Linear vs. Logistic Regression: A Study on Low Birth Weight

Will Bliss

5/10/2022

A large concern for newborn children is their weight at birth. An infant is diagnosed with low birth weight if it weights less than 2500 grams. This is a concern as low birth weight babies are at higher risk of infant mortality and birth defects. During pregnancy, a woman's traits (physically and behaviorally) can lead to an infant having normal birth weight. A study was done by University Baystate Medical Center Springfield, Massachusetts to collect data on 189 women who gave birth. Their medical information was recorded in attempt to recognize which common medical features lead to a woman giving birth to an infant with low or normal birth weight.

For this project, we will apply linear and logistic regression to the dataset. With each type of regression, we will arrive at a model. We will use each model to try to predict the outcome (low birth weight or not) based on the woman's medical features.

For linear regression, we use the `BWT` variable, which is the weight of the newborn in grams. For logistic regression, we use the `LOW` variable. Here, `LOW` $= 1$ when `BWT` ≤ 2500g, and `LOW` $= 0$ when `BWT` > 2500g. In other words, `BWT` is 1 when the newborn is classified as having low birth weight, and 0 otherwise. A glimpse of the data is shown below:

```r
lowbwt <- read.csv("lowbwt.csv")
head(lowbwt)
```

```
##   ID LOW AGE LWT RACE SMOKE PTL HT UI FTV  BWT
## 1  4   1  28 120    3     1   1  0  1   0  709
## 2 10   1  29 130    1     0   0  0  1   2 1021
## 3 11   1  34 187    2     1   0  1  0   0 1135
## 4 13   1  25 105    3     0   1  1  0   0 1330
## 5 15   1  25  85    3     0   0  0  1   0 1474
## 6 16   1  27 150    3     0   0  0  0   0 1588
```

Let's clean the data that we are working with. We:

- remove the `ID` variable
- create dummy variable for race

```r
data <- lowbwt[,-1]

## white: this is when black = 0 & other = 0
## data$WhiteRace <- as.numeric(data$RACE == 1)

## black
data$BlackRace <- as.numeric(data$RACE == 2)
```

```r
## other
data$OtherRace <- as.numeric(data$RACE == 3)

## remove old race variable
data <- data[,-4]
```

Now, lets put $\frac{2}{3}$ of the data into a training dataset, and the remaining one third into a testing dataset:

```r
trainObs <- sample(c(TRUE, FALSE), nrow(data), replace = TRUE, prob = c(0.667, .333))
dataTrain <- data[trainObs,]
dataTest <- data[!trainObs,]
```

We have the following training dataset:

```r
## col 1 (LOW) = binary response
## col 9 (BWT) = numerical response
trainActual <- dataTrain$LOW
testActual <- dataTest$LOW

head(dataTrain)
```

```
##    LOW AGE LWT SMOKE PTL HT UI FTV  BWT BlackRace OtherRace
## 2    1  29 130     0   0  0  1   2 1021         0         0
## 4    1  25 105     0   1  1  0   0 1330         0         1
## 5    1  25  85     0   0  0  1   0 1474         0         1
## 8    1  24 128     0   1  0  0   1 1701         1         0
## 9    1  24 132     0   0  1  0   0 1729         0         1
## 10   1  21 165     1   0  1  0   1 1790         0         0
```

