Customer Team 2

Weihan Weng

2023-11-12

1. Predict y (i.e., the decision to join the club) as a function of the available scoring variables x (gender and all hl...) using a LOGIT model on the Training list. Include an intercept term to account for a base response rate. Keep all coefficients (i.e., do not eliminate coefficients which seems to be statistically insignificant). Hand-in: Report coefficients and p-values.

```
setwd("C:/Users/Weihan Weng/Downloads")

training = read.csv("Data_Estimation_R.csv")

str(training)
```

```
## 'data.frame':    200 obs. of  8 variables:
##  $ id      : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ location: int  1 0 0 1 0 1 1 1 0 0 ...
##  $ hl1     : int  302 221 202 148 43 183 163 474 446 113 ...
##  $ hl2     : int  0 0 9 0 0 0 0 9 0 0 ...
##  $ hl3     : int  0 10 45 15 15 0 0 40 20 0 ...
##  $ hl5     : int  0 12 0 0 0 12 0 0 12 0 ...
##  $ hl6     : int  0 26 13 0 0 0 0 0 26 15 ...
##  $ y       : int  1 0 0 0 0 0 1 1 1 0 ...
```

```
summary(training)
```

```
##        id            location          hl1             hl2
##  Min.   :  1.00   Min.   :0.000   Min.   : 16.0   Min.   : 0.00
##  1st Qu.: 50.75   1st Qu.:0.000   1st Qu.:120.8   1st Qu.: 0.00
##  Median :100.50   Median :0.000   Median :191.5   Median : 0.00
##  Mean   :100.50   Mean   :0.325   Mean   :205.9   Mean   : 4.08
##  3rd Qu.:150.25   3rd Qu.:1.000   3rd Qu.:278.0   3rd Qu.: 9.00
##  Max.   :200.00   Max.   :1.000   Max.   :476.0   Max.   :57.00
##       hl3             hl5             hl6              y
##  Min.   : 0.00   Min.   : 0.00   Min.   : 0.00   Min.   :0.00
##  1st Qu.: 0.00   1st Qu.: 0.00   1st Qu.: 0.00   1st Qu.:0.00
##  Median :10.00   Median : 0.00   Median : 0.00   Median :0.00
##  Mean   :10.65   Mean   : 2.88   Mean   : 6.24   Mean   :0.36
##  3rd Qu.:15.00   3rd Qu.: 0.00   3rd Qu.:13.00   3rd Qu.:1.00
##  Max.   :60.00   Max.   :39.00   Max.   :69.00   Max.   :1.00
```

```
glm.training <- glm(y ~ location + hl1 + hl2 + hl3 + hl5 + hl6,
                    family = binomial(link = "logit"), data = training)
summary(glm.training)
```

```
##
## Call:
## glm(formula = y ~ location + hl1 + hl2 + hl3 + hl5 + hl6, family = binomial(link = "logit"),
##     data = training)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.928404   0.361689  -2.567  0.01026 *
## location    -0.016632   0.344941  -0.048  0.96154
## hl1          0.005733   0.001840   3.115  0.00184 **
## hl2         -0.045830   0.026570  -1.725  0.08455 .
## hl3         -0.068239   0.017004  -4.013 5.99e-05 ***
## hl5          0.004349   0.026228   0.166  0.86830
## hl6         -0.004919   0.017404  -0.283  0.77746
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 261.37  on 199  degrees of freedom
## Residual deviance: 234.26  on 193  degrees of freedom
## AIC: 248.26
##
## Number of Fisher Scoring iterations: 4
```

2. Based on your logit model, score all individuals on in the Testing list (you can do this manually, e.g., in Excel, or adapt the R code from class). This means calculate, for all prospects in the Testing sample, the predicted response rate. Using your model, compute (for each individual):

