TELECOM ParisTech **MDI720**
2017/2018 **Joseph Salmon, François Portier**

DEVOIR FINAL : Modèles linéaires

Pour ce travail vous devez déposer un **unique** fichier au format `nom_prenon.ipynb` sur le site pédagogique du cours MDI 720.

Vous devez charger votre fichier sur ce site (MDI720 > Validation), avant le mercredi 25/10/2017, 23h59, dans l'un des deux dossier qui correspond à votre nom.

La note totale est sur **20** points, répartis comme suit :

— qualité des réponses aux questions : **15** pts,

— qualité de rédaction, de présentation et d'orthographe : **2** pts,

— indentation, style PEP8, commentaires adaptés, etc. : **2** pts,

— absence de bug : **1** pt.

Les personnes qui n'auront pas rendu leur devoir avant la limite obtiendront **zéro** (et aucun travail par mail ne sera accepté).

EXERCICE 1. (Lasso seuillé)

Dans cette section on veut comparer différentes procédures sur la base de données “Leukemia”. On reprend les notations du cours : $X \in \mathbb{R}^p$ est la matrice des variables explicatives (sans *intercept*), $\mathbf{y} \in \mathbb{R}^n$ est le vecteur des observations. On travaillera sans *intercept* (sauf pour pour la question 10), et pour les validations croisées, on utilisera uniquement $CV = 4$ folds. Charger les données de la manière suivante :

```
from sklearn.datasets.mldata import fetch_mldata
dataset_name = 'leukemia'
data = fetch_mldata(dataset_name)
X = data.data
y = data.target
X = X.astype(float)
y = y.astype(float)
```

1) Donner le nombre d'observations et de variables explicatives (*features*) de cette base de données. Appliquer un pré-traitement afin que chaque colonne de X soit dorénavant de variance empirique égale à 1.

2) Appliquer une analyse en composantes principales (ACP) sur la matrice X, et visualiser les variables explicatives en dimension $d = 1$, puis en dimension $d = 2$ en projetant sur les axes principaux qui conviennent. Faire de même avec la méthode TSNE. On affichera les points de deux couleurs différentes selon leur classe.

3) Couper les données en deux ensembles : un pour l'entraînement $(X^{train}, \mathbf{y}^{train})$ et un pour le test $(X^{test}, \mathbf{y}^{test})$. On utilisera 80% des données pour l'entraînement (en utilisant par exemple la fonction `model_selection.train_test_split` de `sklearn`).

4) On définit le Lasso (sans *intercept*) comme en cours par :

$$\hat{\boldsymbol{\theta}}_\lambda^{\text{Lasso}} \in \operatorname*{arg\,min}_{\boldsymbol{\theta} \in \mathbb{R}^p} \left(\frac{1}{2} \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2 + \lambda \|\boldsymbol{\theta}\|_1 \right). \tag{1}$$

Notons que dans la plupart des packages il est défini par

$$\hat{\boldsymbol{\theta}}_{\lambda'}^{\text{Lasso package}} \in \mathop{\arg\min}_{\boldsymbol{\theta}\in\mathbb{R}^p} \left(\frac{1}{2n} \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2 + \lambda' \|\boldsymbol{\theta}\|_1 \right). \tag{2}$$

avec n le nombre d'observations fournies. Trouver mathématiquement λ' en fonction de λ tel que $\hat{\boldsymbol{\theta}}_{\lambda}^{\text{Lasso}} = \hat{\boldsymbol{\theta}}_{\lambda'}^{\text{Lasso package}}$.

5) Utiliser `LassoCV` sur $(X^{train}, \mathbf{y}^{train})$. On utilisera pour cela la grille standard des solveurs, avec $T = 17$ valeurs de paramètres de régularisation (*i.e.,* on choisit pour $t = 0, \dots, T-1$, $\lambda'_t = \lambda'_0 10^{-\delta t/(T-1)}$, avec $\delta = 0.01$ et $\lambda'_0 = \left\|X^\top \mathbf{y}\right\|_\infty / n$, le plus petit λ' tel que $\hat{\boldsymbol{\theta}}_{\lambda'}^{\text{Lasso package}} = 0$). Donner l'erreur de prédiction moyenne (quadratique) obtenue par `LassoCV` sur $(X^{test}, \mathbf{y}^{test})$, *i.e.,* $\|X^{test}\hat{\boldsymbol{\theta}}_{\lambda'}^{\text{Lasso package}} - \mathbf{y}^{test}\|_2^2 / n_{test}$.
   Afficher aussi graphiquement l'erreur de prédiction (quadratique) moyenne obtenue par validation croisée pour chaque paramètre λ' (on pourra utiliser l'attribut `mse_path_` de `LassoCV` ainsi qu'une échelle semi-log avec `semilogx`).
6) Proposer et calculer un estimateur $\hat{\sigma}$ de l'écart type du bruit dans le modèle linéaire considéré.
7) Coder la méthode suivante :

Algorithm 1: Lasso Seuillé

Input: $X^{train}, \mathbf{y}^{train}, \lambda', \tau$
Output: Lasso Seuillé : $\hat{\boldsymbol{\theta}}_{\lambda'}^{\text{th}-\text{Lasso}}$
$\boldsymbol{\theta} \leftarrow \hat{\boldsymbol{\theta}}_{\lambda'}^{\text{Lasso package}}(X^{train}, \mathbf{y}^{train})$
$S = \varnothing$
for $j \in [\![1, p]\!]$ **do**
  if $|\boldsymbol{\theta}_j| > \tau$ **then**
    $S \leftarrow S \cup \{j\}$ (rajout de j aux indices retenus)
$\hat{\boldsymbol{\theta}}_{\lambda'}^{\text{th}-\text{Lasso}} \leftarrow \theta^{\text{OLS}}(X_S^{train}, \mathbf{y}^{train})$ (moindres carrés de $\mathbf{y}^{train}$ sur la matrice extraite de X^{train} en ne gardant que les colonnes d'indices dans S)
return $\hat{\boldsymbol{\theta}}_{\lambda'}^{\text{th}-\text{Lasso}}$

8) Écrire une procédure de validation croisée (avec CV=4 folds) pour la procédure "Lasso Seuillé" sur la double grille en λ' et en τ (prendre seulement 5 valeurs pour τ).
9) Comparer l'erreur de prédiction obtenue sur la partie "test" pour :
   (a) le "Lasso Seuillé" avec validation croisée (de la question précédente)
   (b) le `LassoCV` (de la question 5, sans *intercept*).
   (c) l'estimateur des moindres carrées (sans *intercept*).
10) Reprendre l'ensemble des comparaisons précédentes, mais cette fois en tenant compte de l'*intercept* dans votre démarche.
11) Comparer (sur la partie test) les performances des deux méthodes suivantes :
   (a) le `LassoCV` modifié pour retourner une prédiction valant soit 1 soit -1
   (b) la `LogisticRegressionCV`.

   On utilisera ici l'erreur 0/1 (*i.e.,* la proportion d'erreurs de "classe" faites) comme mesure de performance.