First, let's perform linear regression on the data to try to predict the test data:

```r
## MANUAL Linear regression
xCols <- c(2,3,4,5,6,7,8,10,11)
betaZeroTrain <- matrix(1, ncol = 1, nrow(dataTrain))
XTrain <- as.matrix(cbind(betaZeroTrain, dataTrain[,xCols]))
XTXInverse <- solve(t(XTrain) %*% XTrain)
## dataTrain$LOW is Y vector, would not let me assign it to a variable
betas <- XTXInverse %*% t(XTrain) %*% dataTrain$BWT

YHatTrain <- XTrain %*% betas
trainPredict <- ifelse(YHatTrain <= 2500, 1, 0)

## Compute training error
compareTrain <- data.frame(pred = trainPredict, actual = dataTrain$LOW)
errorTrain <- mean(compareTrain$actual != compareTrain$pred)
errorTrain
```

```
## [1] 0.3333333
```

```r
## Repeat for Test data
betaZeroTest <- matrix(1, ncol = 1, nrow(dataTest))
XTest <- as.matrix(cbind(betaZeroTest, dataTest[,xCols]))

YHatTest <- XTest %*% betas
testPredict <- ifelse(YHatTest <= 2500, 1, 0)

compareTest <- data.frame(pred = testPredict, actual = dataTest$LOW)
errorTest <- mean(compareTest$actual != compareTest$pred)
errorTest
```

```
## [1] 0.2698413
```

```r
successProb <- 1-errorTest
successProb
```

```
## [1] 0.7301587
```

Using linear regression, we were able to correctly classify the child as having low birth weight or not 73.0159 percent of the time. Below is a table of the first 15 observations and predictions:

```r
valuesLinear <- data.frame(Actual = compareTest$actual, Predicted = compareTest$pred)

head(valuesLinear, 15)
```

```
##    Actual Predicted
## 1       1         0
## 2       1         0
## 3       1         0
## 4       1         0
## 5       1         0
## 6       1         1
## 7       1         0
## 8       1         0
## 9       1         0
## 10      1         0
## 11      1         0
## 12      1         1
## 13      1         0
## 14      1         0
## 15      1         0
```

and now the last 15 observations and predictions:

```r
tail(valuesLinear, 15)
```

```
##    Actual Predicted
## 49      0         0
## 50      0         0
## 51      0         0
## 52      0         0
## 53       0          0
## 54       0          0
## 55       0          0
## 56       0          0
## 57       0          0
## 58       0          0
## 59       0          0
## 60       0          0
## 61       0          0
## 62       0          0
## 63       0          0
```

Let's look at the classifications for 1's and 0's

```
## correct proportion of 1's
linearOneLocations <- which(valuesLinear$Actual==1)
correctOnesLin <- mean(valuesLinear$Predicted[linearOneLocations] == 1)
correctOnesLin
```

```
## [1] 0.1578947
```

```
##correct proportion of 0's
linearZeroLocations <- which(valuesLinear$Actual==0)
correctZerosLin <- mean(valuesLinear$Predicted[linearZeroLocations]==0)
correctZerosLin
```

```
## [1] 0.9772727
```

A 1 is correctly classified 15.79% of the time. A 0 is correctly classified 97.73% of the time. While it is less than idea to have a correct prediction of 1's only 15.79% of the time, it is good that we are correctly identifying those with normal birth weight almost every time.

Now, let's try performing logistic regression on the dataset using iterative least squares:

```
X <- dataTrain[,xCols]
## remove numerical birth weight response
dataTrain <- dataTrain[,-9]
beta0 <- rep(1, nrow(dataTrain))
X <- cbind(beta0, X)
X <- as.matrix(X)

Y <- dataTrain$LOW

beta <- as.matrix(rep(0, ncol(X)), ncol = 1)
p <- rep(1/2, nrow(dataTrain))
W <- diag(nrow(dataTrain))

## iterate
while(TRUE){
  z <- X %*% beta + solve(W) %*% (Y - p)
  temp <- beta
  beta <- beta + solve(t(X) %*% W %*% X) %*% t(X) %*% (Y - p)
  for(i in 1:nrow(dataTrain)){
    p[i] <- exp(t(beta) %*% X[i, ]) / (1 + exp(t(beta) %*% X[i, ]))
  }
  W <- diag(p * (1 - p))
  if(abs(temp - beta) < 0.01){
    break
  }
}

## output coefficients
beta
```

```
##                   [,1]
## beta0      2.20987991
## AGE       -0.09282192
## LWT       -0.01475998
## SMOKE      0.72223329
## PTL        0.50267465
## HT         1.39019632
## UI         0.44117399
## FTV        0.14891392
## BlackRace  1.26862344
## OtherRace  0.37482922
```

```
## store the names for later
betanames <- rownames(beta)
```