(a) Prdicted Response Rate

(b) Lift

Hand-in: Results for the first 10 Names

```
testing = read.csv("Data_Holdout_R.csv")
str(testing)
```

```
## 'data.frame':    300 obs. of  8 variables:
##  $ id      : int  201 202 203 204 205 206 207 208 209 210 ...
##  $ location: int  0 1 1 1 0 0 1 0 0 0 ...
##  $ hl1     : int  158 187 313 310 37 78 427 30 286 249 ...
##  $ hl2     : int  0 0 0 0 9 0 9 0 0 0 ...
##  $ hl3     : int  0 0 25 0 0 0 10 0 15 10 ...
##  $ hl5     : int  0 0 0 0 0 0 0 0 0 24 ...
##  $ hl6     : int  13 0 0 0 0 0 26 13 0 26 ...
##  $ y       : int  1 0 0 1 1 1 1 1 0 0 ...
```

```
summary(testing)
```

```
##        id           location         hl1             hl2             hl3
##  Min.   :201.0   Min.   :0.00   Min.   : 17.0   Min.   : 0.00   Min.   : 0.00
##  1st Qu.:275.8   1st Qu.:0.00   1st Qu.:135.0   1st Qu.: 0.00   1st Qu.: 0.00
```

```
##  Median :350.5   Median :0.00   Median :219.0   Median : 0.00   Median :10.00
##  Mean   :350.5   Mean   :0.28   Mean   :216.8   Mean   : 4.18   Mean   :13.28
##  3rd Qu.:425.2   3rd Qu.:1.00   3rd Qu.:291.5   3rd Qu.: 9.00   3rd Qu.:20.00
##  Max.   :500.0   Max.   :1.00   Max.   :474.0   Max.   :33.00   Max.   :70.00
##       hl5             hl6              y
##  Min.   : 0.00   Min.   : 0.000   Min.   :0.0000
##  1st Qu.: 0.00   1st Qu.: 0.000   1st Qu.:0.0000
##  Median : 0.00   Median : 0.000   Median :0.0000
##  Mean   : 3.74   Mean   : 6.253   Mean   :0.3333
##  3rd Qu.: 0.00   3rd Qu.:13.000   3rd Qu.:1.0000
##  Max.   :39.00   Max.   :56.000   Max.   :1.0000
```

```
#Predicting buy/no buy
prediction <- data.frame(
  ID = testing$id,
  ResponseProb = predict(glm.training, testing, type = c("response")),
  ResponsePredict = round(predict(glm.training, testing, type = c("response")), digits = 0)
)

prediction$ActualResponse = testing$y #add actual response
prediction$Lift = prediction$ResponseProb/mean(training$y) #add lift

print(head(prediction, 10))
```

```
##     ID ResponseProb ResponsePredict ActualResponse      Lift
## 1  201    0.4783915               0              1 1.3288654
## 2  202    0.5317304               1              0 1.4770288
## 3  203    0.2980727               0              0 0.8279797
## 4  204    0.6968386               1              1 1.9356628
## 5  205    0.2443930               0              1 0.6788694
## 6  206    0.3819674               0              1 1.0610205
## 7  207    0.5696267               1              1 1.5822965
## 8  208    0.3056892               0              1 0.8491368
## 9  209    0.4225614               0              0 1.1737816
## 10 210    0.4485039               0              0 1.2458442
```

