectif de l'utilisateur : recherche de peu de variables assimilées, par exemple, à des biomarqueurs ou de “beaucoup” de variables dans le cadre d'une tentative de compréhension globale de la structure des données. De même, le nombre de composantes r est choisi avec une valeur réduite afin de construire des représentations graphiques $r \leq 3$ plus élémentaire pour aider à l'interprétation.

En résumé, ce sont donc les capacités d'interprétation d'un problème qui guident concrètement le choix à moins qu'un objectif de construction d'un meilleur modèle de prévision conduisent à une optimisation par validation croisée.

Dans le cas particulier de PLS-DA, la sélection de variables s'opère sur le seul ensemble des variables X et donc un seul paramètre λ est à régler.

Attention, les variables latentes successivement calculées perdent leur propriété de stricte orthogonalité du fait de la pénalisation. Cela ne s'est pas avéré

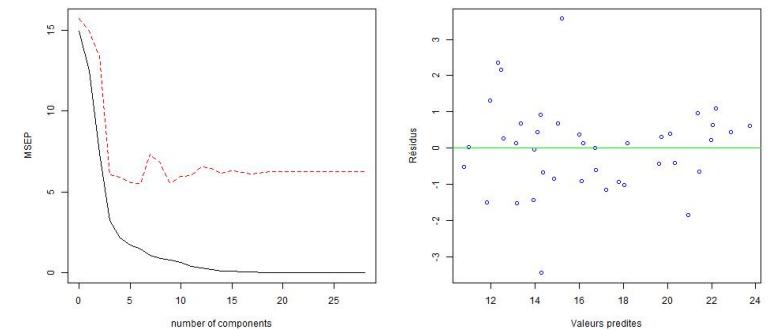


FIGURE 4 – Cookies : Optimisation du nombre de composante en régression PLS par validation croisée et graphe des résidus calculés sur l'échantillon test.

gênant sur les quelques premières dimensions et donc composantes calculées en pratique.

4 Exemples

4.1 PLS1 de données de spectrométrie NIR

Les données (*cookies*) sont celles étudiées par *régression pénalisée*. Comme pour les autres techniques, le paramètre de complexité, ici le nombre de composantes, est optimisé par validation croisée. Le graphe de la figure 4 montre l'évolution de l'erreur quadratique (ou risque) d'apprentissage (en noir) et de celle estimée par validation croisée (en rouge).

Une fois la dimension optimale déterminée, les prévisions des taux de sucre sont calculés pour l'échantillon test afin d'obtenir le graphe des résidus.

4.2 sPLS de données simulées

Le modèle de simulation est celui proposé par (Chun et Keles, 2010)[2]. Les données générées permettent de voir le rôle de la pénalisation dans la sélection des variables en PLS mode canonique. Elles sont constituées de

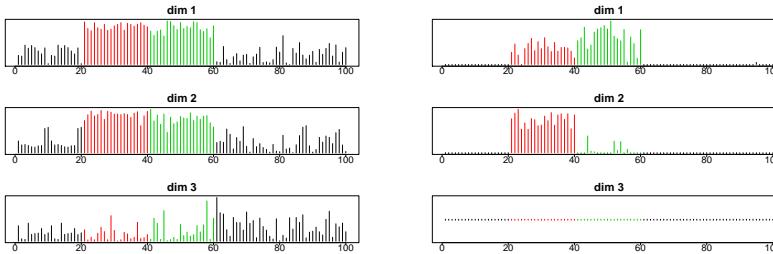


FIGURE 5 – Effet de la pénalisation sur les vecteurs “loading” associés à la matrice \mathbf{X} ; PLS à gauche et sPLS à droite.

CO	RE	OV	BR	PR	CNS	LEU	ME
7	8	6	8	2	9	6	8

TABLE 1 – Effectifs des répartitions des échantillons des lignées cellulaires en 8 types de cancer et 3 types de cellules : épithéliales, mésenchymales, mélanomes

- $n = 40, p = 5000$ (X var.), $q = 50$ (Y var.)
- 20 variables X et 10 variables Y d’effet μ_1
- 20 variables X et 20 variables Y d’effet μ_2

4.3 Analyse canonique par sPLS2

Les données (NCI) concernent 60 lignées cellulaires de tumeurs. L’objectif est de comparer deux plate-formes. Sur la première (*cDNA chip*) ont été observées les expressions de $p = 1375$ gènes tandis que sur la 2ème (*Affymetrix*) ce sont $q = 1517$ gènes qui sont concernés. Une grande majorité des gènes, sont communs aux deux tableaux $\mathbf{X}(60 \times 1375)$ et $\mathbf{Y}(60 \times 1517)$.

Les deux technologies de mesure d’expression des gènes conduisent-elles à des résultats globalement comparables pour l’étude de ces lignées cellulaires cancéreuses ?

4.4 Recherche de bio-marqueurs par sPLS-DA

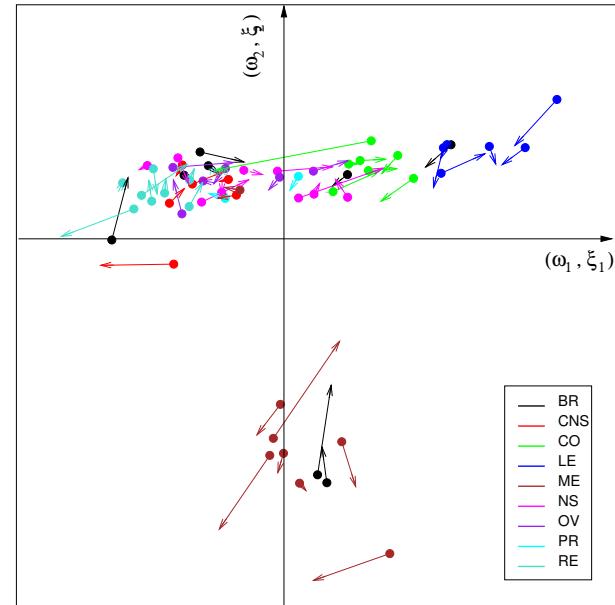


FIGURE 6 – Les “individus “lignées cellulaires” sont représentées dans les deux espaces : (ξ_1, ω_1) vs. (ξ_2, ω_2) . La longueur de chaque vecteur souligne l’impact de la technologie utilisée sur chaque type de cellule.

Les données

Les qualités prédictives de la sPLS-DA peuvent-être comparées avec celles des autres méthodes abordées dans ce cours, notamment celles d'agrégation de modèles. Lê Cao et al. (2011)[3] ont mené cette comparaison systématique sur un ensemble de jeux de données publiques dont des données relatives à la discrimination de 5 types de cancer du cerveau. Les expressions de $p = 6144$ gènes sont observés sur $n = 90$ individus.

La question est donc de savoir si la mesure des expressions d'une sélection de gènes est pertinente pour aider à diagnostiquer ces différents types de cancer. Les gènes les plus discriminants au sens de la régressions sPLS-DA sont représentés par leur coefficients (*loadings*) dans la base des deux premières composantes (figure 7). Le réseau (figure 8) est celui des gènes connus dans la littérature pour intervenir sur ces pathologies.

Parmi les gènes sélectionnés par la sPLS-DA (figure 7), une couleur particulière est attribuée à ceux déjà connus et présents dans le réseau.

5 Robustesse d'une sélection

5.1 Principe

Le grand nombre de variables au regard de la taille de l'échantillon soulève quelques doutes quand à la robustesse ou la stabilité d'une sélection de variables au sein d'un modèle; n'est-elle pas finalement qu'un artefact lié à l'échantillon observé ?

Bach (2008)[1] d'une part, Meinshausen et Bühlmann (2008)[6] d'autre part ont proposé des stratégies pour évaluer cette stabilité, éventuellement l'optimiser en les utilisant pour régler le paramètre de pénalisation. Ils partent de la même idée : étudier les occurrences ou non des sélections des variables dans un modèle pour une pénalisation donnée sur des échantillons bootstrap (Bach, 2008)[1] ou sur des sous-échantillons aléatoires (Meinshausen et Bühlmann, 2010)[6]. Ils étudient ces stratégies dans le cas du modèle linéaire avec pénalisation Lasso et montrent dans ce cas des propriétés asymptotiques de convergence vers la bonne sélection. Bach (2008)[1] s'intéresse à la sélection obtenue par intersection de toutes les sélections sur chacun des échantillons bootstrap tandis que Meinshausen et Bühlmann (2010)[6] compte le nombre de fois où

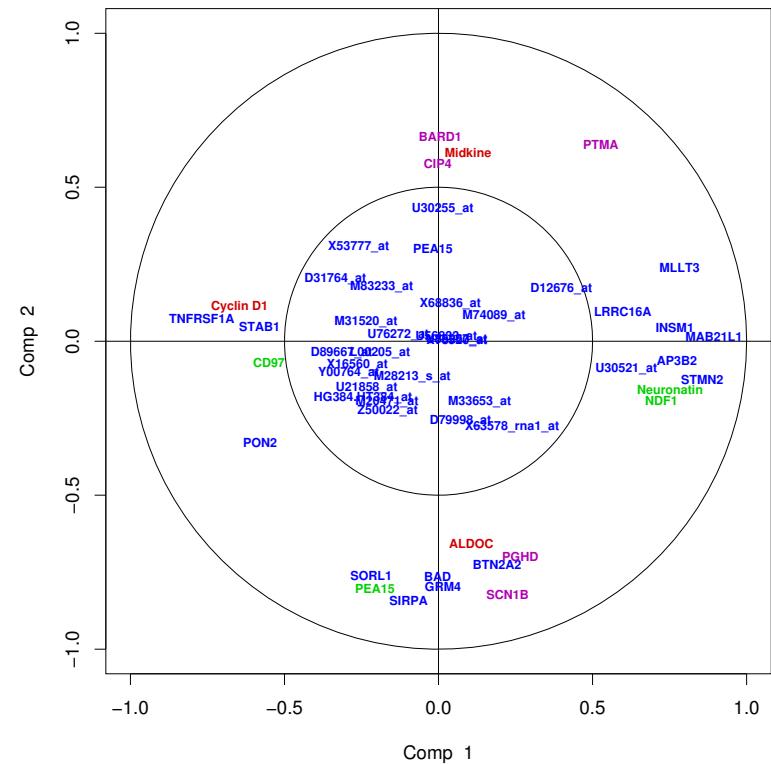


FIGURE 7 – Représentation des individus dans les deux premières composantes

une variables est sélectionnée pour une valeur donnée de la pénalisation.

5.2 Exemple

Le graphique de la figure 9 est obtenu en synthétisant les stratégies précédentes. Sur chacun des 50 échantillons bootstrap, une sPLS-DA est calculée pour différentes valeurs de la pénalisation. On ne s'intéresse ici qu'à la première composante ($h = 1$). Dans ce cas de seuillage doux, la pénalisation revient à fixer le nombre de variables intervenant dans la construction de la première variable latente. La probabilité d'occurrence d'une variable ou gène est tout simplement estimée par le ratio du nombre de fois où elle a été sélectionnée. Quelques variables ou gènes apparaissent assez systématiquement sélectionnés, principalement 4 d'entre eux. Il apparaît que les données observées ne peuvent garantir la sélection que d'un nombre restreint de gènes. Ce constat serait à rapprocher du résultat théorique de Verzelen (2012)[9] dans le cas du modèle gaussien. Celui-ci met en évidence qu'un problème de ultra-haute dimension se manifeste si

$$\frac{2k \log(p/k)}{n} > \frac{1}{2}.$$

Avec les effectifs ($n=90$, $p=6144$) de l'exemple présenté, cette contrainte, dans le cas gaussien, signifierait qu'il est illusoire de vouloir sélectionner plus de 6 gènes. Pour un tout autre modèle, c'est aussi ce que nous signifie le graphique. Seule la considération d'un petit nombre de gènes dont la sélection est relativement stable sur les différents échantillons bootstrap est raisonnable sur ces données compte tenu de la faible taille de l'échantillon.

Références

- [1] F. Bach, *Bolasso : model consistent Lasso estimation through the bootstrap*, Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML) (2008), 33–40.
- [2] H. Chun et S. Keles, *Sparse partial least squares regression for simultaneous dimension reduction and variable selection*, Journal of the Royal Statistical Society : Series B **72** (2010), 3–25.
- [3] K. A. Lê Cao, S. Boistard et P. Besse, *Sparse PLS Discriminant Analysis : biologically relevant feature selection and graphical displays for*

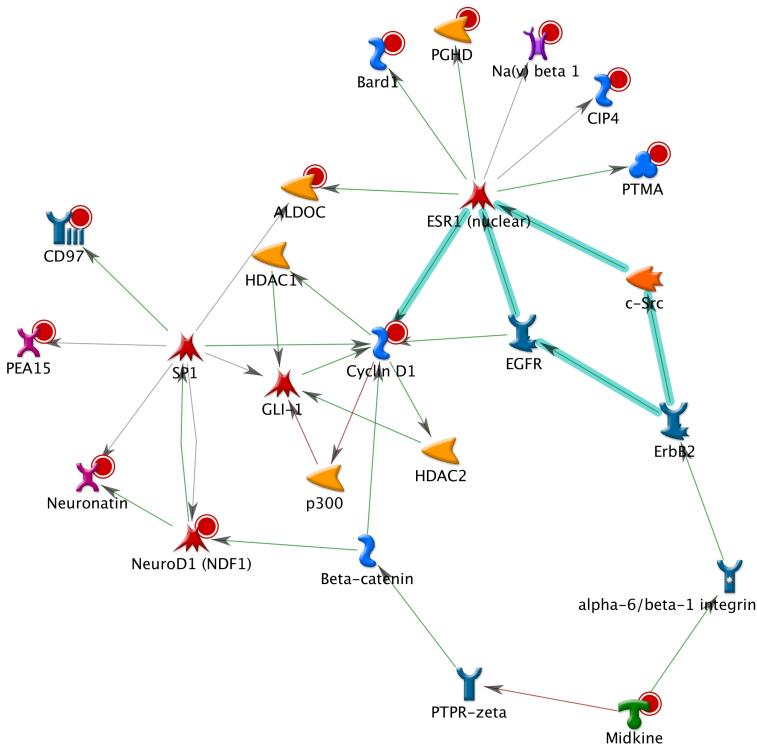


FIGURE 8 – Représentation (Gene Go software) en réseau des gènes déjà identifiés comme liés à ces pathologies de tumeurs cérébrales.

multiclass problems, BMC Bioinformatics **12** (2011), n° 253.

- [4] K. A. Lê Cao, P.G.P Martin, C. Robert-Granié et P. Besse, *Sparse Canonical Methods for Biological Data Integration : application to a cross-platform study*, BMC Bioinformatics **10** (2009), n° 34.
- [5] K. A. Lê Cao, D. Rossouw, C. Robert-Granié et P. Besse, *A sparse PLS for variable selection when integrating Omics data*, Statistical Applications in Genetics and Molecular Biology **7** (2008), n° 35.
- [6] N. Meinshausen et P. Bühlmann, *Stability selection*, Journal of the Royal Statistical Society : Series B **72** (2008), 417–473.
- [7] H. Shen et J.Z. Huang, *Sparse principal component analysis via regularized low rank matrix approximation*, Journal of Multivariate Analysis **99** (2008), 1015–1034.
- [8] M. Tenenhaus, *La régression PLS : théorie et applications*, Technip, 1998.
- [9] Nicolas Verzelen, *Minimax risks for sparse regressions : Ultra-high-dimensional phenomenons*, Electron. J. Statistics **6** (2012), 38–90, <http://arxiv.org/pdf/1008.0526.pdf>.
- [10] H. Wold, *Multivariate analysis*, Academic Press,, 1966.

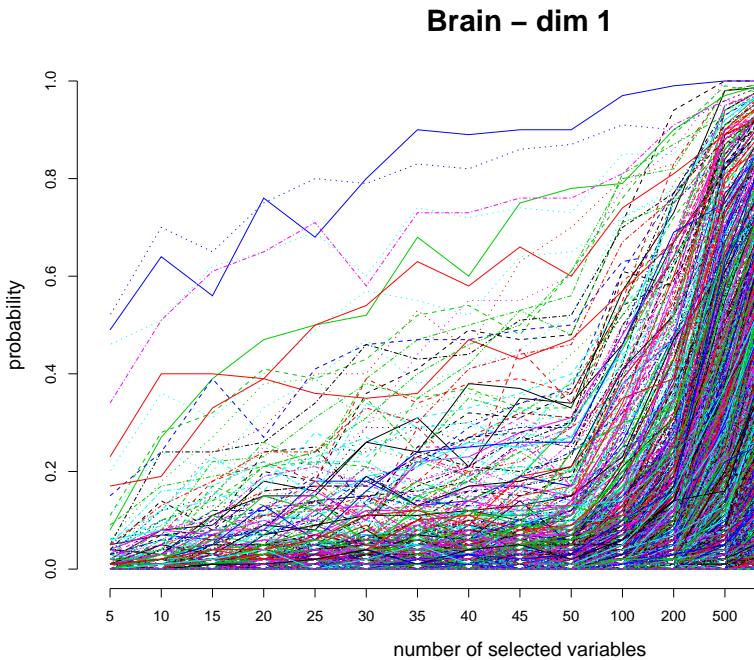


FIGURE 9 – Probabilités de sélection des différentes variables (gènes) sur la première composante en fonction de la valeur de la pénalisation en sPLS-DA.

Analyse Discriminante Décisionnelle

Résumé

Une variable qualitative Y à m modalités est modélisée par p variables quantitatives $X^j, j = 1, \dots, p$. L'objectif est la prévision de la classe d'un ou de nouveaux individus sur lesquels les variables $X^j, j = 1, \dots, p$ sont également observées. Différents modèles d'analyse discriminante décisionnelle sont considérés : règle linéaire et quadratique de décision dans le cas gaussien, règle non paramétrique par k plus proches voisins.

[Retour au plan du cours](#)

1 Introduction

Il s'agit de la modélisation d'une variable qualitative Y à m modalités par p variables quantitatives $X^j, j = 1, \dots, p$ observées sur un même échantillon Ω de taille n . L'objectif de l'analyse discriminante décisionnelle déborde le simple cadre descriptif de l'analyse factorielle discriminante (AFD). Disposant d'individus sur lesquels les X^j sont observées mais pas Y , il s'agit de *décider* de la modalité \mathcal{T}_ℓ de Y (ou de la classe correspondante) de ces individus. L'ADD s'applique donc également à la situation précédente de la régression logistique ($m = 2$) mais aussi lorsque le nombre de classes est plus grand que 2. Les variables explicatives devant être quantitatives, celles qualitatives sont remplacées par des indicatrices.

L'objectif est de définir des *règles de décision* (ou d'affectation); $\mathbf{x} = (x^1, \dots, x^p)$ désigne les observations des variables explicatives sur un individu, $\{\mathbf{g}_\ell; \ell = 1, \dots, m\}$ les barycentres des classes calculés sur l'échantillon et $\bar{\mathbf{x}}$ le barycentre global.

La matrice de covariance empirique se décompose en

$$\mathbf{S} = \mathbf{S}_e + \mathbf{S}_r.$$

où \mathbf{S}_r est appelée variance intraclasse (within) ou résiduelle :

$$\mathbf{S}_r = \bar{\mathbf{X}}_r' \mathbf{D} \bar{\mathbf{X}}_r = \sum_{\ell=1}^m \sum_{i \in \Omega_\ell} w_i (\mathbf{x}_i - \mathbf{g}_\ell)(\mathbf{x}_i - \mathbf{g}_\ell)',$$

et \mathbf{S}_e la variance interclasse (between) ou expliquée :

$$\mathbf{S}_e = \bar{\mathbf{G}}' \bar{\mathbf{D}} \bar{\mathbf{G}} = \bar{\mathbf{X}}_e' \mathbf{D} \bar{\mathbf{X}}_e = \sum_{\ell=1}^m \overline{w_\ell} (\mathbf{g}_\ell - \bar{\mathbf{x}})(\mathbf{g}_\ell - \bar{\mathbf{x}})'.$$

2 Règle de décision issue de l'AFD

2.1 Cas général : m quelconque

DÉFINITION 1. — L'individu x est affectée à la modalité de Y minimisant :

$$d_{\mathbf{S}_r^{-1}}^2(\mathbf{x}, \mathbf{g}_\ell), \ell = 1, \dots, m.$$

Cette distance se décompose en

$$d_{\mathbf{S}_r^{-1}}^2(\mathbf{x}, \mathbf{g}_\ell) = \|\mathbf{x} - \mathbf{g}_\ell\|_{\mathbf{S}_r^{-1}}^2 = (\mathbf{x} - \mathbf{g}_\ell)' \mathbf{S}_r^{-1} (\mathbf{x} - \mathbf{g}_\ell)$$

et le problème revient donc à maximiser

$$\mathbf{g}_\ell' \mathbf{S}_r^{-1} \mathbf{x} - \frac{1}{2} \mathbf{g}_\ell' \mathbf{S}_r^{-1} \mathbf{g}_\ell.$$

Il s'agit bien d'une règle linéaire en \mathbf{x} car elle peut s'écrire : $\mathbf{A}_\ell \mathbf{x} + \mathbf{b}_\ell$.

2.2 Cas particulier : $m=2$

Dans ce cas, la dimension r de l'AFD vaut 1. Il n'y a qu'une seule valeur propre non nulle λ_1 , un seul vecteur discriminant v^1 et un seul axe discriminant Δ_1 . Les 2 barycentres \mathbf{g}_1 et \mathbf{g}_2 sont sur Δ_1 , de sorte que v^1 est colinéaire à $\mathbf{g}_1 - \mathbf{g}_2$.

L'application de la règle de décision permet d'affecter \mathbf{x} à \mathcal{T}_1 si :

$$\mathbf{g}_1' \mathbf{S}_r^{-1} \mathbf{x} - \frac{1}{2} \mathbf{g}_1' \mathbf{S}_r^{-1} \mathbf{g}_1 > \mathbf{g}_2' \mathbf{S}_r^{-1} \mathbf{x} - \frac{1}{2} \mathbf{g}_2' \mathbf{S}_r^{-1} \mathbf{g}_2$$

c'est-à-dire encore si

$$(\mathbf{g}_1 - \mathbf{g}_2)' \mathbf{S}_r^{-1} \mathbf{x} > (\mathbf{g}_1 - \mathbf{g}_2)' \mathbf{S}_r^{-1} \frac{\mathbf{g}_1 + \mathbf{g}_2}{2}.$$

Remarque

La règle de décision liée à l'AFD est simple mais elle est limitée et insuffisante notamment si les variances des classes ne sont pas identiques. De plus, elle ne tient pas compte de l'échantillonnage pour \mathbf{x} : tous les groupes n'ont pas nécessairement la même probabilité d'occurrence.

3 Règle de décision bayésienne

3.1 Introduction

La variable Y , qui indique le groupe d'appartenance d'un individu, prend ses valeurs dans $\{\mathcal{T}_1, \dots, \mathcal{T}_m\}$ et est munie d'une loi de probabilité π_1, \dots, π_m . Les probabilités $\pi_\ell = P[\mathcal{T}_\ell]$ représentent les probabilités *a priori* des classes ou groupes ω_ℓ . Les vecteurs \mathbf{x} des observations des variables explicatives sont supposés suivant, connaissant leur classe, une loi de densité

$$h_\ell(\mathbf{x}) = P[\mathbf{x} | \mathcal{T}_\ell]$$

par rapport à une mesure de référence¹.

3.2 Définition

Une règle de décision est une application δ de Ω dans $\{\mathcal{T}_1, \dots, \mathcal{T}_m\}$ qui, à tout individu, lui affecte une classe connaissant \mathbf{x} . Sa définition dépend du contexte de l'étude et prend en compte la

- connaissance ou non de coûts de mauvais classement,
- connaissance ou non des lois *a priori* sur les classes,
- nature aléatoire ou non de l'échantillon.

Soit $c_{\ell|k}$ le coût du classement dans \mathcal{T}_ℓ d'un individu de \mathcal{T}_k . Le *risque de Bayes* d'une règle de décision δ exprime alors le coût moyen :

$$R_\delta = \sum_{k=1}^m \pi_k \sum_{\ell=1}^m c_{\ell|k} \int_{\{\mathbf{x} \mid \delta(\mathbf{x})=\mathcal{T}_\ell\}} h_k(\mathbf{x}) d\mathbf{x}$$

où $\int_{\{\mathbf{x} \mid \delta(\mathbf{x})=\mathcal{T}_\ell\}} h_k(\mathbf{x}) d\mathbf{x}$ représente la probabilité d'affecter \mathbf{x} à \mathcal{T}_ℓ alors qu'il est dans \mathcal{T}_k .

1. La mesure de Lebesgues pour des variables réelles, celle de comptage pour des variables qualitatives

3.3 Coûts inconnus

L'estimation des coûts n'est pas du ressort de la Statistique et, s'ils ne sont pas connus, on suppose simplement qu'ils sont tous égaux. La minimisation du risque ou règle de Bayes revient alors à affecter tout \mathbf{x} à la classe la plus probable c'est-à-dire à celle qui maximise la probabilité conditionnelle *a posteriori* : $P[\mathcal{T}_\ell \mid \mathbf{x}]$. Par le théorème de Bayes :

$$P[\mathcal{T}_\ell \mid \mathbf{x}] = \frac{P[\mathcal{T}_\ell \text{ et } \mathbf{x}]}{P[\mathbf{x}]} = \frac{P[\mathcal{T}_\ell] \cdot P[\mathbf{x} \mid \mathcal{T}_\ell]}{P[\mathbf{x}]}$$

avec le principe des probabilités totales : $P[\mathbf{x}] = \sum_{\ell=1}^m P[\mathcal{T}_\ell] \cdot P[\mathbf{x} \mid \mathcal{T}_\ell]$.

Comme $P[\mathbf{x}]$ ne dépend pas de ℓ , la règle consiste à choisir \mathcal{T}_ℓ maximisant

$$P[\mathcal{T}_\ell] \cdot P[\mathbf{x} \mid \mathcal{T}_\ell] = \pi_\ell \cdot P[\mathbf{x} \mid \mathcal{T}_\ell];$$

$P[\mathbf{x} \mid \mathcal{T}_\ell]$ est la probabilité d'observer \mathbf{x} au sein de la classe \mathcal{T}_ℓ . Pour une loi discrète, il s'agit d'une probabilité du type $P[\mathbf{x} = \mathbf{x}_k^\ell \mid \mathcal{T}_\ell]$ et d'une densité $h(\mathbf{x} \mid \mathcal{T}_\ell)$ pour une loi continue amis dans tous les cas la notation $h_\ell(\mathbf{x})$ est utilisée.

La règle de décision s'écrit finalement sous la forme :

$$\delta(\mathbf{x}) = \arg \max_{\ell=1, \dots, m} \pi_\ell h_\ell(\mathbf{x}).$$

3.4 Détermination des *a priori*

Les probabilités *a priori* π_ℓ peuvent effectivement être connues : proportions de divers groupes dans une population, de diverses maladies... ; sinon elles sont estimées sur l'échantillon d'apprentissage :

$$\hat{\pi}_\ell = w_\ell = \frac{n_\ell}{n} \quad (\text{si tous les individus ont le même poids})$$

à condition qu'il soit bien un échantillon aléatoire susceptible de fournir des estimations correctes des fréquences. Dans le cas contraire il reste à considérer tous les π_ℓ égaux.

3.5 Cas particuliers

- Dans le cas où les probabilités *a priori* sont égales, c'est par exemple le cas du choix de probabilités non informatives, la règle de décision

bayésienne revient alors à maximiser $h_\ell(\mathbf{x})$ qui est la vraisemblance, au sein de \mathcal{T}_ℓ , de l'observation \mathbf{x} . La règle consiste alors à choisir la classe pour laquelle cette vraisemblance est maximum.

- Dans le cas où $m = 2$, \mathbf{x} est affectée à \mathcal{T}_1 si :

$$\frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} > \frac{\pi_2}{\pi_1},$$

faisant ainsi apparaître un rapport de vraisemblance. D'autre part, l'introduction de coûts de mauvais classement différents selon les classes amène à modifier la valeur limite π_2/π_1 .

Finalement, il reste à estimer les densités conditionnelles $h_\ell(\mathbf{x})$. Les différentes méthodes d'estimation considérées conduisent aux méthodes classiques de discrimination bayésienne objets des sections suivantes.

4 Règle bayésienne avec modèle gaussien

Dans cette section, conditionnellement à \mathcal{T}_ℓ , $\mathbf{x} = (x_1, \dots, x_p)$ est supposée être l'observation d'un vecteur aléatoire gaussien $\mathcal{N}(\boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)$; $\boldsymbol{\mu}_\ell$ est un vecteur de \mathbb{R}^p et $\boldsymbol{\Sigma}_\ell$ une matrice $(p \times p)$ symétrique et définie-positive. La densité de la loi, au sein de la classe \mathcal{T}_ℓ , s'écrit donc :

$$h_\ell(\mathbf{x}) = \frac{1}{\sqrt{2\pi}(\det(\boldsymbol{\Sigma}_\ell))^{1/2}} \exp \left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_\ell)' \boldsymbol{\Sigma}_\ell^{-1} (\mathbf{x} - \boldsymbol{\mu}_\ell) \right].$$

L'affectation de \mathbf{x} à une classe se fait en maximisant $\pi_\ell \cdot h_\ell(\mathbf{x})$ par rapport à ℓ soit encore la quantité :

$$\ln(\pi_\ell) - \frac{1}{2} \ln(\det(\boldsymbol{\Sigma}_\ell)) - \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_\ell)' \boldsymbol{\Sigma}_\ell^{-1} (\mathbf{x} - \boldsymbol{\mu}_\ell).$$

4.1 Hétéroscédasticité

Dans le cas général, il n'y a pas d'hypothèse supplémentaire sur la loi de \mathbf{x} et donc les matrices $\boldsymbol{\Sigma}_\ell$ sont fonction de ℓ . Le critère d'affectation est alors *quadratique* en \mathbf{x} . Les probabilités π_ℓ sont supposées connues mais il est nécessaire d'estimer les moyennes $\boldsymbol{\mu}_\ell$ ainsi que les covariances $\boldsymbol{\Sigma}_\ell$ en maximisant, compte tenu de l'hypothèse de normalité, la vraisemblance. Ceci conduit

à estimer la moyenne

$$\widehat{\boldsymbol{\mu}}_\ell = \mathbf{g}_\ell$$

par la moyenne empirique de \mathbf{x} dans la classe ℓ pour l'échantillon d'apprentissage et $\boldsymbol{\Sigma}_\ell$ par la matrice de covariance empirique $\mathbf{S}_{R\ell}^*$:

$$\mathbf{S}_{R\ell}^* = \frac{1}{n_\ell - 1} \sum_{i \in \Omega_\ell} (\mathbf{x}_i - \mathbf{g}_\ell)(\mathbf{x}_i - \mathbf{g}_\ell)'$$

pour ce même échantillon.

4.2 Homoscédasticité

Les lois de chaque classe sont supposées partager la même structure de covariance $\boldsymbol{\Sigma}_\ell = \boldsymbol{\Sigma}$. En supprimant les termes indépendants de ℓ , le critère à maximiser devient

$$\ln(\pi_\ell) - \frac{1}{2} \boldsymbol{\mu}_\ell' \boldsymbol{\Sigma}_\ell^{-1} \boldsymbol{\mu}_\ell + \boldsymbol{\mu}_\ell' \boldsymbol{\Sigma}_\ell^{-1} \mathbf{x}$$

qui est cette fois *linéaire* en \mathbf{x} . Les moyennes $\boldsymbol{\mu}_\ell$ sont estimées comme précédemment tandis que $\boldsymbol{\Sigma}$ est estimée par la matrice de covariance intraclassé empirique :

$$\mathbf{S}_R^* = \frac{1}{n - m} \sum_{\ell=1}^m \sum_{i \in \Omega_\ell} (\mathbf{x}_i - \mathbf{g}_\ell)(\mathbf{x}_i - \mathbf{g}_\ell)'$$

Si, de plus, les probabilités π_ℓ sont égales, après estimation le critère s'écrit :

$$\overline{\mathbf{x}}_\ell' \mathbf{S}_R^{*-1} \mathbf{x} - \frac{1}{2} \overline{\mathbf{x}}_\ell' \mathbf{S}_R^{*-1} \overline{\mathbf{x}}_\ell.$$

qui coïncide avec le critère initial issu de l'AFD..

4.3 Commentaire

Les hypothèses : normalité, éventuellement l'homoscédasticité, doivent être vérifiées par la connaissance *a priori* du phénomène ou par une étude préalable de l'échantillon d'apprentissage. L'hypothèse d'homoscédasticité, lorsqu'elle est vérifiée, permet de réduire très sensiblement le nombre de paramètres à estimer et d'aboutir à des estimateurs plus fiables car de variance moins élevée. Dans le cas contraire, l'échantillon d'apprentissage doit être de taille importante.

5 Règle bayésienne avec estimation non paramétrique

5.1 Introduction

En Statistique, l'estimation est non paramétrique ou fonctionnelle lorsque le nombre de paramètres à estimer est infini. L'objet statistique à estimer est alors une fonction par exemple de régression $y = f(x)$ ou encore une densité de probabilité h . Dans ce cas, au lieu de supposer que la densité est de type connu (gaussien) dont les paramètres sont estimés, c'est la fonction de densité h qui est directement estimée. Pour tout x de \mathbb{R} , $h(x)$ est donc estimée par $\hat{h}(x)$.

Cette approche très souple a l'avantage de ne pas nécessiter d'hypothèse particulière sur la loi (seulement la régularité de h pour de bonnes propriétés de convergence), en revanche elle n'est applicable qu'avec des échantillons de grande taille d'autant plus que le nombre de dimensions p est grand (*curse of dimensionality*).

Dans le cadre de l'analyse discriminante, ces méthodes permettent d'estimer directement les densités $h_\ell(\mathbf{x})$. On considère ici deux approches : la méthode du noyau et celle des k plus proches voisins.

5.2 Méthode du noyau

Estimation de densité

Soit y_1, \dots, y_n n observations équipondérées d'une v.a.r. continue Y de densité h inconnue. Soit $K(y)$ (le *noyau*) une densité de probabilité unidimensionnelle (sans rapport avec h) et λ un réel strictement positif. On appelle estimation de h par la méthode du noyau la fonction

$$\hat{h}(y) = \frac{1}{n\lambda} \sum_{i=1}^n K\left(\frac{y - y_i}{\lambda}\right).$$

Il est immédiat de vérifier que

$$\forall y \in \mathbb{R}, \hat{h}(y) \geq 0 \quad \text{et} \quad \int_{-\infty}^{+\infty} \hat{h}(y) dy = 1;$$

λ est appelé *largeur de fenêtre* ou paramètre de *lissage*; plus λ est grand, plus l'estimation \hat{h} de h est régulière. Le noyau K est choisi centré en 0, unimodal et symétrique. Les cas les plus usuels sont la densité gaussienne, celle uniforme sur $[-1, 1]$ ou triangulaire : $K(x) = [1 - |x|]\mathbf{1}_{[-1,1]}(x)$. La forme du noyau n'est pas très déterminante sur la qualité de l'estimation contrairement à la valeur de λ .

Application à l'analyse discriminante

La méthode du noyau est utilisée pour calculer une estimation non paramétrique de chaque densité $h_\ell(\mathbf{x})$ qui sont alors des fonctions définies dans \mathbb{R}^p . Le noyau K^* doit donc être choisi multidimensionnel et

$$\hat{h}_\ell(\mathbf{x}) = \frac{1}{n_\ell \lambda^p} \sum_{i \in \Omega_\ell} K^*\left(\frac{\mathbf{x} - \mathbf{x}_i}{\lambda}\right).$$

Un noyau multidimensionnel peut être défini à partir de la densité usuelle de lois : multinormale $\mathcal{N}_p(\mathbf{0}, \Sigma_p)$ ou uniforme sur la sphère unité ou encore par produit de noyaux unidimensionnels :

$$K^*(\mathbf{x}) = \prod_{j=1}^p K(x^j).$$

5.3 k plus proches voisins

Cette méthode d'affectation d'un vecteur \mathbf{x} consiste à enchaîner les étapes décrites dans l'algorithme ci-dessous.

Algorithme des k plus proches voisins (k -nn)

1. Choix d'un entier $k : 1 \geq k \geq n$.
2. Calculer les distances $d_{\mathbf{M}}(\mathbf{x}, \mathbf{x}_i)$, $i = 1, \dots, n$ où \mathbf{M} est la métrique de Mahalanobis c'est-à-dire la matrice inverse de la matrice de variance (ou de variance intraclassé).
3. Retenir les k observations $\mathbf{x}_{(1)}, \dots, \mathbf{x}_{(k)}$ pour lesquelles ces distances sont les plus petites.
4. Compter les nombres de fois k_1, \dots, k_m que ces k observations apparaissent dans chacune des classes.

TABLE 1 – Cancer : estimations des taux d'erreurs de prévision obtenus par différents types d'analyse discriminante

Méthode	apprentissage	validations croisée	test
linéaire	1,8	3,8	3,6
k NN	2,5	2,7	2,9

5. Estimer localement les densités conditionnelles par

$$\hat{h}_\ell(\mathbf{x}) = \frac{k_\ell}{kV_k(\mathbf{x})};$$

où $V_k(\mathbf{x})$ est le volume de l'ellipsoïde $\{\mathbf{z} | (\mathbf{z}-\mathbf{x})' \mathbf{M}(\mathbf{z}-\mathbf{x}) = d_{\mathbf{M}}(\mathbf{x}, \mathbf{x}_{(k)})\}$.

