ST509 - Takehome Final

Submission due on June 19th (Wed) 4:45pm

Problems

1. **(LUM)** Liu et al. (2011) proposed the Large-margin Unified Machine (LUM) by introducing the following loss function, a hybrid version of "exponential" loss and "hinge" loss function:

$$V(u) = \begin{cases} 1-u & \text{if } u < \frac{c}{1+c} \\ \frac{1}{1+c}\left(\frac{a}{(1+a)u-c+a}\right)^a & \text{if } u \geq \frac{c}{1+c} \end{cases}$$

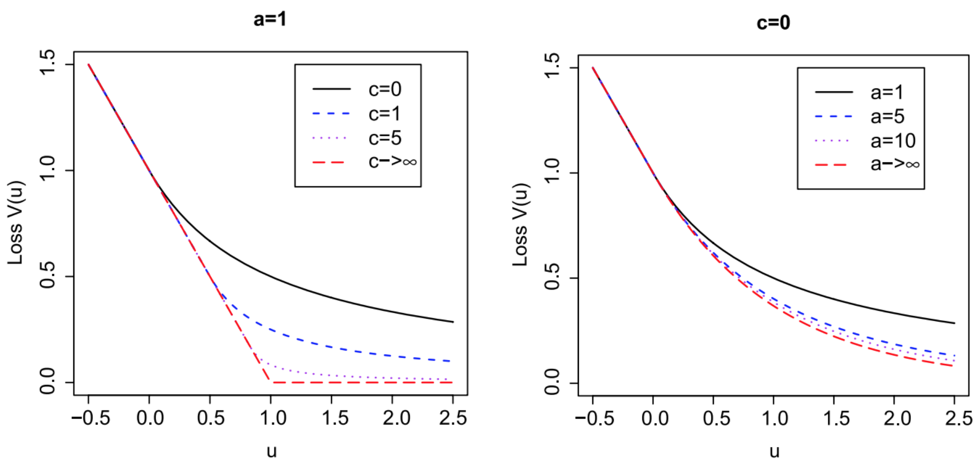

Figure 1: LUM loss function. Left penal: $a = 1, c = 0, 1, 5, \infty$; Right penal: $c = 0, a = 1, 5, 10, \infty$.

Given $y_i \in \{-1, 1\}$ and $x_i \in \mathbb{R}, i = 1, \cdots, n$, the linear LUM seeks a linear decision function $f(x) = \alpha + \beta x$ by solving

$$\min_{\alpha,\beta} \frac{1}{n}\sum V(y_i(\alpha + \beta x_i)) + \frac{\lambda}{2}\beta^2$$

The optimization can be done by applying the coordinate decent algorithm. The updating equations at the $(t+1)$-step are given by

- Intercept α:

$$\hat{\alpha}^{(t+1)} = \underset{\alpha}{\text{argmin}} \sum_{i=1}^{n} V\left(y_i(\alpha + \hat{\beta}^{(t)} x_i)\right)$$

- Slope β:

$$\hat{\beta}^{(t+1)} = \underset{\beta}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{n} V\left(y_i(\alpha^{(t)} + \beta x_i)\right) + \frac{\lambda}{2}\beta^2$$

Updating equations are all one dimensional, and can readily solved via, for example, `optimize()` function in `R`.

Please write your own code to solve the linear LUM with a single predictor via CD algorithm. You may use built-in functions in `R` to solve the one-dimensional CD updating equations above:

```
lum <- function(x, y, a = 1, c = 1, lambda = 1) {
# x: predictor
# y: binary response with {-1, 1} coding
# a, c: constants in LUM loss
# lambda: regularization pamameter

....

est <- c(alpha, beta) # linear LUM estimator
return(est)
}
```

2. **(WSVM)** One way to generalize the SVM is to allow a class weight that controls relative importance between two classes. The linear weighted SVM (WSVM) solves

$$\min_{\beta_0,\boldsymbol{\beta}} \; C\sum_{i=1}^{n} w_i\xi_i + \boldsymbol{\beta}^T\boldsymbol{\beta}$$
$$\text{subject to } y_i(\beta_0 + \boldsymbol{\beta}^T\mathbf{x}_i) \geq 1 - \xi_i \text{ and } \xi_i \geq 0, \forall i = 1, \cdots, n.$$

where $w_i = 1 - \pi$ for $y_i = 1$ and π for $y_i = -1$ with $\pi \in (0, 1)$ being pre-specified.

Please write your own code to solve WSVM:

```
wsvm <- function(x, y, pi = 0.5, C = 1) {
# x: predictor
# y: binary response with {-1, 1} coding
# pi: weight parameter in (0,1)
# C: cost parameter

....

est <- c(beta0, beta) # beta0 for intercept, and beta for the coefficient vector
return(est)
}
```

3. **(EM algorithm for Right-censored Exponential Random Variable)** In the right censoring context, we observe $(Y_1, \delta_1), \cdots, (Y_n, \delta_n)$, where $Y_i = \min(X_i, R_i)$, X_i has the density $f(x; \sigma) = \sigma^{-1} e^{x/\sigma}$ (i.e., Exponential (σ)), R_i is the (non-stochastic) censoring time, and $\delta = I(X_i \leq R_i)$ is censoring indicator. Without loss of generality, we assume that $Y_1, \cdots, Y_{n_u}$ are the uncensored observations and $Y_{n_u+1}, \cdots, Y_n$ are the censored ones. Then the likelihood is given by

$$L(\mathbf{Y}, \boldsymbol{\delta} \mid \sigma) = \sigma^{-n_u} e^{-\sum_{i=1}^n Y_i/\sigma}$$

   (a) Compute the maximum likelihood estimator of σ, $\hat{\sigma}_{\text{MLE}}$.

