«kernel methods» @ AMMI, Rwanda

$\boxed{\text{TP 2}}$ YUNLONG JIAO, 23/01/2019.

Ex 1. (30 min) Derive analytical solutions to the following optimization problems. Then implement them ②&③ in R/Python.

$$\mathcal{D} = \{(\vec{x}_i, y_i)\}_{i=1}^{n} \quad x_i \in \mathbb{R}^p,\ y_i \in \mathbb{R}.$$

① OLS:

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{n}\|y - X\beta\|^2 := J(\beta) \qquad y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n$$

$$X = \begin{pmatrix} x_1^T \\ \vdots \\ x_n^T \end{pmatrix} \in \mathbb{R}^{n \times p}$$

$$\frac{\partial J}{\partial \beta} = -\frac{2}{n} X^T(y - X\beta) = 0 \qquad \underline{\text{Model}}: y_i = \beta^T x_i + \varepsilon_i$$

$$(X^TX)\beta = X^Ty$$

If (X^TX) invertible, $\hat{\beta}^{OLS} = (X^TX)^{-1}X^Ty$

$\Rightarrow$ Prediction: $x^* \in \mathbb{R}^p$ $\quad \hat{y}^* = (x^*)^T \hat{\beta}^{OLS} = (x^*)^T (X^TX) X^T y.$

② Ridge:

$$\min_{\beta} \frac{1}{n}\|y - X\beta\|^2 + \lambda\|\beta\|^2 \quad (\lambda > 0)$$

$$\frac{\partial J}{\partial \beta} = -\frac{2}{n} X^T(y - X\beta) + 2\lambda\beta = 0$$

$$\hat{\beta}^{Ridge} = (X^TX + \lambda n I)^{-1} X^T y \overset{\text{Ex}}{=} X^T(XX^T + \lambda n I)^{-1} y$$

$\Rightarrow$ Prediction: $x^* \in \mathbb{R}^p$.

$$\hat{y}^* = (x^*)^T \hat{\beta}^{Ridge}$$

$$= (x^*)^T (X^TX + \lambda n I)^{-1} X^T y.$$

$$\overset{\text{Ex}}{=} \underbrace{(x^*)^T X^T}_{K_{x^*,X}} \underbrace{(XX^T + \lambda n I)^{-1}}_{K_{XX} + \lambda n I} y$$

$\boxed{\text{GP}}$

③ Weighted Ridge Regression (WRR)

$$\min \frac{1}{n}\sum_i w_i (y_i - \beta^T x_i)^2 + \lambda \|\beta\|^2 \qquad (w_i > 0)$$

$$= \frac{1}{n} (y - X\beta)^T W (y - X\beta) + \lambda \|\beta\|^2 \qquad W = \mathrm{diag}(w_1, \dots, w_n)$$

$$= \frac{1}{n} \| W^{\frac{1}{2}} (y - X\beta)\|^2 + \lambda \|\beta\|^2.$$

$$= \frac{1}{n} \| \tilde{y} - \tilde{X}\beta\|^2 + \lambda \|\beta\|^2. \qquad \tilde{y} = W^{\frac{1}{2}} y, \quad X = W^{\frac{1}{2}} X.$$

Due to Ridge ②.

$$\hat{\beta}^{W\text{-Ridge}} = (X^T W X + \lambda n I)^{-1} X^T W y.$$

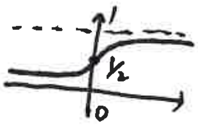
Ex 2: (IRLS for Logistic Ridge Regression.)
(90 min)

$$\mathcal{D} = \{(\vec{x}_i, y_i)\}_{i=1}^{n}. \qquad x_i \in \mathbb{R}^p, \quad y_i \in \{-1, +1\}.$$

$$X \in \mathbb{R}^{n \times p}. \qquad y \in \{-1, +1\}^n.$$

Model: $$\mathbb{P}(y_i \mid \vec{x}_i, \beta) = \begin{cases} \sigma(\beta^T x_i) & \text{if } y = +1 \\ 1 - \sigma(\beta^T x_i) \stackrel{}{=} \sigma(-\beta^T x_i) & \text{if } y = -1 \end{cases} = \sigma(y_i \beta^T x_i)$$

where. sigmoid $$\sigma(a) = \frac{1}{1 + e^{-a}}$$



$$\begin{cases} \sigma(-a) = 1 - \sigma(a) \\ \sigma'(a) = \sigma(a)\,\sigma(-a) \end{cases}$$

LRR: $$\min \frac{1}{n} \sum_i \Big[\underbrace{-\log \mathbb{P}(y_i \mid x_i, \beta)}_{\to\ -LK.}\Big] + \underbrace{\lambda \|\beta\|^2}_{\to \text{ Gaussian Prior.}} := J(\beta)$$

$$= -\frac{1}{n}\sum_i \log \sigma(y_i \beta^T x_i) + \lambda \|\beta\|^2$$

① Derive. $\nabla J(\beta)$, $\nabla^2 J(\beta)$.