Pour $k = 1$, \mathbf{x} est affecté à la classe du plus proche élément.

Comme toute technique, celles présentées ci-dessus nécessitent le réglage d'un paramètre (largeur de fenêtre ou nombre de voisins considérés). Ce choix s'apparente à un choix de modèle et nécessite le même type d'approche à savoir l'optimisation d'un critère (erreur de classement, validation croisée).

6 Exemples

6.1 Cancer du sein

Par principe, l'analyse discriminante s'applique à des variables explicatives quantitatives. Ce n'est pas le cas des données qui sont au mieux ordinaires. Il est clair que construire une fonction de discrimination comme combinaison de ces variables n'a guère de sens. Néanmoins, en s'attachant uniquement à la qualité de prévision sans essayer de construire une interprétation du plan ou de la surface de discrimination, il est d'usage d'utiliser l'analyse discriminante de façon "sauvage". Les résultats obtenus sont résumés dans le tableau 1. L'analyse discriminante quadratique, avec matrice de variance estimée pour chaque classe n'a pas pu être calculée. Une des matrices n'est pas inversible.

TABLE 2 – Ozone : estimations des taux d'erreurs de prévision obtenus par différents types d'analyse discriminante

Méthode	apprentissage	validations croisée	test
linéaire	11,9	12,5	12,0
quadratique	12,7	14,8	12,5

TABLE 3 – Banque : estimations des taux d'erreurs de prévision obtenus par différents types d'analyse discriminante

Méthode	apprentissage	validations croisée	test
linéaire	16,5	18,3	18
quadratique	17,8	22,0	30
k NN	23,5	29,8	29

6.2 Concentration d'ozone

Dans cet exemple aussi, deux variables sont qualitatives : le type de jour à 2 modalités ne pose pas de problème mais remplacer la station par un entier est plutôt abusif. D'ailleurs, la méthode des plus proches voisins ne l'acceptent pas, une transformation des données serait nécessaire.

6.3 Carte visa

Comme pour les données sur le cancer, les données bancaires posent un problème car elles associent différents types de variables. Il est possible de le contourner, pour celles binaires, en considérant quantitative, l'indicatrice de la modalité (0 ou 1). Pour les autres, certaines procédures (DISQUAL pour discrimination sur variables qualitatives) proposent de passer par une analyse factorielle multiple des correspondances pour rendre tout quantitatif mais ceci n'est pas implémenté de façon standard dans les logiciels d'origine américaine.

Pour l'analyse discriminante, R ne propose pas de sélection automatique de variable mais inclut une estimation de l'erreur par validation croisée. Les résultats trouvés sont résumés dans le tableau 3. Seule une discrimination linéaire

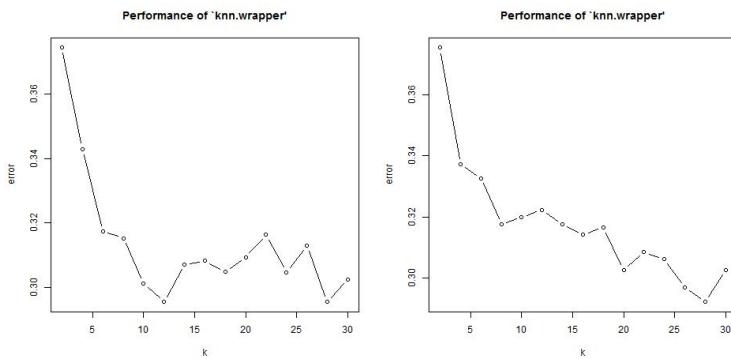


FIGURE 1 – Banque : Deux exécutions de l’optimisation du choix de k par validation croisée.

semble fournir des résultats raisonnables, la recherche d’une discrimination quadratique n’apporte rien pour ces données. De son côté, SAS propose une sélection automatique (procédure stepdisc) mais les résultats obtenus ne sont pas sensiblement meilleurs après sélection.

Le choix de k dans la méthode des k plus proches voisins est souvent délicat ; chaque exécution de l’estimation de l’erreur par validation croisée conduit à des résultats aléatoires et très différents et k optimal oscille entre 10 et 30 (fig. 1) !

Arbres binaires de décision

Résumé

Méthodes de construction d'arbres binaires de décision, modélisant une discrimination (classification tree) ou une régression (regression tree). Principes et algorithmes de construction des arbres, critères d'homogénéité et construction des nœuds, élagage pour l'obtention d'un modèle parcimonieux.

[Retour à l'introduction.](#)

Tous les tutoriels sont disponibles sur le dépôt :

github.com/wikistat

1 Introduction

Les méthodes dites de partitionnement récursif ou de segmentation datent des années 60. Elles ont été formalisées dans un cadre générique de sélection de modèle par Breiman et col. (1984)[1] sous l'acronyme de CART : *Classification and Regression Tree*. Parallèlement Quinlan (1993)[2] a proposé l'algorithme C4.5 dans la communauté informatique. L'acronyme CART correspond à deux situations bien distinctes selon que la variable à expliquer, modéliser ou prévoir est qualitative (discrimination ou en anglais *classification*) ou quantitative (régression).

Très complémentaires des méthodes statistiques plus classiques : analyse discriminante, régression linéaire, les solutions obtenues sont présentées sous une forme graphique simple à interpréter, même pour des néophytes, et constituent une aide efficace pour l'aide à la décision. Elles sont basées sur une séquence récursive de règles de division, coupes ou *splits*.

La figure 1 présente un exemple illustratif d'arbre de classification. Les variables Age, Revenu et Sexe sont utilisées pour discriminer les observations sous la forme d'une structure arborescente. L'ensemble des observations est regroupé à la racine de l'arbre puis chaque division ou coupe sépare chaque nœud en deux nœuds fils plus *homogènes* que le nœud père au sens d'un *critère* à préciser et dépendant du type de la variable Y , quantitative ou qualitative.

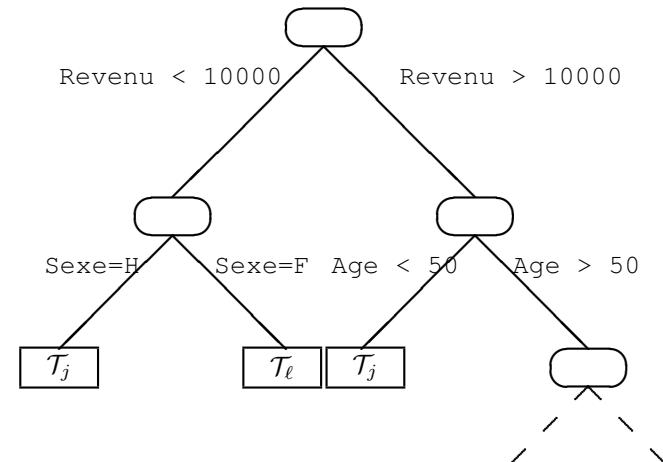


FIGURE 1 – Exemple élémentaire d'arbre de classification.

Lorsque le partitionnement est interrompu, chaque nœud terminal de l'arbre complet ainsi construit devient une feuille à laquelle est attribuée une valeur si Y est quantitative, une classe de Y si elle est qualitative.

La dernière étape consiste en un élagage correspondant à une sélection de modèle afin d'en réduire la complexité dans l'objectif, toujours répété, d'éviter le sur-ajustement. Depuis que Breiman et al. (1984)[1] ont formaliser cette étape, CART connaît un succès important avec un l'atout majeur de la facilité de l'interprétation même pour un néophyte en Statistique. La contrepartie est que ces modèles sont particulièrement instables, très sensibles à des fluctuations de l'échantillon.

Par ailleurs, pour des variables explicatives quantitatives, la construction d'un arbre constitue un *partitionnement dyadique* de l'espace (cf. figure 2). Le modèle ainsi défini manque par construction de régularité même, et surtout, si le phénomène à modéliser est lui-même régulier.

Ces deux aspects ou faiblesses de CART : instabilité, irrégularités, sont à l'origine du succès des méthodes d'ensemble ou d'**agrégation de modèles**.

2 Construction d'un arbre binaire maximal

2.1 Principe

Les données sont constituées de l'observation de p variables quantitatives ou qualitatives explicatives X^j et d'une variable à expliquer Y qualitative à m modalités $\{\mathcal{T}_\ell; \ell = 1 \dots, m\}$ ou quantitative réelle, observées sur un échantillon de n individus.

La construction d'un arbre de discrimination binaire (cf. figure 1) consiste à déterminer une séquence de *nœuds*.

- Un nœud est défini par le choix conjoint d'une variable parmi les explicatives et d'une *division* qui induit une partition en deux classes. Implicitement, à chaque nœud correspond donc un sous-ensemble de l'échantillon auquel est appliquée une dichotomie.
- Une division est elle-même définie par une *valeur seuil* de la variable quantitative sélectionnée ou un partage en deux *groupes des modalités* si la variable est qualitative.
- À la racine ou nœud initial correspond l'ensemble de l'échantillon ; la procédure est ensuite itérée sur chacun des sous-ensembles.

L'algorithme considéré nécessite :

1. la définition d'un *critère* permettant de sélectionner la meilleure division parmi toutes celles *admissibles* pour les différentes variables ;
2. une règle permettant de décider qu'un nœud est terminal : il devient ainsi une *feuille* ;
3. l'affectation de chaque feuille à l'une des classes ou à une valeur de la variable à expliquer.

2.2 Critère de division

Une division est dite *admissible* si aucun des deux nœuds descendants qui en découlent n'est vide. Si la variable explicative est qualitative ordinaire avec m modalités, elle fournit $(m - 1)$ divisions binaires admissibles. Si elle est seulement nominale le nombre de divisions passe à $2^{(m-1)} - 1$. Une variable quantitative se ramène au cas ordinal.

Attention, l'algorithme tend à favoriser la sélection de variables explicatives avec beaucoup de modalités car celles-ci offrent plus de souplesse dans

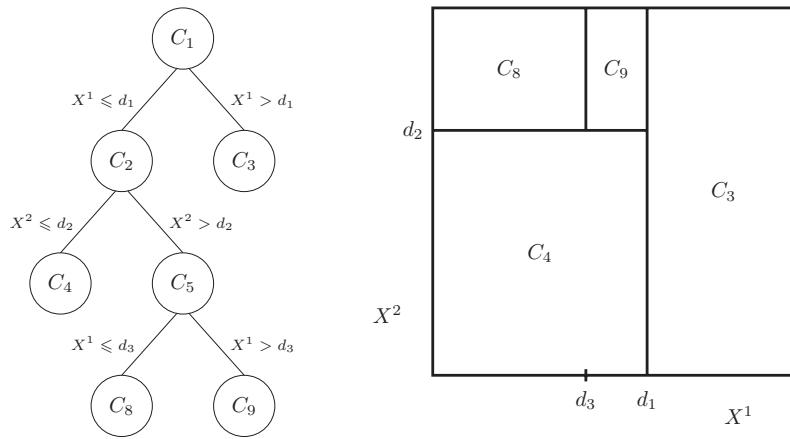


FIGURE 2 – Construction d'un arbre avec variables explicatives quantitatives et pavage dyadique de l'espace. Chaque nœud père engendre deux fils conséquence d'une division ou coupe définie par le choix d'une variable : X^1 ou X^2 et d'une valeur seuil, successivement d_1, d_2, d_3 . À chaque pavé de l'espace ainsi découpé, est finalement associée une valeur ou une classe de Y .

la construction de deux sous groupes. Ces variables sont à utiliser avec parcimonie (e.g. le code postal) car susceptibles de favoriser un sur-apprentissage ; il est souvent préférable de réduire drastiquement le nombre de modalités (e.g. région géographique ou zone urbaine *vs.* zone rurale) par fusion de modalités comme c'est classique en [analyse des correspondances multiple](#).

Le critère de division repose sur la définition d'une fonction d'*hétérogénéité* ou de désordre explicitée dans la section suivante. L'objectif étant de partager les individus en deux groupes les plus homogènes au sens de la variable à expliquer. L'hétérogénéité d'un nœud se mesure par une fonction non négative qui doit être

1. nulle si, et seulement si, le nœud est homogène : tous les individus appartiennent à la même modalité ou prennent la même valeur de Y .
2. Maximale lorsque les valeurs de Y sont équiprobables ou très dispersées.

La division du nœud κ crée deux fils, gauche et droit. Pour simplifier, ils sont notés κ_G et κ_D mais une re-numérotation est nécessaire pour respecter la séquence de sous-arbres qui sera décrite dans la section suivante.

Parmi toutes les divisions admissibles du nœud κ , l'algorithme retient celle qui rend la somme $D_{\kappa_G} + D_{\kappa_D}$ des hétérogénéités des nœuds fils minimales. Ceci revient encore à résoudre à chaque étape k de construction de l'arbre :

$$\max_{\{divisions de X^j; j=1,p\}} D_\kappa - (D_{\kappa_G} + D_{\kappa_D})$$

Graphiquement, la longueur de chaque branche peut être représentée proportionnellement à la réduction de l'hétérogénéité occasionnée par la division.

2.3 Règle d'arrêt

La croissance de l'arbre s'arrête à un nœud donné, qui devient donc terminal ou *feuille*, lorsqu'il est homogène ou lorsqu'il n'existe plus de partition admissible ou, pour éviter un découpage inutilement fin, si le nombre d'observations qu'il contient est inférieur à une valeur seuil à choisir en général entre 1 et 5.

2.4 Affectation

Dans le cas Y quantitative, à chaque feuille ou pavé de l'espace, est associée une valeur : la moyenne des observations associées à cette feuille. Dans le cas qualitatif, chaque feuille ou nœud terminal est affecté à une classe \mathcal{T}_ℓ de Y en considérant le mode conditionnel :

- celle la mieux représentée dans le nœud et il est ensuite facile de compter le nombre d'objets mal classés ;
- la classe *a posteriori* la plus probable au sens bayésien si des probabilités *a priori* sont connues ;
- la classe la moins coûteuse si des coûts de mauvais classement sont donnés.

3 Critères d'homogénéité

Deux cas sont à considérer, les arbres de régression ou de classification.

3.1 Y quantitative

Dans le cas de la régression, l'hétérogénéité du nœud κ est définie par la variance :

$$D_\kappa = \frac{1}{|\kappa|} \sum_{i \in \kappa} (y_i - \bar{y}_\kappa)^2$$

où $|\kappa|$ est l'effectif du nœud κ .

L'objectif est de chercher pour chaque nœud la division, ou plus précisément la variable et la règle de division, qui contribuera à la plus forte décroissance de l'hétérogénéité des nœuds fils à gauche κ_G et à droite κ_D . Ce qui revient à minimiser la variance intraclasse ou encore :

$$\frac{|\kappa_G|}{n} \sum_{i \in \kappa_G} (y_i - \bar{y}_{\kappa_G})^2 + \frac{|\kappa_D|}{n} \sum_{i \in \kappa_D} (y_i - \bar{y}_{\kappa_D})^2.$$

On peut encore dire que la division retenue est celle qui rend, le plus significatif possible, le test de Fisher (analyse de variance) comparant les moyennes entre les deux nœuds fils. Dans leur présentation originale, Breiman et al. (1984)[1] montrent que, dans le cas d'une distribution gaussienne, le raffinement de l'arbre est associé à une décroissance, la plus rapide possible, d'une

déviance, d'où la notation D_κ ou écart à la vraisemblance du modèle gaussien associé.

3.2 Y qualitative

Soit Y variable qualitative à m modalités ou catégories \mathcal{T} numérotées $\ell = 1, \dots, m$. Plusieurs fonctions d'hétérogénéité, ou de désordre peuvent être définies pour un nœud : un critère défini à partir de la notion d'*entropie* ou à partir de la *concentration de Gini*. Un autre critère (CHAID) est basé sur la statistique de test du χ^2 . En pratique, il s'avère que le choix du critère importe moins que celui du niveau d'élagage, c'est souvent *Gini* qui est choisi par défaut mais le critère d'entropie s'interprète encore comme un terme de déviance par rapport à la vraisemblance mais d'un modèle multinomial saturé cette fois.

Entropie

L'hétérogénéité du nœud κ est définie par l'entropie qui s'écrit avec la convention $0 \log(0) = 0$:

$$D_\kappa = -2 \sum_{\ell=1}^m |\kappa| p_\kappa^\ell \log(p_\kappa^\ell)$$

où p_κ^ℓ est la proportion de la classe \mathcal{T}_ℓ de Y dans le nœud κ .

Concentration de Gini

L'hétérogénéité du nœud est définie par :

$$D_\kappa = \sum_{\ell=1}^m p_\kappa^\ell (1 - p_\kappa^\ell).$$

Comme dans le cas quantitatif, il s'agit, pour chaque nœud de rechercher, parmi les divisions admissible, celle qui maximise la décroissance de l'hétérogénéité.

Comme pour l'analyse discriminante décisionnelle, plutôt que des proportions, des probabilités conditionnelles sont définies par la règle de Bayes lorsque les probabilités *a priori* π_ℓ d'appartenance à la ℓ -ième classe sont connues. Dans le cas contraire, les probabilités de chaque classe sont estimées sur l'échantillon et donc les probabilités conditionnelles s'estiment simplement

par les proportions. Enfin, il est toujours possible d'introduire, lorsqu'ils sont connus, des coûts de mauvais classement et donc de se ramener à la minimisation d'un risque bayésien.

4 Élagage de l'arbre optimal

La démarche de construction précédente fournit un arbre A_{\max} à K feuilles qui peut être excessivement raffiné et donc conduire à un modèle de prévision très instable car fortement dépendant des échantillons qui ont permis son estimation. C'est une situation de sur-ajustement à éviter au profit de modèles plus parcimonieux donc plus robuste au moment de la prévision. Cet objectif est obtenu par une procédure d'*élagage* (*pruning*) de l'arbre.

Il s'agit donc de trouver un arbre optimal entre celui trivial réduit à une seule feuille et celui maximal A_{\max} en estimant leur performance par exemple sur un *échantillon de validation*. Tous les sous-arbres sont admissibles mais, comme leur nombre est de croissance exponentielle, il n'est pas envisageable de tous les considérer.

Pour contourner ce verrou, Breiman et col. (1984)[1] ont proposé une démarche consistant à construire une *suite emboîtée de sous-arbres* de l'arbre maximal puis à choisir, seulement *parmi cette suite*, l'arbre optimal qui minimise un risque ou erreur de généralisation. La solution ainsi obtenue est un optimum local mais l'efficacité et la fiabilité sont préférées à l'optimalité.

4.1 Construction de la séquence d'arbres

Pour un arbre A donné, on note K_A le nombre de feuilles ou nœuds terminaux κ , $\kappa = 1, \dots, K_A$ de A ; la valeur de K_A exprime la complexité de A . La qualité d'ajustement d'un arbre A est mesurée par

$$D(A) = \sum_{\kappa=1}^{K_A} D_\kappa$$

où D_κ est l'hétérogénéité de la feuille κ de l'arbre A et donc, selon le cas : la variance interclasse, l'entropie, la concentration de Gini, le nombre de mal classés, la déviance ou le coût de mauvais classement.

La construction de la séquence d'arbres emboîtés repose sur une pénalisa-

tion de la complexité de l'arbre :

$$C(A) = D(A) + \gamma \times K_A.$$

Pour $\gamma = 0$, $A_{\max} = A_{K_A}$ minimise $C(A)$. En faisant croître γ , l'une des divisions de A_{K_A} , celle pour laquelle l'amélioration de D est la plus faible (inférieure à γ), apparaît comme superflue et les deux feuilles obtenues sont regroupées (élaguées) dans le nœud père qui devient terminal ; A_{K_A} devient A_{K_A-1} .

Le procédé est itéré pour la construction de la séquence emboîtée :

$$A_{\max} = A_{K_A} \supset A_{K_A-1} \supset \cdots A_1$$

où A_1 , le nœud racine, regroupe l'ensemble de l'échantillon.

Il est alors facile de tracer le graphe représentant la décroissance ou éboulis des valeurs de D_κ en fonction du nombre croissant de feuilles dans l'arbre ou, c'est équivalent, en fonction de la séquence des valeurs décroissantes du coefficient de pénalisation γ .

4.2 Recherche de l'arbre optimal

Une fois la séquence d'arbres emboités construite, il s'agit d'en extraire celui optimal minimisant un risque ou erreur de généralisation. Si la taille de l'échantillon le permet, l'extraction préalable d'un [échantillon de validation](#) permet une estimation facile de ces risques.

Dans le cas contraire, c'est une stratégie de [validation croisée](#) en V segments qu'il faut mettre en place.

Celle-ci présente dans ce cas une particularité. En effet, à chacun des V échantillons constitués de $V - 1$ segments, correspond une séquence d'arbres différente. L'erreur moyenne n'est pas, dans ce cas, calculée pour chaque sous-arbre avec un nombre de feuilles donné mais pour chaque sous-arbre correspondant à une valeur fixée du coefficient de pénalisation γ issue de la séquence produite initialement par tout l'échantillon. À chacun des V échantillons correspond un arbre différent pour chacune des valeurs de γ mais c'est cette valeur qui est optimisée.

À la valeur de γ minimisant l'estimation de l'erreur de prévision par validation croisée, correspond ensuite l'arbre jugé optimal dans la séquence estimée sur tout l'échantillon d'apprentissage.

Le principe de sélection d'un arbre optimal est donc décrit dans l'algorithme ci-dessous.

Algorithm 1 Sélection d'arbre ou élagage par validation croisée

Construction de l'arbre maximal A_{\max}

Construction de la séquence $A_K \dots A_1$ d'arbres emboîtés associée à une Séquence de valeurs de pénalisation γ_κ

for $v = 1, \dots, V$ **do**

Pour chaque échantillon, estimation de la séquence d'arbres associée la séquence des pénalisations γ_κ

Estimation de l'erreur sur la partie restante de validation de l'échantillon

end for

Calcul de la séquence des moyennes de ces erreurs

L'erreur minimale désigne la pénalisation γ_{Opt} optimale

Retenir l'arbre associé à γ_{Opt} dans la séquence $A_K \dots A_1$

4.3 Remarques pratiques

Sur l'algorithme :

- Les arbres ne requièrent pas d'hypothèses sur les distributions des variables et semblent particulièrement adaptés au cas où les variables explicatives sont nombreuses. En effet, la procédure de *sélection* des variables est *intégrée* à l'algorithme construisant l'arbre et les *interactions* sont implicitement prises en compte.
- Il peut être utile d'associer arbre et régression logistique. Les *premières division* d'un arbre sont utilisées pour construire une *variable synthétique* intégrée à une régression logistique afin de sélectionner les quelques interactions apparaissant comme les plus pertinentes.
- La recherche d'une division est *invariante* par transformation monotone des variables explicatives quantitatives. Cela confère une *robustesse* de l'algorithme vis-à-vis de possibles valeurs atypiques ou de distributions très asymétriques. Seuls les rangs des observations sont considérés par l'algorithme pour chaque variable quantitative.
- *Attention*, cet algorithme suit une stratégie pas à pas hiérarchisée. Il peut, comme dans le cas du choix de modèle pas à pas en régression, passer à coté d'un optimum global ; il se montre par ailleurs très *in-*

stable et donc sensible à des fluctuations d'échantillon. Cette instabilité ou variance de l'arbre est une conséquence de la structure hiérarchique : une erreur de division en début d'arbre est propagée tout au long de la construction.

- Plusieurs variantes ont été proposées puis abandonnées : arbres ternaires plutôt que binaires, règle de décision linéaire plutôt que dichotomique. La première renforce inutilement l'instabilité alors que si une décision ternaire est indispensable, elle est la succession de deux divisions binaires. La deuxième rend l'interprétation trop complexe donc le modèle moins utile.
- Attention* Dans le cas d'une régression, Y est approchée par une fonction étagée. Si Y est l'observation d'un phénomène présentant des propriétés de régularité, ce modèle peut ne pas être approprié ou moins approprié qu'une autre famille de méthodes. En revanche, si Y présente des effets de seuillage, des singularités, ou s'il s'agit de discriminer des classes non connexes, un arbre de décision peut s'avérer plus approprié. Cela renforce l'idée que, sans information précise sur la nature des données (variable Y) à modéliser, il n'y a pas d'autre stratégie qu'essayer plusieurs types de modèles en comparant leurs performances.

Sur les implémentations :

- L'implémentation dans la librairie `rpart` de R mémorise la séquence des divisions ou *découpes compétitives*. Cela permet de préférer des arbres présentant des divisions certes moins optimales mais par exemple moins onéreuses à observer ou plus faciles à interpréter. Cela permet également de considérer des observations avec *données manquantes* pour certaines variables explicatives. Il suffit de déterminer pour chaque nœuds une séquence ordonnée de divisions possibles ou *surrogate splits* qui sont les divisions présentant le moins de *désaccords* avec celle meilleure. Au moment de calculer une prévision, si une donnée manque pour l'application d'une division ou règle de décision, la division suivante est prise en compte jusqu'à ce qu'une décision soit prise à chacun des nœuds rencontrés.
- Attention* : dans la version actuelle (0.19) de Scikit-learn, seul le paramètre de profondeur maximale de l'arbre (`max_depth`) permet de contrôler le sur-apprentissage. Ce paramètre optimisé, encore par validation croisée, ne définit qu'une séquence d'arbres très grossière

(croissance du nombre de feuilles par puissance de 2) par rapport à celle obtenue par pénalisation, feuille à feuille, dans `rpart` de R. Comme souvent dans les librairies de Python, l'efficacité algorithmique prévaut sur le sens "statistique".

5 Exemples

5.1 Concentration d'ozone

Arbre de régression

Un arbre de régression est estimé pour prévoir la concentration d'ozone. La librairie `rpart` du logiciel R prévoit une procédure d'élagage par validation croisée afin d'optimiser le coefficient de pénalisation. L'arbre (figure 3) montre bien quelles sont les variables importantes intervenant dans la prévision. Mais, compte tenu de la hiérarchisation de celles-ci, due à la structure arborescente du modèle, cette liste n'est pas similaire à celle mise en évidence dans le modèle gaussien. On voit plus précisément ici la complexité des interactions entre la prédition par MOCAGE et l'effet important de la température dans différentes situations. Les résidus de l'échantillon test du modèle d'arbre de régression prennent une structure particulière (figure 5) car les observations communes à une feuille terminale sont affectées de la même valeur. Il y a donc une colonne par feuille. La précision de l'ajustement peut s'en trouver altérée ($R^2 = 0,68$) mais il apparaît que ce modèle est moins soumis au problème d'hétéroscédasticité très présent dans le modèle gaussien.

Arbre de discrimination

Un modèle est estimé (figure 4) afin de prévoir directement le dépassement d'un seuil. Il est de complexité similaire à celle de l'arbre de régression mais ne fait pas jouer le même rôle aux variables. La température remplace la prévision MOCAGE de l'ozone comme variable la plus importante. Les prévisions de dépassement de seuil sur l'échantillon test sont sensiblement moins bonnes que celle de la régression, les taux sont de 14,4% avec l'arbre de régression et de 14,5% directement avec l'arbre de discrimination. Les matrices de confusion présentent les mêmes biais que les modèles de régression en omettant un nombre important de dépassements.

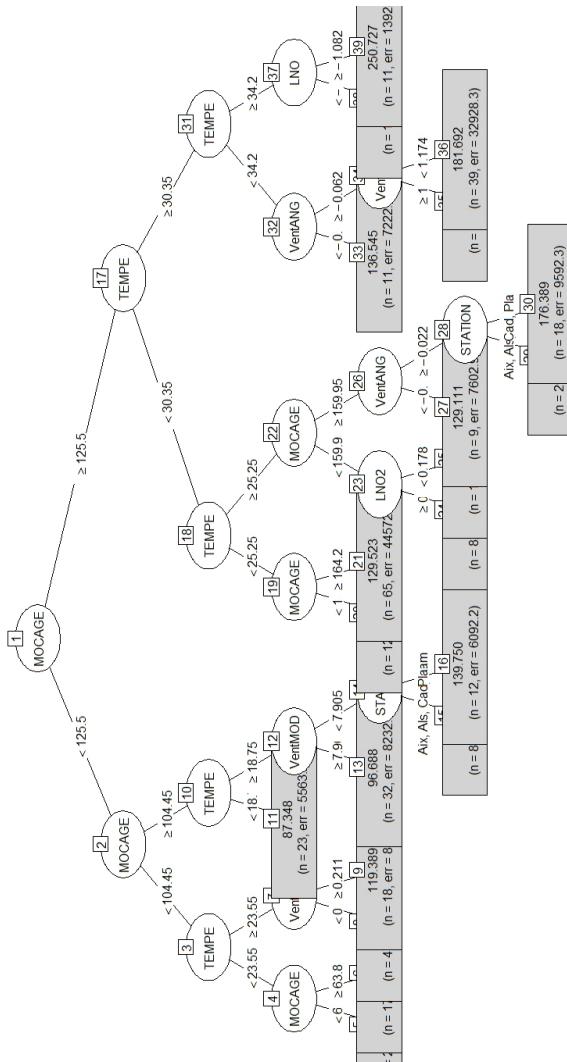


FIGURE 3 – Ozone : arbre de régression élagué par validation croisée (R).

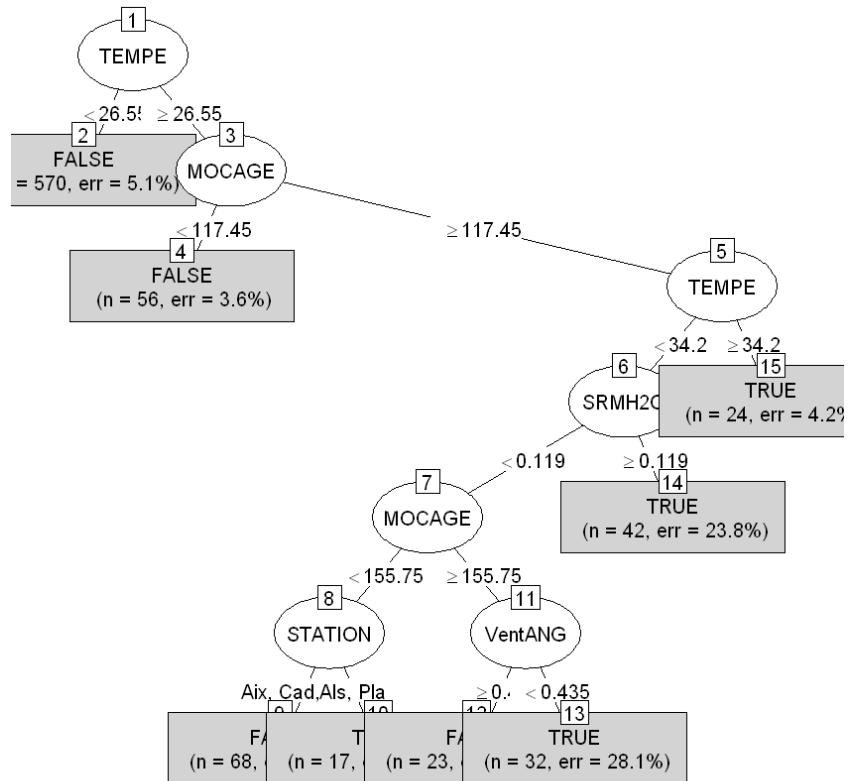


FIGURE 4 – *Ozone : arbre de discrimination élagué par validation croisée (R).*

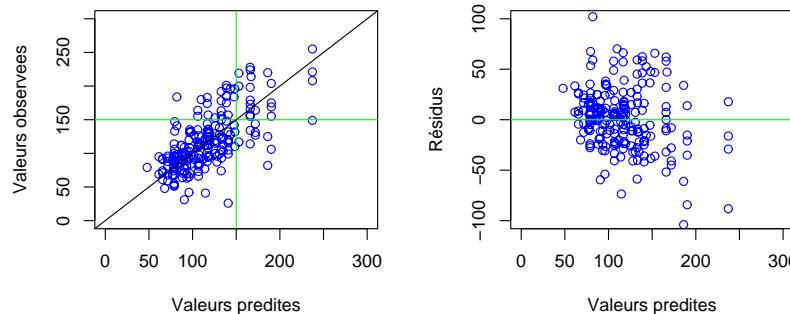


FIGURE 5 – Ozone : Valeurs observées et résidus de l'échantillon test.

5.2 Carte Visa Premier

L'étude des données bancaires s'intéresse soit aux données quantitatives brutes soient à celles-ci après découpage en classes des variables quantitatives. Ce découpage rend des services en régression logistique car le modèle construit s'en trouve plus flexible : plus de paramètres et moins de degrés de liberté, comme l'approximation par des indicatrices (des classes) de transformations non linéaires des variables. Il a été fait "à la main" en prenant les quantiles comme bornes de classe. C'est un usage courant pour obtenir des classes d'effectifs égaux et répartit ainsi au mieux la précision de l'estimation des paramètres mais ce choix n'est pas optimal au regard de l'objectif de prévision. Dans le cas d'un modèle construit à partir d'un arbre binaire, il est finalement préférable de laisser faire celui-ci le découpage en classe c'est-à-dire de trouver les valeurs seuils de décision. C'est la raison pour laquelle, l'arbre est préféablement estimé sur les variables quantitatives et qualitatives initiales.

La librairie `rpart` de R propose d'optimiser l'élagage par validation croisée pour obtenir la figure 6. Cet arbre conduit à un taux d'erreur estimé à 8% sur l'échantillon test, mieux que la régression logistique qui manque de flexibilité sur cet échantillon.

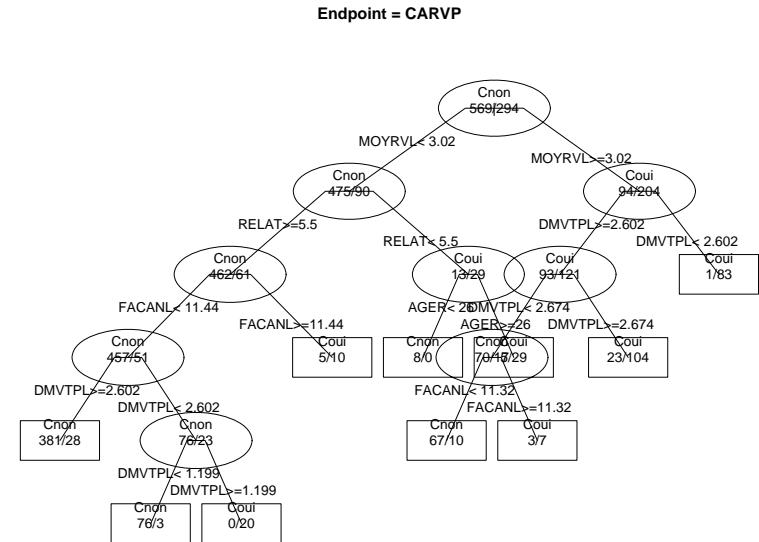


FIGURE 6 – Banque : arbre de décision élagué par validation croisée dans R.

Références

- [1] L. Breiman, J. Friedman, R. Olshen et C. Stone, *Classification and regression trees*, Wadsworth & Brooks, 1984.
- [2] J.R. Quinlan, *C4.5 – Programs for machine learning*, M. Kaufmann, 1993.

Réseaux de neurones

Résumé

Définition et caractéristiques des réseaux de neurones limitée aux perceptrons multicouches spécifiques pour la régression et la classification supervisée. Structure, fonctions de transfert, algorithme d'apprentissage par rétro-propagation du gradient, contrôles du sur-ajustement, introduction à l'apprentissage profond.

[Retour au plan du cours](#)

1 Introduction

1.1 Historique

L'Intelligence Artificielle, branche de l'Informatique fondamentale s'est développée avec pour objectif la simulation des comportements du cerveau humain. Les premières tentatives de modélisation du cerveau sont anciennes et précèdent même l'ère informatique. C'est en 1943 que McCulloch (neurophysiologiste) et Pitts (logicien) ont proposé les premières notions de *neurone formel*. Ce concept fut ensuite mis en réseau avec une couche d'entrée et une sortie par Rosenblatt en 1959 pour simuler le fonctionnement rétinien et tâcher de reconnaître des formes. C'est l'origine du *perceptron*. Cette approche dite *connexioniste* a atteint ses limites technologiques, compte tenu de la puissance de calcul de l'époque, mais aussi théoriques au début des années 70.

L'approche connexioniste à *connaissance répartie* a alors été supplantée par une approche *symbolique* qui promouvait les *systèmes experts à connaissance localisée* dont l'objectif était d'automatiser le principe de l'expertise humaine en associant trois concepts :

- une *base de connaissance* dans laquelle sont regroupées les connaissances d'experts humains sous forme de propositions logiques élémentaires ou plus élaborées en utilisant des quantificateurs (logique du premier ordre).
- une *base de faits* contenant les observations du cas à traiter comme, par exemple, des résultats d'examens, d'analyses de sang, de salive pour

- des applications biomédicales de choix d'un antibiotique,
- un *moteur d'inférence* chargé d'appliquer les règles expertes sur la base de faits afin d'en déduire de nouveaux faits jusqu'à la réalisation d'un objectif comme le choix du traitement d'une infection bactérienne.

Face aux difficultés rencontrées lors de la modélisation des connaissances d'un expert humain, au volume considérable des bases qui en découlaient et au caractère exponentiel de la complexité des algorithmes d'inférence mis en jeu, cette approche s'est éteinte avec les années 80. Il a été montré que les systèmes basés sur le calcul des prédictifs du premier ordre conduisaient à des problèmes NP complets.

