Apprentissage statistique et grande dimension

Fiche de TP n°1

1 Approximation gaussienne dans le modèle de régression

1. Simuler un n-échantillon $\xi_1, \ldots, \xi_n$ de loi normale de variance $\sigma^2 = 0.1$.
2. Soit $f(x) = \min\{x, 1-x\}$ pour $x \in [0,1]$. Soit $\{\varphi_1, \ldots, \varphi_M\}$ un dictionnaire (on choisira par exemple la base des fonctions constantes par morceaux définies par

$$\varphi_k(x) = M^{1/2}\mathbf{1}_{[(k-1)M^{-1}, kM^{-1})}(x), \quad k = 1, \ldots, M.$$

   Simuler le modèle d'observation (X_i, Y_i), $i = 1, \ldots, n$?

$$Y_i = f(X_i) + \xi_i,$$

   avec $X_i = i/n$. Représenter la fonction f et les données sur un même graphique (on pourra choisir n de l'ordre de 100 ou 1000 par exemple).
3. Calculer, pour $k = 1, ..., M$,

$$\theta_n(\varphi_k) = \frac{1}{n}\sum_{i=1}^{n} Y_i \varphi_k(X_i) \quad \text{et} \quad \theta(\varphi_k) = \int_0^1 f(x)\varphi_k(x)dx.$$

4. Construire et représenter graphiquement l'estimateur

$$\widehat{f}_{n,M}(x) = \sum_{k=1}^{M} \theta_n(\varphi_k)\varphi_k(x)$$

   pour différentes valeurs de M (par exemple M variant entre 5 et 50 pour n de l'ordre de grandeur 100 ou 1000).
5. Écrire une fonction qui calcule, pour un M fixé, le critère des moindres carrés

$$MSE(M) = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \widehat{f}_{n,M}(X_i)\right)^2$$

   Modifier ensuite cette fonction pour qu'elle retourne le valeur du critère pour un ensemble de valeurs possibles de M. Expliquer pourquoi la valeur de M qui minimise le critère MSE n'est pas un bon choix.
6. Écrire une fonction qui calcule, pour un M fixé, le critère de validation croisée

$$CV(M) = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \widehat{f}^{[-i]}_{n,M}(X_i)\right)^2$$

   avec $\widehat{f}^{[-i]}_{n,M}$ l'estimateur calculé à partir de l'échantillon $\{Y_j, j \neq i\}$. Modifier ensuite cette fonction pour qu'elle retourne le valeur du critère pour un ensemble de valeurs possibles de M. Choisir la valeur de M qui minimise le critère et tracer l'estimateur correspondant. Attention au temps de calcul ! Commencer par prendre un petit nombre de valeurs de M.

7. (Optionnel) Représenter (par approximation de Monte-Carlo) la fonction d'erreur $M \mapsto \mathbb{E}[\|\widehat{f}_{n,M} - f\|_n^2]$ et retrouver le phénomène d'équilibre biais-variance en jouant sur les paramètres n et M (avec les spécifications précédentes).
8. (Optionnel) Reprendre cette étude pour d'autres choix de fonctions f, par exemple $f(x) = \sin(10x)$. Étudier la précision de cette approximation en fonction de M et n. On choisit $f(x) = x(1-x)$ pour $x \in [0,1]$. Les résultats sont-ils modifiés ? On remplace la base des fonctions constantes par morceaux par la base trigonométrique. Qu'observe-t-on ?
9. (Optionnel) Reprendre l'étude pour des X_i indépendants et uniformes sur $[0,1]$.

2 Régression linéaire ridge et Lasso

L'objectif de cet exercice est de prédire le salaire d'un joueur de baseball en fonction de certaines statistiques associées à ses performances sur l'année.

