Logistic-Lasso Coordinate Descent Algorithm

Jimmy Kelliher (UNI: jmk2303)

2022-03-20

Theory

Lemma 1. Consider the optimization problem

$$\min_{x \in \mathbb{R}} \left\{ \frac{1}{2}(x-b)^2 + c|x| \right\}$$

for $b \in \mathbb{R}$ and $c \in \mathbb{R}_{++}$. It follows that the minimizer is given by

$$\hat{x} = S(b, c),$$

where S is the soft-thresholding operator.

Lemma 2. Consider the optimization problem

$$\min_{\beta_k \in \mathbb{R}} \left\{ \frac{1}{2n} \sum_{i=1}^{n} w_i \left(z_i - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\}$$

for some $k \in \{1, \ldots, p\}$. It follows that the minimizer is given by

$$\hat{\beta}_k = \left(\sum_{i=1}^{n} w_i x_{ik}^2 \right)^{-1} \sum_{i=1}^{n} w_i x_{ik} \left(z_i - \sum_{j \neq k} \beta_j x_{ij} \right).$$

Lemma 3. With $\hat{\beta}_k$ defined as above,

$$\min_{\beta_k \in \mathbb{R}} \left\{ \frac{1}{2n} \sum_{i=1}^{n} w_i \left(z_i - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\} = \min_{\beta_k \in \mathbb{R}} \left\{ \frac{1}{2}(\beta_k - \hat{\beta}_k)^2 + \left(\frac{1}{n} \sum_{i=1}^{n} w_i x_{ik}^2 \right)^{-1} \lambda |\beta_k| \right\}.$$

Proposition. By Lemma 1 and Lemma 3,

$$\arg\min_{\beta_k \in \mathbb{R}} \left\{ \frac{1}{2n} \sum_{i=1}^{n} w_i \left(z_i - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\} = S\left(\hat{\beta}_k, \left(\frac{1}{n} \sum_{i=1}^{n} w_i x_{ik}^2 \right)^{-1} \lambda \right)$$

Praxis

```
data <-
  read_csv("data/breast-cancer.csv") %>%
  mutate(diagnosis = 1 * (diagnosis == "M"))
```

```
## Rows: 569 Columns: 32
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): diagnosis
## dbl (31): id, radius_mean, texture_mean, perimeter_mean, area_mean, smoothne...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Helper Functions

```
# logistic function
logistic <- function(x) 1 / (1 + exp(-x))

# shrinkage function
S <- function(beta, gamma) {
  if(abs(beta) <= gamma) {
    0
  } else if(beta > 0) {
    beta - gamma
  } else {
    beta + gamma
  }
}

# probability adjustment function
p_adj <- function(p, epsilon) {
  if (p < epsilon) {
    0
  } else if(p > 1 - epsilon) {
    1
  } else {
    p
  }
}

# weight adjustment function
w_adj <- function(p, epsilon) {
  if ((p < epsilon) | (p > 1 - epsilon)) {
    epsilon
  } else {
    p * (1 - p)
  }
}
```

Toy Example

```
set.seed(1)
lambda <- 0 #0.0125
epsilon <- 10^(-5)

n    <- 1000
X    <- scale(matrix(rnorm(3 * n), c(n, 3)))
X    <- as.matrix(cbind(rep(1, n), X))
y    <- 1 * (runif(n) > 0.5)

# initialize parameters
beta <- rep(0, ncol(X))

p <- map_dbl(logistic(- X %*% beta), p_adj, epsilon)
w <- map_dbl(p, w_adj, epsilon)
z <- X %*% beta + (y - p) / w

terminate <- 0
iter <- 1
while(terminate < 1) {

  beta_old <- beta

  for(k in 1:ncol(X)) {
    x_k    <- X[ , k]
    x_notk <- X[ , -k]
    b_notk <- beta[-k]

    # un-penalized coefficient update
    b_k_temp <- sum(w * (z - x_notk %*% b_notk) * x_k) / sum(w * x_k^2)
    # shrinkage update
    b_k      <- S(b_k_temp, lambda / mean(w * x_k^2))
    # update beta vector along with other parameters
    beta[k] <- b_k
    #p <- map_dbl(logistic(- X %*% beta), p_adj, epsilon)
    #w <- map_dbl(p, w_adj, epsilon)
    #z <- X %*% beta + (y - p) / w
  }

  iter <- iter + 1

  if(iter == 100 | max(abs(beta - beta_old)) < 10^-10) {
    print(iter)
    terminate <- 1
  }

}
```

```
## [1] 6
```

```
# Estimates from Coordinate Descent
print(beta)
```

```
## [1]  0.104000000  0.020401450  0.006293554 -0.052070535
```

```r
# True estimates from GLM
as.vector(glm(y ~ X[ , -1], family = binomial)$coefficients)
```

```
## [1]  0.104174898  0.020476567  0.006314589 -0.052258099
```

```r
# True estimates from GLMNET
fit <- glmnet(X, y, family = "binomial", standardize = FALSE, lambda = lambda, thresh = 10^-10)
as.vector(fit$beta[ , ncol(fit$beta)])
```

```
## [1]  0.000000000  0.020476567  0.006314589 -0.052258099
```

Test with Actual Data

```r
set.seed(1)
epsilon <- 10^(-5)

n    <- nrow(data)
X    <- data[ , -c(1, 2)]
X    <- as.matrix(cbind(rep(1, n), X))
y    <- data$diagnosis

lambda <- 1 # (max(t(X) %*% y) / n)

# initialize parameters
beta <- rep(0, ncol(X))
p <- map_dbl(logistic(- X %*% beta), p_adj, epsilon)
w <- map_dbl(p, w_adj, epsilon)
z <- X %*% beta + (y - p) / w

terminate <- 0
iter <- 1
while(terminate < 1) {

  beta_old <- beta
  # initially go through all parameters
  K <- 1:ncol(X)
  #if(iter > 1) {
  #  K <- which(beta > 0)
  #}

  for(k in K) {
    x_k    <- X[ , k]
    x_notk <- X[ , -k]
    b_notk <- beta[-k]

    # un-penalized coefficient update
    b_k_temp <- sum(w * (z - x_notk %*% b_notk) * x_k) / sum(w * x_k^2)
    # shrinkage update
    b_k      <- S(b_k_temp, lambda / mean(w * x_k^2))

    # update beta vector along with other parameters
    beta[k] <- b_k
```

```r
    #p <- map_dbl(logistic(- X %*% beta), p_adj, epsilon)
    #w <- map_dbl(p, w_adj, epsilon)
    #z <- X %*% beta + (y - p) / w
  }

  iter <- iter + 1

  if(iter == 1000 | max(abs(beta - beta_old)) < 10^-10) {
    print(iter)
    terminate <- 1
  }

}
```

```
## [1] 905
```

```r
# True estimates from GLMNET
fit <- glmnet(X, y, family = "binomial", standardize = FALSE, lambda = lambda)

# results
tibble(
    Variable   = 1:length(beta)
  , Name       = c("intercept", names(data[ , -c(1, 2)]))
  , Jimmy      = beta
  , GLMNET     = as.vector(fit$beta[ , ncol(fit$beta)])
  , Difference = abs(Jimmy - GLMNET)
) %>%
  filter(Jimmy != 0 | GLMNET != 0) %>%
  knitr::kable()
```

Variable	Name	Jimmy	GLMNET	Difference
4	perimeter_mean	-0.0412133	0.0000000	0.0412133
5	area_mean	0.0011238	-0.0088197	0.0099434
25	area_worst	0.0030613	0.0168840	0.0138226