L'essor technologique et quelques avancées théoriques :

- estimation du gradient par rétro-propagation de l'erreur (Hopkins, 1982),
- analogie de la phase d'apprentissage avec les modèles markoviens de systèmes de particules de la mécanique statistique (verres de spin) par (Hopfield, 1982),

au début des années 80 ont permis de relancer l'approche connexioniste. Celle-ci a connu au début des années 90 un développement considérable si l'on considère le nombre de publications et de congrès qui lui ont été consacrés mais aussi les domaines d'applications très divers où elle apparaît. La motivation initiale de simulation du cortex cérébral a été rapidement abandonnée alors que les méthodes qui en découlaient ont trouvé leur propre intérêt de développement méthodologique et leurs champs d'applications.

Remis en veilleuse depuis le milieu des années 90 au profit d'autres algorithmes d'apprentissage machine ou plutôt statistique : *boosting*, *support vector machine*..., les réseaux de neurones connaissent un regain d'intérêt et même un énorme battage médiatique sous l'appellation d'apprentissage profond (*deep learning*). La taille des bases de données, notamment celles d'images issues d'internet, associée à la puissance de calcul disponible, permettent d'estimer les millions de paramètres de perceptrons accumulant des dizaines voire centaines de couches de neurones aux propriétés très spécifiques. Ce succès médiatique est la conséquence des résultats spectaculaires obtenus par ces réseaux en reconnaissance d'image, jeux de go, traitement du langage naturel...

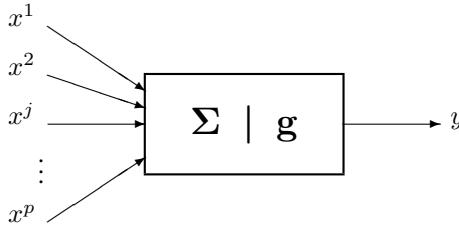


FIGURE 1 – Représentation d'un neurone formel.

1.2 Réseaux de neurones

Un réseau neuronal est l'association, en un graphe plus ou moins complexe, d'objets élémentaires, les *neurones formels*. Les principaux réseaux se distinguent par l'organisation du graphe (en couches, complets...), c'est-à-dire leur architecture, son niveau de complexité (le nombre de neurones, présence ou non de boucles de rétroaction dans le réseau), par le type des neurones (leurs fonctions de transition ou d'activation) et enfin par l'objectif visé : apprentissage supervisé ou non, optimisation, systèmes dynamiques...

1.3 Neurone formel

De façon très réductrice, un neurone biologique est une cellule qui se caractérise par

- des synapses, les points de connexion avec les autres neurones, fibres nerveuses ou musculaires ;
- des dendrites ou entrées du neurones ;
- les axones, ou sorties du neurone vers d'autres neurones ou fibres musculaires ;
- le noyau qui active les sorties en fonction des stimulations en entrée.

Par analogie, le neurone formel est un modèle qui se caractérise par un état interne $s \in \mathcal{S}$, des signaux d'entrée x_1, \dots, x_p et une fonction d'activation

$$s = h(x_1, \dots, x_p) = g\left(\alpha_0 + \sum_{j=1}^p \alpha_j x_j\right) = g(\alpha_0 + \boldsymbol{\alpha}' \mathbf{x}).$$

La fonction d'activation opère une transformation d'une combinaison affine des signaux d'entrée, α_0 , terme constant, étant appelé le biais du neurone. Cette combinaison affine est déterminée par un *vecteur de poids* $[\alpha_0, \dots, \alpha_p]$ associé à chaque neurone et dont les valeurs sont estimées dans la phase d'apprentissage. Ils constituent la *mémoire ou connaissance répartie* du réseau.

Les différents types de neurones se distinguent par la nature g de leur fonction d'activation. Les principaux types sont :

- *linéaire* g est la fonction identité,
- *seuil* $g(x) = \mathbf{1}_{[0,+\infty[}(x)$,
- *sigmoïde* $g(x) = 1/(1 + e^x)$,
- *ReLU* $g(x) = \max(0, x)$ (*rectified linear unit*)
- *radiale* $g(x) = \sqrt{1/2\pi} \exp(-x^2/2)$,
- *stochastiques* $g(x) = 1$ avec la probabilité $1/(1 + e^{-x/H})$, 0 sinon (H intervient comme une température dans un algorithme de recuit simulé),
- ...

Les modèles linéaires, sigmoïdaux, ReLU, sont bien adaptés aux algorithmes d'apprentissage impliquant (cf. ci-dessous) une rétro-propagation du gradient car leur fonction d'activation est différentiable ; ce sont les plus utilisés. Le modèle à seuil est sans doute plus conforme à la réalité biologique mais pose des problèmes d'apprentissage. Enfin le modèle stochastique est utilisé pour des problèmes d'optimisation globale de fonctions perturbées ou encore pour les analogies avec les systèmes de particules (machine de Boltzman).

2 Perceptron multicouche

Nous ne nous intéresserons dans ce cours qu'à une structure élémentaire de réseau, celle dite statique ne présentant pas de boucle de rétroaction et dans un but d'apprentissage supervisé. Les systèmes dynamiques, avec boucle de rétroaction ainsi que les cartes de Kohonen ou cartes auto-organisatrices pour

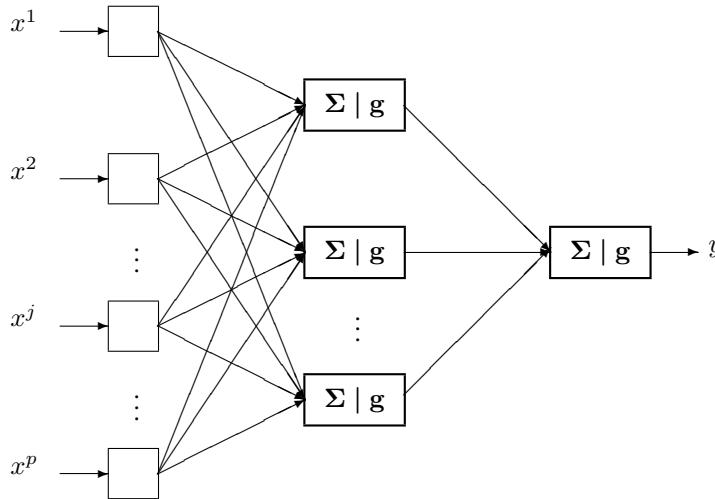


FIGURE 2 – Exemple de perceptron multicouche élémentaire avec une couche cachée et une couche de sortie.

la classification non supervisées ne sont pas abordés.

2.1 Architecture

Le perceptron multicouche (PMC) est un réseau composé de couches successives. Une *couche* est un ensemble de neurones n'ayant pas de connexion entre eux. Une couche d'entrée lit les signaux entrant, un neurone par entrée x_j , une couche en sortie fournit la réponse du système. Selon les auteurs, la couche d'entrée qui n'introduit aucune modification n'est pas comptabilisée. Une ou plusieurs couches cachées participent au transfert.

Dans un perceptron, un neurone d'une couche cachée est connecté en entrée à chacun des neurones de la couche précédente et en sortie à chaque neurone de la couche suivante.

2.2 Fonction de transfert

Par souci de cohérence, les mêmes notations ont été conservées à travers les différents chapitres. Ainsi, les *entrées* d'un réseau sont encore notées X_1, \dots, X_p comme les variables explicatives d'un modèle tandis que les *poids* des entrées sont des paramètres α, β à estimer lors de la procédure d'*apprentissage* et que la *sortie* est la variable Y à expliquer ou cible du modèle.

Un perceptron multicouche réalise donc une transformation des variables d'entrée :

$$Y = f(X_1, \dots, X_p; \alpha)$$

où α est le vecteur contenant chacun des paramètres $\alpha_{j,k,l}$ de la j ème entrée du k ème neurone de la l ème couche; la couche d'entrée ($l = 0$) n'est pas paramétrée, elle ne fait que distribuer les entrées sur tous les neurones de la couche suivante.

Un théorème dit *d'approximation universelle* montre que cette structure élémentaire à une seule couche cachée est suffisante pour prendre en compte les problèmes classiques de modélisation ou apprentissage statistique. En effet, toute fonction régulière peut être approchée uniformément avec une précision arbitraire et dans un domaine fini de l'espace de ses variables, par un réseau de neurones comportant une couche de neurones cachés en nombre fini possédant tous la même fonction d'activation et un neurone de sortie linéaire. *Attention*, ce résultat, qui semble contradictoire avec les structures d'apprentissage profond, est théorique, il masque des difficultés d'apprentissage et de stabilité pour des problèmes complexes en très grande dimension.

De façon usuelle et en régression (Y quantitative), la dernière couche est constituée d'un seul neurone muni de la fonction d'activation identité tandis que les autres neurones (couche cachée) sont munis de la fonction sigmoïde. En classification binaire, le neurone de sortie est muni également de la fonction sigmoïde tandis que dans le cas d'une discrimination à m classes (Y qualitative), ce sont m neurones avec fonction sigmoïde, un par classe, qui sont considérés en sortie.

Ainsi, en régression avec un perceptron à une couche cachée de q neurones et un neurone de sortie, cette fonction s'écrit :

$$\begin{aligned} y = f(\mathbf{x}; \boldsymbol{\alpha}, \boldsymbol{\beta}) &= \beta_0 + \boldsymbol{\beta}' \mathbf{z} \\ \text{avec } z_k &= g(\alpha_{k0} + \boldsymbol{\alpha}_k' \mathbf{x}); k = 1, \dots, q. \end{aligned}$$

2.3 Apprentissage

Supposons que l'on dispose d'une base d'apprentissage de taille n d'observations $(x_i^1, \dots, x_i^p; y_i)$ des variables explicatives X^1, \dots, X^p et de la variable à prévoir Y . Considérons le cas le plus simple de la régression avec un réseau constitué d'un neurone de sortie linéaire et d'une couche à q neurones dont les paramètres sont optimisés par moindres carrés. Ceci se généralise à toute fonction perte dérivable et donc à la discrimination à m classes.

L'apprentissage est l'estimation des paramètres $\boldsymbol{\alpha}_{j=0,p;k=1,q}$ et $\boldsymbol{\beta}_{k=0,q}$ par minimisation de la fonction perte quadratique (ou d'un fonction d'entropie en classification) :

$$Q(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{i=1}^n Q_i = \sum_{i=1}^n [y_i - f(\mathbf{x}; \boldsymbol{\alpha}, \boldsymbol{\beta})]^2.$$

Différents algorithmes d'optimisation sont proposés, ils sont généralement basés sur une évaluation du gradient par rétro-propagation.

2.3.1 Rétro-propagation de l'erreur

Il s'agit donc dévaluer la dérivée de la fonction coût en une observation et par rapport aux différents paramètres. Soit $z_{ki} = g(\alpha_{k0} + \boldsymbol{\alpha}_k' \mathbf{x}_i)$ et $\mathbf{z}_i = \{z_{i1}, \dots, z_{iq}\}$. Les dérivées partielles de la fonction perte quadratique s'écrivent :

$$\begin{aligned} \frac{\partial Q_i}{\partial \beta_k} &= -2(y_i - \phi(x_i))(\boldsymbol{\beta}' \mathbf{z}_i)z_{ki} = \delta_i z_{ki} \\ \frac{\partial Q_i}{\partial \alpha_{kp}} &= -2(y_i - \phi(x_i))(\boldsymbol{\beta}' \mathbf{z}_i)\beta_k f'(\boldsymbol{\alpha}_k' \mathbf{x}_i)x_{ip} = s_{ki}x_{ip}. \end{aligned}$$

Les termes δ_i et s_{ki} sont respectivement les termes d'erreur du modèle courant à la sortie et sur chaque neurone caché. Ces termes d'erreur vérifient les

équations dites de rétro-propagation :

$$s_{ki} = g'(\boldsymbol{\alpha}_k' \mathbf{x}_i)\beta_k \delta_i$$

dont les termes sont évaluée sen deux passes. Une *passe avant*, avec les valeurs courantes des poids : l'application des différentes entrées \mathbf{x}_i au réseau permet de déterminer les valeurs ajustées $\hat{f}(\mathbf{x}_i)$. La *passe retour* permet ensuite de déterminer les δ_i qui sont *rétro-propagés* afin de calculer les s_{ki} et ainsi obtenir les évaluations des gradients.

2.3.2 Algorithmes d'optimisation

Sachant évaluer les gradients, différents algorithmes, plus ou moins sophistiqués, sont implémentés. Le plus élémentaire est une utilisation itérative du gradient : en tout point de l'espace des paramètres, le vecteur gradient de Q pointe dans la direction de l'erreur croissante. Pour faire décroître Q il suffit donc de se déplacer en sens contraire. Il s'agit d'un algorithme itératif modifiant les poids de chaque neurone selon :

$$\begin{aligned} \boldsymbol{\beta}_k^{(r+1)} &= \boldsymbol{\beta}_k^{(r)} - \tau \sum_{i=1}^n \frac{\partial Q_i}{\partial \beta_k^{(r)}} \\ \boldsymbol{\alpha}_{kp}^{(r+1)} &= \boldsymbol{\alpha}_{kp}^{(r)} - \tau \sum_{i=1}^n \frac{\partial Q_i}{\partial \alpha_{kp}^{(r)}}. \end{aligned}$$

Le coefficient de proportionnalité τ est appelé le *taux d'apprentissage*. Il peut être fixe, à déterminer par l'utilisateur, ou encore varier en cours d'exécution selon certaines heuristiques. Il paraît en effet intuitivement raisonnable que, grand au début pour aller plus vite, ce taux décroisse pour aboutir à un réglage plus fin au fur et à mesure que le système s'approche d'une solution.

Si l'espace mémoire est suffisant, une version accélérée de l'algorithme fait intervenir à chaque itération un ensemble (*batch*) d'observations pour moyenner les gradients et mises à jour des poids.

Bien d'autres méthodes d'optimisation ont été adaptées à l'apprentissage d'un réseau : méthodes du gradient avec second ordre utilisant une approximation itérative de la matrice hessienne (algorithme BFGS, de Levenberg-Marquardt) ou encore une évaluation implicite de cette matrice par la méthode

Algorithm 1 Rétro propagation élémentaire du gradient

Initialisation des poids b_{jkl} par tirage aléatoire selon une loi uniforme sur $[0, 1]$.

Normaliser dans $[0, 1]$ les données d'apprentissage.

while $Q > \text{errmax}$ ou $\text{niter} < \text{itermax}$ **do**

Ranger la base d'apprentissage dans un nouvel ordre aléatoire.

for chaque élément $i = 1, \dots, n$ de la base **do**

Calculer $\varepsilon(i) = y_i - f(x_i^1, \dots, x_i^p; (b)(i-1))$ en propageant les entrées vers l'avant.

L'erreur est rétro-propagée dans les différentes couches afin d'affecter à chaque entrée une responsabilité dans l'erreur globale.

Mise à jour de chaque poids $b_{jkl}(i) = b_{jkl}(i-1) + \Delta b_{jkl}(i)$

end for

end while

dite du gradient conjugué. La littérature sur le sujet propose quantités de recettes destinées à améliorer la vitesse de convergence de l'algorithme ou bien lui éviter de rester collé à une solution locale défavorable. D'autres heuristiques proposent d'ajouter un terme d'inertie afin d'éviter des oscillations de l'algorithme.

D'autres algorithmes encore sont des versions adaptatives. Lorsque de nouvelles observations sont proposées une à une au réseau. Dans ce dernier type d'algorithme, des propriétés de dynamique markovienne (processus ergodique convergeant vers la mesure stationnaire) impliquent une convergence presque sûre : la probabilité d'atteindre une précision fixée *a priori* tend vers 1 lorsque la taille de l'échantillon d'apprentissage tend vers l'infini.

On pourra se reporter à l'abondante littérature sur le sujet pour obtenir des précisions sur les algorithmes d'apprentissage et leurs nombreuses variantes. Il est important de rappeler la liste des choix qui sont laissés à l'utilisateur. En effet, même si les logiciels proposent des valeurs par défaut, il est fréquent que cet algorithme connaisse quelques soucis de convergence.

2.4 Contrôle de la complexité

Régularisation

Dans les réseaux élémentaires, une option simple pour éviter le sur-apprentissage consiste à introduire une terme de pénalisation ou régularisation, comme en régression *ridge*, dans le critère à optimiser. Celui-ci devient alors : $Q(\theta) + \gamma \|\theta\|^2$. Plus la valeur du paramètre γ (*decay*) est importante et moins les poids des entrées des neurones peuvent prendre des valeurs chaotiques contribuant ainsi à limiter les risques de sur-apprentissage.

Choix des paramètres

L'utilisateur doit donc déterminer

- les variables d'entrée et la variable de sortie ; leur faire subir comme pour toutes méthodes statistiques, d'éventuelles transformations, normalisations.
- L'architecture du réseau : le nombre de couches cachées qui correspond à une aptitude à traiter des problèmes de non-linéarité, le nombre de neurones par couche cachée. Ces deux choix conditionnent directement le nombre de paramètres (de poids) à estimer et donc la complexité du modèle. Ils participent à la recherche d'un bon compromis biais/variance c'est-à-dire à l'équilibre entre qualité d'apprentissage et qualité de prévision.
- Trois autres paramètres interviennent également sur ce compromis : le nombre maximum d'itérations, l'erreur maximum tolérée et un terme éventuel de régularisation *ridge* (*decay*).
- Le taux d'apprentissage ainsi qu'une éventuelle stratégie d'évolution de celui-ci.
- la taille des ensembles ou *batchs* d'observation considérés à chaque itération.

En pratique, tous ces paramètres ne peuvent être réglés simultanément par l'utilisateur. Celui-ci est confronté à des choix concernant principalement le contrôle du sur-apprentissage : limiter le nombre de neurones ou la durée d'apprentissage ou encore augmenter le coefficient de pénalisation de la norme des paramètres. Ceci nécessite de déterminer un mode d'estimation de l'erreur : échantillon validation ou test, validation croisée ou bootstrap.

Une stratégie simple et sans doute efficace consiste à introduire un nombre

plutôt grand de neurones puis à optimiser le seul paramètre de régularisation (*decay*) par validation croisée.

2.5 Remarques

Les champs d'application des PMC sont très nombreux : discrimination, prévision d'une série temporelle, reconnaissance de forme... Ils sont en général bien explicités dans les documentations des logiciels spécialisés.

Les critiques principales énoncées à l'encontre du PMC concernent les difficultés liés à l'apprentissage (temps de calcul, taille de l'échantillon, localité de l'optimum obtenu) ainsi que son statut de boîte noir. En effet, contrairement à un modèle de discrimination ou un arbre, il est *a priori* impossible de connaître l'influence effective d'une entrée (une variable) sur le système dès qu'une couche cachée intervient. Néanmoins, des techniques de recherche de sensibilité du système à chacune des entrées permettent de préciser les idées et, éventuellement de simplifier le système en supprimant certaines des entrées.

En revanche, ils possèdent d'indéniables qualités lorsque l'absence de linéarité et/ou le nombre de variables explicatives (images) rendent les modèles statistiques traditionnelles inutilisables. Leur flexibilité par l'introduction de couches spécifiques en apprentissage profond, alliée à une procédure d'apprentissage intégrant la pondération (le choix) des variables comme de leurs interactions peuvent les rendre très efficaces.

3 Exemples

Les réseaux de neurones étant des boîtes noires, les résultats fournis ne sont guère explicites et ne conduisent donc pas à des interprétations peu informatives du modèle. Seule une étude des erreurs de prévisions et, dans le cas d'une régression, une étude des résidus, permet de se faire une idée de la qualité du modèle.

3.1 Cancer du sein

La prévision de l'échantillon test par un réseau de neurones conduit à la matrice de confusion ci-dessous et donc une erreur estimée de 3%.

benign malignant

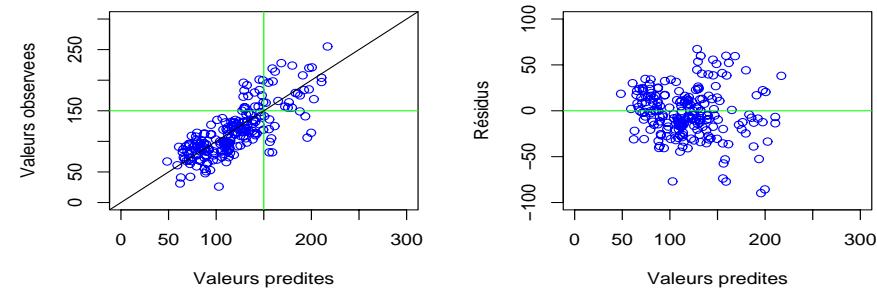


FIGURE 3 – Ozone : Valeurs observées et résidus de l'échantillon test en fonction des valeurs prédites par un réseau de 10 neurones

FALSE	83	1
TRUE	3	50

3.2 Concentration d'ozone

La comparaison des résidus (figure 3) montre que le problème de non-linéarité qui apparaissait sur les modèles simples (MOCAGE, régression linéaire) est bien résolu et que ces résidus sont plutôt moins étendus, mais le phénomène d'hétéroscédasticité est toujours présent quelque soit le nombre de neurones utilisés. Il a été choisi relativement important (10) et conduit donc à un bon ajustement ($R^2 = 0,77$) mais devra être réduit pour optimiser la prévision.

L'optimisation des paramètres d'un réseau de neurones est instable comme pour les proches voisins car chaque exécution de l'estimation de l'erreur par validation croisée fournit des résultats différents. Elle est en plus très compliquée par le nombre de paramètres à optimiser : nombre de neurones sur la couche (*size*), pénalisation (*decay*), nombre d'itérations. Une fonction de la librairie e1071 permet de faire varier à la fois la taille et la pénalisation et fournit des graphiques élégants (figure 4) mais les exécutions sont très longues et les résultats pas toujours pertinents. Le plus efficace semble être de fixer

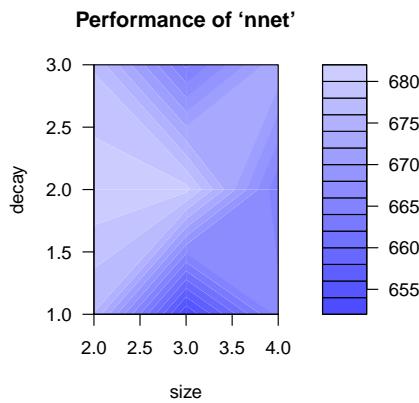


FIGURE 4 – Ozone : optimisation des paramètres (pénalisation et nombre de neurones) par validation croisée.

“assez grands” la taille (nombre de neurones) et le nombre d’itérations pour se focaliser sur le seul réglage de la pénalisation.

Comme pour les arbres de décision, les réseaux de neurones ne proposent pas de modèles très efficaces sur cet exemple. Les taux d’erreur de prévision du dépassement du seuil sont de 14,4% à partir du modèle quantitatif et de 15,6% avec une prévision directement qualitative. Les courbes ROC estimées sur l’échantillon test permettent de comparer les méthodes. Dans ce cas et pour l’échantillon test concerné, la méthode la plus efficace (figure 5) pour prévoir le dépassement du pic d’ozone est un réseau de neurone modélisant la concentration plutôt que la prévision directe du dépassement (logit ou réseau qualitatif).

3.3 Données bancaires

Une fonction de la librairie `e1071`, pratique mais très chronophage, propose une automatisation de l’optimisation des paramètres (*decay*, nombre de neurones). Elle produit une carte de type contour permettant d’évaluer “à l’œil” les valeurs optimales. La prévision de l’échantillon test par ce réseau de neu-

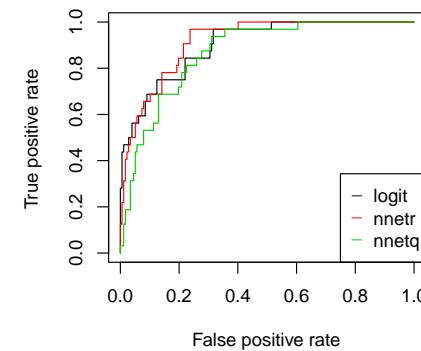


FIGURE 5 – Ozone : courbes ROC pour la régression logistique et les réseaux de neurones.

rones conduit à la matrice de confusion ci-dessous et donc une erreur estimée de 21,5% :

pred.vistest	FALSE	TRUE
FALSE	110	16
TRUE	27	47

4 Introduction à l’apprentissage profond

Les techniques associées sont simplement introduites dans ce document, elles sont développées dans celui associé au cours de *Statistique en grande dimension*.

4.1 Préambule

Pendant les années 90s et le début des années 2000, le développement de l’apprentissage machine s’est focalisé sur les algorithmes de machines à vecteurs supports et ceux d’agrégation de modèles. Pendant une relative mise en

veilleuse du développement de la recherche sur les réseaux de neurones, leur utilisation est restée présente de même qu'une veille attendant le développement de la puissance de calcul et celle des grandes bases de données, notamment d'images.

Le renouveau de la recherche dans ce domaine est dû à Yoshua Bengio et Yan Le Cun qui a tenu à jour un [célèbre site](#) dédié à la reconnaissance des caractères manuscrits de la base MNIST. La liste des publications listées sur ce site témoigne de la lente progression de la qualité de reconnaissance, de 12% avec un simple perceptron à 1 couche jusqu'à moins de 0,3% en 2012 par l'introduction et l'amélioration incrémentale d'une couche de neurones spécifique appelée *convolutional neural network* (ConvNet). L'étude de ces données qui ont servi de benchmark pour la comparaison de très nombreuses méthodes sert maintenant de données jouet pour beaucoup de tutoriels des environnements dédiés (*TensorFlow*, *Keras*, *pyTorch*, *Caffe*...)

Schématiquement, trois grandes familles de réseaux d'apprentissage profond sont développées avec des ambitions industrielles.

convolutional neural networks (ConvNet) pour l'analyse d'images.

long-short term memory (LSTM) lorsqu'une dimension temporelle ou plus généralement des propriétés d'autocorrélation sont à prendre en compte pour le traitement du signal ou encore l'analyse du langage naturel.

autoEncoder decoder ou réseau *diabolo* en apprentissage non supervisé pour, par exemple, le débruitage d'images ou signaux, la détection d'anomalies.

Seul le premier point est développé pour illustrer les principaux enjeux.

4.2 Reconnaissance d'images

Cette couche de neurones (ConvNet) illustrée par la figure 6 ou plutôt un empilement de ces couches introduit des propriétés spécifiques d'*invariance par translation*. Ces propriétés sont indispensables à l'objectif de reconnaissance de caractères et plus généralement d'images qui peuvent être vues sous des angles différents. C'est dans ce domaine que les résultats les plus spectaculaires ont été obtenus tandis que l'appellation *deep learning* était avancée afin d'accompagner le succès grandissant et le battage médiatique associé.

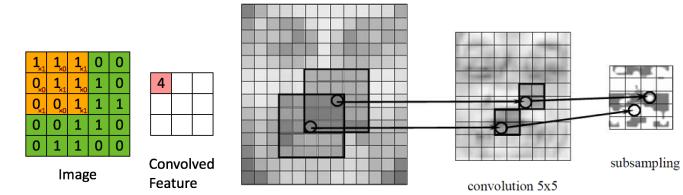


FIGURE 6 – Principe élémentaire d'une couche de convolution et application à une image.

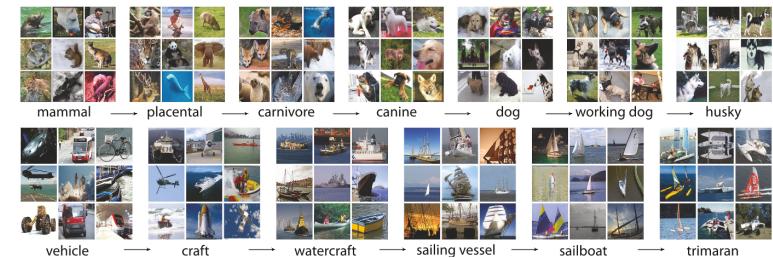


FIGURE 7 – Échantillon de la base ImageNet.

La communauté de reconnaissance d'images (figure 7) se confronte chaque année depuis 2010 sur un jeu de données issues d'une base d'images labelisées : 15 millions d'images, 22000 catégories hiérarchisées. De cette base sont extraits 1,2 millions d'images pour l'apprentissage avec 1000 catégories. Les participants au concours doivent prévoir la catégorie de 15000 images de l'échantillon test. Ce projet à l'initiative de l'Université Stanford est largement sponsorisé par Google.

Comme pour les données de reconnaissance de caractères, une progression largement empirique a conduit à l'introduction et au succès d'un réseau empilant des couches de neurones aux propriétés particulières. Cette progression est retracée dans le tableau 8. C'est en 2012 qu'une équipe utilise pour la

2012 Teams	%error	2013 Teams	%error	2014 Teams	%error
Supervision (Toronto)	15.3	Clarifai (NYU spinoff)	11.7	GoogLeNet	6.6
ISI (Tokyo)	26.1	NUS (singapore)	12.9	VGG (Oxford)	7.3
VGG (Oxford)	26.9	Zeiler-Fergus (NYU)	13.5	MSRA	8.0
XRCE/INRIA	27.0	A. Howard	13.5	A. Howard	8.1
UvA (Amsterdam)	29.6	OverFeat (NYU)	14.1	DeeperVision	9.5
INRIA/LEAR	33.4	UvA (Amsterdam)	14.2	NUS-BST	9.7
		Adobe	15.2	TTIC-ECP	10.2
		VGG (Oxford)	15.2	XYZ	11.2
		VGG (Oxford)	23.0	UvA	12.1

FIGURE 8 – Classements successifs (Le Cun 2016) des équipes participant au concours ImageNet. En rouge, celles utilisant des neurones profonds.

première fois un réseau de neurones profond contrairement à des traitements spécifiques et ad'hoc de l'analyse d'images utilisées jusque là. L'amélioration était telle que toutes les équipes ont ensuite adopté cette technologie pour une succession d'améliorations empiriques. En 2016 une équipe propose un réseau à 152 couches et atteint un taux d'erreur de 3%, mieux que les 5% d'un expert humain.

Ce concours est depuis lors abandonné au profit de problèmes plus complexes de reconnaissance de scènes associant plusieurs objets ou thèmes.

4.3 Couches pour l'apprentissage profond

Construire un réseau d'apprentissage profond consiste à empiler des couches de neurones aux propriétés spécifiques rapidement résumées ci-dessous. Le choix de du type, de l'ordre, de la complexité de chacune de ces couches ainsi que du nombre est complètement empirique et l'aboutissement de très nombreuses expérimentations nécessitant des moyens de calculs et bases de données considérables.

fully connected Couche classique de perceptron et dernière couche d'un

réseau profond qui opère la discrimination finale entre par exemple des images à reconnaître. Les couches précédentes construisant, extrayant, des caractéristiques (*features*) de celles-ci.

convolution opère une convolution sur le signal d'entrée en associant une réduction de dimension (cf. figure 6).

pooling réduction de dimension en remplaçant un sous-ensemble des entrées (sous-image) par une valeur, généralement le max.

normalisation identique au précédent avec une opération de centrage et / ou de normalisation des valeurs.

drop out les paramètres estimés sont les possibilités de supprimer des neurones d'une couche afin de réduire la dimension.

...

4.4 Utilisation rudimentaire

Sans bases de données très volumineuse et moyens de calcul substantiels il est illusoire de vouloir apprendre un réseau profond impliquant l'estimation de millions de paramètres. Une mise en œuvre simple sur des données spécifiques consiste à :

- Identifier un réseau ou modèle existant appris sur des données similaires. Pour les images, considérer par exemple les versions des réseaux *inception* de *tensorFlow* ou *AlexNet* de *Caffe*.
- Supprimer la dernière couche du modèle dédiée à la classification,
- Apprendre les poids de cette seule dernière couche sur les données spécifiques.

Agrégation de modèles

Résumé

Les algorithmes décrits sont basés sur des stratégies adaptatives (boosting, gradient boosting) ou aléatoires (bagging, random forest) permettant d'améliorer l'ajustement par une combinaison ou agrégation d'un grand nombre de modèles tout en évitant ou contrôlant le sur-ajustement. Définitions, optimisation et principes d'utilisation de ces algorithmes.

[Retour au plan du cours](#)

1 Introduction

Deux types d'algorithmes sont abordés. Ceux reposant sur une construction aléatoire d'une famille de modèles : *bagging* pour *bootstrap aggregating* (Breiman 1996)[2] et les forêts aléatoires (*random forests*) de Breiman (2001)[4] qui propose une amélioration du *bagging* spécifique aux modèles définis par des arbres binaires (CART). Ceux basés sur le *boosting* (Freund et Shapiro, 1996)[8] et qui reposent sur une construction *adaptative*, déterministe ou aléatoire, d'une famille de modèles. Ces algorithmes se sont développés à la frontière entre apprentissage machine (*machine learning*) et Statistique. De nombreux articles comparatifs montrent leur efficacité sur des exemples de données simulées et surtout pour des problèmes réels complexes (Fernandez-Delgado et al. 2014)[7]). Elles participent régulièrement aux solutions gagnantes des concours de prévisions de type [Kaggle](#) et leurs propriétés théoriques sont un thème de recherche toujours actif.

Les principes du *bagging* ou du *boosting* s'appliquent à toute méthode de modélisation (régression, CART, réseaux de neurones) mais n'ont d'intérêt, et réduisent sensiblement l'erreur de prévision, que dans le cas de modèles *instables*, donc plutôt non linéaires. Ainsi, l'utilisation de ces algorithmes n'a guère de sens avec la régression multilinéaire ou l'analyse discriminante. Ils sont surtout mis en œuvre en association avec des arbres binaires comme modèles de base. En effet, l'instabilité déjà soulignée des arbres apparaît alors comme une propriété nécessaire à la réduction de la variance par agrégation de

modèles.

La présentation des ces algorithmes toujours en évolution nécessite une mise à jour continue entraînant une progression chronologique des versions historiques (*bagging*, *adaboost*) à l'*extrem gradient boosting*. Ce choix ou plutôt l'adaptation à cette contrainte n'est sans doute pas optimal mais présente finalement quelques vertus pédagogiques en accompagnant l'accroissement de la complexité des algorithmes présentés.

2 Famille de modèles aléatoires

2.1 Bagging

Principe et algorithme

Soit Y une variable à expliquer quantitative ou qualitative, X^1, \dots, X^p les variables explicatives et $f(\mathbf{x})$ un modèle fonction de $\mathbf{x} = \{x^1, \dots, x^p\} \in \mathbb{R}^p$. On note n le nombre d'observations et

$$\mathbf{z} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$$

un échantillon de loi F .

Considérant B échantillons indépendants notés $\{\mathbf{z}_b\}_{b=1,B}$, une prévision par *agrégation de modèles* est définie ci-dessous dans le cas où la variable à expliquer Y est :

- quantitative : $\hat{f}_B(\cdot) = \frac{1}{B} \sum_{b=1}^B \hat{f}_{\mathbf{z}_b}(\cdot)$,
- qualitative : $\hat{f}_B(\cdot) = \arg \max_j \text{card} \left\{ b \mid \hat{f}_{\mathbf{z}_b}(\cdot) = j \right\}$.

Dans le premier cas, il s'agit d'une simple moyenne des résultats obtenus pour les modèles associés à chaque échantillon, dans le deuxième, un *comité* de modèles est constitué pour *voter* et *élire* la réponse la plus probable. Dans ce dernier cas, si le modèle retourne des probabilités associées à chaque modalité comme en régression logistique ou avec les arbres de décision, il est aussi simple de calculer des moyennes de ces probabilités.

Le principe est élémentaire, moyennant les prévisions de plusieurs modèles indépendants permet de *réduire la variance* et donc de réduire l'erreur de prévision.

Cependant, il n'est pas réaliste de considérer B échantillons indépendants.

Cela nécessiterait généralement trop de données. Ces échantillons sont donc remplacés par B répliques d'échantillons *bootstrap* obtenus chacun par n tirages avec remise selon la mesure empirique \hat{F} . Ceci conduit à l'algorithme ci-dessous.

Algorithm 1 Bagging

```

Soit  $\mathbf{x}_0$  à prévoir et
 $\mathbf{z} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$  un échantillon
for  $b = 1$  à  $B$  do
  Tirer un échantillon bootstrap  $\mathbf{z}_b^*$ .
  Estimer  $\hat{f}_{\mathbf{z}_b}(\mathbf{x}_0)$  sur l'échantillon bootstrap.
end for
Calculer l'estimation moyenne  $\hat{f}_B(\mathbf{x}_0) = \frac{1}{B} \sum_{b=1}^B \hat{f}_{\mathbf{z}_b}(\mathbf{x}_0)$  ou le résultat du vote.
  
```

Erreur out-of-bag

Il est naturel et techniquement facile d'accompagner ce calcul par une estimation *out-of-bag* (*o.o.b.*) de l'erreur de prévision car sans biais, ou plutôt pessimiste, comme en validation croisée.

Erreur *o.o.b.* : Pour chaque observation (y_i, \mathbf{x}_i) considérer les seuls modèles estimés sur un échantillon *bootstrap* ne contenant pas cette observation (à peu près 1/3). Prévoir la valeur \hat{y} comme précédemment (moyenne ou vote) et calculer l'erreur de prévision associée ; moyenner sur toute les observations.

Utilisation

En pratique, CART est souvent utilisée comme méthode de base pour construire une famille de modèles c'est-à-dire d'arbres binaires. L'effet obtenu, par moyenage d'arbres, est une forme de "lissage" du pavage de l'espace des observations pour la construction des règles de décision. Trois stratégies d'élagage sont possibles :

1. laisser construire et garder un arbre complet pour chacun des échantillons en limitant le nombre minimale (5 par défaut) d'observation par feuille ;
2. construire un arbre d'au plus q feuilles ou de profondeur au plus q ;
3. construire à chaque fois l'arbre complet puis l'élaguer par validation croisée.