3. Sort the Testing list in decreasing order of lift.

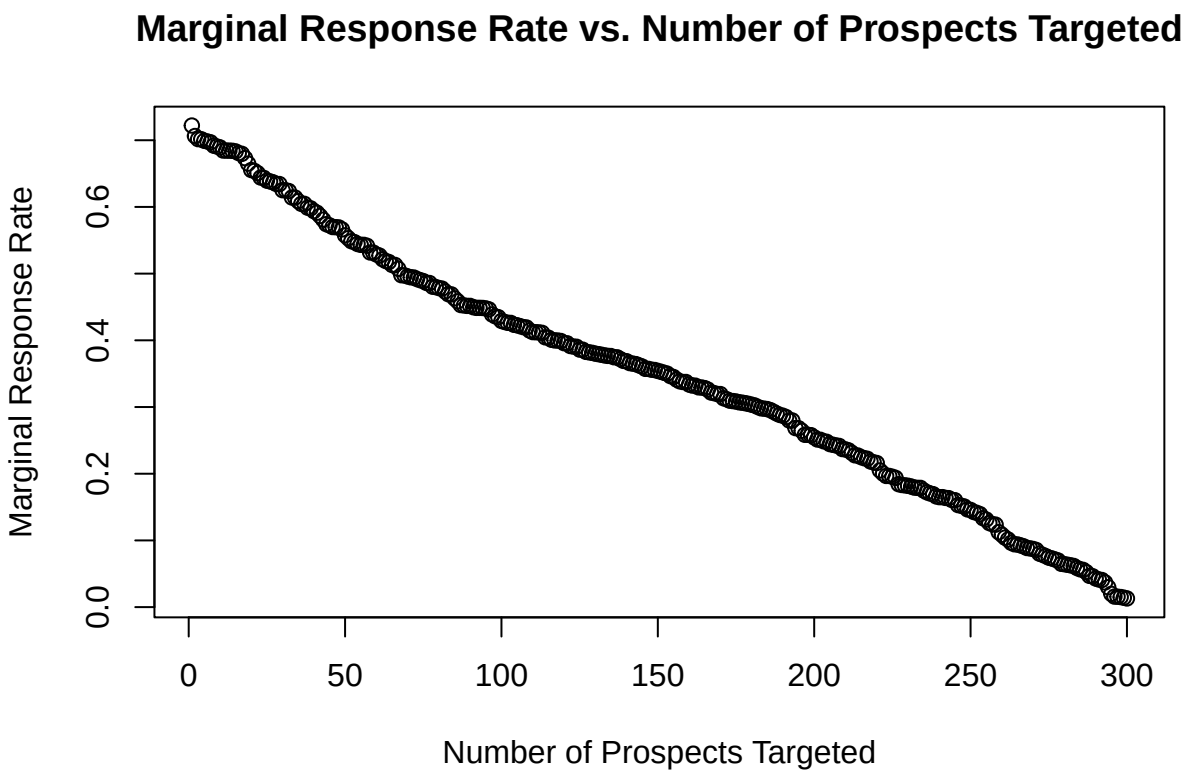
```
prediction.sorting <- prediction[order(prediction$Lift, decreasing = TRUE), ]

print(head(prediction.sorting, 10))
```

```
##      ID ResponseProb ResponsePredict ActualResponse     Lift
## 131 331    0.7220329               1              0 2.005647
## 192 392    0.7063219               1              1 1.962005
## 20  220    0.7016614               1              1 1.949059
## 160 360    0.7015425               1              1 1.948729
## 101 301    0.6991360               1              0 1.942045
## 285 485    0.6980484               1              0 1.939023
## 4   204    0.6968386               1              1 1.935663
## 291 491    0.6918510               1              0 1.921808
## 298 498    0.6906274               1              1 1.918409
## 132 332    0.6892722               1              1 1.914645
```

4. Plot Marginal Response Rate vs. Number of Prospects Targeted

```
plot(prediction.sorting$ResponseProb,
     main = "Marginal Response Rate vs. Number of Prospects Targeted",
     xlab = "Number of Prospects Targeted", ylab = "Marginal Response Rate")
```



5. We know that average CLV is $30 and the solicitation cost is $12. Based on the Marginal Cost Rule determine who the CD club should send invitations to from the Testing list.

$12/30 = 0.4$, so the firm should target as long as the marginal response rate exceeds 0.4.

The CD club should send the invitations to these IDs:

331 392 220 360 301 485 204 491 498 332 342 243 444 309 446 401 219 320 438 400 269 455 251 258 300 313 456 343 215 356 443 275 493 227 241 357 462 217 225 338 268 482 264 417 337 379 207 441 273 257 376 494 330 293 327 415 500 202 380 366 324 329 325 230 428 463 233 214 292 261 354 397 222 351 224 460 290 419 289 201 394 256 458 427 277 344 420 368 450 212 488 299 478 210 470 403 474 461 359 459 311 483 391 288 209 270 439 254 294 495 339 422 365 370 414 431

