Test-Implementation-Segment-Specific-05-31-2021

Sixtus Dakurah

07/01/2021

Define the variables:

B : Total amount available for disbursement.

$S_i, \quad i = 1, 2, ..., s$: Denote segment i with a total of s segments.

L : Loan amount (This will be an average value).

P : The price variable.

Step 1: Fit the logistic model for the given segment.

The design matrix is of the form:

$X = [j, x_2, x_3]$ where $l = 2$: Price, $l = 3$: Loan Amount.

The model for predicting the probability of booking is of the form:

$$\pi(X, \beta) = [\sigma(z_i), ..., \sigma(z_m)]' \tag{1}$$

Where $\sigma(z_k) = [1 + exp\{-z_k\}]^{-1}$ and $z_k = x_k\beta$, $k = 1, ..., m$ the number of examples.

An expressive form for $x_k\beta = \beta_0 + \beta_2 x_{k,2} + \beta_3 x_{k,3}$

We now build a logistic model using the data from segment 1

```
# <- data.sim %>% filter(x1 == 's1')
#glm.segment1.fit <- glm(y~x1 + x2 + x3 + I(x1*x2) +
#I(x1*x3), data = data.segment1, family = binomial)
#glm.segment1.fit <- glm(y~x1 + x2, data = data.sim, family = binomial)
#summary(glm.segment1.fit)
```

Extract the coefficients:

```
#seg1.intercept <- ((glm.segment1.fit)$coefficients)[[1]]
#coef2 <- ((glm.segment1.fit)$coefficients)[[2]]
#coef3 <- ((glm.segment1.fit)$coefficients)[[3]]
#seg2.intercept <- seg1.intercept +  ((glm.segment1.fit)$coefficients)[[2]]
#seg3.intercept <- seg1.intercept +  ((glm.segment1.fit)$coefficients)[[3]]
#price.coef <- ((glm.segment1.fit)$coefficients)[[4]]
```

The fitted model has the form:

$\hat{\sigma}(z_k) = [1 + exp\{-z_k\}]^{-1}$

$\pi(x_2, \hat{\beta}) = [\hat{\sigma}(z_1), ..., \hat{\sigma}(z_k)]'$

```
# get the predictions
#segment1.pred.probs <- predict(glm.segment1.fit, type = "response")
#data.sim <- data.sim %>%
#  mutate(pred.probs = segment1.pred.probs,
#         pred.y = as.factor(ifelse(pred.probs > 0.5, 's', 'f')))
#head(data.sim)
```

```
#table(data.sim$pred.y, data.sim$y)
```

Step 2: We now optimize over the price variable.

```
# Compute the total amount available for disbursement
#(B <- sum(data.sim$x3)) # this will now be the average loan amount.
#B <- 100
```

Denote the profit as:

$\rho(x_2, x_4) = x_2 - x_4$ where the new variable x_4 is the cost of booking.

The expected profit is then given as:

$$E\left[\rho(x_2, x_4)|x_2\right] = \sum_{k=1}^{m} \pi(x_{\{k,2\}}, \hat{\beta})\rho(x_{\{k,2\}}, x_{\{k,4\}}) \tag{2}$$

We also have the expected loan amount of the form:

$$E\left[x_3|x_2\right] = \sum_{k=1}^{m} \pi(x_{\{k,2\}}, \hat{\beta}) * x_{k,3} \tag{3}$$

The optimization problem is now of the form:

$$argmin_{x_2} E\left[\rho(x_2, x_4)|x_2\right] \qquad s.t. \quad E\left[x_3|x_2\right] < B_{avg} \tag{4}$$

~~This optimization problem is not well-defined and a solution can not be obtained if we employ single-variable optimization.~~

~~This will be feasible if we resort to vector-valued optimization in which case the $\{k\}$ in (2) and (3) will be mute. But that too will not be useful in this context as we're not looking to optimize for all known price points.~~

~~A modification that will make optimization feasible~~

Let $\rho(x_{\{k,2\}}, x_{\{k,4\}}) = x_{\{k,2\}} - x_{\{k,4\}}$ be the profit for a randomly chosen price point.