La première stratégie semble en pratique un bon compromis entre volume des calculs et qualité de prévision. Chaque arbre est alors affecté d'un faible biais et d'une grande variance mais la moyenne des arbres réduit avantageusement

celle-ci. En revanche, l'élagage par validation croisée pénalise les calculs sans, en pratique, gain substantiel de qualité.

Cet algorithme a l'avantage de la simplicité, il s'adapte et se programme facilement quelque soit la méthode de modélisation mise en œuvre. Il pose néanmoins quelques problèmes :

- temps de calcul pour évaluer un nombre suffisant d'arbres jusqu'à ce que l'erreur de prévision *out-of-bag* ou sur un échantillon validation se stabilise et arrête si elle tend à augmenter ;
- nécessiter de stocker tous les modèles de la combinaison afin de pouvoir utiliser cet outil de prévision sur d'autres données,
- l'amélioration de la qualité de prévision se fait au détriment de l'interprétabilité. Le modèle finalement obtenu devient une *boîte noire*.

2.2 Forêts aléatoires

Motivation

Dans les cas spécifique des modèles d'arbres binaires de décision (CART), Breiman (2001)[4] propose une amélioration du *bagging* par l'ajout d'une composante aléatoire. L'objectif est donc de rendre plus *indépendants* les arbres de l'agrégation en ajoutant du hasard dans le choix des variables qui interviennent dans les modèles. Depuis la publication initiale de l'algorithme, cette méthode a beaucoup été testée, comparée (Fernandez-Delgado et al. 2014)[7], (Caruana et al. 2008[5]), analysée. Elle devient dans beaucoup d'articles d'apprentissage machine la méthode à battre en matière de qualité de prévision alors que ses propriétés théoriques de convergence, difficiles à étudier, commencent à être publiées (Scornet et al. 2015)[13]. Néanmoins elle peut conduire aussi à de mauvais résultats notamment lorsque le problème sous-jacent est linéaire et donc qu'une simple régression PLS conduit à de bonnes prévisions même en grande dimension. C'est le cas par exemple de données de **spectrométrie en proche infra-rouge** (NIR).

Plus précisément, la variance de la moyenne de B variables indépendantes, identiquement distribuées, chacune de variance σ^2 , est σ^2/B . Si ces variables sont identiquement distribuées mais en supposant qu'elles sont corrélées deus à deux de corrélation ρ , Breiman (2001)[4] montre que la variance de la moyenne

devient :

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2.$$

Comme dans le cas indépendant, le 2ème terme décroît avec B mais le premier limite considérablement l'avantage du *bagging* si la corrélation est élevée. C'est ce qui motive principalement la *randomisation* introduite dans l'algorithme ci-dessous afin de réduire ρ entre les prévisions fournies par chaque modèle.

Algorithme

Le *bagging* est appliqué à des arbres binaires de décision en ajoutant un tirage aléatoire de m variables explicatives parmi les p .

Algorithm 2 Forêts Aléatoires

Soit \mathbf{x}_0 à prévoir et

$\mathbf{z} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ un échantillon

for $b = 1$ à B **do**

Tirer un échantillon bootstrap \mathbf{z}_b^*

Estimer un arbre sur cet échantillon avec **randomisation** des variables : la recherche de chaque division optimale est précédée d'un tirage aléatoire d'un sous-ensemble de m prédicteurs.

end for

Calculer l'estimation moyenne $\widehat{f}_B(\mathbf{x}_0) = \frac{1}{B} \sum_{b=1}^B \widehat{f}_{\mathbf{z}_b}(\mathbf{x}_0)$ ou le résultat du vote.

Paramètres de l'algorithme

La stratégie d'élagage peut, dans le cas des forêts aléatoires, être plus élémentaire qu'avec le *bagging* en se limitant à des arbres de taille q relativement réduite voire même triviale avec $q = 2$ (*stump*). En effet, avec le seul *bagging*, des arbres limités à une seule fourche risquent d'être très semblables (fortement corrélés) car impliquant les mêmes quelques variables apparaissant comme les plus explicatives. Dans la stratégie par défaut de l'algorithme, c'est simplement le nombre minimum d'observation par noeuds qui limite la taille de l'arbre, il est fixé à par défaut à 5. Ce sont donc des arbres plutôt complets qui sont considérés, chacun de faible biais mais de variance importante.

La sélection aléatoire d'un nombre réduit de m prédicteurs potentiels à chaque étape de construction d'un arbre, accroît significativement la variabilité en mettant en avant nécessairement d'autres variables. Chaque modèle de base est évidemment moins performant, sous-optimal, mais, l'union faisant la force, l'agrégation conduit finalement à de bons résultats. Le nombre m de variables tirées aléatoirement peut, selon les exemples traités, être un paramètre sensible avec des choix par défaut pas toujours optimaux :

- $m = \sqrt{p}$ dans un problème de classification,
- $m = p/3$ dans un problème de régression.

Comme pour le *bagging*, l'évaluation itérative de l'erreur *out-of-bag* permet de contrôler le nombre B d'arbres de la forêt de même qu'éventuellement optimiser le choix de m . C'est néanmoins une procédure de validation croisée qui est préféablement opérée pour optimiser m .

Importance des variables

Comme pour tout modèles construit par agrégation ou boîte noire, il n'y a pas d'interprétation directe. Néanmoins des informations pertinentes sont obtenues par le calcul et la représentation graphique d'indices proportionnels à l'*importance* de chaque variable dans le modèle agrégé et donc de sa participation à la régression ou à la discrimination. C'est évidemment d'autant plus utile que les variables sont très nombreuses. Deux critères sont ainsi proposés pour évaluer l'importance de la j ème variable.

- Le premier (*Mean Decrease Accuracy*) repose sur une permutation aléatoire des valeurs de cette variable. Plus la qualité, estimée par une l'erreur *out-of-bag*, de la prévision est dégradée par la permutation des valeurs de cette variable, plus celle-ci est importante. Une fois le b -ème arbre construit, l'échantillon *out-of-bag* est prédit par cet arbre et l'erreur estimée enregistrée. Les valeurs de la j -ème variable sont aléatoirement permutées dans l'échantillon *out-of-bag* et l'erreur à nouveau calculée. La décroissance de la qualité de prévision est moyennée sur tous les arbres et utilisée pour évaluer l'importance de la variable j dans la forêt. Il s'agit donc d'une mesure globale mais indirecte de l'influence d'une variable sur la qualité des prévisions.
- Le deuxième (*Mean Decrease Gini*) est local, basé sur la décroissance de l'hétérogénéité définie à partir du critère de Gini ou de celui d'entropie. L'importance d'une variable est alors une somme pondérée des

décroissances d'hétérogénéité induites lorsqu'elle est utilisée pour définir la division associée à un nœud.

Implémentations

- L'implémentation la plus utilisée est celle de la librairie `randomForest` de R qui ne fait qu'interfacer le programme original développé en Fortran77 par Léo Breiman et Adele Cutler qui maintient le [site](#) dédié à cet algorithme.
- Une alternative en R, plus efficace en temps de calcul surtout avec un volume important de données, consiste à utiliser la librairie `ranger`.
- Le site du logiciel `Weka` développé à l'université Waikato de Nouvelle Zélande propose une version en Java.
- Une version très efficace et proche de l'algorithme original est disponible dans la librairie `Scikit-learn` de Python.
- Une autre version adaptée aux données massives est proposée dans la librairie `MLLib` de `Spark`, technologie développée pour interfaçer différentes architectures matérielles/logicielles avec des systèmes de gestion de fichiers de données distribuées (Hadoop). En plus du nombre d'arbres, de la profondeur maximum des arbres et du nombre de variables tirées au hasard à chaque recherche de division optimale pour construire un nœud, cette implémentation ajoute "innocemment" deux paramètres : `subsamplingRate` et `maxBins` nantis d'une valeur par défaut. Ces paramètres jouent un rôle important, certes pour réduire drastiquement le temps de calcul, mais, en contre partie, pour restreindre la précision de l'estimation. Ils règlent l'équilibre entre temps de calcul et précision de l'estimation comme le ferait un échantillonnage des données.

`subsamplingRate = 1.0` sous échantillonne comme son nom l'indique avant la construction de chaque arbre. Avec la valeur par défaut, c'est la version classique des forêts aléatoires avec B Bootstrap mais si ce taux est faible, ce sont des échantillons de taille réduites mais plus distincts ou indépendants pour chaque arbre qui sont tirés. La variance est d'autant réduite (arbre plus indépendants) mais le biais augmente car chaque arbre est moins bien estimé sur un petit lot.

`maxBins=32` est le nombre maximum de modalités qui sont consi-

dérées pour une variable qualitative ou encore le nombre de valeurs possibles pour une variable quantitative. Seules les modalités les plus fréquentes d'une variable qualitative sont prises en compte, les autres sont automatiquement regroupées dans une modalité autre. Comme précédemment, le temps de recherche d'une meilleure division est évidemment largement influencé par le nombre de modalités ou encore le nombre de valeurs possibles d'une variable quantitative en opérant des découpages en classe. Réduire le nombre de valeurs possibles est finalement une autre façon de réduire la taille de l'échantillon mais, il serait sans doute opportun de mieux contrôler ces paramètres notamment en guidant les regroupements des modalités pour éviter des contre sens. Ne vaut-il pas mieux sous-échantillonner préalablement les données plutôt que de restreindre brutalement le nombre de valeurs possibles ?

Autres utilisations

Devenu le *couteau suisse* de l'apprentissage, les forêts aléatoires sont utilisées à différentes fins (consulter le [site dédié](#)) :

- Similarité ou proximité entre observations. Après la construction de chaque arbre, incrémenter la similarité ou proximité de deux observations qui se trouvent dans la même feuille. Sommer sur la forêt, normaliser par le nombre d'arbres. Un **positionnement multidimensionnel** peut représenter ces similarités ou la matrice des dissimilarités qui en découle.
- Détection d'observations atypiques multidimensionnelles (*outliers*) ou de "nouveautés" (*novelties*) pour signifier qu'une observation n'appartient pas aux classes connues. Un critère d'"anormalité" par rapport à une classe est basé sur la notion précédente de proximités (faible) d'une observation aux autres observations de sa classe.
- Classification non supervisée. Si aucune variables Y n'est à modéliser, l'idée est de se ramener au cas précédent en simulant des observations constituant une deuxième classe synthétique. à partir de celles connues (première classe). Pour ce faire, chaque colonne (variable) est aléatoirement permutée détruisant ainsi la structure de corrélation entre les variables. Une forêt est estimée pour modéliser la variable ainsi créée puis les mêmes approches : matrice de dissimilarités, classification non

supervisée à partir de cette matrice, positionnement multidimensionnel, détection d'observations atypiques, sont développées.

- **Imputation de données manquantes.**
- Modèles de durée de vie (*survival forest*).

3 Famille de modèles adaptatifs

3.1 Principes du *Boosting*

Le *boosting* diffère des approches précédentes par ses origines et ses principes. L'idée initiale, en apprentissage machine, était d'améliorer les compétences d'un *faible classifieur* c'est-à-dire celle d'un modèle de discrimination dont la probabilité de succès sur la prévision d'une variable qualitative est légèrement supérieure à celle d'un choix aléatoire. L'idée originale de Schapire de 1990 a été affinée par Freund et Schapire (1996)[8] qui ont décrit l'algorithme original *adaBoost* (*Adaptive boosting*) pour la prévision d'une variable binaire. De nombreuses études ont ensuite été publiées pour adapter cet algorithme à d'autres situations : k classes, régression, paramètre de *shrinkage* et rendre compte de ses performances sur différents jeux de données. Ces tests ont montré le réel intérêt pratique de ce type d'algorithme pour réduire sensiblement la variance (comme le *bagging*) mais aussi le biais de prévision comparativement à d'autres approches. En effet, comme les arbres sont identiquement distribués par *bagging*, l'espérance de B arbres est la même que l'espérance d'un arbre. Cela signifie que le biais d'arbres agrégés par *bagging* est le même que celui d'un seul arbre. Ce n'est plus le cas avec le *boosting*.

Cet algorithme fut considéré comme la meilleure méthode *off-the-shelf* c'est-à-dire ne nécessitant pas un long prétraitement des données ni un réglage fin de paramètres lors de la procédure d'apprentissage. Néanmoins, l'évolution vers de nouvelles versions (*extrem gradient boosting*) plus performantes en terme de prévision a conduit à multiplier le nombre de paramètres à régler et optimiser.

Le *boosting* adopte le même principe général que le *bagging* : construction d'une famille de modèles qui sont ensuite agrégés par une moyenne pondérée des estimations ou un vote. Il diffère nettement sur la façon de construire la famille qui est dans ce cas récurrente : chaque modèle est une version *adaptive* du précédent en donnant plus de poids, lors de l'estimation suivante,

aux observations mal ajustées ou mal prédictes. Intuitivement, cet algorithme concentre donc ses efforts sur les observations les plus difficiles à ajuster tandis que l'agrégation de l'ensemble des modèles réduit le risque de sur-ajustement.

Les algorithmes de *boosting* proposés diffèrent par différentes caractéristiques :

- la façon de pondérer c'est-à-dire de renforcer l'importance des observations mal estimées lors de l'itération précédente,
- leur objectif selon le type de la variable à prédire Y : binaire, qualitative à k classes, réelles ;
- la fonction perte, qui peut être choisie plus ou moins robuste aux valeurs atypiques, pour mesurer l'erreur d'ajustement ;
- la façon d'agrégier, ou plutôt pondérer, les modèles de base successifs.

La littérature sur le sujet présente donc de très nombreuses versions de cet algorithme dont le dernier avatar, proposé dans une librairie XGBoost (*extrem gradient boosting*) connaît beaucoup de succès dans les concours de prévision Kaggle.

Cette section propose une présentation historique, car pédagogique, d'adaBoost à XGBoost.

3.2 Algorithme de base

Décrivons la version originale du *boosting* pour un problème de discrimination élémentaire à deux classes en notant δ la fonction de discrimination à valeurs dans $\{-1, 1\}$. Dans cette version, le modèle de base retourne l'identité d'une classe, il est encore nommé adaBoost discret. Il est facile de l'adapter à des modèles retournant une valeur réelle comme une probabilité d'appartenance à une classe.

Les poids de chaque observations sont initialisés à $1/n$ pour l'estimation du premier modèle puis évoluent à chaque itération donc pour chaque nouvelle estimation. L'importance d'une observation w_i est inchangée si elle est bien classée, elle croît sinon proportionnellement au défaut d'ajustement du modèle. L'agrégation finale des prévisions : $\sum_{m=1}^M c_m \delta_m(\mathbf{x}_0)$ est une combinaison pondérée par les qualités d'ajustement de chaque modèle. Sa valeur absolue appelée *marge* est proportionnelle à la confiance que l'on peut attribuer à son signe qui fournit le résultat de la prévision. Attention, un contrôle doit être ajouté en pratique pour bien vérifier que le classifieur de base est faible

Algorithm 3 adaBoost ou (*adaptive boosting*)

Soit \mathbf{x}_0 à prévoir et

$\mathbf{z} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ un échantillon

Initialiser les poids $\mathbf{w} = \{w_i = 1/n ; i = 1, \dots, n\}$.

for $m = 1$ à M **do**

Estimer δ_m sur l'échantillon pondéré par \mathbf{w} .

Calculer le taux d'erreur apparent :

$$\hat{\mathcal{E}}_p = \frac{\sum_{i=1}^n w_i \mathbf{1}\{\delta_m(\mathbf{x}_i) \neq y_i\}}{\sum_{i=1}^n w_i}.$$

Calculer les logit : $c_m = \log((1 - \hat{\mathcal{E}}_p)/\hat{\mathcal{E}}_p)$.

Calculer les nouvelles pondérations :

$$w_i \leftarrow w_i \cdot \exp [c_m \mathbf{1}\{\delta_m(\mathbf{x}_i) \neq y_i\}] ; i = 1, \dots, n.$$

end for

Résultat du vote : $\hat{f}_M(\mathbf{x}_0) = \text{signe} \left[\sum_{m=1}^M c_m \delta_m(\mathbf{x}_0) \right]$.

mais pas mauvais à savoir que c_m garde bien des valeurs positives ; que le taux d'erreur apparent ne soit pas supérieur à 50%.

Ce type d'algorithme est utilisé avec un arbre (CART) comme modèle de base. De nombreuses applications montrent que si le classifieur faible est un arbre trivial à deux feuilles (*stump*), adaBoost fait mieux qu'un arbre sophistiqué pour un volume de calcul comparable : autant de feuilles dans l'arbre que d'itérations dans adaBoost. Hastie et col. (2001)[11] discutent la meilleure stratégie d'élagage applicable à chaque modèle de base. Ils le comparent avec le niveau d'interaction requis dans un modèle d'analyse de variance. Le cas $q = 2$ correspondant à la seule prise en compte des effets principaux. Empiriquement ils recommandent une valeur comprise entre 4 et 8.

De nombreuses adaptations ont été proposées à partir de l'algorithme initial. Elles font intervenir différentes fonctions pertes offrant des propriétés de robustesse ou adaptées à une variable cible Y quantitative ou qualitative à plusieurs classes : adaBoost M1, M2, MH ou encore MR. Schapire (2002)[12] liste une bibliographie détaillée.

3.3 Version aléatoire

À la suite de Freund et Schapire (1996)[8], Breiman (1998)[3] développe aussi, sous le nom d'*Arcing* (adaptively resample and combine), une version aléatoire, et en pratique très proche, du *boosting*. Elle s'adapte à des classifieurs pour lesquels il est difficile voire impossible d'intégrer une pondération des observations dans l'estimation. Ainsi plutôt que de jouer sur les pondérations, à chaque itération, un nouvel échantillon est tiré avec remise, comme pour le bootstrap, mais selon des probabilités inversement proportionnelles à la qualité d'ajustement de l'itération précédente. La présence des observations difficiles à ajuster est ainsi renforcée pour que le modèle y consacre plus d'attention. L'algorithme *adaBoost* précédent est facile à adapter en ce sens en regardant celui développé ci-dessous pour la régression et qui adopte ce point de vue.

3.4 Pour la régression

Différentes adaptations du *boosting* ont été proposées pour le cas de la régression, c'est-à-dire lorsque la variable à prédire est quantitative. Voici l'algorithme de Drucker (1997) dans la présentation de Gey et Poggi (2002)[10] qui en étudient les performances empiriques en relation avec CART. Freund

et Schapire (1996) ont proposé *adaBoost.R* avec le même objectif tandis que le point de vue de Friedman (2002)[9] est décrit dans la section suivante par l'algorithme de *gradient boosting machine*.

Algorithm 4 Boosting pour la régression

Soit \mathbf{x}_0 à prévoir et

$\mathbf{z} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ un échantillon

Initialiser \mathbf{p} par la distribution uniforme $\mathbf{p} = \{p_i = 1/n ; i = 1, \dots, n\}$.

for $m = 1$ à M **do**

Tirer avec remise dans \mathbf{z} un échantillon \mathbf{z}_m^* suivant \mathbf{p} .

Estimer \hat{f}_m sur l'échantillon \mathbf{z}_m^* .

Calculer à partir de l'échantillon initial \mathbf{z} :

$$l_m(i) = l(y_i, \hat{f}_m(\mathbf{x}_i)) \quad i = 1, \dots, n; \quad (l : \text{fonction perte})$$

$$\widehat{\mathcal{E}}_m = \sum_{i=1}^n p_i l_m(i);$$

$$w_i = g(l_m(i)) p_i. \quad (g \text{ continue non décroissante})$$

Calculer les nouvelles probabilités : $p_i \leftarrow \frac{w_i}{\sum_{i=1}^n w_i}$.

end for

Calculer $\hat{f}(\mathbf{x}_0)$ moyenne ou médiane des prévisions $\hat{f}_m(\mathbf{x}_0)$ pondérées par des coefficients $\log(\frac{1}{\beta_m})$.

Précisions :

- Dans cet algorithme la fonction perte l peut être exponentielle, quadratique ou, plus robuste, la valeur absolue. Le choix usuel de la fonction quadratique est retenu par Gey et Poggi (2002)[10].
- Notons $L_m = \sup_{i=1, \dots, n} l_m(i)$ le maximum de l'erreur observée par le modèle \hat{f}_m sur l'échantillon initial. La fonction g est définie par :

$$g(l_m(i)) = \beta_m^{1-l_m(i)/L_m} \quad (1)$$

$$\text{avec } \beta_m = \frac{\widehat{\mathcal{E}}_m}{L_m - \widehat{\mathcal{E}}_m}. \quad (2)$$

- Comme pour adaBoost discret, une condition supplémentaire est ajoutée

tée à l'algorithme. Il est arrêté ou réinitialisé à des poids uniformes si l'erreur se dégrade trop : si $\widehat{\mathcal{E}}_m < 0.5L_m$.

L'algorithme génère M prédicteurs construits sur des échantillons bootstrap \mathbf{z}_m^* dont le tirage dépend de probabilités \mathbf{p} mises à jour à chaque itération. Cette mise à jour est fonction d'un paramètre β_m qui est un indicateur de la performance, sur l'échantillon \mathbf{z} , du m -ième prédicteur estimé sur l'échantillon \mathbf{z}_m^* . La mise à jour des probabilités dépend donc à la fois de cet indicateur global β_m et de la qualité relative $l_m(i)/L_m$ de l'estimation du i -ème individu. L'estimation finale est enfin obtenue à la suite d'une moyenne ou médiane des prévisions pondérées par la qualité respective de chacune de ces prévisions. Gey et Poggi (2002)[10] conseille la médiane afin de s'affranchir de l'influence de prédicteurs très atypiques.

3.5 Modèle additif pas à pas

Le bon comportement du *boosting* par rapport à d'autres techniques de discrimination est difficile à expliquer ou justifier par des arguments théoriques. À la suite d'une proposition de Breiman en 1999 (rapport technique) de considérer le *boosting* comme un algorithme global d'optimisation, Hastie et col. (2001)[11] présentent le *boosting* dans le cas binaire sous la forme d'une approximation de la fonction f par un modèle additif construit pas à pas :

$$\widehat{f}(\mathbf{x}) = \sum_{m=1}^M c_m \delta(\mathbf{x}; \gamma_m)$$

est cette combinaison où c_m est un paramètre, δ le classifieur (faible) de base fonction de \mathbf{x} et dépendant d'un paramètre γ_m . Si l est une fonction perte, il s'agit, à chaque étape, de résoudre :

$$(c_m, \gamma_m) = \arg \min_{(c, \gamma)} \sum_{i=1}^n l(y_i, \widehat{f}_{m-1}(\mathbf{x}_i) + c\delta(\mathbf{x}_i; \gamma))$$

$\widehat{f}_m(\mathbf{x}) = \widehat{f}_{m-1}(\mathbf{x}) + c_m \delta(\mathbf{x}; \gamma_m)$ est alors une amélioration de l'ajustement précédent.

Dans le cas d'*adaBoost* pour l'ajustement d'une fonction binaire, la fonction perte utilisée est $l(y, f(\mathbf{x})) = \exp[-yf(\mathbf{x})]$. il s'agit donc de ré-

soudre :

$$\begin{aligned} (c_m, \gamma_m) &= \arg \min_{(c, \gamma)} \sum_{i=1}^n \exp \left[-y_i (\widehat{f}_{m-1}(\mathbf{x}_i) + c\delta(\mathbf{x}_i; \gamma)) \right]; \\ &= \arg \min_{(c, \gamma)} \sum_{i=1}^n w_i^m \exp [-cy_i \delta(\mathbf{x}_i; \gamma)] \\ \text{avec } w_i^m &= \exp [-y_i \widehat{f}_{m-1}(\mathbf{x}_i)]; \end{aligned}$$

w_i^m ne dépendant ni de c ni de γ , il joue le rôle d'un poids fonction de la qualité de l'ajustement précédent. Quelques développements complémentaires montrent que la solution du problème de minimisation est obtenue en deux étapes : recherche du classifieur optimal puis optimisation du paramètre c_m .

$$\begin{aligned} \gamma_m &= \arg \min_{\gamma} \sum_{i=1}^n \mathbf{1}\{y_i \neq \delta(\mathbf{x}_i; \gamma)\}, \\ c_m &= \frac{1}{2} \log \frac{1 - \widehat{\mathcal{E}}_p}{\mathcal{E}_p} \end{aligned}$$

avec $\widehat{\mathcal{E}}_p$ erreur apparente de prévision tandis que les w_i sont mis à jour avec :

$$w_i^{(m)} = w_i^{(m-1)} \exp [-c_m].$$

On montre ainsi qu'*adaBoost* approche f pas à pas par un modèle additif en utilisant une fonction perte exponentielle tandis que d'autres types de *boosting* sont définis sur la base d'une autre fonction perte :

$$\begin{aligned} \text{adaBoost } l(y, f(\mathbf{x})) &= \exp[-yf(\mathbf{x})], \\ \text{LogitBoost } l(y, f(\mathbf{x})) &= \log_2(1 + \exp[-2yf(\mathbf{x})]), \\ L^2\text{Boost } l(y, f(\mathbf{x})) &= (y - f(\mathbf{x}))^2/2. \end{aligned}$$

D'autres fonctions pertes sont envisageables pour, en particulier, un algorithme plus robuste face à un échantillon d'apprentissage présentant des erreurs de classement dans le cas de la discrimination ou encore des valeurs atypiques (*outliers*) dans le cas de la régression. Hastie et col. (2001)[11] comparent les intérêts respectifs de plusieurs fonctions pertes. Celles jugées robustes (entropie en discrimination, valeur absolue en régression) conduisent à des algorithmes plus compliqués à mettre en œuvre.

3.6 Boosting et gradient adaptatif

Préambule

Dans le même esprit d'approximation adaptative, Friedman (2002)[9] a proposé sous l'acronyme MART (*multiple additive regression trees*) puis sous celui de GBM (*gradient boosting models*) une famille d'algorithmes basés sur une fonction perte supposée convexe et différentiable notée l . Le principe de base est le même que pour adaBoost, construire une séquence de modèles de sorte que chaque étape, chaque modèle ajouté à la combinaison, apparaisse comme un pas vers une meilleure solution. La principale innovation est que ce pas est franchi dans la direction du *gradient de la fonction perte*, afin d'améliorer les propriétés de convergence. Une deuxième idée consiste à approcher le gradient par un *arbre de régression* afin d'éviter un sur-apprentissage.

Le modèle adaptatif pas-à-pas précédent :

$$\hat{f}_m(\mathbf{x}) = \hat{f}_{m-1}(\mathbf{x}) + c_m \delta(\mathbf{x}; \gamma_m)$$

est transformé en une descente de gradient :

$$\hat{f}_m = \hat{f}_{m-1} - \gamma_m \sum_{i=1}^n \nabla_{f_{m-1}} l(y_i, f_{m-1}(x_i)).$$

Plutôt que de chercher un meilleur classifieur comme avec adaBoost, le problème se simplifie en la recherche d'un meilleur pas de descente γ :

$$\min_{\gamma} \sum_{i=1}^n \left[l\left(y_i, f_{m-1}(x_i) - \gamma \frac{\partial l(y_i, f_{m-1}(x_i))}{\partial f_{m-1}(x_i)}\right) \right].$$

Algorithme

L'algorithme ci-dessous décrit le cas de la régression, il peut être adapté à celui de la classification.

L'algorithme est initialisé par un terme constant c'est-à-dire encore un arbre à une feuille. Les expressions du gradient reviennent simplement à calculer les termes r_{mj} , pour chaque nœud du modèle et chaque observation de ce nœud, à l'étape précédente. Le pas de descente γ est optimisé pour obtenir chaque mise à jour du modèle. Un algorithme de discrimination est similaire calculant autant de probabilités que de classes à prévoir.

Algorithm 5 Gradient Tree Boosting pour la régression

Soit \mathbf{x}_0 à prévoir

$$\text{Initialiser } \hat{f}_0 = \arg \min_{\gamma} \sum_{i=1}^n l(y_i, \gamma)$$

for $m = 1$ à M **do**

$$\text{Calculer } r_{mi} = - \left[\frac{\partial l(y_i, f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)} \right]_{f=f_{m-1}} ; i = 1, \dots, m$$

Ajuster un arbre de régression δ_m aux couples $(\mathbf{x}_i, r_{mi})_{i=1, \dots, n}$

Calculer γ_m en résolvant : $\min_{\gamma} \sum_{i=1}^n l(y_i, f_{m-1}(\mathbf{x}_i) + \gamma \delta_m(\mathbf{x}_i))$.

Mise à jour : $\hat{f}_m(\mathbf{x}) = \hat{f}_{m-1}(\mathbf{x}) + \gamma_m \delta_m(\mathbf{x})$

end for

Résultat : $\hat{f}_M(\mathbf{x}_0)$.

Sur-ajustement et rétrécissement

Friedman (2002)[9] a également proposé une version de *stochastic gradient boosting* incluant un sous-échantillonnage aléatoire à chaque étape afin de construire comme en *bagging* une séquence de prédicteurs plus indépendants. Le taux de sous-échantillonnage est un autre paramètre à optimiser, il est présent dans la version de GBM implémentée dans *Scikit-learn*.

Une autre proposition consiste à ajouter un coefficient η de rétrécissement (*shrinkage*). Compris entre 0 et 1, celui-ci pénalise l'ajout d'un nouveau modèle dans l'agrégation et ralentit la convergence.

$$\hat{f}_m(\mathbf{x}) = \hat{f}_{m-1}(\mathbf{x}) + \eta \gamma_m \delta_m(\mathbf{x}).$$

Si sa valeur est petite ($< 0,1$) cela conduit à accroître le nombre d'arbres mais entraîne généralement une amélioration de la qualité de prévision. Le *boosting* est un algorithme qui peut effectivement converger exactement, donc éventuellement vers une situation de sur-apprentissage. En pratique, cette convergence peut être rendue suffisamment lente pour être mieux contrôlée.

Dans cette dernière version de l'algorithme, il est conseillé de *contrôler* le nombre d'itérations par un échantillon de validation ou par validation croisée.

En résumé, ce sont trois paramètres qu'il est d'usage de contrôler ou optimiser lors de la mise en œuvre de l'algorithme de *gradient boosting* implanté en R (*gbm*) ou en python (*scikit-learn*). La nomenclature est celle de Python (*scikit-learn*) mais les mêmes paramètres se retrouvent en R.

- la profondeur maximale des arbres : `max_depth`,
- le coefficient de rétrécissement : `learning_rate` ou `shrinkage`,
- le nombre d'arbres ou d'itérations `n_estimators`.

Auxquels s'ajoute le taux de sous-échantillonnage si le choix est fait du *stochastic gradient boosting*.

Outre le choix de la fonction perte les implémentations en R ou dans `scikit-learn` proposent plusieurs autres paramètres :

- `max_features` : nombre de variables tirés à la recherche de chaque division comme dans *random forest*,
- `min_samples_split` : nombre minimal d'observations nécessaires pour diviser un noeud,
- `min_samples_leaf` ou `min_weight_fraction_leaf` : nombre minimal d'observations dans chaque feuille pour conserver une division,
- `max_leaf_node` : contrôle comme `max_depth` la complexité d'un arbre.
- `subsample` : part d'échantillon tiré aléatoirement à chaque étape.
- ... consulter la [documentation en ligne](#).

Ces paramètres jouant des rôles redondants sur le contrôle du sur-apprentissage il n'est pas nécessaire de tous les optimiser. [Certains sites](#) proposent des stratégies testées sur les concours Kaggle pour conduire l'optimisation.

Comme pour les forêts aléatoire, des critères d'*importance des variables* sont ajoutés au *boosting* afin d'apporter quelques pistes de compréhension du modèle.

3.7 Extrem Gradient Boosting

Prélude

Plus récemment, Chen et Guestrin (2016)[6] ont proposé un dernier avatar du *boosting* avec l'*extrem gradient boosting*. La complexification est très sensible notamment avec le nombre de paramètres qu'il est nécessaire de prendre en compte dans l'optimisation de l'algorithme. Les coûts de calcul deviendraient assez rédhibitoires, aussi la librairie associée (XGBoost), disponible en R, Python, Julia, Scala et dans des environnements distribués (Spark) offre une parallélisation efficace des calculs avec notamment la possibilité d'accéder à la carte graphique (GPU) de l'ordinateur et propose une version approchée

lorsque les données massives sont distribuées.

Mais, ce qui encourage réellement à s'intéresser à cette version du *boosting*, c'est son utilisation assez systématique dans les solutions gagnantes des concours Kaggle de prévision. Très schématiquement, si les données du concours sont des images, ce sont des algorithmes d'apprentissage profond qui l'emportent, sinon ce sont des combinaisons sophistiqués (usines à gaz) d'algorithmes incluant généralement XGBoost (cf. figure 1).

Il n'est pas dit que ces solutions gagnantes de type "usines à gaz" soient celles à rendre opérationnelles ou à industrialiser mais il semble nécessaire d'intégrer XGBoost dans la boîte du *data scientist* des outils à comparer. Même sans disposer des moyens de calcul adaptés à une optimisation de tous les paramètres, la mise en œuvre n'utilisant que les valeurs par défaut et une optimisation sommaire avec `caret`, qui offre des possibilités de parallélisation, même sous Windows, peut améliorer les solutions.

L'algorithme XGBoost inclut plusieurs fonctionnalités chacune associée avec un ou des paramètres supplémentaires à optimiser, en plus de ceux déjà décrit avec une nomenclature différente (`learning_rate`, `subsample`, `max_features...`) pour le *gradient boosting*. XGBoost est schématiquement décrit dans le cas de la régression. Se reporter à l'[article original](#) de Chen et Guestrin (2016)[6] pour des précisions, notamment sur les astuces d'implémentation et sur la dernière [documentation en ligne](#) pour la liste complète des paramètres. Un site propose des stratégies d'optimisation étudiées empiriquement sur les concours Kaggle.

Terme de régularisation

Une nouvelle fonction objectif \mathcal{L} est considérée en complétant la fonction perte convexe différentiable l par un terme de régularisation :

$$\mathcal{L}(f) = \sum_{i=1}^n l(\hat{y}_i, y_i) + \sum_{m=1}^M \Omega(\delta_m)$$

avec

$$\Omega(\delta) = \alpha|\delta| + \frac{1}{2}\beta\|\mathbf{w}\|^2,$$

où $|\delta|$ est le nombre de feuilles de l'arbre de régression δ et \mathbf{w} le vecteur des valeurs attribuées à chacune de ses feuilles.

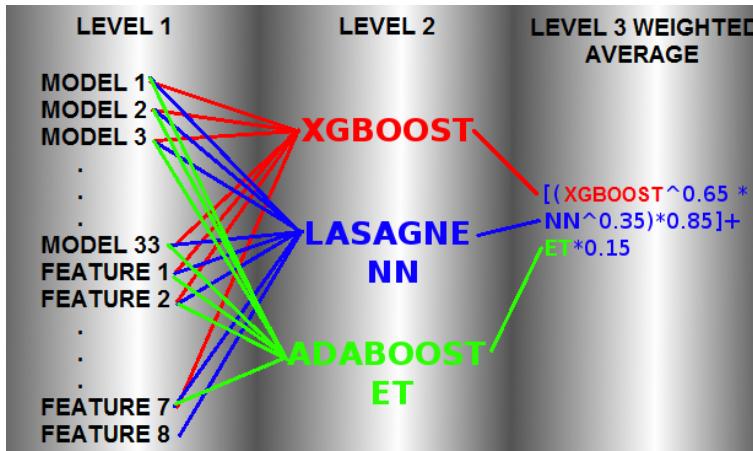


FIGURE 1 – Solution gagnante d'un concours kaggle : *Identify people who have a high degree of Psychopathy based on Twitter usage. Combinaison pondérée des combinaisons (boosting, réseaux de neurones) de trente trois modélisations (random forest, boosting, k plus proches voisins...) et 8 nouvelles variables (features) ad hoc.*

Cette pénalisation ou régularisation limite l'ajustement de l'arbre ajouté à chaque étape et contribue à éviter un sur-ajustement. Notamment lorsque des observations affectées d'erreurs importantes sont présentes, augmenter le nombre d'itérations peut provoquer une dégradation de la performance globale plutôt qu'une amélioration. Le terme Ω s'interprète comme une combinaison de régularisation *ridge* de coefficient β et de pénalisation Lasso de coefficient α .

Approximation du gradient

Une astuce consiste à approcher le gradient par un développement de Taylor au second ordre. Il suffit alors de sommer, pour chaque feuille, les valeurs des dérivées première et seconde de la fonction perte pour évaluer la fonction objectif pénalisées et maximiser sa décroissance lors de la recherche des divisions. La simplification introduite permet une parallélisation efficace de la construction des arbres.

Recherche des divisions

Lorsque les données sont volumineuses, éventuellement distribuées, l'algorithme original *glouton* de recherche d'une meilleure division nécessite trop de temps de calcul ou peut saturer l'espace mémoire. Une version approchée consiste à découper les variables quantitatives en classes en utilisant les quantiles de la distribution comme bornes. C'est un procédé similaire à celui utilisé dans l'implémentation des forêts aléatoires de la librairie `MLlib` de `Spark`. Dans le cas du *gradient boosting* c'est plus compliqué, Chen et Guestrin (2016)[6] détaillent un algorithme pour des quantiles pondérés.