   (b) EM algorithm can be applied to solve this problem. Toward this, compute

$$Q(\sigma, \sigma^{(v)}, \mathbf{Y}, \boldsymbol{\delta}) = \mathbb{E}_{\sigma^{(v)}} \left\{ \log L_C(\sigma \mid \mathbf{X}) \mid \mathbf{Y}, \boldsymbol{\delta} \right\}$$

   where the complete likelihood $\log L_C(\sigma|\mathbf{X})$ is given by

$$\log L_C(\sigma|\mathbf{X}) = \sum_{i=1}^n \log f(X_i|\sigma).$$

   Finally, show that the corresponding EM updating equation is given by

$$\sigma^{(v+1)} = \frac{1}{n} \left\{ \sum_{i=1}^n Y_i + (n - n_u)\sigma^{(v)} \right\}.$$

   Hint: The conditional density of X_i given $X_i > R_i$ is $\sigma^{-1} \exp\{-(x - R_i)/\sigma\} I(x > R_i)$.

   (c) Write your own code to obtain the MLE of σ via EM algorithm:

```
em <- function(y, delta, ....) {
# y: observed survival time
# delta: censoring indicator
# ....: other factors required in EM Algorithm

....

return(hat.sigma) # hat.sigma is the mle of sigma
}
```

4. **(CI for Poisson Regression)** $y_i|\mathbf{x}_i \sim Poisson\left(\mu(\mathbf{x}_i;\tilde{\boldsymbol{\beta}})\right)$ where

$$\log\left\{\mu(\mathbf{x}_i;\tilde{\boldsymbol{\beta}})\right\} = \beta_0 + \mathbf{x}_i^T\boldsymbol{\beta}, \qquad i = 1, \cdots, n.$$

with $\mathbf{x}_i = (x_{i1}, \cdots, x_{ip})^T$ and $\tilde{\boldsymbol{\beta}} = (\beta_0, \boldsymbol{\beta}^T)^T$ with $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_p)^T$. Given a set of data $\{\mathbf{x}_i, y_i\}, i = 1, \cdots, n$, the log-likelihood $\ell(\beta)$ is given by

$$\ell(\tilde{\boldsymbol{\beta}}) \propto \sum_{i=1}^{n}\left[y_i \log\left\{\mu(\mathbf{x}_i;\tilde{\boldsymbol{\beta}})\right\} - \mu(\mathbf{x}_i;\tilde{\boldsymbol{\beta}})\right] \tag{1}$$

(a) Assume the following prior distribution for $\beta_j, j = 0, 1, \cdots, p$:

$$\beta_j \sim N(0, 1000^2),\ j = 0, 1, \cdots, p$$

Please write your own code to get posterior samples of $\beta_j, j = 0, 1, \cdots, p$ via MCMC such as MH within Gibbs Sampler:

```
mcmc <- function(x, y, n.samples = 10000, ....) {
# x: n * p predictor matrix
# y: response vector in the Poisson regression
# n.samples: number of posterior samples to be obtained.
# ....: other factors required in MCMC

...

return(samples) # n.sample * (p + 1) matrix of posterior samples.
}
```

(b) Write your own code to obtain a Bayesian CI of $\beta_j, j = 0, 1, \cdots, p$ from the object obtained from `mcmc()` function.

```
bayes.ci <- function(obj, alpha = 0.05) {
# obj: object from the mcmc() function
# alpha: confidence level

...

return(c(lower, upper)) # lower and upper limit of the Bayesian CI
}
```

(c) Write your own code to obtain a bootstrap percentile CI of $\beta_j, j = 0, 1, \cdots, p$. You may use `glm` function to fit the model:

```
boot.ci <- function(x, y, B = 500, alpha = 0.05, ....) {
# x: n * p predictor matrix
# y: response vector in the Poisson regression
# B: number of bootstrap repetitions
# alpha: confidence level
...

return(c(lower, upper)) # lower and upper limit of the bootstrap CI
}
```

Submission Rules (Important!)

- You must send me the following by e-mail (`sjshin@korea.ac.kr`, and cc to your email).

  i) "report (in pdf)" that contains answers of the problem sets;

  ii) "single R file" that contains four functions ONLY

- **Due date: 6/19 (Wed) 4:45pm**
  - If I get your mail after 6/19 (Sat) 4:45pm, you will lose 30% of the credits you earned.
  - If I get your mail after 6/19 (Sat) 5:00pm, NO credit!
- Additional rules:
  - Subject line of the email: `ST509_Final_StuduentID`

    (ex: `ST509_Final_2019150010`)
  - File name of your report: `ST509_Final_StuduentID.pdf`
  - Your report must be sent in a pdf format. Handwriting is okay, but you have to scan it in a pdf format.
  - File name of your code: `ST509_Final_StuduentID.R`
  - All functions should be included in a single R file.
- If you do NOT strictly follow these rules above, you additionally lose 5% of your credits.

References

Liu, Y., Zhang, H. H. and Wu, Y. (2011). Hard or soft classification? large-margin unified machines, *Journal of the American Statistical Association* **106**(493): 166–177.