The expected profit is now given as:

$$E\left[\rho(x_{\{k,2\}}, x_{\{k,4\}})|x_{\{k,2\}}\right] = \pi(x_{\{k,2\}}, \hat{\beta})\rho(x_{\{k,2\}}, x_{\{k,4\}}) \tag{5}$$

We also have the expected loan amount of the form:

$$E\left[x_{\{k,3\}}|x_{\{k,2\}}\right] = \pi(x_{\{k,2\}}, \hat{\beta}) * x_{k,3} \tag{6}$$

We now have the optimization problem:

$$argmin_{x_{\{k,2\}}} E\left[\rho(x_{\{k,2\}}, x_{\{k,4\}})|x_{\{k,2\}}\right] \qquad s.t. \quad E\left[x_{\{k,3\}}|x_{\{k,2\}}\right] < B_{avg} \tag{7}$$

The Lagrangian is of the form:

$$F(x_{k,2}, \lambda) = E\left[\rho(x_{\{k,2\}}, x_{\{k,4\}})|x_{\{k,2\}}\right] - \lambda\left[E\left[x_{\{k,3\}}|x_{\{k,2\}}\right] - B_{avg}\right] \tag{8}$$

For ease of manipulation let:

$f(x_{k,2}) = E\left[\rho(x_{\{k,2\}}, x_{\{k,4\}})|x_{\{k,2\}}\right]$ and $c(x_{k,2}) = E\left[x_{\{k,3\}}|x_{\{k,2\}}\right] - B_{avg}$

Using Newton's method, we have the following derivations:

Opt 1: Quadratic Taylor series expansion for some chosen starting values $x^o_{k,2}, \lambda^o$:

$$F(x_{k,2}, \lambda) \approx F(x^o_{k,2}, \lambda^o) + (x_{k,2} - x^o_{k,2}) \left.\frac{\partial F}{\partial x_{k,2}}\right|_0 + (\lambda - \lambda^o) \left.\frac{\partial F}{\partial \lambda}\right|_0 + \frac{1}{2}(x_{k,2} - x^o_{k,2})^2 \left.\frac{\partial^2 F}{\partial x^2_{k,2}}\right|_0 + (x_{k,2} - x^o_{k,2}) * (\lambda - \lambda^o) \left.\frac{\partial^2 F}{\partial x_{k,2}\partial \lambda}\right|_0 \tag{9}$$

Opt 2: Inserting (8) into (9), we have:

$$\begin{aligned} F(x_{k,2}, \lambda) \approx F(x^o_{k,2}, \lambda^o) + (x_{k,2} - x^o_{k,2}) \left\{ \left.\frac{\partial f}{\partial x_{k,2}}\right|_0 - \lambda^o \left.\frac{\partial c}{\partial x_{k,2}}\right|_0 \right\} - (\lambda - \lambda^o) c(x^o_{k,2}) + \\ \frac{1}{2}(x_{k,2} - x^o_{k,2})^2 \left\{ \left.\frac{\partial^2 f}{\partial x^2_{k,2}}\right|_0 - \lambda \left.\frac{\partial^2 c}{\partial x^2_{2,k}}\right|_0 \right\} - (x_{k,2} - x^o_{k,2}) * (\lambda - \lambda^o) \left.\frac{\partial c}{\partial x_{k,2}}\right|_0 \end{aligned} \tag{10}$$

Note: the last part, the derivative is w.r.t. only the $x_{k,2}$, as that of λ goes to 1.