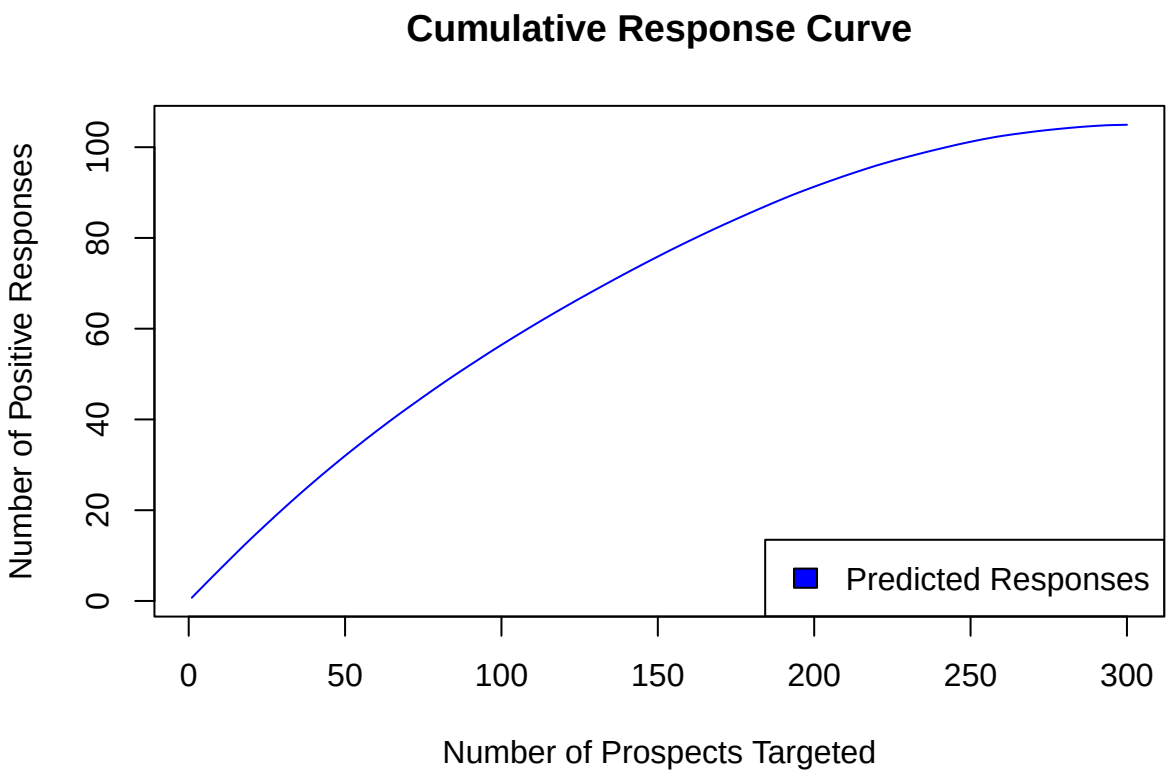
6. Compute the Cumulative Sum (aka running sum) for the Predicted Response Rates in decreasing order for the Testing list. Plot the curve for Number of Positive Responses vs. Number of Prospects Targeted.

```
CummlativeSum <- data.frame(matrix(ncol = 2, nrow = 300))
CummlativeSum$y_predicted = cumsum(prediction.sorting$ResponseProb)

plot(CummlativeSum$y_predicted,
     main = "Cumulative Response Curve",
```

```
     xlab = "Number of Prospects Targeted", ylab = "Number of Positive Responses",
     type = "l",
     col = "blue")

legend("bottomright",
       c("Predicted Responses"),
       fill = c("blue"))
```



7. The CD club has only 40 items of the collector's edition of "Pink Floyd's The Wall". Based on the Limited Supply Rule, which prospects (and how many) on the Testing list should the CD club send an invitation to?

```
supply <- 40

# Initialize index variable
last_index_less_than_40 <- 0

# Loop to find the index of the last value less than 40
for (i in seq_along(CummlativeSum$y_predicted)) {
  if (CummlativeSum$y_predicted[i] < supply) {
    last_index_less_than_40 <- i
  }
}

# Print the result
print(paste("If there are only", supply, "items, they should send",
```

```
            last_index_less_than_40, "invitations."))
```

```
## [1] "If there are only 40 items, they should send 64 invitations."
```