1. Installer/charger les packages *glmnet* et *ISLR*. Charger le jeu de données *Hitters* dans R.
2. Explorer le jeu de données. Quelle est sa taille ? Quelles sont les variables ? Les individus ? Y'a t-il beaucoup de données manquantes ? À l'aide de la fonction *na.omit()*, extraire du jeu de données les individus pour lesquels certaines données sont manquantes.
3. La fonction *glmnet()* permet, entre autres, d'estimer les paramètres d'un modèle linéaire expliquant le salaire en fonction des autres prédicteurs en régularisant le critère des moindres carrés. Nous considérons ici des pénalités de type "ridge" ou Lasso.
   (a) Création des variables :

   ```
   x = model.matrix(Salary~., Hitters)[,-1]
   y = Hitters$Salary
   ```

   (b) Consulter l'aide de la fonction *glmnet()*. Pour estimer les paramètres du modèle pour différentes valeurs de *lambda* (prises sur une grille allant de lambda = 10^{10} à lambda = 10^{-2}), la syntaxe est

   ```
   grid = 10^seq(10, -2, length = 100)
   ridge.mod = glmnet(x, y, alpha=0, lambda = grid)
   ```

   Le paramètre alpha indique quelle méthode est utilisée pour calculer les coefficients (alpha = 0 pour la régression ridge, alpha = 1 pour le Lasso). La commande *coef(ridge.mod)* permet d'avoir accès aux coefficients ainsi calculés.
4. Séparer l'échantillon en deux parties : un échantillon d'apprentissage *x.train*, *y.train* qui comprend les observations d'environ 70% des individus et que nous utiliserons pour estimer les coefficients du modèle ; et un échantillon de test *x.test*, *y.test* qui comprend les observations des individus qui ne sont pas dans l'échantillon d'apprentissage et que nous utiliserons pour évaluer la qualité de l'estimation.
5. Estimer les paramètres du modèle en utilisant uniquement les données de l'échantillon d'apprentissage.
6. À partir de ces estimations, prédire y pour les individus de l'échantillon de test. On pourra utiliser la syntaxe suivante (qui donne les prédictions pour lambda = 4) :

   ```
   ridge.pred = predict(ridge.mod, s=4, newx= x.test)
   ```

   Essayer différentes valeurs de pour le paramètre lambda. Que remarquez-vous ?
7. Nous souhaitons choisir correctement le paramètre lambda. Séparer l'échantillon d'apprentissage en deux parties :

— Un premier échantillon comprenant environ 90% de l'échantillon d'apprentissage à partir duquel nous allons calculer l'estimateur pour différentes valeurs de lambda.
— Un second échantillon contenant le reste de l'échantillon d'apprentissage qui nous servira à évaluer les performances des différents modèles obtenus.

Tracer l'erreur quadratique moyenne en fonction de lambda (on fera varier lambda entre 10 et 500).

La fonction *cv.glmnet()* répète cette opération 10 fois en sélectionnant le premier sous-échantillon de façon à ce que chaque individu soit sélectionné une (et une seule) fois. À la fin de l'opération, nous obtenons une prédiction de y pour tous les individus de l'échantillon et pour tous les lambda de la grille et nous sélectionnons la valeur de lambda pour laquelle l'erreur quadratique moyenne est minimale. Exécuter les commandes suivantes :

```
cv.out = cv.glmnet(x.train, y.train, alpha=0)
plot(cv.out)
cv.out$lambda.min
```

Nous obtenons ainsi la valeur de lambda optimale selon le critère de validation croisée 10-fold [1].

8. Calculer l'erreur quadratique moyenne sur l'échantillon de test à partir des coefficients estimés sur l'échantillon d'apprentissage pour cette valeur de lambda. Tracer les salaires prédits en fonction des salaires observés. Que pensez-vous de la qualité de prédiction ? On pourra également étudier les résidus.
9. Maintenant que l'on a calculé la valeur optimale de lambda, estimer les coefficients du modèle en utilisant cette fois tout l'échantillon. Interpréter les coefficients obtenus.
10. Répéter la même opération avec l'estimateur Lasso. Comparer la qualité des prédictions des deux estimateurs sur l'échantillon de test (qui doit être le même pour les deux méthodes). Quel est l'avantage principal du Lasso par rapport à la régression ridge ?