Données manquantes

La gestion de données manquantes ou de zéros instrumentaux anormaux est prise en compte de façon originale dans l'implémentation de `XGBoost`. Celui-ci propose à chaque division une direction par défaut si une donnée est manquante. Pour palier cette absence, le gradient est calculé sur les seules valeurs présentes.

Enfin d'autres spécificités de l'implémentation améliorent les performances de l'algorithme : stockage mémoire en block compressé pour paralléliser les opérations très nombreuses de tris, tampons mémoire pour réduire les accès

disque, blocs de données compressés distribués (*sharding*) sur plusieurs disques.

En plus des paramètres du *gradient boosting* dont certains change de nom par rapport à l'implémentation de Scikit-learn, d'autres paramètres sont à optimiser avec [quelques précisions](#) :

- alpha : coefficient de pénalisation Lasso (l_1),
- lambda : coefficient de régularisation de type *ridge* (l_2),
- tree_method : sélection, automatique par défaut, de l'algorithme exacte glouton ou celui approché par découpage des variables quantitatives guidé par le paramètre
- sketch_eps : qui contrôle le nombre de classes.
- scale_pos_weight : utile pour des classes déséquilibrées.
- Autres paramètres techniques liés à l'optimisation du temps de calcul.

Interprétation

L'interprétabilité des arbres de décision sont une des raisons de leur succès. Leur lecture ne nécessite pas de compétences particulières en statistique. Cette propriété est évidemment perdue par l'agrégation d'arbres ou de tout autre modèle. Néanmoins, surtout si le nombre de variables est très grand, il est important d'avoir une indication de l'importance relative des variables entrant dans la modélisation. Des critères d'importance des variables sont disponibles comme dans le cas des forêts aléatoires.

4 Super apprenti

D'autres stratégies ont été proposées dans l'optique d'une prévision "brute", sans sélection de variables ou objectif d'interprétation du modèle. Elles procèdent en deux principales étapes :

- la première consiste à estimer un ensemble de modèles variés appartenant à celles des méthodes précédentes, de la régression au *boosting* en passant par les réseaux de neurones.
- la deuxième construit une combinaison linéaire convexe (*super learner* Van der Laan et al. (2007)[14]) ou une régression locale, à partir des modèles précédents (COBRA de Biau et al. (2013)[1]) ou encore une architecture sophistiquée et très empirique utilisée dans les concours de prévision (cf. figure 1).

La dernière solution de "type usine à gaz" ne conduit généralement pas à des

approches industrialisables et présentent des risques de supr-apprentissage, même sur l'échantillon test masqué lorsque des centaines voire milliers de concurrents s'affrontent avec des solutions ne différant qu'à la 2ème ou 3ème décimale du critère. Une part de hasard et donc d'artefact ne peut être exclue pour apparaître dans les toutes premières places.

Les autres solutions de combinaisons de modèles présentent des garanties plus théoriques.

4.1 Super learner

Le principe de l'approche proposée par van der Laan et al. (2007) [14] est simple, il s'agit de calculer une combinaison convexe ou moyenne pondérée de plusieurs prévisions obtenues par plusieurs modèles. Les paramètres de la combinaison sont optimisés en minimisant un critère de validation croisée. La méthode est implémentée dans la librairie SuperLearner de R où toutes les combinaisons de méthodes ne sont pas possibles, seule une liste prédéfinie est implémentée à cette date (juin 2014) : `glm`, `random forest`, `gbm`, `mars`, `svm`. Son emploi est illustré dans le [scénario](#) d'analyse de données (QSAR) issues de criblage virtuel de molécules.

4.2 COBRA

Biau et al. (2013)[1] proposent de combiner une collection de m fonctions de régression $\hat{f}_k(k = 1, m)$ en tenant compte de la proximité entre les données d'apprentissage avec l'observation à prévoir. Plus précisément, la prévision en $\hat{y}x$ est obtenue à partir de m prévisions comme la *moyenne non pondérée des observations* y_i dont les prévisions par $\alpha * m$ machines (α entre 0 et 1) sont dans les boules de rayon ε centrées en chaque $\hat{f}_k(x_i)$. Ils montrent que, asymptotiquement, cette combinaison d'estimateurs fait au moins aussi bien, au sens du risque quadratique ou erreur L^2 de prévision, que la meilleure des fonctions de régression de la collection.

Principe

La principale originalité de COBRA par rapport aux techniques d'agrégation de modèles précédentes, est que cette méthode n'opère pas une moyenne de prévisions mais une moyenne d'observations : celles les plus proches des prévisions d'une famille de modèles ou de m machines. COBRA opère donc

une forme de régression non-paramétrique avec une fonction *noyau* ou une notion de voisinage très complexe car elle dépend des prévisions d'une famille de machines. Biau et al. (2013) explique le principe par un exemple jouet repris dans la figure 2.

Connaissant un ensemble d'apprentissage (x_i, y_i) , les quantités f_1 et f_2 sont estimées. La prévision en x_0 est construite de la façon suivante. Une sélection des observations est opérée, ce sont celles qui vérifient pour un seuil ε choisi :

$$m = 1, 2 : |f_m(x_i) - f_m(x_0)| \leq \varepsilon.$$

La simple moyenne des observations sélectionnées par unanimité fournit la prévision. Ce principe d'unanimité peut être relâché en acceptant qu'une proportion réduite α des M machines satisfassent la contrainte sur les observations.

Sous des hypothèses que les machines sont bornées, Biau et al. (2013) montrent que le risque “collectif” est borné par le plus petit risque de toutes les machines plus un terme d'ordre $\ell^{-2 \frac{2}{M+2}}$.

Illustration

Comme pour le *Super learner* cette approche est testée dans le [scénario](#) d'analyse de données (QSAR) issues de criblage virtuel de molécules. La librairie R COBRA implémente cette méthode en proposant une procédure d'optimisation des paramètres α et ε . Tout type de modèle de régression peut être inclus dans la procédure COBRA, il est conseillé d'en utiliser des très “variés” linéaires et surtout non linéaires afin d'optimiser les chances du succès.

Bien entendu, même les faibles capacités d'interprétation de certains méthodes comme *random forest* avec les critères d'importance de variables ne sont plus conservées.

5 Exemples

5.1 Cancer du sein

La prévision de l'échantillon test par ces algorithmes à des taux d'erreurs estimés de 4,4 et 2,2% pour cet exemple et avec les échantillons (apprentissage et test) tirés.

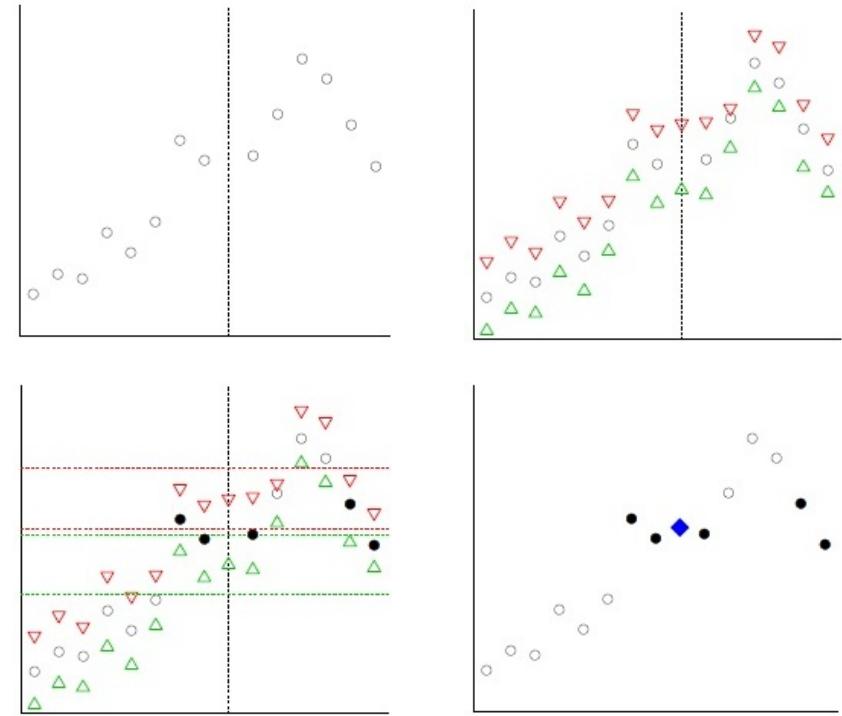


FIGURE 2 – COBRA : De gauche à droite et de bas en haut : L'ensemble d'apprentissage (Y fonction de X) ; il faut prévoir la valeur sur la ligne pointillée. Les estimations de chaque observation par deux machines (rouge et verte). Une tolérance ($\pm \varepsilon$ à optimiser) détermine les observations retenues pour chaque machine autour de la valeur à prévoir. La prévision est la moyenne (en bleu) des observations (en noir) sélectionnées pour toutes les machines à l'étape précédente.

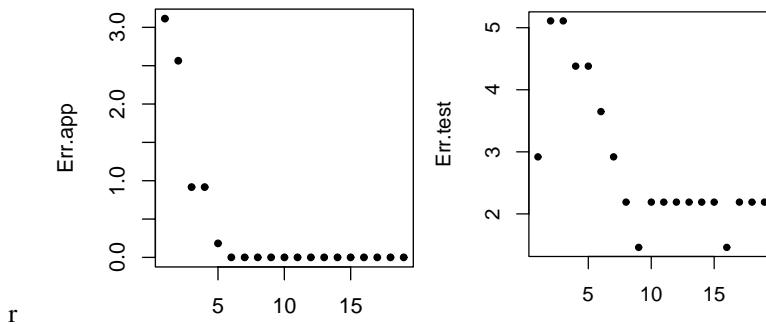


FIGURE 3 – Cancer : Évolution des taux d'erreur (%) sur les échantillons d'apprentissage et de test en fonction du nombre d'arbres dans le modèle avec adaBoost.

Il est remarquable de noter l'évolution des erreurs d'ajustement et de test (figure 3) en fonction du nombre d'arbres estimés par adaBoost. L'erreur d'apprentissage arrive rapidement à 0 tandis que celle de test continue à décroître avant d'atteindre un seuil. Cet algorithme est donc relativement robuste au sur-apprentissage avant, éventuellement, de se dégrader pour des raisons, sans doute, de précision numérique. Ce comportement a été relevé dans beaucoup d'exemples dans la littérature.

5.2 Concentration d'ozone

Malgré une bonne prévision quantitative, la prévision du dépassement de seuil reste difficile pour l'algorithme des forêts aléatoires. Par une régression ou une discrimination, le taux d'erreur obtenu est le même (12,5%) sur le même échantillon test et d'autres expérimentations sont nécessaires pour dépasser, ou non, les différentes méthodes. Il semble que, à travers plusieurs exemples, l'amélioration apportée à la prévision par des algorithmes d'agrégation de modèles soit nettement plus probante dans des situations difficiles c'est-à-dire avec beaucoup de variables explicatives et des problèmes de multicollinearité.

Comme les réseaux de neurones, les algorithmes d'agrégation de modèles sont des boîtes noires. Néanmoins dans le cas des forêts, les critères d'importance donnent des indications sur le rôle de celles-ci. Les voici ordonnées par

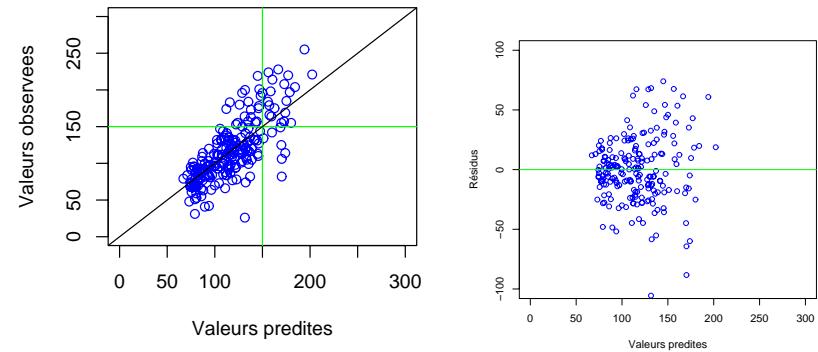


FIGURE 4 – Ozone : Valeurs observées et résidus de l'échantillon test en fonction des valeurs prédites par une forêt aléatoire

ordre croissant du critère basé sur celui de Gini pour la construction des arbres.

jour	station	lno	lno2	vmodule	s_rmh2o	o3_pr	TEMPE
2.54	13.58	21.78	23.33	24.77	31.19	43.87	67.66

Les variables prépondérantes sont celles apparues dans la construction d'un seul arbre.

5.3 Données bancaires

Les arbres, qui acceptent à la fois des variables explicatives qualitatives et quantitatives en optimisant le découpage des variables quantitatives, se prêtent bien au traitement des données bancaires. On a vu qu'un seul arbre donnait des résultats semble-t-il très corrects. Naturellement les forêts constitués d'arbres se trouvent également performantes sur ces données en gagnant en stabilité et sans trop se poser de problème concernant l'optimisation de paramètres. Les TPs décrivent également les résultats proposés par les algorithmes de bagging et de boosting sur les arbres en faisant varier certains paramètres comme le shrinkage dans le cas du boosting.

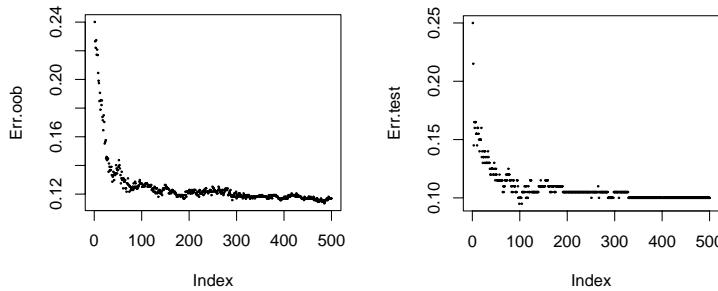


FIGURE 5 – Banque : Évolution du taux de mal classés estimés "out-of-bag" et sur l'échantillon test en fonction du nombre d'arbres intervenant dans la combinaison de modèles.

Les graphiques de la figure 5 montrent bien l'insensibilité des forêts au sur-apprentissage. Les taux d'erreurs estimés, tant par bootstrap (out-of-bag), que sur un échantillon test, se stabilisent au bout de quelques centaines d'itérations. Il est même possible d'introduire dans le modèle toutes les variables quantitatives et qualitatives, avec certaines dupliquées, en laissant l'algorithme faire son choix. Cet algorithme conduit à un taux d'erreur de 10,5% sur l'échantillon test avec la matrice de confusion :

	Cnon	Coui
Cnon	126	11
Coui	10	53

tandis que les coefficients d'importance :

QSMOY	FACANL	RELAT	DMVTPL	QCREDL	MOYRVL
20.97	26.77	29.98	36.81	40.31	50.01

mettent en évidence les variables les plus discriminantes. De son côté, le boosting (sans shrinkage) fournit des résultats comparables avec un taux d'erreur

de 11%, le gradient boosting atteint 9,5% et XGBoost 8,8% sans gros efforts d'optimisation.

Références

- [1] G. Biau, A. Fischer, B. Guedj et J. D. Malley, *COBRA : A Nonlinear Aggregation Strategy*, Journal of Multivariate Analysis **146** (2016), 18–28.
- [2] L. Breiman, *Bagging predictors*, Machine Learning **26** (1996), n° 2, 123–140.
- [3] _____, *Arcing classifiers*, Annals of Statistics **26** (1998), 801–849.
- [4] _____, *Random forests*, Machine Learning **45** (2001), 5–32.
- [5] Rich. Caruana, N. Karampatziakis et A. Yessenalina, *An Empirical Evaluation of Supervised Learning in High Dimensions*, Proceedings of the 25th International Conference on Machine Learning (New York, NY, USA), ICML '08, ACM, 2008, p. 96–103, ISBN 978-1-60558-205-4.
- [6] Tianqi Chen et Carlos Guestrin, *XGBoost : A Scalable Tree Boosting System*, Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, 2016, p. 785–794.
- [7] Manuel Fernandez-Delgado, Eva Cernadas, Senen Barro et Dinani Amorim, *Do we Need Hundreds of Classifiers to Solve Real World Classification Problems ?*, IEEE Trans. Pattern Anal. Mach. Intell. **15** (2014), 3133–31.
- [8] Y. Freund et R.E. Schapire, *Experiments with a new boosting algorithm*, Machine Learning : proceedings of the Thirteenth International Conference, Morgan Kaufman, 1996, San Francisco, p. 148–156.
- [9] J. H. Friedman, *Stochastic gradient boosting*, Computational Statistics and Data Analysis **38** (2002), .
- [10] S. Gey et J. M. Poggi, *Boosting and instability for regression trees*, Rap. tech. 36, Université de Paris Sud, Mathématiques, 2002.
- [11] T. Hastie, R. Tibshirani et J Friedman, *The elements of statistical learning : data mining, inference, and prediction*, Springer, 2009, Second edition.
- [12] R. Schapire, *The boosting approach to machine learning. An overview*, MSRI workshop on non linear estimation and classification, 2002, p. .

- [13] E. Scornet, G. Biau et J. P. Vert, *Consistency of random forests*, The Annals of Statistics (2015), à paraître.
- [14] M. J. van der Laan, E. C. Polley et A. E. Hubbard, *Super learner*, Statistical Applications in Genetics and Molecular Biology **6 :1** (2007).

Machines à vecteurs supports

Résumé

Recherche d'un hyperplan, dit de marge optimale (vaste), pour la séparation de deux classes dans un espace hilbertien défini par un noyau reproduisant associé au produit scalaire de cet espace. Estimation de l'hyperplan dans le cas linéaire et séparable ; les contraintes actives du problème d'optimisation déterminent les vecteurs supports. Extension au cas non linéaire par plongement dans un espace hilbertien à noyau reproduisant. Extension au cas non séparable par pénalisation.

[Retour au plan du cours](#)

1 Introduction

Les *Support Vector Machines* souvent traduit par l'appellation de Séparateur à Vaste Marge (SVM) sont une classe d'algorithmes d'apprentissage initialement définis pour la discrimination c'est-à-dire la prévision d'une variable qualitative initialement binaire. Ils ont été ensuite généralisés à la prévision d'une variable quantitative. Dans le cas de la discrimination d'une variable dichotomique, ils sont basés sur la recherche de l'*hyperplan de marge optimale* qui, lorsque c'est possible, classe ou sépare correctement les données tout en étant le plus éloigné possible de toutes les observations. Le principe est donc de trouver un classifieur, ou une fonction de discrimination, dont la capacité de généralisation (qualité de prévision) est la plus grande possible.

Cette approche découle directement des travaux de Vapnik en théorie de l'apprentissage à partir de 1995. Elle s'est focalisée sur les propriétés de généralisation (ou prévision) d'un modèle en contrôlant sa complexité. Voir à ce sujet la [vignette](#) sur l'estimation d'un risque et la section introduisant la dimension de Vapnik-Chernovenkis comme indicateur du pouvoir séparateur d'une famille de fonctions associé à un modèle et qui en contrôle la complexité. Le principe fondateur des SVM est justement d'intégrer à l'estimation le contrôle de la complexité c'est-à-dire le nombre de paramètres qui est associé dans ce cas au nombre de vecteurs supports. L'autre idée directrice de Vapnik dans

ce développement, est d'éviter de substituer à l'objectif initial : la discrimination, un ou des problèmes qui s'avèrent finalement plus complexes à résoudre comme par exemple l'estimation non-paramétrique de la densité d'une loi multidimensionnelle en analyse discriminante.

Le principe de base des SVM consiste de ramener le problème de la discrimination à celui, linéaire, de la recherche d'un hyperplan optimal. Deux idées ou astuces permettent d'atteindre cet objectif :

- La première consiste à définir l'hyperplan comme solution d'un problème d'optimisation sous contraintes dont la fonction objectif ne s'exprime qu'à l'aide de produits scalaires entre vecteurs et dans lequel le nombre de contraintes "actives" ou vecteurs supports contrôle la complexité du modèle.
- Le passage à la recherche de surfaces séparatrices non linéaires est obtenu par l'introduction d'une fonction noyau (*kernel*) dans le produit scalaire induisant implicitement une transformation non linéaire des données vers un espace intermédiaire (*feature space*) de plus grande dimension. D'où l'appellation couramment rencontrée de machine à noyau ou *kernel machine*. Sur le plan théorique, la fonction noyau définit un espace hilbertien, dit auto-reproduisant et isométrique par la transformation non linéaire de l'espace initial et dans lequel est résolu le problème linéaire.

Cet outil devient largement utilisé dans de nombreux types d'application et s'avère un concurrent sérieux des algorithmes les plus performants (agrégation de modèles). L'introduction de noyaux, spécifiquement adaptés à une problématique donnée, lui confère une grande flexibilité pour s'adapter à des situations très diverses (reconnaissance de formes, de séquences génomiques, de caractères, détection de spams, diagnostics...). À noter que, sur le plan algorithmique, ces algorithmes sont plus pénalisés par le nombre d'observations, c'est-à-dire le nombre de vecteurs supports potentiels, que par le nombre de variables. Néanmoins, des versions performantes des algorithmes permettent de prendre en compte des bases de données volumineuses dans des temps de calcul acceptables.

Le livre de référence sur ce sujet est celui de Schölkopf et Smola (2002)[2]. De nombreuses introduction et présentations des SVM sont accessibles sur des sites comme par exemple : www.kernel-machines.org. Guermeur et Paugam-Moisy (1999)[1] en proposent une en français.

2 Principes

2.1 Problème

Comme dans toute situation d'apprentissage, on considère une variable Y à prédire mais qui, pour simplifier cette introduction élémentaire, est supposée dichotomique à valeurs dans $\{-1, 1\}$. Soit $\mathbf{X} = X^1, \dots, X^p$ les variables explicatives ou prédictives et $\phi(\mathbf{x})$ un modèle pour Y , fonction de $\mathbf{x} = \{x^1, \dots, x^p\} \in \mathbb{R}^p$. Plus généralement on peut simplement considérer la variable \mathbf{X} à valeurs dans un ensemble \mathcal{F} .

On note

$$\mathbf{z} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$$

un échantillon statistique de taille n et de loi F inconnue. L'objectif est donc de construire une estimation $\hat{\phi}$ de ϕ , fonction de \mathcal{F} dans $\{-1, 1\}$, de sorte que la probabilité :

$$P(\phi(\mathbf{X}) \neq Y)$$

soit minimale.

Dans ce cas (Y dichotomique), le problème se pose comme la recherche d'une frontière de décision dans l'espace \mathcal{F} des valeurs de \mathbf{X} . De façon classique, un compromis doit être trouvé entre la *complexité* de cette frontière, qui peut s'exprimer aussi comme sa capacité à pulvériser un nuage de points par la VC dimension, donc la capacité d'*ajustement* du modèle, et les qualités de *généralisation* ou prévision de ce modèle. Ce principe est illustré par la figure 1.

2.2 Marge

La démarche consiste à rechercher, plutôt qu'une fonction $\hat{\phi}$ à valeurs dans $\{-1, 1\}$, une fonction réelle f dont le signe fournira la prévision :

$$\hat{\phi} = \text{signe}(f).$$

L'erreur s'exprime alors comme la quantité :

$$P(\phi(\mathbf{X}) \neq Y) = P(Yf(\mathbf{X}) \leq 0).$$

De plus, la valeur absolue de cette quantité $|Yf(\mathbf{X})|$ fournit une indication sur la confiance à accorder au résultat du classement.

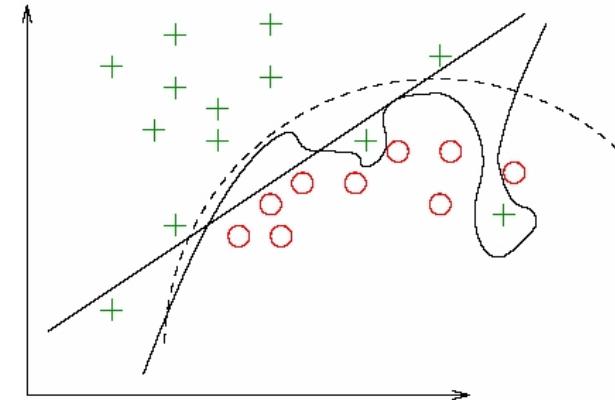


FIGURE 1 – Sous-ajustement linéaire et sur-ajustement local (proches voisins) d'un modèle quadratique.

On dit que $Yf(\mathbf{X})$ est la *marge* de f en (\mathbf{X}, Y) .

2.3 Espace intermédiaire

Une première étape consiste à transformer les valeurs de \mathbf{X} , c'est-à-dire les objets de \mathcal{F} par une fonction Φ à valeurs dans un espace \mathcal{H} intermédiaire (*feature space*) muni d'un *produit scalaire*. Cette transformation est fondamentale dans le principe des SVM, elle prend en compte l'éventuelle non linéarité du problème posé et le ramène à la résolution d'une séparation linéaire. Ce point est détaillé dans une section ultérieure. Traitons tout d'abord le cas linéaire c'est-à-dire le cas où Φ est la fonction identité.

3 Séparateur linéaire

3.1 Hyperplan séparateur

La résolution d'un problème de séparation linéaire est illustré par la figure 2. Dans le cas où la séparation est possible, parmi tous les hyperplans solutions pour la séparation des observations, on choisit celui qui se trouve le plus "loin"

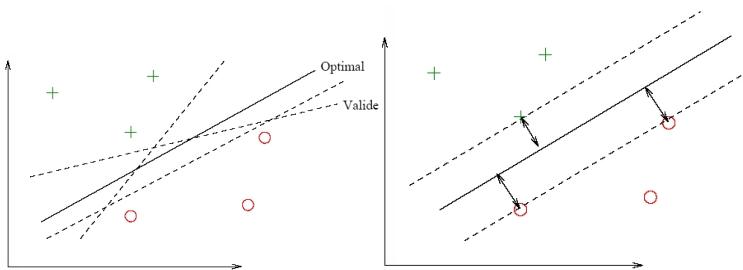


FIGURE 2 – Recherche d'un hyperplan de séparation optimal au sens de la marge maximale.

possible de tous les exemples, on dit encore, de *marge* maximale.

Dans le cas linéaire, un hyperplan est défini à l'aide du produit scalaire de \mathcal{H} par son équation :

$$\langle \mathbf{w}, \mathbf{x} \rangle + b = 0$$

où \mathbf{w} est un vecteur orthogonal au plan tandis que le signe de la fonction

$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b$$

indique de quel côté se trouve le point \mathbf{x} à prédire. Plus précisément, un point est bien classé si et seulement si :

$$y f(\mathbf{x}) > 0$$

mais, comme le couple (\mathbf{w}, b) qui caractérise le plan est défini à un coefficient multiplicatif près, on s'impose :

$$y f(\mathbf{x}) \geq 1.$$

Un plan (\mathbf{w}, b) est un séparateur si :

$$y_i f(\mathbf{x}_i) \geq 1 \quad \forall i \in \{1, \dots, n\}.$$

La distance d'un point \mathbf{x} au plan (\mathbf{w}, b) est donnée par :

$$d(\mathbf{x}) = \frac{|\langle \mathbf{w}, \mathbf{x} \rangle + b|}{\|\mathbf{w}\|} = \frac{|f(\mathbf{x})|}{\|\mathbf{w}\|}$$

et, dans ces conditions, la marge du plan a pour valeur $\frac{2}{\|\mathbf{w}\|^2}$. Chercher le plan séparateur de marge maximale revient à résoudre le problème ci-dessous d'optimisation sous contraintes (problème primal) :

$$\begin{cases} \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{avec } \forall i, y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1. \end{cases}$$

Le problème dual est obtenu en introduisant des multiplicateurs de Lagrange. La solution est fournie par un *point-selle* $(\mathbf{w}^*, b^*, \boldsymbol{\lambda}^*)$ du lagrangien :

$$L(\mathbf{w}, b, \boldsymbol{\lambda}) = 1/2 \|\mathbf{w}\|_2^2 - \sum_{i=1}^n \lambda_i [y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1].$$

Ce point-selle vérifie en particulier les conditions :

$$\lambda_i^* [y_i (\langle \mathbf{w}^*, \mathbf{x}_i \rangle + b^*) - 1] = 0 \quad \forall i \in \{1, \dots, n\}.$$

Les *vecteurs support* sont les vecteurs \mathbf{x}_i pour lesquels la contrainte est active, c'est-à-dire les plus proches du plan, et vérifiant donc :

$$y_i (\langle \mathbf{w}^*, \mathbf{x}_i \rangle + b^*) = 1.$$

Les conditions d'annulation des dérivées partielles du lagrangien permettent d'écrire les relations que vérifient le plan optimal, avec les λ_i^* non nuls seulement pour les points supports :

$$\mathbf{w}^* = \sum_{i=1}^n \lambda_i^* y_i \mathbf{x}_i \quad \text{et} \quad \sum_{i=1}^n \lambda_i^* y_i = 0.$$

Ces contraintes d'égalité permettent d'exprimer la formule duale du lagrangien :

$$W(\boldsymbol{\lambda}) = \sum_{i=1}^n \lambda_i - \frac{1}{2} \sum_{i,j=1}^n \lambda_i \lambda_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle.$$

Pour trouver le point-selle, il suffit alors de maximiser $W(\boldsymbol{\lambda})$ avec $\lambda_i \geq 0$ pour tout $i \in \{1, \dots, n\}$. La résolution de ce problème d'optimisation quadratique de

taille n , le nombre d'observations, fournit l'équation de l'hyperplan optimal :

$$\sum_{i=1}^n \lambda_i^* y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b^* = 0 \quad \text{avec} \quad b^0 = -\frac{1}{2} [\langle \mathbf{w}^*, \mathbf{s}_{v_{class+1}} \rangle + \langle \mathbf{w}^*, \mathbf{s}_{v_{class-1}} \rangle]$$

Pour une nouvelle observation \mathbf{x} non apprise présentée au modèle, il suffit de regarder le signe de l'expression :

$$f(\mathbf{x}) = \sum_{i=1}^n \lambda_i^* y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b^*$$

pour savoir dans quel demi-espace cette forme se trouve, et donc quelle classe il faut lui attribuer.

3.2 Cas non séparable

Lorsque les observations ne sont pas séparables par un plan, il est nécessaire d'"assouplir" les contraintes par l'introduction de termes d'erreur ξ_i qui en contrôlent le dépassement :

$$y_i \langle \mathbf{w}, \mathbf{x}_i \rangle + b \geq +1 - \xi_i \quad \forall i \in \{1, \dots, n\}.$$

Le modèle attribue ainsi une réponse fausse à un vecteur \mathbf{x}_i si le ξ_i correspondant est supérieur à 1. La somme de tous les ξ_i représente donc une borne du nombre d'erreurs.

Le problème de minimisation est réécrit en introduisant une pénalisation par le dépassement de la contrainte :

$$\begin{cases} \min \frac{1}{2} \|\mathbf{w}\|^2 + \delta \sum_{i=1}^n \xi_i \\ \forall i, y_i \langle \mathbf{w}, \mathbf{x}_i \rangle + b \geq +1 - \xi_i \end{cases}$$

Remarques

- Le paramètre δ contrôlant la pénalisation est à régler. Plus il est grand et plus cela revient à attribuer une forte importance à l'ajustement. Il est le paramètre qui ajuste le compromis entre bon ajustement et bonne généralisation.
- Le problème dans le cas non séparable se met sous la même forme duale que dans le cas séparable à une différence près : les coefficients λ_i sont tous bornés par la constante δ de contrôle de la pénalisation.

- De nombreux algorithmes sont proposés pour résoudre ces problèmes d'optimisation quadratique. Certains, proposant une décomposition de l'ensemble d'apprentissage, sont plus particulièrement adaptés à prendre en compte un nombre important de contraintes lorsque n , le nombre d'observation, est grand.
- On montre par ailleurs que la recherche des hyperplans optimaux répond bien au problème de la "bonne" généralisation. On montre aussi que, si l'hyperplan optimal peut être construit à partir d'un petit nombre de vecteurs supports, par rapport à la taille de la base d'apprentissage, alors la capacité en généralisation du modèle sera grande, indépendamment de la taille de l'espace.
- Plus précisément, on montre que, si les \mathbf{X} sont dans une boule de rayon R , l'ensemble des hyperplans de marge fixée δ a une VC-dimension bornée par

$$\frac{R^2}{\delta^2} \quad \text{avec } \|\mathbf{x}\| \leq R.$$

- L'erreur par validation croisée (*leave-one-out*) est bornée en moyenne par le nombre de vecteurs supports. Ces bornes d'erreur sont bien relativement prédictives mais néanmoins trop pessimistes pour être utiles en pratique.

4 Séparateur non linéaire

4.1 Noyau

Revenons à la présentation initiale du problème. Les observations faites dans l'ensemble \mathcal{F} (en général \mathbb{R}^p) sont considérées comme étant transformées par une application non linéaire Φ de \mathcal{F} dans \mathcal{H} muni d'un produit scalaire et de plus grande dimension.

Le point important à remarquer, c'est que la formulation du problème de minimisation ainsi que celle de sa solution :

$$f(\mathbf{x}) = \sum_{i=1}^n \lambda_i^* y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b^*$$

ne fait intervenir les éléments \mathbf{x} et \mathbf{x}' que par l'intermédiaire de *produits scalaires* : $\langle \mathbf{x}, \mathbf{x}' \rangle$. En conséquence, il n'est pas nécessaire d'expliquer la transfor-

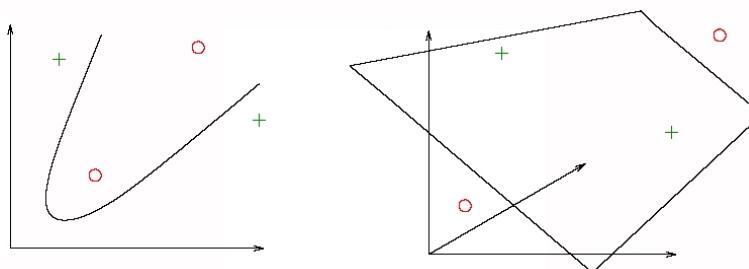


FIGURE 3 – Rôle de l'espace intermédiaire dans la séparation des données.

mation Φ , ce qui serait souvent impossible, à condition de savoir exprimer les produits scalaires dans \mathcal{H} à l'aide d'une fonction $k : \mathcal{F} \times \mathcal{F} \rightarrow \mathbb{R}$ symétrique appelée *noyau* de sorte que :

$$k(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle.$$

Bien choisi, le noyau permet de matérialiser une notion de “proximité” adaptée au problème de discrimination et à sa structure de données.

Exemple

Prenons le cas trivial où $\mathbf{x} = (x_1, x_2)$ dans \mathbb{R}^2 et $\Phi(\mathbf{x}) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$ est explicite. Dans ce cas, \mathcal{H} est de dimension 3 et le produit scalaire s'écrit :

$$\begin{aligned} \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle &= x_1^2 x_1'^2 + 2x_1 x_2 x_1' x_2' + x_2^2 x_2'^2 \\ &= (x_1 x_1' + x_2 x_2')^2 \\ &= \langle \mathbf{x}, \mathbf{x}' \rangle^2 \\ &= k(\mathbf{x}, \mathbf{x}'). \end{aligned}$$

Le calcul du produit scalaire dans \mathcal{H} ne nécessite pas l'évaluation explicite de Φ . D'autre part, le plongement dans $\mathcal{H} = \mathbb{R}^3$ peut rendre possible la séparation linéaire de certaines structures de données (cf. figure 3).

4.2 Condition de Mercer

Une fonction $k(., .)$ symétrique est un noyau si, pour tous les \mathbf{x}_i possibles, la matrice de terme général $k(\mathbf{x}_i, \mathbf{x}_j)$ est une matrice définie positive c'est-à-dire

quelle définit une matrice de produit scalaire.

Dans ce cas, on montre qu'il existe un espace \mathcal{H} et une fonction Φ tels que :

$$k(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle.$$

Malheureusement, cette condition théorique d'existence est difficile à vérifier et, de plus, elle ne donne aucune indication sur la construction de la fonction noyau ni sur la transformation Φ . La pratique consiste à combiner des noyaux simples pour en obtenir des plus complexes (multidimensionnels) associés à la situation rencontrée.

4.3 Exemples de noyaux

- Linéaire

$$k(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle$$

- Polynômial

$$k(\mathbf{x}, \mathbf{x}') = (c + \langle \mathbf{x}, \mathbf{x}' \rangle)^d$$

- Gaussien

$$k(\mathbf{x}, \mathbf{x}') = e^{-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}}$$

Beaucoup d'articles sont consacrés à la construction d'un noyau plus ou moins exotique et adapté à une problématique posée : reconnaissance de séquences, de caractères, l'analyse de textes... La grande flexibilité dans la définition des noyaux, permettant de définir une notion adaptée de similitude, confère beaucoup d'efficacité à cette approche à condition bien sûr de construire et tester le bon noyau. D'où apparaît encore l'importance de correctement évaluer des erreurs de prévision par exemple par validation croisée.

Attention, les SVM à noyaux RBF gaussiens, pour lesquels, soit on est dans le cas séparable, soit la pénalité attribuée aux erreurs est autorisée à prendre n'importe quelle valeur, ont une VC-dimension infinie.