Opt 3: That the maximum is achieved at $x_{k,2}$ requires the necessary condition:

$$\frac{\partial F}{\partial x_{k,2}} = \left.\frac{\partial f}{\partial x_{k,2}}\right|_0 + (x_{k,2} - x^o_{k,2}) \left\{ \left.\frac{\partial^2 f}{\partial x^2_{k,2}}\right|_0 - \lambda \left.\frac{\partial^2 c}{\partial x^2_{2,k}}\right|_0 \right\} - \lambda \left.\frac{\partial c}{\partial x_{k,2}}\right|_0 = 0 \tag{11}$$

Opt 4: We now derive the gradient update rules

We can easily see that:

$$\Delta x_{k,2} = \frac{\left\{ \lambda \left.\frac{\partial c}{\partial x_{k,2}}\right|_0 - \left.\frac{\partial f}{\partial x_{k,2}}\right|_0 \right\}}{\left\{ \left.\frac{\partial^2 f}{\partial x^2_{k,2}}\right|_0 - \lambda \left.\frac{\partial^2 c}{\partial x^2_{2,k}}\right|_0 \right\}} \tag{12}$$

Similarly for a first order expansion of the constraints, we can solve a new value of λ as follows:

$$c(x_{k,2}) = c(x^o_{k,2}) + \Delta x_{k,2} \left.\frac{\partial c}{\partial x_{k,2}}\right|_{x^o_{k,2}} = 0 \tag{13}$$

If we put (12) into (13), we have the following value of λ:

$$\lambda = \frac{\left\{ \frac{\partial f}{\partial x_{k,2}}\Big|_0 \right\}}{\left\{ \frac{\partial c}{\partial x_{k,2}}\Big|_0 \right\}} - \frac{\left\{ \frac{\partial^2 f}{\partial x_{k,2}^2}\Big|_0 - \lambda \frac{\partial^2 c}{\partial x_{2,k}^2}\Big|_0 \right\} * c(x_{k,2}^o)}{\left\{ \frac{\partial c}{\partial x_{k,2}}\Big|_0 \right\}^2} \tag{14}$$

Putting this into (12) will give final update rule.

Step 3: Algorithm

The individual components for (12) are derived as follows:

$$\begin{aligned} \left\{ \frac{\partial f}{\partial x_{k,2}}\Big|_0 \right\} &= \left\{ \frac{\partial \left[\pi(x_{\{k,2\}}, \hat{\beta})\rho(x_{\{k,2\}}, x_{\{k,4\}})\right]}{\partial x_{k,2}}\Bigg|_0 \right\} \\ &= \left\{ \frac{\partial \left[\pi(x_{\{k,2\}}, \hat{\beta})\right]}{\partial x_{k,2}} \rho(x_{\{k,2\}}, x_{\{k,4\}}) + \pi(x_{\{k,2\}}, \hat{\beta}) \frac{\partial \left[\rho(x_{\{k,2\}}, x_{\{k,4\}})\right]}{\partial x_{k,2}}\Bigg|_0 \right\} \\ &= \left\{ \hat{\beta}_2 exp\{-z_k\} \left[\hat{\sigma}(z_k)\right]^2 * \rho(x_{\{k,2\}}, x_{\{k,4\}}) + \pi(x_{\{k,2\}}, \hat{\beta})\Big|_0 \right\} \end{aligned} \tag{15}$$

$$\begin{aligned} \left\{ \frac{\partial^2 f}{\partial x_{k,2}^2}\Bigg|_0 \right\} &= \left\{ \frac{\partial}{\partial x_{k,2}} \left[\hat{\beta}_2 exp\{-z_k\}\hat{\sigma}(z_k) * \rho(x_{\{k,2\}}, x_{\{k,4\}}) + \pi(x_{\{k,2\}}, \hat{\beta})\right]\Bigg|_0 \right\} \\ &= \left(-\hat{\beta}_2^2 exp\{-z_k\} * \left[\hat{\sigma}(z_k)\right]^2 + 2\hat{\beta}_2^2 \left[exp\{-z_k\}\right]^2 * \left[\hat{\sigma}(z_k)\right]^3 \right) * \rho(x_{\{k,2\}}, x_{\{k,4\}}) + \\ &+ \hat{\beta}_2 exp\{-z_k\} \left[\hat{\sigma}(z_k)\right]^2 + \hat{\beta}_2 exp\{-z_k\} \left[\hat{\sigma}(z_k)\right]^2 |_0 \end{aligned} \tag{16}$$