$$\nabla_\beta J(\beta) = -\frac{1}{n}\sum \frac{\sigma(y_i \beta^T x_i)\,\sigma(-y_i \beta^T x_i)}{\sigma(y_i \beta^T x_i)} y_i x_i + 2\lambda\beta$$

$$\boxed{P(\beta) = \mathrm{diag}[\sigma(-y_i \beta^T x_i)]_n}$$

$$= -\frac{1}{n} \sum y_i \underbrace{\sigma(-y_i \beta^T x_i)}\, x_i + 2\lambda\beta.$$

$$(x_1 | \dots | x_n) (\diagdown) \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}.$$

$$=: -\frac{1}{n} X^T P(\beta)\, y + 2\lambda\beta$$

$$\nabla^2 J(\beta) = \frac{1}{n} \sum \underbrace{\sigma(y_i \beta^T x_i)\, \sigma(-y_i \beta^T x_i)}\, x_i x_i^T + 2\lambda I$$

$$=: \frac{1}{n} X^T W(\beta) X + 2\lambda I$$

$(x_1 | \ldots | x_n)(\diagdown)(||)$

$$\boxed{W(\beta) = diag\left[\sigma(\beta^T x_i)\,\sigma(-\beta^T x_i)\right]_n}$$

② Show that each step of the Newton-Raphson solver of LRR is equivalent to WRR. (Ex 1.③)

Gradient Descent:	Newton-Raphson solver:
$\min_x f(x) \quad f \in C^1$.	$\min f(x) \quad f \in C^2$
– Initialize $x := x^{(0)}$	– Initialize $x := x^{(0)}$
– For $k = 0, \ldots, L$, or until convergence (*)	– For $k = 0, \ldots, L$, or until convergence, (*).
$x^{(k+1)} = x^{(k)} - t\, \nabla f(x^{(k)})$ (t → step size.)	$x^{(k+1)} = x^{(k)} - (\nabla^2 f(x^{(k)}))^{-1} \nabla f(x^{(k)})$.
at k-th step.	at k-th step:
$f(x) \approx f(x^{(k)}) + (x - x^{(k)})^T \nabla f(x^{(k)}) + \frac{1}{2t} \Vert x - x^{(k)} \Vert^2 := f_{g_1}(x)$	$f(x) \approx f(x^{(k)}) + (x - x^{(k)})^T \nabla f(x^{(k)}) + \frac{1}{2}(x - x^{(k)})^T \nabla^2 f(x^{(k)})(x - x^{(k)}) := f_{g_2}(x)$.
$\min_x f_{g_1}(x)$ (around $x^{(k)}$)	$\min_x f_{g_2}(x)$ (around $x^{(k)}$).
$\Rightarrow x^{(k+1)} = x^{(k)} - t\, \nabla f(x^{(k)})$	$\Rightarrow x^{(k+1)} = x^{(k)} - (\nabla^2 f(x^{(k)}))^{-1} \nabla f(x^{(k)})$.

(*) Stopping criteria was too simplified here...!

(+) Newton is rarely used, consider Conjugate Gradient (Truncated Newton), BFGS. (Quasi-Newton)

$$\beta^{new} \leftarrow \beta^{old} - [\nabla^2 J(\beta^{old})]^{-1} \nabla J(\beta^{old}).$$

LRR: $\min J(\beta) := -\frac{1}{n} \sum_i \log \sigma(y_i \beta^T x_i)$.

at k-th step:

$$(J(\beta) \approx)\ \tilde{J}(\beta) := J(\beta^{(k)}) + (\beta - \beta^{(k)})^T \nabla J(\beta^{(k)}) + \frac{1}{2}(\beta - \beta^{(k)})^T \nabla^2 J(\beta^{(k)})(\beta - \beta^{(k)})$$

where $\beta^T \nabla J(\beta^{(k)}) = -\frac{1}{n} \beta^T X^T P y + 2\lambda \beta^T \beta^{(k)}$

$$\boxed{P := P(\beta^{(k)})}.$$

$-\beta^T \nabla^2 J(\beta^{(k)}) \beta^{(k)} = -\frac{1}{n} \beta^T X^T W X \beta^{(k)} - 2\lambda \beta^T \beta^{(k)}$

$\frac{1}{2} \beta^T \nabla^2 J(\beta^{(k)}) \beta = \frac{1}{2n} \beta^T X^T W X \beta + \lambda \beta^T \beta$

$$\boxed{W := W(\beta^{(k)})}$$

$$\Rightarrow\ 2J_q(\beta) = \frac{1}{n}\beta^T X^T W X \beta - \frac{2}{n}\beta^T X^T W \underbrace{\boxed{\left(X\beta^{(k)} + W^{-1} P y\right)}}_{=:\, z} + 2\lambda\|\beta\|^2 + C$$

$$= \frac{1}{n}(z - X\beta)^T W (z - X\beta) + 2\lambda\|\beta\|^2 + \tilde{C}$$

③ Implement solveLRR(...) in R/ Python.

Set:

$$[f_i] = [(\beta^{(k)})^T x_i] \qquad i=1,\dots,n.$$

$$[W_i] = \mathrm{diag}[\sigma(f_i)\sigma(-f_i)]$$

$$[P_i] = \mathrm{diag}[\sigma(-y_i f_i)]$$

$$[z_i] = [f_i + y_i P_i W_i^{-1}] = [f_i + y_i / \sigma(y_i f_i)]$$

Call: solveWRR(z, X, W, 2λ). at each update.