4.4 SVM pour la régression

Les SVM peuvent également être mis en œuvre en situation de régression, c'est-à-dire pour l'approximation de fonctions quand Y est quantitative. Dans le cas non linéaire, le principe consiste à rechercher une estimation de la fonction par sa décomposition sur une base fonctionnelle. la forme générale des

fonctions calculées par les SVM se met sous la forme :

$$\phi(\mathbf{x}, \mathbf{w}) = \sum_{i=1}^{\infty} w_i v_i(\mathbf{x}).$$

Le problème se pose toujours comme la minimisation d'une fonction coût mais, plutôt que d'être basée sur un critère d'erreur quadratique (moindre carrés), celle-ci s'inspire des travaux de Huber sur la recherche de modèle robustes et utilise des écarts absolus.

On note $|\cdot|_\epsilon$ la fonction qui est paire, continue, identiquement nulle sur l'intervalle $[0, \epsilon]$ et qui croît linéairement sur $[\epsilon, +\infty]$. La fonction coût est alors définie par :

$$E(\mathbf{w}, \gamma) = \frac{1}{n} \sum_{i=1}^n |y_i - \phi(\mathbf{x}_i, \mathbf{w})|_\epsilon + \gamma \|\mathbf{w}\|^2$$

où γ est, comme en régression *ridge*, un paramètre de régularisation assurant le compromis entre généralisation et ajustement. De même que précédemment, on peut écrire les solutions du problèmes d'optimisation. Pour plus de détails, se reporter à Schölkopf et Smola (2002)[2]. Les points de la base d'apprentissage associés à un coefficient non nul sont là encore nommés vecteurs support.

Dans cette situation, les noyaux k utilisés sont ceux naturellement associés à la définition de bases de fonctions. Noyaux de splines ou encore noyau de Dériclet associé à un développement en série de Fourier sont des grands classiques. Ils expriment les produits scalaires des fonctions de la base.

5 Exemples

Comme pour les réseaux de neurones, l'optimisation des SVM qui, en plus du choix de noyau, peut comporter de 1 à 3 paramètres (pénalisation et éventuels paramètres du noyau) est délicate. La figure 4 montre 3 résultats de validation croisée pour le simple noyau linéaire dans le cas des données NIR.

5.1 Cancer du sein

La prévision de l'échantillon test par un Séparateur à Vaste marge conduit à la matrice de confusion :

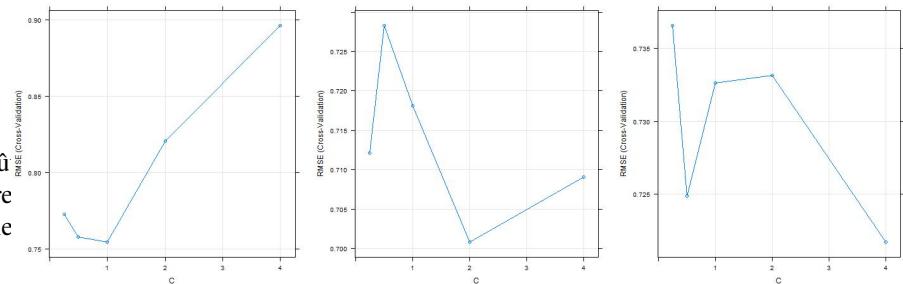


FIGURE 4 – Cookies : trois exécutions de la validation croisée estimant l'erreur en fonction de la pénalisation d'un noyau linéaire.

ign malignant		
benign	83	1
malignant	3	50

et donc une erreur estimée de 3%.

5.2 Concentration d'ozone

Un modèle élémentaire avec noyau par défaut (gaussien) et une pénalisation de 2 conduit à une erreur de prévision estimée à 12,0% sur l'échantillon test. La meilleure prévision de dépassement de seuil sur l'échantillon test initial est fournie par des SVM d' ε -régression. Le taux d'erreur est de 9,6% avec la matrice de confusion suivante :

	0	1
FALSE	161	13
TRUE	7	27

Ce résultat serait à confirmer avec des estimations systématiques de l'erreur. Les graphiques de la figure 5 montre le bon comportement de ce prédicteur. Il souligne notamment l'effet "tunnel" de l'estimation qui accepte des erreurs autour de la diagonale pour se concentrer sur les observations plus éloignées donc plus difficiles à ajuster.

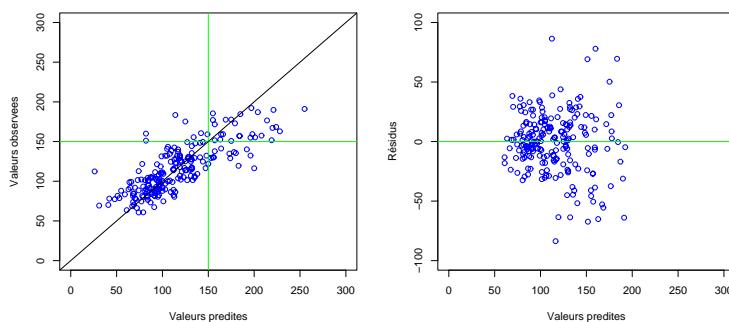


FIGURE 5 – Ozone : Valeurs observées et résidus en fonction des valeurs pré-dites pour l'échantillon test.

5.3 Données bancaires

Les données bancaires posent un problème car elles mixent variables quantitatives et qualitatives. Celles-ci nécessiteraient la construction de noyaux très spécifiques. Leur traitement par SVM n'est pas détaillé ici.

Références

- [1] Y. Guermeur et H. Paugam-Moisy, *Théorie de l'apprentissage de Vapnik et SVM, Support Vector Machines*, Apprentissage automatique (M. Sebban et G. Venturini, réds.), Hermès, 1999, p. 109–138.
- [2] B Schölkopf et A Smola, *Learning with Kernels Support Vector Machines, Regularization, Optimization and Beyond*, MIT Press, 2002.

Imputation de données manquantes

Résumé

Cette vignette présente les différents types de problèmes soulevés par la question très fréquente en pratique d'occurrences de données manquantes, que ce soit pour des données matricielles ou longitudinales. Les méthodes d'imputation de données manquantes sont décrites ; les plus rudimentaires : LOCF, imputation par la médiane, la moyenne..., de même que celles par modélisation ou apprentissage statistique : régression et régression locale, *k-nn*, régression PLS, SVD, Random Forest ou encore par imputation multiple. Ces méthodes sont illustrées et leur efficacité comparée sur trois jeux de données.

[Retour au plan du cours](#)

1 Introduction

Malgré la quantité croissante de données disponibles et l'émergence du Big Data, les problématiques de données manquantes restent très répandues dans les problèmes statistiques et nécessitent une approche particulière. Ignorer les données manquantes peut entraîner, outre une perte de précision, de forts biais dans les modèles d'analyse.

Les données sont constituées de p variables quantitatives ou qualitatives (Y_1, \dots, Y_p) observées sur un échantillon de n individus. Il existe des données manquantes représentées par la matrice M dite d'*indication des valeurs manquantes* [13] dont la forme dépend du type de données manquantes.

Nous commencerons par donner une définition des données manquantes en définissant plusieurs types de données manquantes et en étudiant les répartitions possibles. Nous verrons ensuite quelques d'approches qui nécessitent la suppression de données puis nous proposerons un certain nombre de méthodes de complémentation, sans souci d'exhaustivité.

2 Typologie des données manquantes

2.1 Types de données manquantes

Afin d'aborder correctement l'imputation des données manquantes il faut en distinguer les causes, les données manquantes n'arrivant pas toujours par un pur hasard. Une typologie a été développée par Little & Rubin en 1987 [13], les répartissant en 3 catégories :

- **Missing completely at random (MCAR)** : Une donnée est MCAR, c'est à dire manquante de façon complètement aléatoire si la probabilité d'absence est la même pour toutes les observations. Cette probabilité ne dépend donc que de paramètres extérieurs indépendants de cette variable. Par exemple : "si chaque participant à un sondage décide de répondre à la question du revenu en lançant un dé et en refusant de répondre si la face 6 apparaît" [1]. A noter que si la quantité de données MCAR n'est pas trop importante, ignorer les cas avec des données manquantes ne biaiserait pas l'analyse. Une perte de précision dans les résultats est toutefois à prévoir.
- **Missing at random (MAR)** : Le cas des données MCAR est tout de même peu courant. Il arrive souvent que les données ne manquent pas de façon complètement aléatoire. Si la probabilité d'absence est liée à une ou plusieurs autres variables observées, on parle de *missingness at random* (MAR). Il existe des méthodes statistiques appropriées qui permettront d'éviter de biaiser l'analyse (voir 4)
- **Missing not at random (MNAR)** : La donnée est manquante de façon non aléatoire (MNAR) si la probabilité d'absence dépend de la variable en question. Un exemple répandu [1][9] est le cas où des personnes avec un revenu important refusent de le dévoiler. Les données MNAR induisent une perte de précision (inhérente à tout cas de données manquantes) mais aussi un biais qui nécessite le recours à une analyse de sensibilité.

2.2 Répartition des données manquantes

Soit $Y = (y_{ij}) \in \mathbb{R}^{n \times p}$ la matrice rectangulaire des données pour p variables Y_1, \dots, Y_p et n observations. Considérons $M = (m_{ij})$ la matrice d'*indication des valeurs manquantes* [13], qui va définir la répartition des données manquantes. On considérera alors 3 types de répartition :

1. Les valeurs manquantes **univariées**. C'est à dire que pour une variable

Y_k seulement, si une observation y_{ki} est manquante, alors il n'y aura plus d'observation de cette variable. Une illustration est donnée Figure 1a.

2. Les valeurs manquantes sont dites **monotones** si Y_j manquante pour un individu i implique que toutes les variables suivantes $\{Y_k\}_{k>j}$ sont manquantes pour cet individu (Figure 1b). L'indicateur de données manquantes M est alors un entier $M \in (1, 2, \dots, p)$ pour chaque individu, indiquant le plus grand j pour lequel Y_j est observé.
3. Les valeurs manquantes sont **non monotones** (ou **arbitraires**), comme le représente la Figure 1c. Dans ce cas, on définit la matrice de valeurs manquantes par $M = (m_{ij})$ avec $m_{ij} = 1$ si y_{ij} est manquant et zéro sinon.

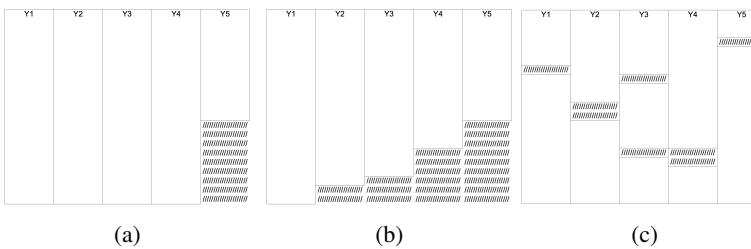


FIGURE 1 – Répartitions des données manquantes. (a) univariées, (b) monotones et (c) arbitraires/non monotones

Cette répartition est valable pour les données longitudinales (voir Figure 2). La répartition monotone correspond alors à une censure à droite.

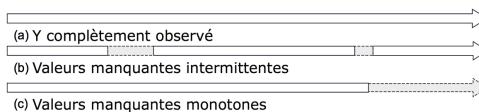


FIGURE 2 – Répartitions des données manquantes pour des variables longitudinales. (a) jeu complet, (b) arbitraires/non monotones et (c) monotones

2.3 Probabilité d'absence

La probabilité d'absence selon le type de données manquantes (MCAR, MAR, MNAR) peut alors être exprimé en fonction de M [13]. Les données sont divisées en deux selon la matrice M d'indication des données manquantes. On définit donc $Y_{obs} = Y\mathbb{1}_{\{M=0\}}$ les données observées et $Y_{mis} = Y\mathbb{1}_{\{M=1\}}$ les données manquantes telles que $Y = \{Y_{obs}, Y_{mis}\}$. Le mécanisme des données manquantes est caractérisé par la distribution conditionnelle de M sachant Y donnée par $p(M|Y)$.

- Dans le cas des données **MCAR** l'absence de données ne dépend pas des valeurs de Y donc

$$p(M|Y) = p(M) \text{ pour tout } Y. \quad (1)$$

- Considérons à présent le cas **MAR**. Soit Y_{obs} la partie observée du jeu de données et Y_{mis} les données manquantes. MAR signifie que l'absence de données dépend uniquement de Y_{obs} :

$$p(M|Y) = p(M|Y_{obs}) \text{ pour tout } Y_{mis}. \quad (2)$$

- Enfin, les données sont **MNAR** si la distribution de M dépend aussi de Y_{mis} .

Exemple pour un échantillon aléatoire univarié

Soit $Y = (y_1, \dots, y_n)^\top$ où y_i est l'observation d'une variable aléatoire pour l'individu i , et $M = (M_1, \dots, M_n)$ où $M_i = 0$ pour les données observées et $M_i = 1$ pour les données manquantes. On suppose également que la distribution conjointe est indépendante des individus. Alors

$$p(Y, M) = p(Y)p(M|Y) = \prod_{i=1}^n p(y_i) \prod_{i=1}^n p(M_i|y_i) \quad (3)$$

où $p(y_i)$ est la densité de y_i et $p(M_i|y_i)$ est la densité d'une loi de Bernoulli pour l'indicateur binaire M_i avec la probabilité $\mathbb{P}(M_i = 1|y_i)$ que y_i soit manquante.

Si $\mathbb{P}(M_i = 1|y_i) = \alpha$ avec α une constante qui ne dépend pas de y_i alors c'est un cas MCAR (ou dans ce cas aussi MAR). Si $\mathbb{P}(M_i = 1|y_i)$ dépend de y_i , le mécanisme de données manquantes est MNAR.

3 Analyse sans complémentation

3.1 Méthodes avec suppression de données

Dans certains cas, l'analyse est possible sans imputer les données manquantes. En général, on se reporte à deux méthodes “classiques” :

- **L'analyse des cas concrets**, qui consiste à ne considérer que les individus pour lesquels toutes les données sont disponibles, i.e. en supprimant les lignes comportant des valeurs manquantes. C'est ce qui est fait automatiquement avec R (`na.action=na.omit`). Cette méthode, on le voit bien Figure 3, risque de supprimer trop de données et d'augmenter de beaucoup la perte de précision. De plus, si les données ne sont pas MCAR, retirer des observations va induire un biais dans l'analyse puisque le sous-échantillon des cas représentés par les données manquantes ne sont pas forcément représentatifs de l'échantillon initial.

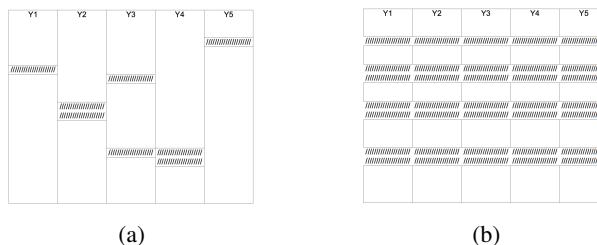


FIGURE 3 – Répartitions des données manquantes. (a) données d'origine, valeurs manquantes arbitraires, (b) observations restantes en analyse des cas complets

- **L'analyse des cas disponibles**. Afin d'éviter de supprimer trop de données, il est possible de faire de la suppression par paires (*pairwise deletion*) ou analyse des cas disponibles (*available-case analysis*). Différents aspects du problème sont alors étudiés avec différents sous-échantillons. Cependant, les différentes analyses ne seront pas nécessairement compatibles entre elles.

L'analyse des cas disponibles correspond aussi au cas où une variable est supprimée du jeu de données à cause de sa trop grande quantité de valeurs manquantes.

3.2 Méthodes qui tolèrent les données manquantes

Si la plupart des méthodes d'analyse suppriment automatiquement les données manquantes, certaines les tolèrent. C'est le cas par exemple des arbres **CART** qui considèrent des *surrogate splits* ou divisions de substitution : Au moment du split d'un nœud, plusieurs couples variables / seuil “optimaux” sont considérés et mémorisés. Au moment de l'utilisation, si la donnée est manquante pour une observation, ce n'est pas la meilleure division qui est utilisée mais celle juste après lui est substituée [7].

4 Méthodes d'imputation

Cette section donne un aperçu non exhaustif des méthodes de complémentation les plus courantes. On considère un jeu de données constitué de p variables quantitatives ou qualitatives (Y_1, \dots, Y_p) observées sur un échantillon de n individus. On définit la matrice M d'indication des valeurs manquantes par $m_{ij} = \mathbb{1}_{\{y_{ij} \text{ manquante}\}}$

4.1 Complémentation stationnaire

Il existe plusieurs complétements stationnaires possibles. On peut par exemple choisir de compléter par la valeur la plus fréquemment représentée (*Concept Most Common Attribute Value Fitting*, CMCF [14]) ou plus simplement par la dernière valeur connue (*Last observation carried forward*, LOCF) :

$$(y_{ij})_{mis} = y_{i^*j^*} = \{y_{i^*j}|m_{i^*j} = 0, j < j^*\} \quad (4)$$

Cette méthode peut sembler trop “naïve” mais est souvent utilisée pour poser les bases d'une comparaison entre méthodes de complémentation.

4.2 Complémentation par une combinaison linéaire des observations

Une autre technique répandue consiste à remplacer toutes les valeurs manquantes par une combinaison linéaire des observations. On retiendra le cas d'imputation par la moyenne :

$$(y_{ij})_{mis} = y_{i^*j^*} = \bar{Y}_{j^*} \quad (5)$$

ou par la médiane :

$$(y_{ij})_{mis} = y_{i^*j^*} = \tilde{Y}_{j^*} \quad (6)$$

Mais ce cas se généralise à toute combinaison linéaire pondérée des observations.

Au lieu d'utiliser toutes les valeurs disponibles, il est possible de se restreindre à des méthodes qui sélectionnent les valeurs les plus influentes. Par exemple, on présente ci-dessous des méthodes d'agrégation locale ou de régression ainsi que des algorithmes combinant différents aspects.

4.3 Méthode des plus proches voisins (KNN)

La complétion par k plus proches voisins (*k-nearest neighbors* ou KNN) consiste à suivre l'algorithme suivant :

Algorithme des k plus proches voisins (k -nn)

1. Choix d'un entier $k : 1 \geq k \geq n$.
2. Calculer les distances $d(Y_{i^*}, Y_i), \quad i = 1, \dots, n$
3. Retenir les k observations $Y_{(i_1)}, \dots, Y_{(i_k)}$ pour lesquelles ces distances sont les plus petites.
4. Affecter aux valeurs manquantes la moyenne des valeurs des k voisins :

$$(y_{ij})_{mis} = y_{i^*j^*} = \frac{1}{k} (Y_{(i_1)} + \dots + Y_{(i_k)}) \quad (7)$$

Comme pour la [classification par KNN](#), la méthode des plus proches voisins nécessite le choix du paramètre k par optimisation d'un critère. De plus, la notion de distance entre les individus doit être choisie avec précaution. On considérera usuellement la distance Euclidienne ou de Mahalanobis.

4.4 Régression locale

La régression locale (en anglais *LOcal regrESSion* : LOESS) [15] permet également d'imputer des données manquantes. Pour cela, un polynôme de degré faible est ajusté autour de la donnée manquante par moindres carrés pondérés, en donnant plus de poids aux valeurs proches de la donnée manquante.

Soit Y_{i^*} une observation à laquelle il manque q valeurs manquantes. On impute ces données manquantes par régression locale en suivant l'algorithme ci-après.

Algorithme LOESS

1. Obtention des k plus proches voisins $Y_{(i_1)}, \dots, Y_{(i_k)}$
2. Création des matrices $A \in \mathbb{R}^{k \times (n-q)}$, $B \in \mathbb{R}^{k \times q}$ et $w \in \mathbb{R}^{(n-q) \times 1}$ de sorte que :
 - Les lignes de A correspondent aux voisins privés des valeurs aux indices des données manquantes de Y_{i^*}
 - Les colonnes de B correspondent aux valeurs des voisins aux indices des données manquantes de Y_{i^*}
 - Le vecteur w correspond aux $(n - q)$ valeurs observées de Y_{i^*} : $w_j = (y_{i^*j})_{obs}$
3. Résolution du problème des moindres carrés

$$\min_{x \in \mathbb{R}^k} \| A^\top x - w \| \quad (8)$$

où $\| \cdot \|$ est la norme quadratique de \mathbb{R}^k .

4. Le vecteur u des données manquantes s'exprime alors par

$$u = B^\top x = B^\top (A^\top)^{-1} w \quad (9)$$

avec $(A^\top)^{-1}$ la matrice pseudo-inverse de A^\top .

4.5 Algorithme NIPALS

L'algorithme NIPALS (*Nonlinear Iterative Partial Least Squares*) est une méthode itérative proche de la [régression PLS](#), utilisée pour estimer les éléments d'une analyse en composantes principales d'un vecteur aléatoire de dimension finie. Cet algorithme peut être adapté pour l'imputation de données manquantes [3]. Soit $Y = (Y_1, \dots, Y_p)$ tel que $\forall i \in 1, \dots, p, E(Y_i) = 0$

(chaque colonne de la matrice est centrée). L'expansion de Y en termes de composantes principales et de facteurs principaux est donnée par

$$Y = \sum_{h=1}^q \xi_h u_h \quad (10)$$

où $q = \dim L_2(Y)$ et $\{\xi_h\}_{h=1,\dots,q}$ sont les composantes principales et $\{u_h\}_{h=1,\dots,q}$ les vecteurs principaux de l'ACP de Y . Donc pour chaque variable Y_i on a

$$Y_i = \sum_{h=1}^q \xi_h u_h(i) \quad (11)$$

L'idée étant que pour chaque h , $u_h(i)$ représente la pente de la régression linéaire de Y_i sur la composante ξ_h . L'algorithme NIPALS va permettre d'obtenir $\{\hat{\xi}_h\}_{h=1,\dots,q}$ et $\{\hat{u}_h\}_{h=1,\dots,q}$ les approximations de $\{\xi_h\}_{h=1,\dots,q}$ et $\{u_h\}_{h=1,\dots,q}$.

Algorithme NIPALS

1. $Y^0 = Y$
2. Pour $h = 1, \dots, q$ faire
 - (a) $\xi_h = Y_1^{h-1}$
 - (b) Tant que u_h n'a pas convergé faire
 - i. Pour $i = 1, \dots, p$ faire

$$u_h(i) = \frac{\sum_{j:y_{ji}, \xi_h(j) \text{ existe}} y_{ji}^{h-1} \xi_h(j)}{\sum_{j:\xi_h(j) \text{ existe}} \xi_h^2(j)}$$

- ii. Normaliser u_h
- iii. Pour $i = 1, \dots, N$ faire

$$\xi_h(i) = \frac{\sum_{j:y_{ij} \text{ existe}} y_{ij}^{h-1} u_h(j)}{\sum_{j:y_{ij} \text{ existe}} u_h^2(j)}$$

- (c) $Y^h = Y^{h-1} - \xi_h u'_h$

On pourra alors approximer les données manquantes par

$$(\hat{y}_{ij})_{mis} = \sum_{h=1}^q \hat{\xi}_h(i) \hat{u}_h(j) \quad (12)$$

4.6 Par décomposition en valeurs singulières (SVD)

4.6.1 Cas où il y a suffisamment de données observées

S'il y a bien plus de données observées que de données manquantes, on sépare le jeu de données Y en deux groupes : d'un côté Y^c avec les observations complètes et de l'autre Y^m comprenant les individus pour lesquels certaines données manquent. On considère alors la décomposition en valeurs singulières (SVD) tronquée du jeu complet [6] :

$$\hat{Y}_J^c = U_J D_J V_J^\top \quad (13)$$

où D_J est la matrice diagonale comprenant les J premières valeurs singulières de Y^c . Les valeurs manquantes sont alors imputées par régression :

$$\min_{\beta \in \mathbb{R}^J} \sum_{i \text{ observées}} \left(Y_{i*} - \sum_{j=1}^J v_{lj} \beta_j \right)^2 \quad (14)$$

Soit V_J^* la version tronquée de V_J , c'est à dire pour laquelle les lignes correspondant aux données manquantes de la ligne Y_{i*} sont supprimées. Une solution du problème 14 est alors

$$\hat{\beta} = (V_J^{*\top} V_J^*)^{-1} V_J^{*\top} Y_{i*} \quad (15)$$

La prédiction des données manquantes est donc donnée par

$$Y_{i*} = V_J^{(*)} \hat{\beta} \quad (16)$$

où $V_J^{(*)}$ est le complément de V_J^* dans V_J .

Comme pour KNN, cette méthode nécessite le choix du paramètre J . On se ramènera alors à un problème de minimisation :

$$\min_{X \text{ de rang } J} \|Y^c - X\|_F \quad (17)$$

avec $\|\cdot\|_F$ la norme de Frobenius.

4.6.2 Cas où il y a trop de données manquantes

Si les données manquantes sont trop nombreuses, cela induira un biais important dans le calcul de la base de décomposition. De plus, il arrive qu'il y ait au moins une donnée manquante pour toutes les observations. Dans ce cas, il faut résoudre le problème suivant :

$$\min_{U_J, V_J, D_J} \| Y - m - U_J D_J V_J^\top \|_* \quad (18)$$

où $\| \cdot \|_*$ somme les carrés des éléments de la matrice, en ignorant les valeurs manquantes. m est le vecteur des moyennes des observations. La résolution de ce problème suit l'algorithme suivant :

Algorithme de Complétion par SVD

1. Créer une matrice Y^0 pour laquelle les valeurs manquantes sont complétées par la moyenne,
2. Calculer la SVD solution du problème (18) pour la matrice complétée Y^i . On crée ainsi Y^{i+1} en remplaçant les valeurs manquantes de Y par celles de la régression.
3. Itérer l'étape précédente jusqu'à ce que $\| Y^i - Y^{i+1} \| / \| Y^i \| < \epsilon$, seuil arbitraire (souvent à 10^{-6})

4.7 Utilisation de Forêts aléatoires

Stekhoven et Bühlmann (2011)[5] ont proposé une méthode de complétion basée sur les **forêts aléatoires** appelée **MissForest**. Un package R éponyme lui est associée. Cette méthode nécessite une première imputation “naïve”, par défaut une complétion par la moyenne, afin d'obtenir un échantillon d'apprentissage complet. Puis une série de forêts aléatoires sont ajustées jusqu'à la première dégradation du modèle.

Pour formaliser cela, on décompose le jeu de données initial en quatre parties. Pour chaque variable Y^s , $s = 1, \dots, S$ dont les valeurs manquantes sont indexées par $i_{mis}^s \subseteq \{1, \dots, n\}$, on définit

1. y_{obs}^s les valeurs observées dans Y^s

2. y_{mis}^s les valeurs manquantes dans Y^s
3. $X^s = Y \setminus Y^s$ l'ensemble des régresseurs de Y^s parmi lesquels on considère
 - (a) x_{obs}^s les régresseurs observés pour $i_{obs}^s = \{i, \dots, n\} \setminus i_{mis}^s$
 - (b) x_{mis}^s les régresseurs manquants pour i_{mis}^s

La méthode suit alors l'algorithme suivant :

Algorithme MissForest

1. Première complétion “naïve” des valeurs manquantes.
2. Soit k le vecteur des indices de colonnes de Y triées par quantité croissante de valeurs manquantes ;
3. **Tant que** γ n'est pas atteint **faire**
 - (a) Y_{imp}^{old} = matrice précédemment imputée
 - (b) **Pour** s dans k **faire**
 - i. Ajuster $y_{obs}^{(s)} \sim x_{obs}^{(s)}$ par forêt aléatoire
 - ii. Prédire $y_{mis}^{(s)}$ avec les régresseurs $x_{mis}^{(s)}$
 - iii. Y_{imp}^{new} est la nouvelle matrice complétée par les valeurs prédites $y_{mis}^{(s)}$
 - (c) mettre à jour le critère γ

Avec un critère d'arrêt γ atteint dès que la différence entre la matrice de données nouvellement imputé et la précédente augmente pour la première fois. La différence de l'ensemble des variables continues est définie comme

$$\Delta_N = \frac{\sum_{j \in N} (Y_{imp}^{new} - Y_{imp}^{old})^2}{\sum_{j \in N} (Y_{imp}^{new})^2} \quad (19)$$

En cas de variables qualitatives on définit la différence par

$$\Delta_F = \frac{\sum_{j \in F} \sum_{i=1}^n \mathbb{1}_{Y_{imp}^{new} \neq Y_{imp}^{old}}}{\#NA} \quad (20)$$

4.8 Inférence Bayésienne

Soit θ la réalisation d'une variable aléatoire et soit $p(\theta)$ sa distribution *a priori*. La distribution *a posteriori* est donc donnée par :

$$p(\theta|Y_{obs}) \propto p(\theta)f(Y_{obs}; \theta) \quad (21)$$

La méthode de *data augmentation* de Tanner et Wong (1987) [10] simule de manière itérative des échantillons aléatoires des valeurs manquantes et des paramètres du modèle, compte tenu des données observées à chaque itération, constituée d'une étape d'imputation (I) et d'une étape "postérieure" (P).

Soit $\theta^{(0)}$ un tirage initial obtenu à partir d'une approximation de la distribution *a posteriori* de θ . Pour une valeur $\theta^{(t)}$ de θ à un instant t

Imputation (I) : soit $Y_{mis}^{(t+1)}$ avec une densité $p(Y_{mis}|Y_{obs}, \theta^{(t)})$

Postérieure (P) : soit $\theta^{(t+1)}$ avec une densité $p(\theta|Y_{obs}, Y_{mis}^{(t)})$

Cette procédure itérative finira par obtenir un tirage de la distribution conjointe de $(Y_{mis}, \theta|Y_{obs})$ lorsque $t \rightarrow +\infty$

4.9 Imputation multiple

L'imputation multiple consiste, comme son nom l'indique, à imputer plusieurs fois les valeurs manquantes afin de combiner les résultats pour diminuer l'erreur (le bruit) due à l'imputation [4]. Cela permet également de définir une mesure de l'incertitude causée par la complétion.

Le maintien de la variabilité d'origine des données se fait en créant des valeurs imputées qui sont basés sur des variables corrélées avec les données manquantes et les causes d'absence. L'incertitude est prise en compte en créant des versions différentes de données manquantes et l'observation de la variabilité entre les ensembles de données imputées.

4.10 Amelia II

Amelia II est un programme d'imputation multiple développé en 2011 par James Honaker et al [8]. Le modèle s'appuie sur une hypothèse de normalité : $Y \sim \mathcal{N}_k(\mu, \Sigma)$, et nécessite donc parfois des transformations préalables des données.

Soit M la matrice d'indication des données manquantes et $\theta = (\mu, \Sigma)$ les

paramètres du modèle. Une autre hypothèse est que les données sont **MAR** donc

$$p(M|Y) = p(M|Y_{obs}) \quad (22)$$

La vraisemblance $p(Y_{obs}|\theta)$ s'écrit alors

$$p(Y_{obs}, M|\theta) = p(M|Y_{obs})p(Y_{obs}|\theta) \quad (23)$$

Donc

$$L(\theta|Y_{obs}) \propto p(Y_{obs}|\theta) \quad (24)$$

Or en utilisant la propriété itérative de l'espérance

$$p(Y_{obs}|\theta) = \int p(Y|\theta)dY_{mis} \quad (25)$$

On obtient donc la loi *a posteriori*

$$p(\theta|Y_{obs}) \propto p(Y_{obs}|\theta) = \int p(Y|\theta)dY_{mis} \quad (26)$$

L'algorithme EMB d'Amelia II combine l'algorithme EM classique (du **maximum de vraisemblance**) avec une approche **bootstrap**. Pour chaque tirage, les données sont estimées par *bootstrap* pour simuler l'incertitude puis l'algorithme EM est exécuté pour trouver l'estimateur *a posteriori* $\hat{\theta}_{MAP}$ pour les données *bootstrap*. Les imputations sont alors créées par tirage de Y_{mis} selon sa distribution conditionnelle sur Y_{obs} et des tirages de θ .

5 Exemple

5.1 Fraudes sur la consommation en gaz

Les différentes méthodes de complétion ont été testées et comparées sur un exemple de détection de fraudes sur la consommation en gaz. Soit $Y \in \mathbb{R}^{N \times 12}$ tel que y_{ij} soit la consommation de gaz de l'individu i au mois j . La répartition des données manquantes est non monotone et on fait l'hypothèse de données MAR. Après une transformation en log afin d'approcher la normalité, la complétion a été effectuée. Les résultats ont été comparés avec un échantillon test de 10% des données, préalablement retiré du set.

Ce jeu de données réel comporte au moins une valeur manquante par individu, et au total 50.4% des données sont manquantes. Si on ne considère que la consommation mensuelle individuelle, sans variables exogènes, on obtient la répartition des erreurs de chaque méthode représentée Figure 4.

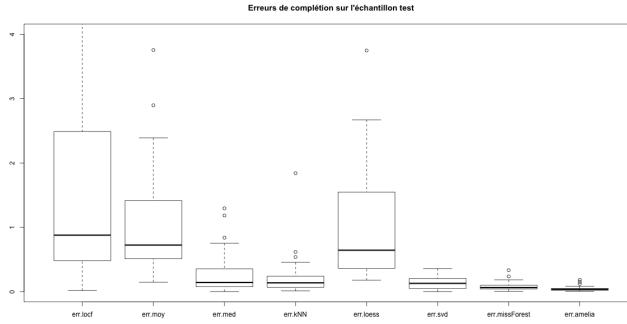


FIGURE 4 – Fraudes - Erreurs de complétion sur un échantillon test

5.2 Encours Boursiers Parisiens (EBP)

On s'intéresse aux cours des **actifs boursiers** sur la place de Paris de 2000 à 2009. On considère 252 cours d'entreprises ou indices régulièrement cotés sur cette période. En se limitant au cas MCAR, on crée artificiellement de plus en plus de données manquantes à imputer. Pour 10% de données manquantes, une comparaison des méthodes d'imputations est donnée Figure 5. Trois méthodes se détachent visiblement : SVD, missForest et AmeliaII.

La robustesse de ces méthodes a été testée en augmentant graduellement la quantité de données manquantes. Les résultats sont donnés Figure 6 pour AmeliaII et Figure 7 pour missForest.

5.3 Maladies Coronariennes (CHD)

La plupart des méthodes d'imputation de sont définies que pour des variables quantitatives. Mais certaines méthodes présentées ci-dessus permettent d'imputer des données qualitatives, voire hétérogènes. C'est le cas de LOCF, KNN et missForest qui ont donc été testées sur un

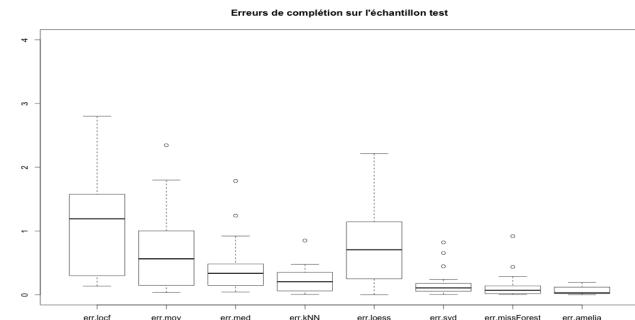


FIGURE 5 – EBP - Erreurs de complétion sur un échantillon test de 10%

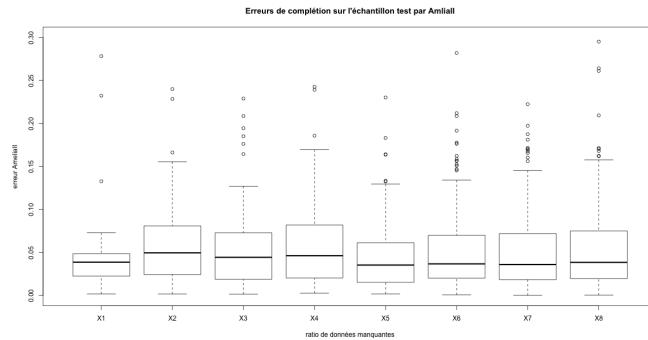


FIGURE 6 – EBP - Erreurs de complétion sur un échantillon test par AmeliaII quand la quantité de valeurs manquantes augmente

jeu de données de référence sur les problèmes de compléTION [11]. Les données ont été acquises par Detranao et al. (1989) [12] et mises à disposition par Bache et Lichman (2013)[2]. Elles se présentent sous la forme d'une matrice d'observations médicales $Y \in \mathbb{R}^{N \times 14}$ de 14 variables hétérogènes pour N patients. Le jeu de données contient donc des variables quantitatives (age, pression, cholestérol, fréquence cardiaque maximale, oldpeak) et qualitatives (sexe, douleur,

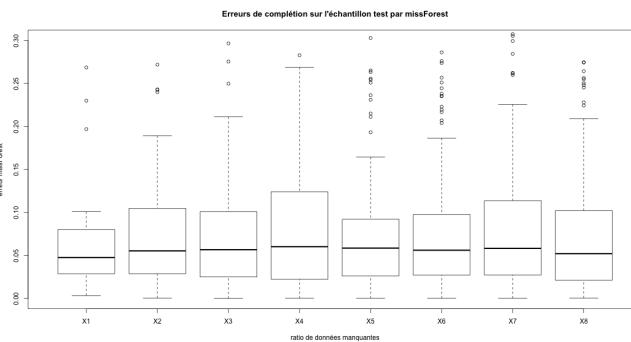


FIGURE 7 – EBP - Erreurs de complétiion sur un échantillon test par missForest quand la quantité de valeurs manquantes augmente

sucré, cardio, angine, pente du pic, nombre de vaisseaux cardiaques, thalassémie, absence/présence de maladie cardiaque).

En se limitant toujours au cas MCAR, on crée artificiellement de plus en plus de données manquantes à imputer. L’adéquation de l’imputation est donnée par la moyenne de l’erreur en valeur absolue dans le cas des données quantitatives et par la distance de Hamming dans le cas des données qualitatives. Les résultats sont représentés Figure 8.