$$\begin{aligned} \left\{ \frac{\partial c}{\partial x_{k,2}}\Big|_0 \right\} &= \left\{ \frac{\partial \left[\left[\pi(x_{\{k,2\}}, \hat{\beta}) * x_{k,3}\right] - B\right]}{\partial x_{k,2}}\Bigg|_0 \right\} \\ &= \left\{ \hat{\beta}_2 exp\{-z_k\} \left[\pi(x_{\{k,2\}}, \hat{\beta})\right]^2 * x_{k,3}\Big|_0 \right\} \end{aligned} \tag{17}$$

$$\begin{aligned} \left\{ \frac{\partial^2 c}{\partial x_{k,2}^2}\Bigg|_0 \right\} &= \left\{ \hat{\beta}_2 \frac{\partial exp\{-z_k\}}{\partial x_{k,2}} * \left[\pi(x_{\{k,2\}}, \hat{\beta})\right]^2 * x_{k,3} + \hat{\beta}_2 exp\{-z_k\} * \frac{\partial}{\partial x_{k,2}} \left[\pi(x_{\{k,2\}}, \hat{\beta})\right]^2 * x_{k,3}\Big|_0 \right\} \\ &= \left\{ -\hat{\beta}_2^2 exp\{-z_k\} * \left[\pi(x_{\{k,2\}}, \hat{\beta})\right]^2 * x_{k,3} + 2\hat{\beta}_2^2 exp\{-2z_k\} * \left[\pi(x_{\{k,2\}}, \hat{\beta})\right]^3 * x_{k,3}\Big|_0 \right\} \end{aligned} \tag{18}$$

We can now optimize as follows:

Input: $\hat{\beta}_0^*, \hat{\beta}_2$ the estimated coefficients from the logistic model.

Output: $x_{\{k,2\}}$ the optimized price variable.

-1 Initialize points $x^o_{\{k,2\}}$ and λ^o.

-2 Whiles not (11) do:

- Compute (15), (16), (17) and (18).

- Obtain a new $x_{\{k,2\}}$ using (12) and (14).

Step 4: Implementation

```
# Initialize the price variables for the three segments:
#x21 <- mean((data.sim %>% filter(x1=="s1"))$x2)
#x22 <- mean((data.sim %>% filter(x1=="s2"))$x2)
#x23 <- mean((data.sim %>% filter(x1=="s3"))$x2)
#x31 <- mean((data.sim %>% filter(x1=="s1"))$x3)
#x32 <- mean((data.sim %>% filter(x1=="s2"))$x3)
#x33 <- mean((data.sim %>% filter(x1=="s3"))$x3)
# Initialize the lambda
#lambda <- 0.2
```

```
#K <- 20
#loan_amount_vector <- c(B, B, B)
#booking_cost <- 10
#variables_vector <- c(1, 0, 0, x21)
#coefficients_vector <- c(seg1.intercept, coef2, coef3, price.coef)
```