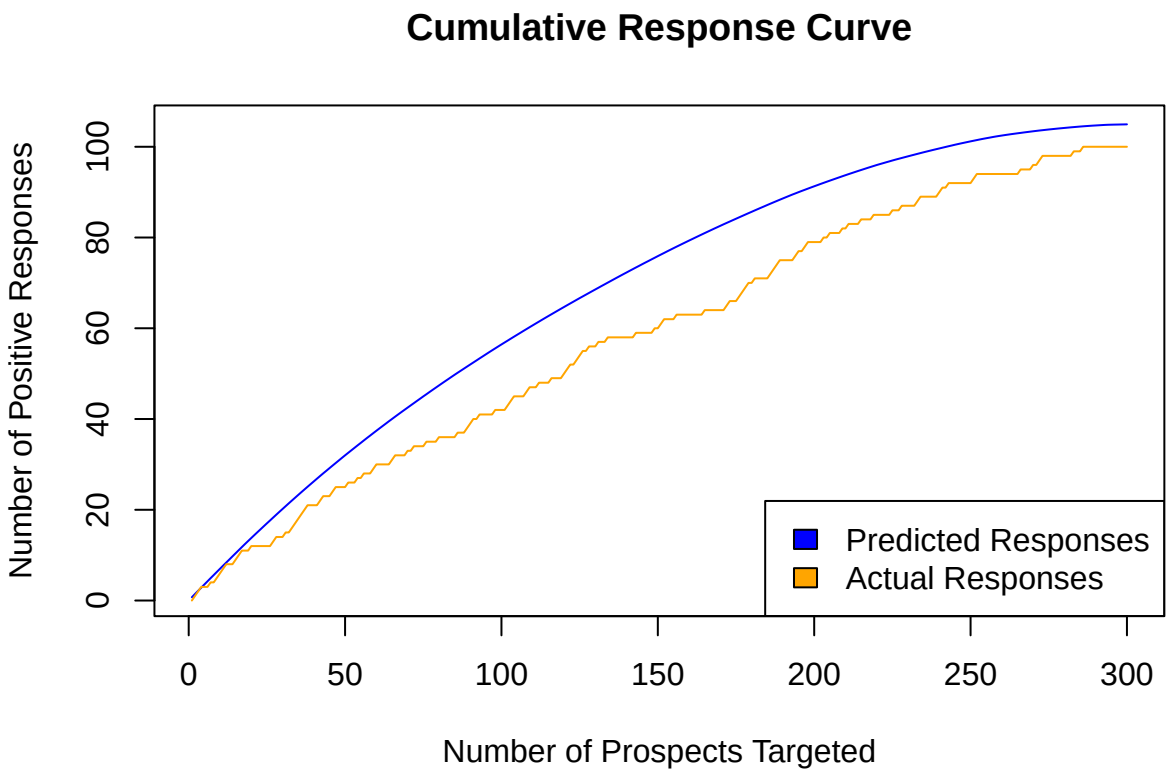
8. Compute the Cumulative Sum (aka running sum) for the Actual Response Rate (recall this is either 0 or 1) in decreasing order of Predicted Response Rate. Plot the curve for curve for number of Actual Positive Responses vs. Number of Prospects Targeted. Superimpose on this the curve obtained in step 6 above. Using the chart, comment on the differences between the Actual Response Rates and the Predicted Response Rates for the prospects in the Testing list. What is the impact on your results in step 7?

```
CummlativeSum <- data.frame(matrix(ncol = 2, nrow = 300))
CummlativeSum$y_predicted = cumsum(prediction.sorting$ResponseProb)
CummlativeSum$y_real = cumsum(prediction.sorting$ActualResponse)

plot(CummlativeSum$y_predicted,
     main = "Cumulative Response Curve",
     xlab = "Number of Prospects Targeted", ylab = "Number of Positive Responses",
     type = "l",
     col = "blue")

lines(CummlativeSum$y_real,
      col = "orange",
      type = "l")

legend("bottomright",
       c("Predicted Responses", "Actual Responses"),
       fill = c("blue", "orange"))
```




```
invitations <- 64

# Find the 64th value in CummlativeSum$y_real
Acknowledgment <- CummlativeSum$y_real[invitations]

# Print the result
print(paste("If the firm sent", invitations, "invitations, the actual responses would be",
            Acknowledgment, "responses."))
```

```
## [1] "If the firm sent 64 invitations, the actual responses would be 30 responses."
```

We can see that the number of actual responses is lower than prediction.To get the maximum expected response, companies can send more invitations based on predictions. The actual response rate is generally a little below our predicted response rate. This points out that our model overestimates the response rate.

Based on the prediction, you are targeting 64 prospects since you have only 40 items, but the real response turns out less sales if you target 64 prospects only. Accordingly, we have a higher probability of sending more than 64 invitations to meet the 40-item standard.