Références

- [1] Gelman A. et Hill J., *Data Analysis Using Regression and Multilevel/Hierarchical Models*, chap. 25, p. 529–563, Cambridge University Press, 2007.
- [2] K. Bache et M. Lichman, *UCI Machine Learning Repository*, 2013, <http://archive.ics.uci.edu/ml>.
- [3] Preda C., Saporta G. et Hedi Ben Hadj Mbarek M., *The NIPALS algorithm for missing functional data*, Romanian Journal of Pure and Applied Mathematics **55** (2010), n° 4, 315–326.
- [4] Rubin D.B., *Multiple Imputation for Nonresponse in Surveys*, Wiley, 1987.
- [5] Stekhoven D.J. et Bühlmann P., *MissForest - nonparametric missing value imputation for mixed-type data*, Bioinformatics Advance Access (2011).
- [6] Hastie et al, *Imputing Missing Data for Gene Expression Arrays*, Rap. tech., Division of Biostatistics, Stanford University, 1999.
- [7] A. J. Feelders, *Handling Missing Data in Trees : Surrogate Splits or Statistical Imputation.*, PKDD, Lecture Notes in Computer Science, t. 1704, Springer, 1999, p. 329–334.

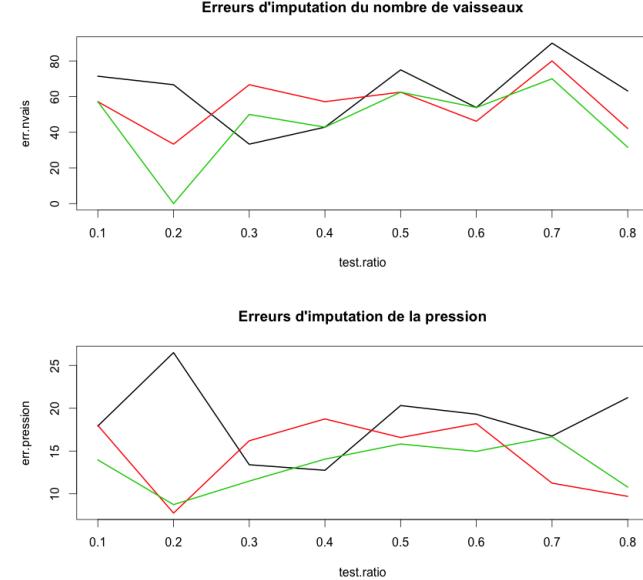


FIGURE 8 – CHD - Erreurs de complétiion sur un échantillon test par LOCF (noir), KNN (rouge) et missForest (vert) quand la quantité de valeurs manquantes augmente, pour une variable qualitative (au dessus) et quantitative (en dessous)

- [8] Honaker J., King G. et Blackwell M., *Amelia II : A Program for Missing Data*, Journal of statistical software **45** (2011), n° 7.
- [9] Glasson Cicignani M. et Berchtold A., *Imputation de Donnees Manquantes : Comparaison de Differentes Approches*, 42e Journees de Statistique, 2010.
- [10] Tanner M.A. et Wong W.H., *The Calculation of Posterior Distributions by Data Augmentation*, Journal of the American Statistical Association **82** (1987), n° 398, 528–540.
- [11] Setiawan N.A., Venkatachalam P.A. et Hani A.F.M., *A Comparative Study of Imputation Methods to Predict Missing Attribute Values in Coronary Heart Disease Data Set*, 4th Kuala Lumpur International Conference on Biomedical Engineering 2008 (University of Malaya Department of Biomedical Engineering Faculty of Engineering, réd.), t. 21, Springer Berlin Heidelberg, 2008, p. 266–269.
- [12] Detrano R., Janosi A., Steinbrunn W., Pfisterer M., Schmid J., Sandhu S., Guppy K., Lee S. et Froelicher V., *International Application of a New Probability Algorithm for the Diagnosis of Coronary Artery Disease*, American Journal of Cardiology **64** (1989), 304–310.
- [13] Little R.J.A. et Rubin D.B., *Statistical Analysis with Missing Data*, Wiley series in probability and statistics, 1987.
- [14] Grzymala Busse J. W., Grzymala Busse W. J. et Goodwin L. K., *Coping With Missing Attribute Values Based on Closest Fit in Preterm Birth Data : A Rough Set Approach*, Computational Intelligence **17** (2001), 425–434.
- [15] Cleveland W.S. et Devlin S.J., *Locally-Weighted Regression : An Approach to Regression Analysis by Local Fitting*, Journal of the American Statistical Association **83** (1988), n° 403, 596–610.

Détection d'anomalies

Résumé

Présentation schématique du cadre général des méthodes de détection d'anomalies multidimensionnelles ou défauts, fraudes, défaillances... Description plus avancée de celles non supervisées et plus précisément non paramétriques, sans hypothèses sur la distribution des variables ou de celles des résidu à un modèle : LOF, OCC SVM et Random Forest pour la détection d'observations atypiques. Illustration sur les données de prévision du dépassement du seuil d'ozone.

[Retour au plan du cours](#)

1 Introduction

1.1 Historique

La détection d'anomalies (*outliers*) est en enjeu ancien et majeur des applications industrielles de la Statistique notamment pour la détection d'une défaillance ou défaut de fabrication. Historiquement très présente dans les services de suivi de la qualité par contrôle statistique des procédés (*Statistical Process Control*), la détection d'anomalies utilise des techniques largement répandues et imposées par la normalisation : diagramme boîte (cf. figure 1), dépassement d'un seuil fixé par un expert, tests séquentiels et *cartes de contrôles*, tests paramétriques de *discordance* (Rosner, 1983)[10].

1.2 Objectif

Le but initial du contrôle de qualité prend de l'ampleur avec la recherche d'une explication voire même idéalement d'une *prévision de la défaillance* en vue d'une *maintenance prédictive*. De plus, l'afflux de données issues d'une multitude de capteurs ou objets connectés impose de remplacer l'approche unidimensionnelle par une conception *multidimensionnelle* de l'anomalie (cf. figure 1) pour prendre en compte volume, variété, vélocité des données. Par ailleurs, le même but de détection d'anomalies se décline en beaucoup objec-

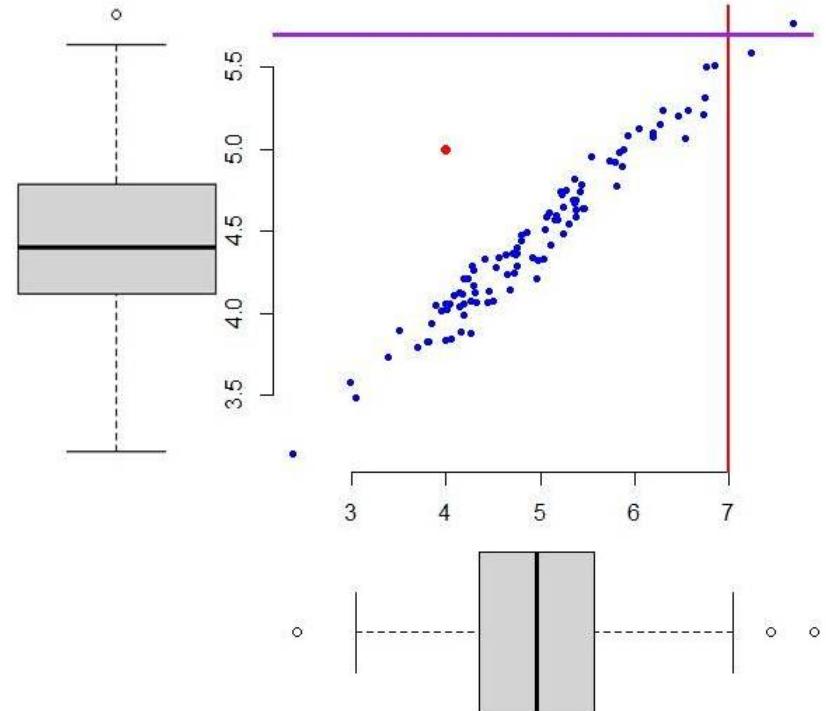


FIGURE 1 – Exemple trivial : une anomalie détectée simultanément par les diagrammes en boîte unidimensionnels semble moins atypique que celle en rouge au regard de la distribution bidimensionnelle.

tifs analogues de détection dans d'autres domaines que celui strict de la production industrielle : alarme, défaut, fraude, intrusion... Cela concerne donc tous les secteurs de production de données et d'informations.

L'objectif de cette vignette est de tenter de donner un aperçu synthétique des méthodes les plus utilisées tout en les situant dans un schéma global des approches et stratégies de détection d'anomalies.

2 Aperçu des méthodes de détection

2.1 Choix opérés

Le rapide tour d'horizon des méthodes et technologies concernées se focalise sur celles récentes, multidimensionnelles, non paramétriques et implémentées dans les librairies facilement accessibles. Se reporter à la bibliographie et aux normes usuelles pour aborder les approches unidimensionnelles par cartes de contrôles et multidimensionnelles paramétriques.

Attention, la théorie des *valeurs extrêmes* qui estime des probabilités d'occurrences d'événements rares en lien avec des lois de probabilités spécifiques est un tout autre problème pas du tout abordé.

2.2 Taxonomie de la détection d'anomalies

Quelque soit la méthode utilisée, une *anomalie des données* est toujours définie, implicitement ou explicitement relativement à un *modèle sous-jacent*. Le choix de la méthode et donc de ce modèle dépend complètement du contexte, de l'objectif visé, des données disponibles, de leurs propriétés. Voici un aperçu schématique de quelques possibilités. Se référer à Aggarwal (2017)[1] pour approfondir la réflexion.

La question de la détection d'anomalie peut être abordée de deux points de vue :

Supervisé : à condition de disposer d'une bases de données historiques contenant d'une part des observations jugées correctes à opposer à d'autres observations identifiées par un expert comme étant des anomalies. Apprendre à identifier, expliquer, prévoir ces anomalies se ramène alors à un objectif classique de classification binaire supervisée et renvoie aux méthodes exposées dans les autres vignettes. Attention,

la classe des anomalies est très généralement (heureusement) sous-représentées, engendrant les problèmes classiques de déséquilibres des classes de la variable à prédire. Problème dont la résolution doit prendre en compte le coût relatif entre celui d'une détections à tort et celui de la non détection ; rapport dépendant du contexte métier de l'étude.

Non-supervisé : Dans le cas où aucune donnée ou caractéristique définit l'anomalie, la vaste littérature parle aussi de classification à une classe (*One Class Classification, OCC*) ou détection de nouveauté (*novety detection*). Ces dernières appellations introduisent une nuance dans l'objectif. Il s'agit soit de la détection d'anomalies (*outliers*) dans un ensemble de données, soit de déterminer si une nouvelle observation est cohérente ou non avec les données déjà disponibles regroupées en une seule classe. Néanmoins les mêmes techniques, objets de cette vignette, peuvent être utilisées dans les deux cas.

Attention, même dans ce cas non-supervisé, il est très utile voire indispensable de disposer d'un historique, ou de simulations réalistes, relatant des cas authentifiés d'anomalies. Ils permettent de valider un choix de méthode et même optimiser un seuil de détection.

Comme déjà écrit, une observation est considérée anormale ou **atypique par rapport à un modèle**. De façon schématique et pour structurer la présentation, ce modèle peut-être :

- *paramétrique*, très généralement gaussien (e.g. modèle de mélanges),
- *non paramétrique* défini à partir des données (e.g. voisinage au sens des k plus proches voisins).

D'autre part, ce *modèle* peut être

- relatif à la présence d'une *variable cible* à expliquer, *modéliser*, prévoir par régression ou discrimination.
- Dans le cas contraire, il est celui de la *densité de probabilité* ou distribution multidimensionnelle des variables.

Par ailleurs, certaines méthodes prennent en compte des mélanges de variables mixtes : quantitatives ou qualitatives, d'autres ne sont adaptées qu'à des variables quantitatives, notamment les méthodes paramétriques. Il est nécessaire dans ce dernier cas de rendre quantitatives les variables qualitatives susceptibles d'être présentes : remplacement par des variables indicatrices, composantes de l'**AFCM**.

2.3 Cas non supervisé

Voici un résumé succinct et quelques exemples spécifiques au cas *non-supervisé*; il n'existe pas d'ensemble d'observations identifiées *a priori* comme des anomalies.

Cas paramétrique

La loi des variables explicatives, ou celle des résidus à un modèle, est supposée explicitement ou implicitement gaussienne multidimensionnelle.

Relatif au modèle d'une variable cible Dans le cas d'un modèle linéaire gaussien, les observations atypiques ou mal ajustées sont, par exemple, celles présentant de grands résidus studentisés ou de grandes valeurs de coefficients h_{ii} sur la diagonale de $\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$ où \mathbf{X} est la matrice de design du modèle. Néanmoins, la détection d'observations atypiques rejoint plutôt celle des *observations influentes*, par exemple à l'aide de la *distance de Cook*, bien connue déjà en **régression linéaire simple**. Les autres observations mal ajustées peuvent en effet passer inaperçues sans risque pour la prévision.

Sans variable cible à modéliser une généralisation du T-test de student a été proposée par Hotelling (1931)[7]. L'anormalité à une densité multidimensionnelle peut être caractérisée par la distance (D_M) de Mahalanobis estimée à partir de la matrice inverse de celle de covariance empirique : (\mathbf{S}^{-1}) de la distribution. La librairie `mvoutlier` de R (voir la bibliographie associée) calcule des quantiles au sens de D_M .

Une autre approche (Ruiz-Gazen et Caussinus ; 2007)[5], basée sur l'*analyse*

en composantes principales, utilise une estimation robuste de (\mathbf{S}^{-1}) :

$$\begin{aligned}\bar{\mathbf{x}} &= \sum_{i=1}^n \mathbf{x}_i \\ \mathbf{S} &= \frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})' \\ w_i &= \exp(-\beta/2 \|\mathbf{x}_i - \bar{\mathbf{x}}\|_{\mathbf{S}^{-1}}^2) \\ \mathbf{R} &= \frac{\sum_{i=1}^n \beta_i (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})'}{\sum_{i=1}^n \beta_i}\end{aligned}$$

L'ACP calculée en utilisant pour métrique de l'espace des individus celle de matrice $\mathbf{M} = \mathbf{R}^{-1}$ à pour effet de faire ressortir les observations atypiques au sens de cette distance sur le premier plan factoriel. L'hypothèse de normalité n'est pas explicite pour exécuter l'ACP mais seule une distribution sous-jacente approximativement gaussienne peut conduire à des résultats raisonnables. Des distributions trop exotiques ou des structures de liaisons non linéaires complexes entre les variables peuvent sévèrement perturber les représentations.

Cas non Paramétrique

Il n'y pas d'hypothèse sur la distribution multidimensionnelle des variables, celle-ci est estimée localement de différentes façon.

Relatif au modèle d'une variable cible *Random forest* (Breiman, 2001)[2] inclut une solution originale adaptée à la prise en compte de variables mixtes.

Sans variable cible à modéliser C'est dans ce dernier cas que le plus de méthodes ont été proposées et la littérature la plus vaste. Quelques mots clés : LOF, GLOSH, OCC SVM... pour des variables quantitatives ou rendues quantitatives, *random forest* pour variables mixtes.

Il ne s'agit évidemment pas d'une *liste exhaustive* des méthodes et algorithmes disponibles. Ainsi les méthodes de **classification non supervisée** (CAH, k -means,...) sont aussi candidates à la détection d'anomalie lorsqu'une

ou des classes se réduisent à une seule observation ou que des observations restent en marge (DBSCAN). Consulter Aggarwal (2017)[1] pour avoir un aperçu plus vaste et plus détaillé d'un ensemble des méthodes et de leur justification.

Le choix opéré par la suite met l'accent sur les méthodes les plus généralement utilisables donc acceptant des variables mixtes, et facilement accessibles dans les librairies classiques (R et pour certaines en Python). Les méthodes choisies, non paramétriques, sont illustrées sur les données de prévision de dépassement du seuil d'ozone par adaptation statistique.

3 Méthodes non-paramétriques

Cette section développe donc les situations dans lesquelles

- il n'existe pas de base de données suffisamment renseignée des anomalies pour les apprendre,
- et qui ne nécessitent pas d'hypothèse de nature probabiliste (normalité) sur la distribution des variables ou de celle des résidus à un modèle.

Deux cas sont donc considérés selon qu'une anomalie est définie par rapport à un modèle explicatif ou prédictif d'une variable cible ou celui d'estimation non paramétrique de la distribution des observations.

3.1 Anomalie par rapport à un modèle prédictif

Il s'agit donc d'identifier des observations atypiques par rapport à un modèle expliquant une variable Y (régression ou discrimination) par des variables mixtes quantitatives ou qualitatives et sans hypothèse sur leur distribution. Breiman (2001)[2] propose de la faire en définissant une notion de *proximité* ou similarité puis de distance entre les observations participant à l'apprentissage d'une forêt aléatoire.

Une matrice de similarité entre les observations prises deux à deux qui ont participé à l'apprentissage d'une forêt aléatoire est simplement obtenue en comptant le nombre de fois où deux observations appartiennent à la même feuille d'un arbre. Ces effectifs sont ensuite normalisés par le nombre d'arbres de la forêt pour obtenir une matrice symétrique, positive dont les termes sont bornés par 1, les valeurs de la diagonale.

Breiman (2001)[2] définit ensuite un *score d'anomalie* d'une observation relativement à sa classe. Soit \bar{P} la somme des carrés des proximités de l'obser-

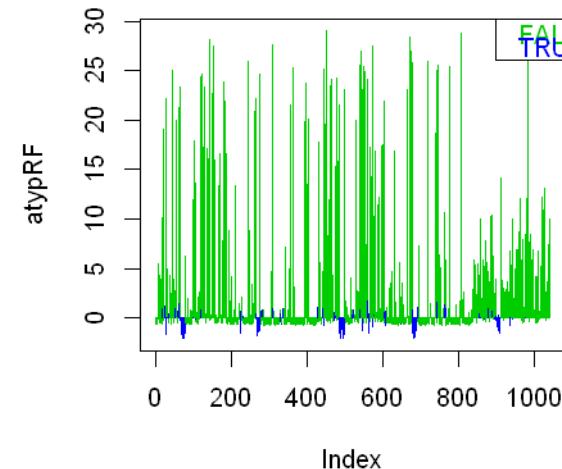


FIGURE 2 – Ozone : Score d'anomalie des observations au sens des forêts aléatoires et par rapport à la prévision du dépassement du seuil d'ozone. En bleu dépassement, en vert pas de dépassement.

vation en question à toutes les observations de sa classe. Le score est le rapport du nombre d'observations divisé par \bar{P} puis normalisé en soustrayant la médiane et divisant par *MAD* (*mean absolute deviation* : la médiane des écarts absolu à la médiane).

La figure 2 représente les scores d'anomalies pour chaque observation au sens de cette définition lors de la prévision de dépassement du pic d'ozone. Seules des observations sans dépassement de seuil conduisent à des scores élevés ; observations dont les conditions météorologiques s'apparentent à celles observées lors d'un dépassement.

Le graphique suivant figure 3 représentent les observations jugées atypiques dans le premier plan factoriel de l'analyse en composantes principales

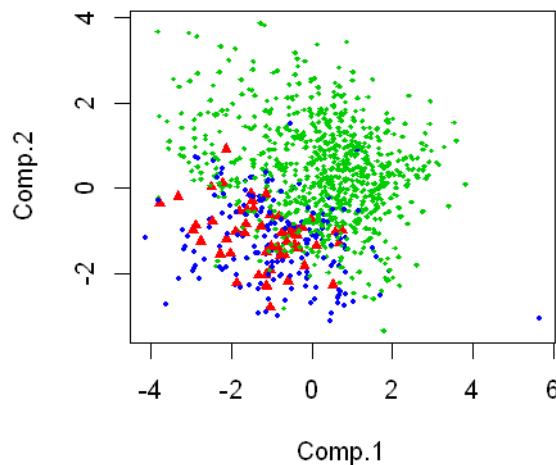


FIGURE 3 – Ozone : observations atypiques (en rouge) au sens du critère issue d'une forêt aléatoire expliquant la variable de dépassement (bleu vs. vert) du seuil d'ozone.

(ACP); ce sont celles avec un score supérieur à un seuil choisi arbitrairement à 20. Sans surprise, ces observations sans dépassement se projettent sur le premier plan de l'ACP dans le proche voisinage des observations avec dépassement.

Attention, supprimer ou modifier les observations atypiques à un modèle sans justification serait totalement contraire à l'éthique. L'objectif est avant tout de les identifier car ce sont celles, les plus susceptibles d'être la conséquence d'une erreur (à confirmer) de mesure, de libellé, ou encore une anomalie, défaillance ou tentative de fraude, d'intrusion, selon le contexte.

3.2 Anomalie par rapport à une distribution

Il s'agit dans ce cas d'évaluer l'incohérence ou l'*isolement* d'une ou de plusieurs observations par rapport à l'ensemble des autres observations. Plusieurs approches sont décrites sous les appellations de *One Class Classification* ou *novelty detection*.

Densité locale

Notations Beaucoup de travaux sont basés sur une estimation locale de la densité des observations en considérant leurs distances mutuelles. Dans cette section, les données sont un ensemble D d'individus x issus des observations de n vecteurs \mathbf{x} de \mathbb{R}^p muni d'une métrique (L_1, L_2, L_p, \dots) définissant une distance $d(\mathbf{x}, \mathbf{y})$. Les données peuvent également être, directement, la connaissance d'une matrice $\mathbf{D}_{n \times n}$ de distances des individus, observations ou instances pris 2 à 2 dans D .

L'anomalie ou l'*isolement* d'une observation est apprécié par la proximité des points de son voisinage. Ramaswamy et al. (2000)[9] ordonnent les observations \mathbf{x} de D selon la distance : $\text{Ddist}_k(x)$ de leur k -ième voisin. Les plus grandes valeurs désignent les observations les plus atypiques. Knorr et al. 2000)[8] proposent une autre approche en considérant atypique une observation x si un grand pourcentage, à fixer, des observations y de D est à une distance $d(x, y)$ plus grande qu'une borne minimale également à fixer. Les auteurs se focalisent également sur la complexité des algorithmes proposés.

LOF De très nombreux aménagements ont été proposés à ces versions de base notamment pour stabiliser les résultats ou pour réduire la sensibilité à des situations singulières comme des mélanges de distributions présentant des niveaux de densité très différents. La version la plus populaire est le LOF (*local outlier factor*; Breunig et al. 2000)[3] basé sur des notions proches de l'algorithme DBSCAN (*density-based spatial clustering of applications with noise*; Ester et al. 1996)[6] de *classification non supervisée* sans pour autant chercher des classes.

Breuning et al. (2000)[3] commencent par définir deux quantités. La k -distance, notée Dist_k , plus complexe que celle ci-dessus afin de prendre en compte les possibles équidistances entre les observations.

$\text{Dist}_k(x)$ est égale à $d(x, y)$ pour une observation y de sorte que pour au moins k observations y' de D : $d(x, y') \leq d(x, y)$ et que pour au plus $k - 1$ observations y' de D : $d(x, y') < d(x, y)$.

$V_k(x)$, le voisinage de k -distance de x est l'ensemble des observations vérifiant :

$$V_k(x) = \{x \in D \mid d(x, y) \leq \text{Dist}_k(x)\}.$$

Du fait de possibles équidistances entre les observations, le cardinal de $V_k(x)$ peut être plus grand que k .

La distance d'atteignabilité (*reachability distance*) entre deux observations est définie ci-dessous. Attention, malgré une appellation communément admise, cette quantité n'est pas une *distance* car elle n'est pas symétrique.

$$\text{RDist}_k(x, y) = \max \text{Dist}_k(y), d(x, y).$$

$\text{RDist}_k(x)$ est la distance $d(x, y)$ si y est assez éloigné de x mais, si x est dans le k -voisinage de y , $\text{RDist}_k(x)$ est minoré par $\text{Dist}_k(y)$. Les observations du k -voisinage de y sont considérées équidistantes afin d'apporter une forme de lissage, contrôlé par le paramètre k , à la conception des critères.

La densité locale d'atteignabilité (*local reachability density* en x) est ensuite définie par l'inverse de la moyenne de la distance d'atteignabilité dans le k -voisinage de x :

$$\text{LRDens}(x) = 1 / \left(\frac{\sum_{y \in V_k(x)} \text{RDist}_k(x, y)}{\text{card}(V_k(x))} \right).$$

Finalement :

$$\text{LOF}_k(x) = \frac{\sum_{y \in V_k(x)} \text{LRDens}(y)}{\text{card}(V_k(x))}$$

La valeur du LOF est difficile voire impossible à interpréter dans l'absolu. Une valeur de 1 correspond à une observation dans la norme de la distribution, mais une borne au delà de laquelle une observation est atypique n'est pas explicite, cela dépend du contexte et des dispersions relatives.

Motivées par les critiques, des variantes sont proposées afin d'y remédier. *LoOP* (*Local Outlier Probability*) tente d'être moins sensible au choix

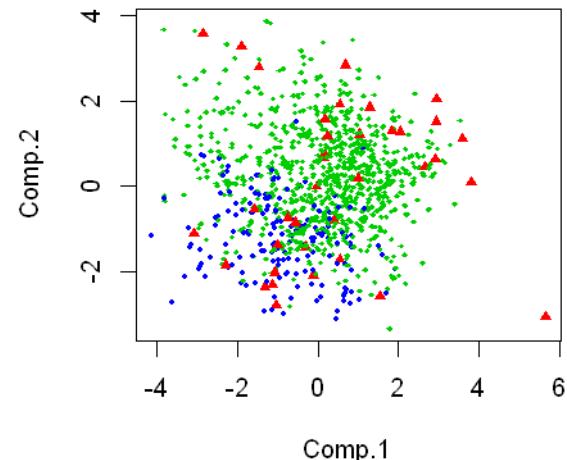


FIGURE 4 – Ozone : Anomalies au sens du LOF (Local Outlier Factor).

de k et est normalisé dans l'intervalle $[0, 1]$. L'*Interpreting and Unifying Outlier Scores* est présentée comme une amélioration du précédent. D'autres approches tentent une démarche similaire à l'agrégation de modèles : échantillons bootstrap et *bagging* des critères, combinaison de critères différents. Enfin, le *Global-Local Outlier Score from Hierarchies (GLOSH)* (Campello et al. 2015)[4] est basé sur une version hiérarchique de l'algorithme DBSCAN plutôt que sur DBSCAN comme le LOF.

Comme souvent, il faudra un peu de temps et d'expérimentations pour qu'une sélection naturelle opère entre toutes les approches publiées et retienne le critère garantissant un meilleur compromis entre pertinence des résultats et complexité algorithmique.

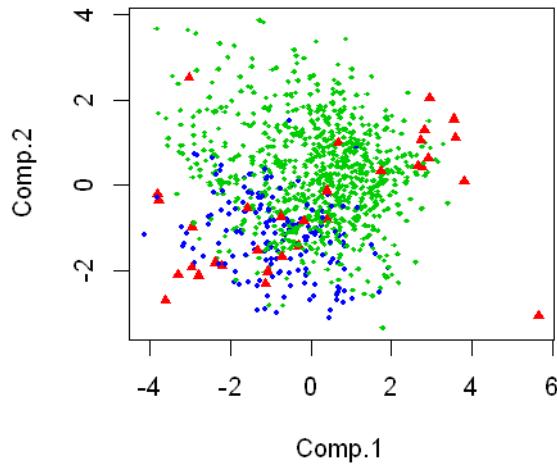


FIGURE 5 – Ozone : Anomalies au sens du GLOSH (Global-Local Outlier Score from Hierarchies).

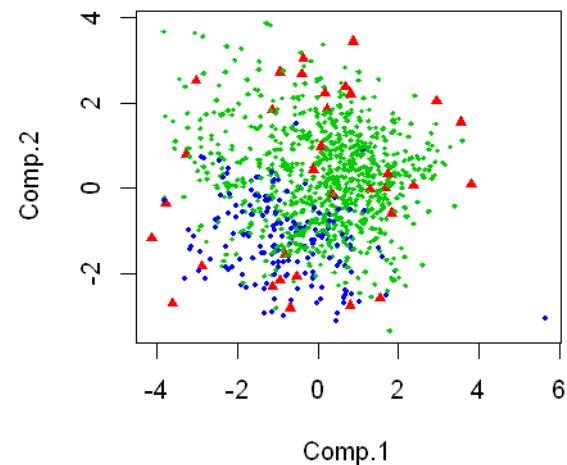


FIGURE 6 – Ozone : Anomalies au sens de OCC SVM (One Class Classification SVM).

OCC SVM

Une autre approche cherche une enveloppe, ou un support des observations jugées normales, définie par des séparateurs à vaste marge (*One Class Classification SVM*) (Schölkopf et al. 1999)[11]. Le principe consiste à poser le problème d'optimisation des SVM avec pour objectif de séparer les données, toutes les observations, de l'origine, dans l'espace de représentation (*feature space*) en maximisant la marge, à savoir la distance entre l'hyperplan et l'origine. La solution produit une fonction binaire qui vaut +1 dans la plus petite région captant les données et -1 ailleurs. Le paramètre de pénalisation à optimiser établit un équilibre entre la régularité de la frontière et la proportion d'observations considérées comme atypiques.

OCC RF

Enfin, la version de *random forest* (Breiman, 2001) pour la classification non supervisée est adaptée de façon très spécifique à l'objectif visé de détection d'anomalies.

Forêt aléatoire non supervisée La version non-supervisée des forêts aléatoires est une application de la notion de proximité entre les observations. Par défaut, lorsqu'aucune variable explicative ou cible n'est fournie à l'algorithme, celui-ci génère deux classes d'observations. La première désigne les observations initiales, la deuxième est obtenue par *permutation aléatoire des valeurs de chaque colonne*. Chaque colonne ou variable possède les mêmes propriétés de centrage et dispersion mais, dans ce deuxième jeu de données, la structure de corrélation ou de liaison entre les variables est évacuée.

La première classe, données initiales sont les observations normales, la deuxième classe constitue un ensemble d'observations synthétiques d'atypiques ou anomalies par rapport à la distribution des données initiales.

À l'issue de ces simulations, un forêt est apprise sur ces données en cherchant à ajuster au mieux la variable classe ainsi construite. Il en découle comme précédemment la construction d'une mesure de proximité, donc de distance, entre les observations deux à deux. La matrice de distance obtenue est enfin utilisée dans un algorithme de [classification ascendante hiérarchique](#) pour l'approche non supervisée issue d'une forêt aléatoire et de [positionnement multidimensionnel](#) pour une représentation plane de ces distances.

Anomalies selon les forêts aléatoires Cette même matrice de proximités entre les observations deux à deux est utilisée pour construire le score d'anomalie de chaque observation comme dans la section 3.1. Ce score indique donc pour chaque observation sa plus ou moins grande proximité avec toutes les autres observations de la classe normale.

Comme précédemment, un seuil arbitraire est choisi et les observations atypiques sont projetées dans le premier plan factoriel de l'ACP. Avec des résultats très différents de ceux de la section 3.1.

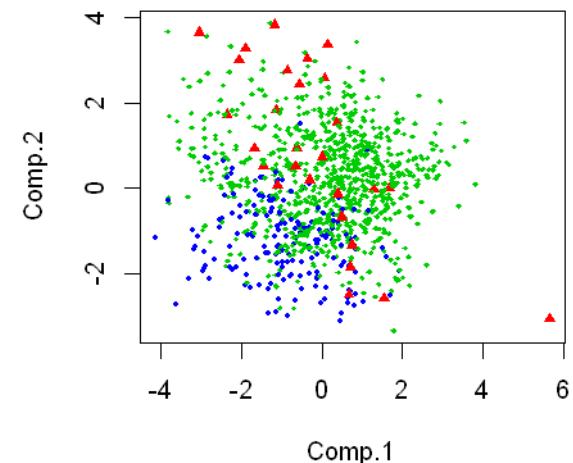


FIGURE 7 – Ozone : observations atypiques (en rouge) au sens du critère issue d'une forêt aléatoire non supervisée c'est-à-dire sans chercher à modéliser le dépassement de seuil.

4 Conclusion

La détection d'anomalies est un problème complexe sans solution uniformément meilleure car le choix de la méthode à utiliser dépend largement du contexte, des propriétés des variables et données observées, ainsi que de l'objectif poursuivi. La recherche d'anomalies sur les données de prévision de dépassement de seuil d'ozone montre, sur cet exemple, que chaque méthode projette sa conception de ce qu'est une anomalie. Néanmoins, il est probable et rassurant que les différentes méthodes vont s'accorder sur la détection d'observations très atypiques pas ou peu présentes dans les données de concentration d'ozone. D'autre part, cf. Aggarwal (2017)[1] pour une revue, de nouvelles approches cherchent à conjuguer ou agréger plusieurs méthodes de détection d'anomalies pour conduire à des résultats plus robustes comme en apprentissage avec le *bagging* ou le *boosting*.

Naturellement, la différence est encore plus marquée entre les deux types d'anomalies détectées par les forêts aléatoires ; anomalies par rapport à un modèle expliquant le dépassement ou par rapport à la distribution globales des observations.

En conséquence, même sans mettre en œuvre une approche supervisée, il est important de disposer d'un historique, sinon de simulations, identifiant des anomalies afin de pouvoir rétrospectivement évaluer l'efficacité de la ou des méthodes tout en optimisant la valeur du paramètre de sensibilité. Que l'approche soit paramétrique ou non, ce paramètre est toujours présent.

Cette très courte présentation n'aborde pas les problèmes plus complexes pouvant émerger. La prise en compte de signaux, courbes, images (*autoencoder et deep learning*), chemins sur un graphe, nécessite d'adapter ou sélectionner la bonne base de représentation (Fourier, splines, ondelettes) ou encore la bonne distance entre les observations concernées, avant de mettre en œuvre l'une des méthodes ci-dessus. Un traitement en ligne des données nécessitent la mise en place de décision séquentielle ou adaptative ; autant de sujets de recherche en cours.

Références

Références

- [1] Charu C. Aggarwal, *Outlier Analysis*, Springer Publishing Company, Incorporated, 2013, ISBN 1461463955, 9781461463955.
- [2] L. Breiman, *Random forests*, Machine Learning **45** (2001), 5–32.
- [3] Markus Breunig, Hans Peter Kriegel, Raymond T. Ng et Jörg Sander, *LOF : Identifying Density-Based Local Outliers*, PROCEEDINGS OF THE 2000 ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, ACM, 2000, p. 93–104.
- [4] Ricardo J. G. B. Campello, Davoud Moulavi, Arthur Zimek et Jörg Sander, *Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection*, ACM Trans. Knowl. Discov. Data **10** (2015), n° 1, 5 :1–5 :51.
- [5] Henri Caussinus et Anne Ruiz-Gazen, *Classification and generalized principal component analysis*, Selected contributions in data analysis and classification (Brito, Bertrand, Cucumel et Carvalho, réds.), Stud. Classification Data Anal. Knowledge Organ., Springer, 2007, p. 539–548, <https://hal.archives-ouvertes.fr/hal-00635541>.
- [6] Martin Ester, Hans Peter Kriegel, Jörg Sander et Xiaowei Xu, *A density-based algorithm for discovering clusters in large spatial databases with noise*, AAAI Press, 1996, p. 226–231.
- [7] Harold Hotelling, *The Generalization of Student's Ratio*, Ann. Math. Statist. **2** (1931), n° 3, 360–378, <https://doi.org/10.1214/aoms/1177732979>.
- [8] Edwin M. Knorr, Raymond T. Ng et Vladimir Tucakov, *Distance-based Outliers : Algorithms and Applications*, The VLDB Journal **8** (2000), n° 3-4, 237–253, ISSN 1066-8888, <http://dx.doi.org/10.1007/s007780050006>.
- [9] Sridhar Ramaswamy, Rajeev Rastogi et Kyuseok Shim, *Efficient Algorithms for Mining Outliers from Large Data Sets*, Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (New York, NY, USA), SIGMOD '00, ACM, 2000, p. 427–438, ISBN 1-58113-217-4, <http://doi.acm.org/10.1145/342009.335437>.

- [10] Bernard Rosner, *Percentage Points for a Generalized ESD Many-Outlier Procedure*, Technometrics **25** (1983), n° 2, 165–172.
- [11] Bernhard Schölkopf, Robert C. Williamson, Alex J. Smola, John Shawe-Taylor et John C. Platt, *Support Vector Method for Novelty Detection*, Advances in Neural Information Processing Systems 12 (S. A. Solla, T. K. Leen et K. Müller, réds.), MIT Press, 2000, p. 582–588, <http://papers.nips.cc/paper/1723-support-vector-method-for-novelty-detection.pdf>.

En guise de conclusion

Résumé

Résumer les grandes lignes de ce cours dans une vue synthétique : méthodes et stratégies dans l'objectif d'une comparaison globale des méthodes sur les différents jeux de données (cancer, pollution, carte visa). Il évoque enfin les pièges fréquents de telles démarches et revient sur la place du statisticien ou data scientist.

[Retour au plan du cours](#)

1 Stratégies du data mining

Les chapitres précédents décrivent les outils de base du prospecteur ou scientifique des données tandis que les logiciels en proposent une intégration plus ou moins complète, plus ou moins conviviale de mise en œuvre. En pratique, l'enchaînement de ces techniques permet la mise en place de *stratégies de fouille* bien définies. Celles-ci dépendent essentiellement des *types* de variables considérés et des *objectifs* poursuivis.

