

TP no 6 - modèles linéaires pénalisés et glmnet

Master parcours SSD - UE Apprentissage Statistique I

1 Exercice 1 - prise en main du package glmnet

Dans cet exercice nous allons illustrer l'utilisation du package `glmnet` sur un problème de classification. Pour cela nous travaillerons sur le jeu de données "South Africa Heart Disease", utilisé à fins illustratives dans le livre *Elements of Statistical Learning* que l'on peut télécharger ici : <https://web.stanford.edu/~hastie/ElemStatLearn/>. Nous nous limiterons ici à un problème de classification binaire, mettant donc en jeu un modèle de régression logistique, mais la "vignette" du package `glmnet` illustre son utilisation de manière bien plus complète¹.

1. Charger le jeu de données.
 - Le jeu de données est contenu dans le fichier texte `SAheart.data`. Notons que la première colonne du fichier contient le nom des lignes et qu'il convient de les interpréter en tant que telle.
2. Mettre en forme le jeu de données
 - (a) extraire la variable réponse, qui est contenue dans le champ `chd`
 - (b) transformer le descripteur qualitatif `famhist` en descripteur quantitatif 0/1
 - (c) standardiser les descripteurs
3. Construire un modèle lasso et représenter le chemin de régularisation obtenu
 - On construit le modèle avec la fonction `glmnet` et on représente le chemin de régularisation obtenu avec la fonction `plot.glmnet`.
 - Notons que par défaut la fonction `glmnet` considère 100 valeurs du paramètre de régularisation définies automatiquement (se référer à la documentation pour davantage de précisions).
4. Faire de même pour une pénalité "ridge".
 - Il suffit pour cela de modifier le paramètre α qui définit la pénalité "elastic-net" :

$$\Omega(w) = \alpha \|w\|_1 + \frac{1 - \alpha}{2} \|w\|_2^2.$$

- Le paramètre α vaut par défaut 1, ce qui correspond à un modèle lasso. Se référer à la documentation pour davantage de précisions.

2 Exercice 2 - lasso, ridge et elastic-net

Dans cet exercice nous allons illustrer le compromis entre performances de prédiction et interprétabilité du modèle que l'on observe fréquemment quand on applique les pénalités Lasso et Ridge à des données de haute dimension (et corrélées) : le Lasso permettant d'obtenir un modèle interprétable, mais la pénalité ridge offrant souvent de meilleures performances de prédiction. Nous illustrerons également que la pénalité elastic-net peut s'avérer intéressante dans ce contexte.

Pour cela nous travaillerons sur le jeu de données "Golub", un jeu de données fondateur pour l'analyse de données de puces à ADN pour l'oncologie, dont on trouve un descriptif ici : https://web.stanford.edu/~hastie/CASI_files/DATA/leukemia.html.

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

1. Charger le jeu de données
 - Le jeu de données est stocké dans deux fichiers texte `X.txt` et `y.txt` contenant respectivement les observations (72 échantillons en dimension 3571) et les réponses associées (codées ici en -1/1).
2. Visualiser les chemins de régularisation des modèles lasso et ridge.
3. Réaliser une expérience de validation croisée en utilisant la fonction `cv.glmnet()` et afficher les résultats avec la fonction `plot.cv.glmnet()`. A quoi correspondent les deux droites verticales représentées en pointillés ?
 - On utilisera l'option `type.measure = 'class'` pour considérer l'erreur de classification comme critère de performance (se référer à la documentation pour davantage de détails).
4. Afin de mieux interpréter les résultats, représenter sur une même figure l'évolution des performances de validation croisée obtenue par les deux modèles quand le paramètre de régularisation varie. Quel modèle offre les meilleures performances ?
 - On se contentera de représenter le taux de bonne classification moyen observé dans les différentes folds de validation croisée, stocké dans le champ `cvm` de l'objet renvoyé par `cv.glmnet()`.
5. Reproduire cette analyse en considérant une pénalité elastic-net afin de considérer un compromis entre lasso et ridge. Commenter les résultats.
 - Il suffit pour cela de modifier le paramètre α de la fonction `cv.glmnet()` et de le faire varier entre 0 (ridge) et 1 (Lasso, la valeur par défaut). On pourra par exemple considérer une grille définie par pas de 0.2.
6. Enfin, dresser un bilan des résultats obtenus en représentant l'évolution (1) des meilleures performances de validation croisée et (2) du nombre de variables sélectionnées, en fonction de α . Commenter les résultats. Quelle valeur de α retiendriez-vous ? Comparer le chemin de régularisation correspondant à celui du lasso.
 - Pour chaque modèle on retiendra la valeur `lambda.1se` proposée par la fonction `cv.glmnet()` comme "meilleur" paramètre de régularisation.
 - Notons que le champ `nzero` de l'objet renvoyé par `cv.glmnet()` donne le nombre de coefficients non-nuls pour chaque valeur de λ considérée (l'ensemble de ces valeurs étant stockées dans le champ `lambda`).