3 Estimation de la variance

On considère le modèle de régression

$$Y_i = f(X_i) + \xi_i, \quad i = 1, \dots, n$$

où les ξ_i sont i.i.d. de variance commune σ^2 et $X_i = i/n$. On ne suppose pas σ^2 connu. Lorsque la fonction f (le paramètre inconnu) est lisse, l'estimateur de Rice défini par

$$\widehat{\sigma}_n^2 = \frac{1}{2(n-1)} \sum_{i=2}^{n} (Y_i - Y_{i-1})^2$$

est un candidat naturel pour l'estimation de σ^2.

Préliminaires méthodologiques

Dans toute la suite, on trouvera souvent la question "Étudier les performances de l'estimateur en fonction de n" et d'un (ou plusieurs) autre(s) paramètre(s) noté(s) ici μ. Cela signifie que l'on sait simuler une variable aléatoire de la forme $Z_n^{(\mu)}$, où par exemple

$$Z_n^{(\mu)} = \|\widehat{\vartheta}_{n,\mu} - \vartheta\|^2$$

1. Nous avons vu dans l'exercice précedent la validation croisée leave-one-out ou n-fold

et ϑ_n est un estimateur de ϑ dépendant de μ (μ peut être égal à la taille du dictionaire, le niveau de variance à travers les observations que l'on a injectées dans l'estimateur, la régularité de la fonction sous-jacente inconnue, etc.). Quand la théorie nous dit que

$$\mathbb{E}[Z_n^{(\mu)}] \leq Cn^{-p(\mu)}, \qquad (\star),$$

nous proposons le protocole suivant pour mettre en évidence ce résultat :

— On se fixe un nombre de répliques de Monte-Carlo K (par exemple $K = 30$).

— Pour $\ell = 1, \ldots, K$, on simule la perte de l'estimateur pour μ fixé (choisi), c'est-à-dire $Z_{n,\ell}^{(\mu)}$ et pour différentes valeurs de n (par exemple $n = 10^k$, $k = 1, \ldots 6$ ou 7 voire plus).

— On construit

$$n \rightsquigarrow K^{-1} \sum_{\ell=1}^{K} Z_{n,\ell}^{(\mu)}, \quad \text{pour} \quad n = 10^k, k = 1, 2, \ldots$$

qui approche $\mathbb{E}[Z_n^{(\mu)}]$ par la loi des grands nombres, c'est la méthode de Monte-Carlo.

— On représente graphiquement $\log K^{-1} \sum_{\ell=1}^{K} Z_{n,\ell}^{(\mu)}$ en fonction de $\log n$ (ce que l'on appelle un log-log plot).

— On trace la droite des moindres carrés pour le nuage de points

$$(\log n, K^{-1} \sum_{\ell=1}^{K} Z_{n,\ell}^{(\mu)}), n = 10^k, k = 1, \ldots, 2...$$

La pente (négative) obtenue est une estimation de $-p(\mu)$.

— On peut répéter l'expérience pour plusieurs valeurs de μ.

1. Etudier les propriétés asymptotiques de $\widehat{\sigma}_n^2$ lorsque f est constante (mais inconnue). On admettra que $\widehat{\sigma}_n^2$ converge lorsque f satisfait une propriété de type hölderienne.
2. Simuler le modèle d'observation (X_i, Y_i), $i = 1, \leq n$ pour des bruits ξ_i gaussiens standard, avec $\sigma^2 = 1$ et
$$f(x) = \max\{x, (1-x)\}^\beta$$
pour différentes valeurs de $\beta > 0$.
3. Etudier en fonction de n et de β la vitesse de convergence de l'estimateur de Rice.