Define the price indexes

```
INTR_INDEX = 1
PRICE_INDEX = 2
LAMT_INDEX = 3
```

Build the utility functions here

```
# Function to compute the exponent of the linear form:
compute_exp_lin_form <- function(variables_vector, coefficients_vector){
  # compute the linear form
  ln.form <- variables_vector*coefficients_vector
  return(exp(-sum(ln.form)))
}

# Unit test
#print(compute_exp_lin_form(variables_vector, coefficients_vector))
```

```
# Function to compute the estimated predicted probabilities:
compute_sigma_hat <- function(variables_vector, coefficients_vector){
  # compute the linear form
  ln.form <- variables_vector*coefficients_vector
  return <- (1 + compute_exp_lin_form(variables_vector, coefficients_vector) )^(-1)
}
# Unit test
#print(compute_sigma_hat(variables_vector, coefficients_vector))
```

```
# Function to compute the 1st partial derivative of f w.r.t. the price: (15)
compute_first_pd_of_f <- function(booking_cost, variables_vector, coefficients_vector){
  # first compute sigma of z-k using (1)
  price_coefficient  = coefficients_vector[PRICE_INDEX] # it's assumed this is in the third place
  lin_form <- compute_exp_lin_form(variables_vector, coefficients_vector)
  sigma_value <- compute_sigma_hat(variables_vector, coefficients_vector)
```

```
  # Implement the equation corresponding to (15)
  return_value <- (price_coefficient*lin_form*(sigma_value^2)*
                     (variables_vector[PRICE_INDEX] - booking_cost)) + sigma_value
  #print(variables_vector)
  #print(paste("New First PD of f in CFPDF: ", return_value))
  return(return_value)
}
#compute_first_pd_of_f(booking_cost, variables_vector, coefficients_vector)
```

```
# Function to compute the 2nd partial derivative of f w.r.t. the price: (16)
compute_second_pd_of_f <- function(booking_cost, variables_vector, coefficients_vector){
  # first compute sigma of z-k using (1)
  price_coefficient  = coefficients_vector[PRICE_INDEX] # it's assumed this is in the third place
  lin_form <- compute_exp_lin_form(variables_vector, coefficients_vector)
  sigma_value <- compute_sigma_hat(variables_vector, coefficients_vector)
  # Implement the equation corresponding to (16)
  first_half <- ( ((-price_coefficient^2)*lin_form*sigma_value^2) +
                    (2*(price_coefficient^2)*(lin_form^2)*(sigma_value^3))
                  )*(variables_vector[PRICE_INDEX] - booking_cost)
  second_half <- (price_coefficient*lin_form*sigma_value^2
                  )+(price_coefficient*lin_form*(sigma_value^2))

  return_value <- first_half + second_half
  #print(paste("New Second PD of f in CSPDF: ", return_value))
  return(return_value)
}

# Function to compute the 1st partial derivative of c w.r.t. the price: (17)
compute_first_pd_of_c <- function(loan_amount_vector, variables_vector,
                                  coefficients_vector){
  # Note: Loan amount will now simply contain a single item.
  # first compute sigma of z-k using (1)
  price_coefficient  = coefficients_vector[PRICE_INDEX] # it's assumed this is in the third place
  # first do for segment 1
  var_vec1 = variables_vector
  lin_f1 <- compute_exp_lin_form(var_vec1, coefficients_vector)
  sigma_v1 <- compute_sigma_hat(var_vec1, coefficients_vector)
  seg <- (price_coefficient*lin_f1*(sigma_v1)^2)*loan_amount_vector[1]

  return_value <- seg
  #print(paste("New PD of C in CFPDC: ", return_value))
  return(return_value)
}

# Function to compute the 1st partial derivative of c w.r.t. the price: (17)
compute_second_pd_of_c <- function(loan_amount_vector, variables_vector,
                                   coefficients_vector){
  # first compute sigma of z-k using (1)
  price_coefficient  = coefficients_vector[PRICE_INDEX] # it's assumed this is in the third place
  # first do for segment 1
  var_vec1 = variables_vector
  lin_f1 <- compute_exp_lin_form(var_vec1, coefficients_vector)
  sigma_v1 <- compute_sigma_hat(var_vec1, coefficients_vector)
```

```
  seg <- (
    (-price_coefficient^2)*lin_f1*(sigma_v1)^2)*loan_amount_vector[1] +