Types de variables

Explicatives L'ensemble des p variables explicatives ou prédictives est noté X , il est constitué de variables

- $X_{\mathbb{R}}$ toutes quantitatives¹,
- X_E toutes qualitatives,
- $X_{\mathbb{RUE}}$ un mélange de qualitatives et quantitatives.

À expliquer La variable à expliquer ou à prédire ou *cible* (target) peut être

- Y quantitative,
- Z qualitative à 2 modalités,
- T qualitative.

1. Une variable explicative qualitative à 2 modalités (0,1) peut être considérée comme quantitative ; c'est l'indicatrice des modalités.

Objectifs

Trois objectifs principaux sont poursuivis dans les applications classiques de fouille / science des données :

1. **Exploration multidimensionnelle** ou réduction de dimension : production de graphes, d'un sous-ensemble de variables représentatives X_r , d'un ensemble de composantes C_q préalables à une autre technique.
2. **Classification** (clustering) ou segmentation : production d'une variable qualitative T_r .
3. **Modélisation (Y ou Z)/Discrimination (Z ou T)** production d'un modèle de prévision de Y (resp. Z, T).

D'autres méthodes plus spécifiques à certains objectifs peuvent apparaître : détection d'atypiques ou d'anomalies, imputation, préalables ou non à la modélisation.

Outils

Les méthodes utilisables se classent en fonction de leur objectif et des types de variables prédictives et cibles.

Exploration

ACP $X_{\mathbb{R}}$ et \emptyset
 AFCM X_E et \emptyset
 AFD $X_{\mathbb{R}}$ et T

Classification

CAH $X_{\mathbb{R}}$ et \emptyset
 NuéeDyn $X_{\mathbb{R}}$ et \emptyset
 ...

Modélisation

1. Modèle linéaire généralisé
 RLM $X_{\mathbb{R}}$ et Y
 ANOVA X_E et Y
 ACOVA $X_{\mathbb{RUE}}$ et Y

- Rlogi X_{RUE} et Z
- Lglin X_T et T
- 2. Analyse discriminante
ADpar/nopar X_R et T
- 3. Classification and regression Tree
ArbReg X_{RUE} et Y
ArbCla X_{RUE} et T
- 4. Réseaux neuronaux
percep X_{RUE} et Y ou T
- 5. Agrégation de modèles
Bagging X_{RUE} et Y ou T
RandFor X_{RUE} et Y ou T
Boosting X_{RUE} et Y ou T
- 6. Support Vector Machine
SVM-R X_{RUE} et Y
SVM-C X_{RUE} et T

Stratégies

Les stratégies classiques de la fouille de données consistent à enchaîner les étapes suivantes :

1. **Extraction** de l'entrepôt des données éventuellement par sondage pour renforcer l'effort sur la qualité des données plutôt que sur la quantité.
2. **Exploration**
 - Tri à plat, et étude des distributions : transformation, recodage éventuel des variables quantitatives, regroupement de modalités des variables qualitatives, élimination de variables (trop de données manquantes, quasi constantes redondantes...). Gestion des données manquantes et valeurs atypiques.
 - Étude bivariée, recherche d'éventuelles relations non linéaires, de variables redondantes, d'incohérences.
 - Étude multivariée, représentations en dimension réduite (ACP, AFCM) et classification non-supervisée par classification ascendante hiérarchique (CAH), ou *kmeans* ou stratégie mixte.
3. **Apprentissage** : régression ou discrimination (classification supervisée).
 - Itérer les étapes suivantes :

- (a) Extraction d'un échantillon *test*,
- (b) Estimation, optimisation (validation croisée) des modèles pour chacune des méthodes utilisables.
- (c) Prévision de l'échantillon test.
- Comparer les distributions et moyennes des erreurs de prévision, éventuellement les courbes ROC.
- Choisir une méthode et le modèle associé de complexité "optimale" et le ré-estimer sur l'ensemble de l'échantillon.
- 4. **Exploitation** du modèle sur l'ensemble des données et diffusion des résultats.

2 Comparaison des résultats

La procédure décrite ci-dessus a été systématiquement mise en œuvre en automatisant dans R ou Python l'extraction aléatoire d'un échantillon test et les estimations, optimisations des différents modèles. Les codes sont disponibles sous forme de scénarios sur le site [wikiwat](#). La librairie *caret* (Kuhn, 2008)[1] et celle *Scikit-learn* de Python s'avèrent très efficaces pour mettre en œuvre cette démarche. L'optimisation des paramètres est réalisée par validation croisée.

Chaque échantillon test fournit donc une estimation sans biais de l'erreur de prévision. La distribution de ces erreurs est alors représentée par des diagrammes en boîtes. En discrimination binaire, des courbes ROC complètent les résultats. Les figures suivantes synthétisent les résultats pour les données de cancer du sein, de chromatographie NIR (cookies), de prévision du pic d'ozone et enfin bancaires (appétence carte visa premier). d'autres exemples sont traitées sur le dépôt [wikiwat](#).

3 Pièges

Les principaux pièges qui peuvent être rencontrés au cours d'une prospection peuvent être le résultat d'un *acharnement* en quête de sens (*data snooping*). Cela signifie qu'à force de creuser, contrairement à un prospecteur minier à la recherche de diamants bien réels, le prospecteur en données disposant d'un grand nombre de variables finit bien, en mode exploratoire, par trouver des relations semblant hautement significatives. Par exemple, au seuil classique, 5% des tests sont, à tort, significatifs et conduisent à des "faux positifs" ou des fausses corrélations. Il suffit donc d'en faire beaucoup, de croiser beaucoup de variables, pour nécessairement trouver du "sens" dans des données. Encore une fois, il est préférable d'éviter le fonctionnement "Shadock" (cf. figure 10) : *je n'ai qu'une chance sur un milliard de réussir; je me dépêche donc de rater le plus d'essais possibles*.

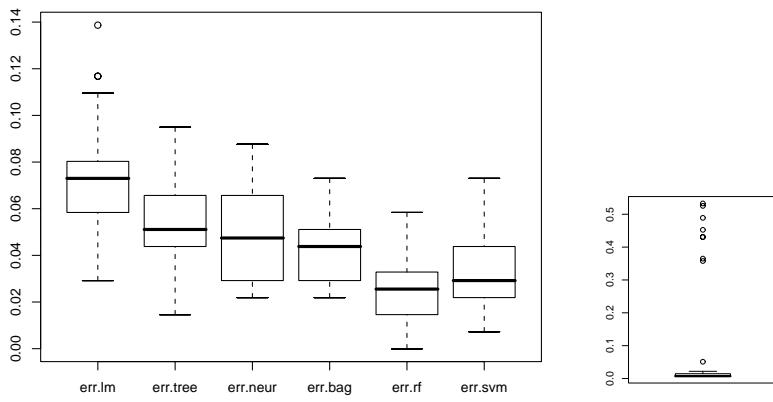


FIGURE 1 – Cancer : Diagrammes boîtes des taux d’erreurs. Le boosting est mis de côté pour des problèmes d’échelle et de comportement erratique provenant d’une focalisation extrême sur une observation imprévisible.

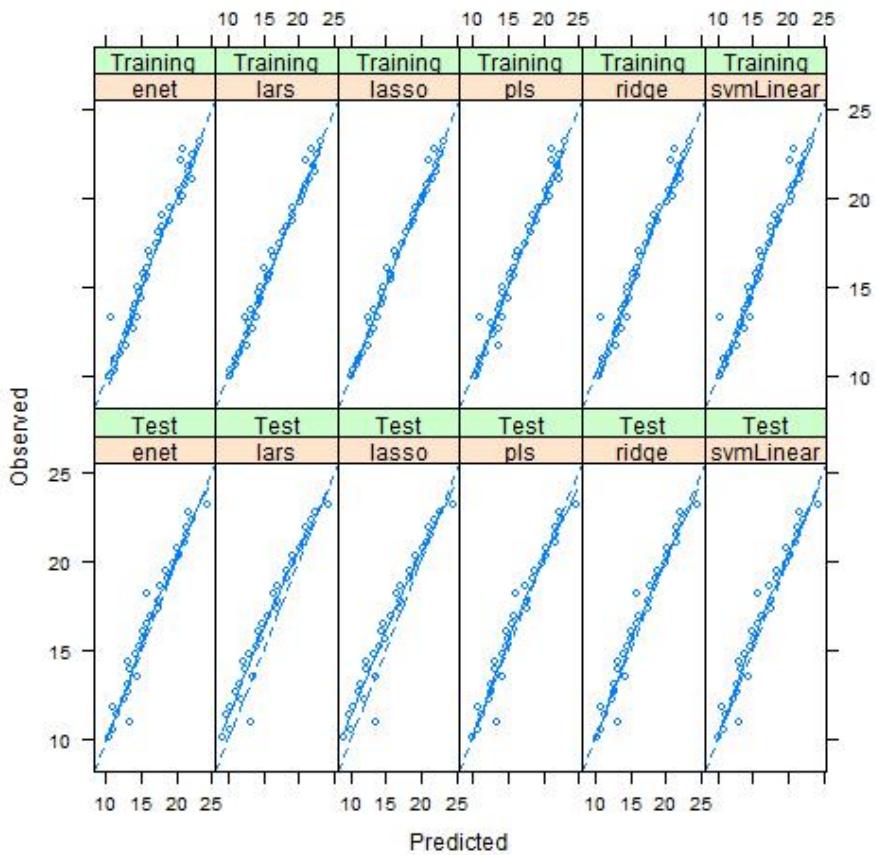


FIGURE 2 – Cookies : Résidus (apprentissage et test) des différents modèles mettant en évidence la forte linéarité des données ainsi que les aspects volontairement atypiques de l’échantillon test original.

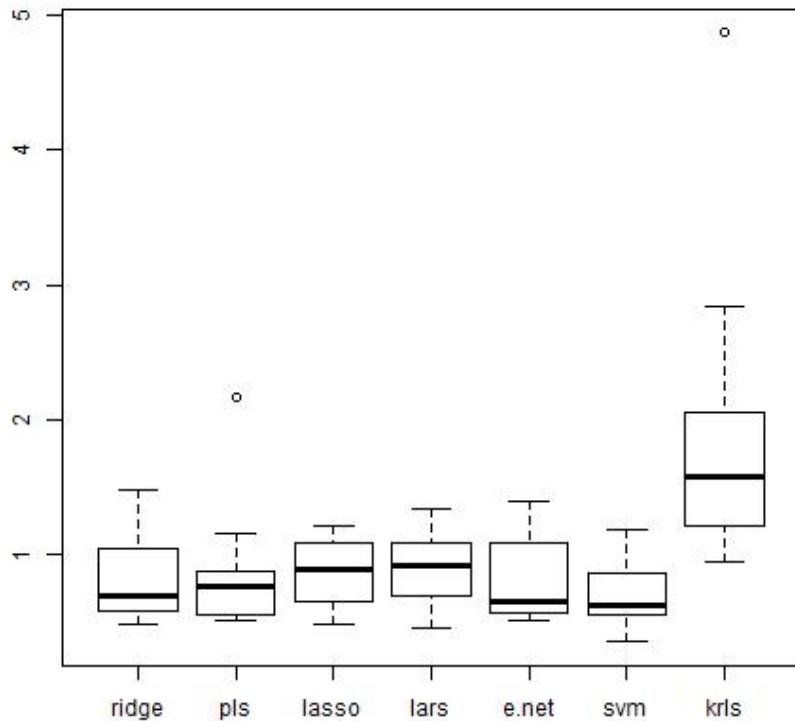


FIGURE 3 – Cookies : Diagrammes boîtes très proches des méthodes linéaires alors que les méthodes non-linéaires ne sont pas retenues car inefficaces. Les SVM (noyau linéaire) conduisent à la meilleure moyenne (0.70) devant la régression ridge (0.84), elastic net (0.85), lasso, PLS (0.86)

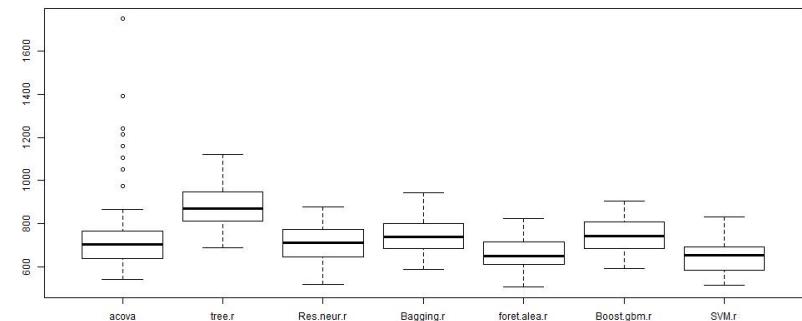


FIGURE 4 – Ozone : Diagrammes boîtes des taux d'erreurs en régression. Meilleur comportement des SVM avec noyau linéaire (649) devant random forest (666). L'analyse de covariance quadratique conduit à une moyenne élevée (774) mais reste utile pour l'interprétation.

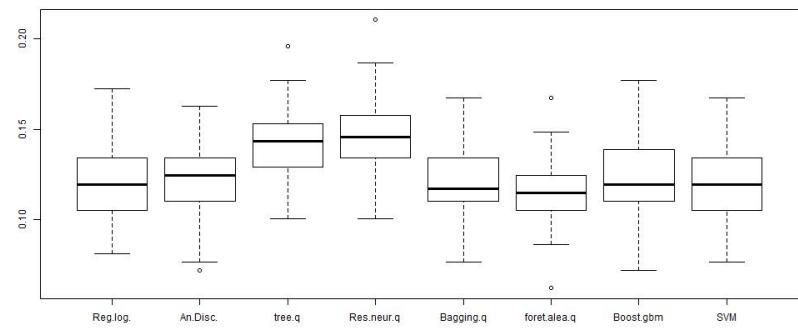


FIGURE 5 – Ozone : Diagrammes boîtes des taux d'erreurs pour la prévision des dépassements de seuil. En moyenne, les deux stratégies (prévision en régression ou directement du dépassement) sont finalement équivalentes pour les meilleures méthodes. Les moyennes se répartissent entre 11 % (random forest) et 14%.

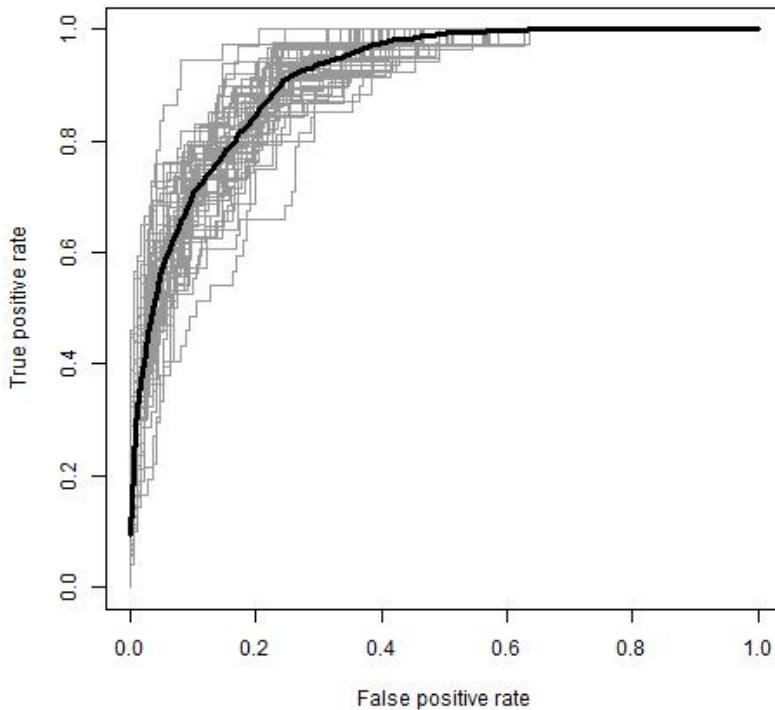


FIGURE 6 – Ozone : Attention, l'échantillon test est petit et les courbes ROC sont fortement dispersées. Il est important d'en calculer une moyenne sur les 50 échantillons tests.

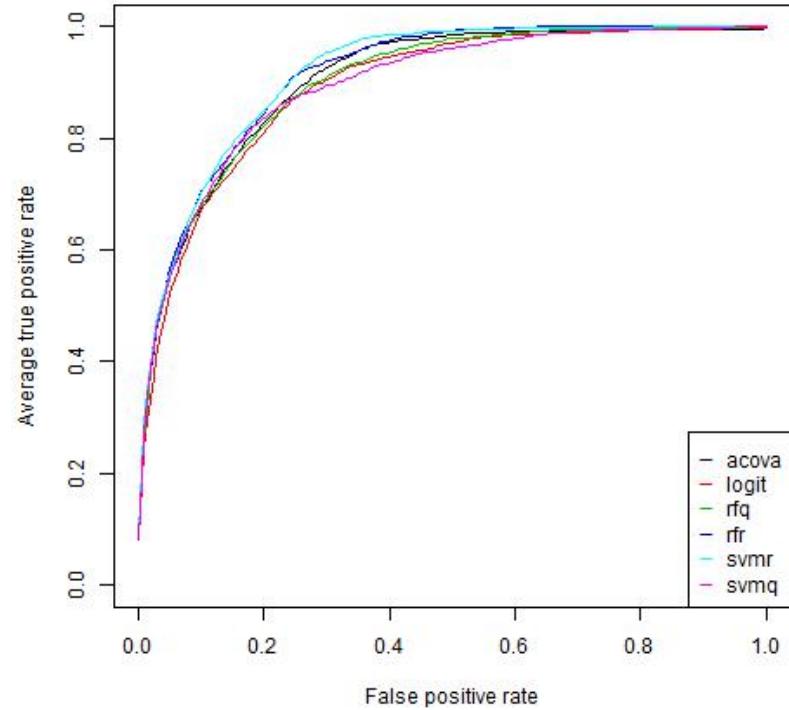


FIGURE 7 – Ozone : Les courbes ROC moyennes, qui permettraient de déterminer un seuil de déclenchement d'alerte, soulignent les meilleures comportements des SVM et de Random forest après régression.

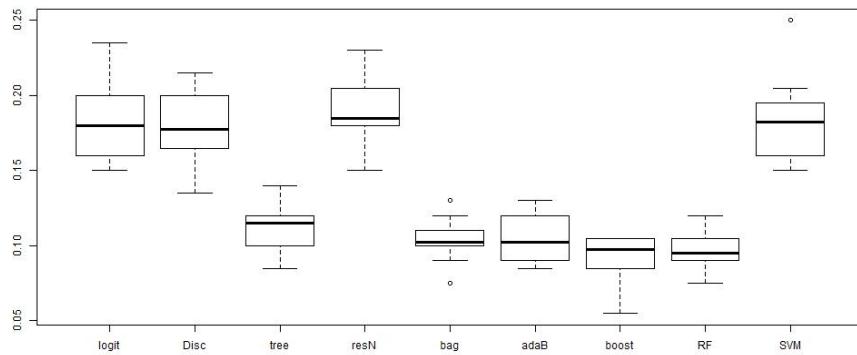


FIGURE 8 – Banque : Diagrammes boîtes des taux d’erreurs. En moyenne, les méthodes basées sur des arbres l’emportent nettement avec celle d’agrégation de modèles (boosting 9%, ranfom forest et bagging 10 %) devant un arbre seul (11 %) très utile pour l’interprétation.

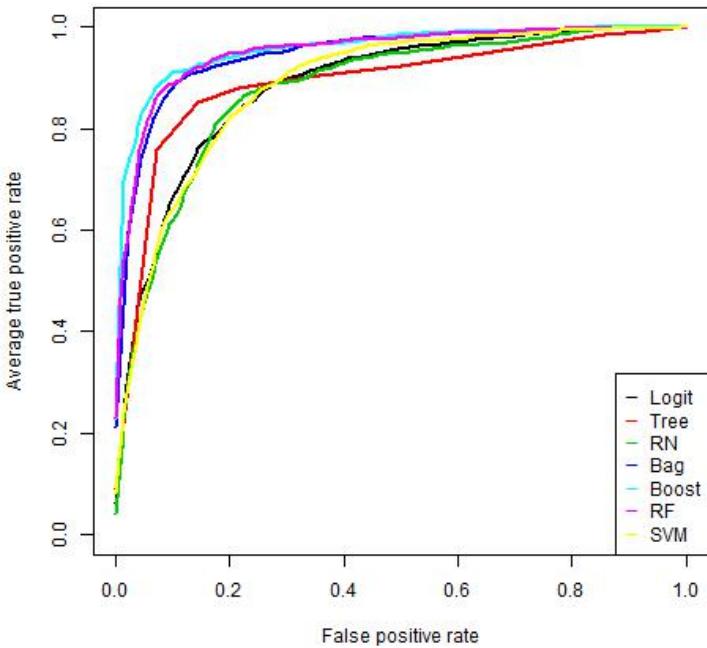


FIGURE 9 – banque : Les courbes ROC moyennes insistent sur le très bon comportement des agrégations de modèles (boosting, random forest, bagging) pour une très grande variété de choix de seuils contrairement à un arbre de discrimination dont les qualité se détériorent pour des seuils faibles.

En phase de modélisation, une sur-paramétrisation ou un sur-ajustement du modèle peut parfaitement expliquer des données sans pour autant que les résultats soient extrapolables ou généralisables à d'autres données que celles étudiées. Les résultats de prévision seront donc entachés d'une forte erreur relative liée à la variance des estimations des paramètres. C'est toujours le problème de trouver un bon compromis entre le biais d'un modèle plus ou moins faux et la variance des estimateurs. Nous insistons donc sur les indispensables phases de choix de modèles et comparaison des méthodes.

4 Rôle du statisticien

4.1 Des compétences multiples

Une bonne pratique du *Data Mining* nécessite de savoir articuler toutes les méthodes entrevues dans ce document. Rude tâche, qui ne peut être entreprise qu'à la condition d'avoir très bien spécifié les objectifs et buts de l'étude. On peut noter que certaines méthodes poursuivent les mêmes objectifs prédictifs. Dans les bons cas, données bien structurées, elles fourniront des résultats très similaires, dans d'autres, une méthode peut se révéler plus efficace compte tenu de la taille de l'échantillon ou géométriquement mieux adaptée à la topologie des groupes à discriminer ou encore en meilleure interaction avec les types des variables. Ainsi, il peut être important et efficace de découper en classes des variables prédictives quantitatives afin d'approcher de façon sommaire une version *non-linéaire* du modèle par une combinaison de variables indicatrices. Cet aspect est par exemple important en régression logistique ou avec un perceptron mais inutile avec des arbres de décisions qui intègrent ce découpage en classes dans la construction du modèle (seuils optimaux). D'autre part, les méthodes ne présentent pas toutes les mêmes facilités d'interprétation. Il n'y a pas de meilleur choix *a priori*, seule l'expérience et un protocole de *test* soigné permettent de se déterminer. C'est la raison pour laquelle la librairie *caret* de R ne font pas de choix et offrent ces méthodes en parallèle pour mieux s'adapter aux données, aux habitudes de chaque utilisateur.

La librairie *caretEnsemble* comme les fonctionnalités *pipeline* de *Scikit-learn* permettent également de combiner des méthodes dans une architecture complexe telle qu'elles se retrouvent généralement dans les solutions gagnantes des concours *Kaggle*. Un ensemble de modèles construisent des prévisions ou variables qui sont ajoutées comme nouvelles variables à l'entrée d'autres algorithmes. C'est également une des explications du succès de l'apprentissage profond..



FIGURE 10 – Shadoks : Tant qu'à pomper, autant que cela serve à quelque chose !

4.2 De l'utilité du statisticien

Le travail demandé déborde souvent du rôle d'un statisticien car la masse et la complexité des données peuvent nécessiter le développement d'interfaces et d'outils graphiques sophistiqués permettant un accès aisés aux données, comme à des résultats, par l'utilisateur finale à l'aide par exemple d'un simple navigateur sur l'intranet de l'entreprise. Néanmoins, au delà de ces aspects plus "informatiques", l'objectif principal reste une "quête de sens" en vue de faciliter les prises de décision tout en préservant la fiabilité. Ainsi, la présence ou le contrôle d'une expertise statistique reste incontournable car la méconnaissance des limites et pièges des méthodes employées peut conduire à des aberrations discréditant la démarche et rendant caducs les investissements consentis.

4.3 Vers le Big Data

Le volume des données générées et stockées par les entreprises industrielles et celles du e-commerce font franchir une nouvelle étape. Nous passons du TéraOctet au PétaOctet. Comme expliqué rapidement en introduction, cette nouvelle étape engendre de nouvelles approches tant pour les architectures des bases de données, la parallélisation des calculs, que pour les algorithmes et méthodes mises en œuvre.

D'un point de vue informatique, une connaissance du nouveau standard *Hadoop*² est vivement souhaitée. Il permet la création d'applications distribuées et "échelonnables" (*scalables*) sur des milliers de nœuds pour gérer des pétaoctets de données. Le principe est de découper et paralléliser (distribution) des tâches en lots de données afin de réduire linéairement le temps (scalable) de calcul en fonction du nombre de nœuds. *Hadoop* devient l'outil de référence du web mining et l'e-commerce.

D'un point de vue statistique / mathématique, le nouveau défi est la construction de bases de représentation fonctionnelle et de modèles pertinents pour aborder et prendre en compte des structures de données complexes : géolocalisation sur des graphes, signaux en temps réels, images 3D, séquences... Chaque problème, surtout industriel, nécessite une approche spécifique issue d'une recherche originale dans le cadre souvent d'une thèse, par exemple CIFRE, qu'un d'un développement d'ingénierie classique. Dans le cas de flots de données, l'aide à la décision devient adaptative ou séquentielle.

Références

- [1] Max Kuhn, *Building Predictive Models in R Using the caret Package*, Journal of Statistical Software **28** (2008), n° 5.

2. Crée en 2009 et développé en Java par Doug Cutting au sein des projets de la fondation des logiciels libres Apache. Il est inspiré des principes de MapReduce de Google.

Introduction au bootstrap

Résumé

Présentation succincte du principe du bootstrap.

Retour au [plan du cours](#)

1 Introduction

La motivation du *bootstrap*¹ (Efron, 1982 ; Efron et Tibshirani, 1993) est d'approcher par simulation (*Monte Carlo*) la distribution d'un estimateur lorsque l'on ne connaît pas la loi de l'échantillon ou, plus souvent lorsque l'on ne peut pas supposer qu'elle est gaussienne. L'objectif est de remplacer des hypothèses probabilistes pas toujours vérifiées ou même invérifiables par des simulations et donc beaucoup de calcul.

Le principe fondamental de cette technique de ré-échantillonnage est de substituer à la distribution de probabilité inconnue F , dont est issu l'échantillon d'apprentissage, la distribution empirique \widehat{F} qui donne un poids $1/n$ à chaque réalisation. Ainsi on obtient un échantillon de taille n dit *échantillon bootstrap* selon la distribution empirique \widehat{F} par n tirages aléatoires avec remise parmi les n observations initiales.

Il est facile de construire un grand nombre d'échantillons bootstrap sur lesquels calculer l'estimateur concerné. La loi simulée de cet estimateur est une approximation asymptotiquement convergente sous des hypothèses raisonnables² de la loi de l'estimateur. Cette approximation fournit ainsi des estimations du biais, de la variance, donc d'un risque quadratique, et même des intervalles de confiance de l'estimateur sans hypothèse (normalité) sur la vraie loi.

1. Cette appellation est inspirée du baron de Münchhausen (Rudolph Erich Raspe) qui se sortit de sables mouvants par traction sur ses *tirants de bottes*. En France “bootstrap” est parfois traduit par *à la Cyrano* (acte III, scène 13) en référence à ce héros qui prévoyait d'atteindre la lune en se plaçant sur une plaque de fer et en itérant le jet d'un aimant.

2. Échantillon indépendant de même loi et estimateur indépendant de l'ordre des observations.

1.1 Principe du *plug-in*

Soit $\mathbf{x} = \{x_1, \dots, x_n\}$ un échantillon de taille n issue d'une loi inconnue F sur (Ω, \mathcal{A}) . On appelle *loi empirique* \widehat{F} la loi discrète des singuliers (x_1, \dots, x_n) affectés des poids $1/n$:

$$\widehat{F} = \sum_{i=1}^n \delta_{x_i}.$$

Soit $A \in \mathcal{A}$, $P_F(A)$ est estimée par :

$$\widehat{(P)}_F(A) = P_{\widehat{F}}(A) = \sum_{i=1}^n \delta_{x_i}(A) = \frac{1}{n} \text{Card } x_i \in A.$$

De manière plus générale, soit θ un paramètre dont on suppose que c'est une fonction de la loi F . on écrit donc $\theta = t(F)$. Par exemple, $\mu = E(F)$ est un paramètre de F suivant ce modèle. Une *statistique* est une fonction (mesurable) de l'échantillon. Avec le même exemple :

$$\widehat{\mu} = \bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

et \bar{x} est la statistique qui estime μ . On dit que c'est un estimateur “*plug-in*” et, plus généralement,

DÉFINITION 1. — *On appelle estimateur plug-in d'un paramètre θ de F , l'estimateur obtenu en remplaçant la loi F par la loi empirique :*

$$\widehat{\theta} = t(\widehat{F}).$$

comme dans le cas de l'estimation de μ : $\widehat{\mu} = E(\widehat{F}) = \bar{x}$.

1.2 Estimation de l'écart-type de la moyenne

Soit X une variable aléatoire réelle de loi F . On pose :

$$\mu_F = E_F(X), \quad \text{et} \quad \sigma_F^2 = \text{Var}_F(X) = E_F[(X - \mu_F)^2];$$

Ce qui s'écrit :

$$X \sim (\mu_F, \sigma_F^2).$$

Soit (X_1, \dots, X_n) n variables aléatoires i.i.d. suivant aussi la loi F . Posons $\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i$. Cette variable aléatoire a pour espérance μ_F et pour variance σ_F^2/n . On dit aussi que la statistique

$$\bar{X} \sim (\mu_F, \sigma_F^2/n).$$

Remarquons qu'en moyennant plusieurs valeurs ou observations, on réduit la variance inhérente à une observation. De plus, sous certaines conditions sur la loi F et comme résultat du théorème de la limite centrale, \bar{X} converge en loi vers la loi normale.

L'estimateur plug-in de σ_F est défini par :

$$\begin{aligned}\hat{\sigma}^2 &= \widehat{\sigma_F}^2 = \sigma_{\hat{F}}^2 = \text{Var}_{\hat{F}}(X) \\ &= E_{\hat{F}}[(X - E_{\hat{F}}(X))^2] = \frac{1}{n} \sum_{i=1}^n (X_i - \bar{X})^2.\end{aligned}$$

L'estimateur plug-in de σ_F est (légèrement) différent de celui du maximum de vraisemblance. L'estimateur plug-in est en général biaisé mais il a l'avantage d'être simple et de pouvoir s'appliquer à tout paramètre θ même lorsque l'on ne peut pas calculer la vraisemblance du modèle.

2 Estimation bootstrap d'un écart-type

Soit $\hat{\theta} = s(x)$ un estimateur quelconque (M.V. ou autre) de θ pour un échantillon x donné. On cherche à apprécier la précision de $\hat{\theta}$ et donc à estimer son écart-type.

2.1 Échantillon bootstrap

Avec les mêmes notations, \hat{F} est la distribution empirique d'un échantillon $x = \{x_1, \dots, x_n\}$.

DÉFINITION 2. — On appelle échantillon bootstrap de x un échantillon de taille n noté

$$x^* = \{x_1^*, \dots, x_n^*\}$$

suivant la loi \hat{F} ; x^* est un ré-échantillon de x avec remise.

2.2 Estimation d'un écart-type

DÉFINITION 3. — On appelle estimation bootstrap de l'écart-type $\widehat{\sigma}_F(\hat{\theta})$ de $\hat{\theta}$, son estimation plug-in : $\sigma_{\hat{F}}(\hat{\theta})$.

Mais, à part dans le cas très élémentaire où, comme dans l'exemple ci-dessus, θ est une moyenne, il n'y a pas de formule explicite de cet estimateur. Une approximation de l'estimateur bootstrap (ou plug-in) de l'écart-type de $\hat{\theta}$ est obtenue par une simulation (Monte-Carlo) décrite dans l'algorithme ci-dessous.

Pour un paramètre θ et un échantillon x donnés, on note $\hat{\theta} = s(x)$ l'estimation obtenue sur cet échantillon. Une réplication bootstrap de $\hat{\theta}$ est donnée par : $\hat{\theta}^* = s(x^*)$.

ALGORITHME 1 : Estimation de l'écart-type

Soit x un échantillon et θ un paramètre.

for $b = 1$ à B **do**

Sélectionner 1 échantillon bootstrap $x^{*b} = \{x_1^{*b}, \dots, x_n^{*b}\}$. par tirage avec remise dans x .

Estimer sur cet échantillon : $\hat{\theta}^*(b) = s(x^{*b})$.

end for

Calculer l'écart-type de l'échantillon ainsi construit :

$$\hat{\sigma}_B^2 = \frac{1}{B-1} \sum_{b=1}^B (\hat{\theta}^*(b) - \hat{\theta}^*(.))^2$$

$$\text{avec } \hat{\theta}^*(.) = \frac{1}{B} \sum_{b=1}^B (\hat{\theta}^*(b)).$$

$\hat{\sigma}_B$ est l'approximation bootstrap de l'estimation plug-in recherchée de l'écart-type de $\hat{\theta}$.

2.3 Estimation du biais

Avec les mêmes notations :

$$\theta = t(F) \quad \text{et} \quad \hat{\theta} = s(x),$$

le biais d'un estimateur s'exprime comme

$$\mathcal{B}_F(\hat{\theta}) = E_F[s(\mathbf{x})] - t(F).$$

Un estimateur est sans biais si $E[\hat{\theta}] = \theta$. Le biais est aussi une mesure de la précision d'un estimateur et on a vu que, généralement, les estimateurs plug-in étaient biaisés.

DÉFINITION 4. — *On appelle estimateur bootstrap du biais, l'estimateur plug-in :*

$$\widehat{\mathcal{B}}_F(\hat{\theta}) = \mathcal{B}_{\widehat{F}}(\hat{\theta}) = E_{\widehat{F}}[s(\mathbf{x}^*)] - t(\widehat{F}).$$

Comme pour l'écart-type, il n'existe généralement pas d'expression analytique et il faut avoir recours à une approximation par simulation.

ALGORITHME 2 : Estimation bootstrap du biais

Soit \mathbf{x} un échantillon et θ un paramètre.

for $b = 1$ à B **do**

Sélectionner 1 échantillon bootstrap $\mathbf{x}^{*b} = \{x_1^{*b}, \dots, x_n^{*b}\}$. par tirage avec remise dans \mathbf{x} .

Estimer sur cet échantillon la réplication bootstrap de $\hat{\theta}$: $\hat{\theta}^*(b) = s(\mathbf{x}^{*b})$.

end for

Approcher $E_{\widehat{F}}[s(\mathbf{x}^*)]$ par $\hat{\theta}^*(.) = \frac{1}{B} \sum_{b=1}^B (\hat{\theta}^*(b))$

L'approximation bootstrap du biais est : $\widehat{\mathcal{B}}_B(\hat{\theta}) = \hat{\theta}^*(.) - \hat{\theta}$.

3 Compléments

En résumé, on peut dire que le bootstrap repose sur une hypothèse très élémentaire : $\hat{\theta}^*$ se comporte par rapport à $\hat{\theta}$ comme $\hat{\theta}$ par rapport à θ . La connaissance de $\hat{\theta}^*$ (distribution, variance, biais...) renseigne alors sur celle de $\hat{\theta}$.

Beaucoup d'autres compléments sont à rechercher dans la littérature et en particulier dans Efron et Tibshirani (1993). Il est ainsi possible de définir des intervalles de confiance bootstrap en considérant la distribution et les quantiles de $\hat{\theta}^*$ ou même encore des tests à partir des versions bootstrap de leur statistique.

Le bootstrap rapidement décrit ici est dit "non-paramétrique" car la loi empirique \widehat{F} est une estimation non-paramétrique de F . Dans le cas où F serait connue à un paramètre près, il existe également une version dite *paramétrique* du bootstrap.

Pour des estimateurs plus compliqués (fonctionnels) comme dans le cas de la régression non-paramétrique par noyau ou spline, il est facile de construire graphiquement une enveloppe bootstrap de l'estimateur à partir de réplications de l'échantillon. Celle-ci fournit généralement une bonne appréciation de la qualité de l'estimateur obtenu. Attention, dans le cas de la régression il est en principe plus justifié de répliquer le tirage sur les *résidus* plutôt que sur les observations. Ce sont les résidus qui sont en effet supposés i.i.d. et qui vérifient donc les hypothèses nécessaires mais cette approche devient très sensible à l'hypothèse sur la validité du modèle. Il est finalement d'usage de considérer un échantillon bootstrap issu des données initiales (Efron et Tibshirani) :

$$\mathbf{z}^{*b} = \{(\mathbf{x}_1^{*b}, y_1^{*b}), \dots, (\mathbf{x}_n^{*b}, y_n^{*b})\};$$

c'est ce qui a été choisi dans ce document.

Enfin, l'estimation bootstrap est justifiée par des propriétés asymptotiques (convergence en loi) lorsque le nombre de réplications (B) croît conjointement avec la taille de l'échantillon (n). Comme la loi empirique \widehat{F} converge (en loi) vers celle théorique, la distribution du paramètre $\hat{\theta} = t(\widehat{F})$ converge (en loi) vers celle théorique de $\theta = t(F)$.