    ( 2*(price_coefficient^2)*(lin_f1^2)*(sigma_v1^2)*loan_amount_vector[1]   )
  return_value <- seg
  #print(paste("New PD of C in CFPDC: ", return_value))
  return(return_value)
}

compute_c <- function(loan_amount_vector, variables_vector, coefficients_vector, B){
  # first compute sigma of z-k using (1)
  price_coefficient  = coefficients_vector[PRICE_INDEX] # it's assumed this is in the third place
  # first do for segment 1
  var_vec1 = variables_vector
  sigma_v1 <- compute_sigma_hat(var_vec1, coefficients_vector)
  seg <- (sigma_v1)*loan_amount_vector[1]

  # Implement the equation corresponding to (15)
  return_value <- seg
  #print(variables_vector)
  #print(paste("Computed C in CC: ", return_value))
  return(return_value)
}
```

```
# Function to compute the new lambda
compute_new_lambda <- function(loan_amount_vector, booking_cost,
                               variables_vector, coefficients_vector, B){
  price_var = variables_vector[PRICE_INDEX]
  first_half = (
    compute_first_pd_of_f(
    booking_cost, variables_vector,coefficients_vector
    ) )/compute_first_pd_of_c(loan_amount_vector, variables_vector, coefficients_vector)
  second_half = (
    (
      compute_second_pd_of_f(
      booking_cost, variables_vector, coefficients_vector
      ) - compute_second_pd_of_c(loan_amount_vector, variables_vector, coefficients_vector)
    )*compute_c(
        loan_amount_vector, variables_vector, coefficients_vector, B
        ) )/(
          compute_first_pd_of_c(loan_amount_vector, variables_vector, coefficients_vector
                                )^2)
  return_value = first_half - second_half
  #print(paste("New Lambda in CNL: ", return_value))
  return(return_value)
}
# Function to compute the new price
compute_new_price <- function(
  lambda, loan_amount_vector, booking_cost, variables_vector, coefficients_vector
  ){
  price_var = variables_vector[PRICE_INDEX]
  first_half=(
  lambda*compute_first_pd_of_c(loan_amount_vector, variables_vector, coefficients_vector)
    ) - compute_first_pd_of_f(booking_cost, variables_vector, coefficients_vector)
```

```
  #print(paste("First Half in CNP: ", first_half))
  second_half=compute_second_pd_of_f(booking_cost, variables_vector, coefficients_vector) -
    (lambda * compute_second_pd_of_c(loan_amount_vector, variables_vector,
                                     coefficients_vector) )
  #print(paste("Second Half in CNP: ", second_half))
  return_value = (first_half/second_half) #+ price_var
  #print(variables_vector)
  #print(paste("New Price in CNP: ", return_value))
  return(return_value)
}

# Function to check if the minimum has been achieved: (11)
check_convergence <- function(new_price, lambda, loan_amount_vector, booking_cost, variables_vector, coe
  price_var = variables_vector[PRICE_INDEX]
  first_half = - (
  lambda*compute_first_pd_of_c(loan_amount_vector, variables_vector, coefficients_vector)
    ) + compute_first_pd_of_f(booking_cost, variables_vector, coefficients_vector)
  #print(paste("First Half in CNP: ", first_half))
  second_half=compute_second_pd_of_f(booking_cost, variables_vector, coefficients_vector) -
    (lambda * compute_second_pd_of_c(loan_amount_vector, variables_vector,
                                     coefficients_vector) )
  #print(paste("Second Half in CNP: ", second_half)
  return_value = (first_half) + (new_price-price_var)*second_half
  return(return_value)
}
```

```
#data_values_s1 <- data_values_s1 %>% mutate(segment = factor(rep(1, K)))
#data_values_s2 <- data_values_s2 %>% mutate(segment = factor(rep(2, K)))
#data_values_s3 <- data_values_s3 %>% mutate(segment = factor(rep(3, K)))
#head(data_values_s1)
```

```
#merged_data_values <- rbind(data_values_s1, data_values_s2, data_values_s3)
#head(merged_data_values)
```

```
#library(ggplot2)
#ggplot(data = merged_data_values) + geom_line(aes(x = index, y = price, color = segment))
```

```
#ggplot(data = merged_data_values) + geom_line(aes(x = index, y = lambda, color = segment))
```

```
#ggplot(data = merged_data_values) + geom_line(aes(x = index, y = secp, color = segment))
```