Now, we use bootstrap method to get the mean value of the coefficients:

```
## bootstrap
M <- 500
N <- nrow(dataTrain)
coefs <- list()

for(k in 1:M){
  Xboot <- matrix(rep(0), nrow = N, ncol = ncol(X))
  index <- sample(1:N, N, replace = TRUE)
  Y <- dataTrain[index ,1]

  for(j in 1:N){
    Xboot[j, ] <- X[index[j],]
  }

  beta <- as.matrix(rep(0, ncol(Xboot)), ncol = 1)
  p <- rep(1/2, nrow(dataTrain))
  W <- diag(nrow(dataTrain))

  ## iterate
  while(TRUE){
    z <- Xboot %*% beta + solve(W) %*% (Y - p)
    temp <- beta
    beta <- beta + solve(t(Xboot) %*% W %*% Xboot) %*% t(Xboot) %*% (Y - p)
    for(i in 1:nrow(dataTrain)){
      p[i] <- exp(t(beta) %*% Xboot[i, ]) / (1 + exp(t(beta) %*% Xboot[i, ]))
    }
    W <- diag(p * (1 - p))
    if(abs(temp - beta) < 0.01){
      break
    }
  }

  coefs[[k]] <- beta
}

means <- c()
for(i in 1:10){
  means <- c(means, mean(unlist(lapply(coefs, '[[', i))))
}
means <- data.frame(means)
rownames(means) <- betanames

## round to avoid scientific notation
round(means, 7)
```

```
##                means
## beta0      2.5910813
## AGE       -0.1008961
## LWT       -0.0170503
## SMOKE      0.7934139
## PTL        0.5946929
## HT         1.5187613
## UI         0.4267172
## FTV        0.1410788
## BlackRace  1.3733801
## OtherRace  0.3639543
```

Now that we have our estimated coefficient values, we calculate the outcome for each test observation. For the logistic regression, we will classify the predicted outcome as 1 if $P(\text{LOW} = 1) \geq 0.5$, and 0 otherwise.

```
YHatTrainLogit <- XTrain %*% beta
predictionTrain <- ifelse(exp(YHatTrainLogit)/(1+exp(YHatTrainLogit)) >= .5, 1, 0)
logitTrain <- mean(dataTrain$LOW == predictionTrain)
logitTrain
```

```
## [1] 0.6984127
```

```
YHatTestLogit <- XTest %*% beta
predictionTest <- ifelse(exp(YHatTestLogit)/(1+exp(YHatTestLogit)) >= .5, 1, 0)
logitTest <- mean(dataTest$LOW == predictionTest)
logitTest
```

```
## [1] 0.6507937
```

Using logistic regression, we were able to correctly classify the child as having low birth weight or not 65.0794 percent of the time. Below is a table of the first 15 observations and predictions:

```r
valuesLogit <- data.frame(Actual = dataTest$LOW, Predicted = predictionTest)

head(valuesLogit, 15)
```

```
##    Actual Predicted
## 1       1         0
## 3       1         1
## 6       1         0
## 7       1         0
## 11      1         0
## 12      1         1
## 15      1         0
## 20      1         0
## 23      1         1
## 33      1         0
## 34      1         0
## 35      1         1
## 43      1         1
## 45      1         0
## 46      1         0
```

and now the last 15 observations and predictions:

```r
tail(valuesLogit, 15)
```

```
##     Actual Predicted
## 154      0         0
## 155      0         0
## 157      0         0
## 159      0         0
## 160      0         0
## 163      0         0
## 167      0         0
## 168      0         0
## 170      0         0
## 171      0         0
## 174      0         0
## 176      0         0
## 179      0         0
## 182      0         0
## 188      0         0
```

```r
## correct proportion of 1's
logitOneLocations <- which(valuesLogit$Actual==1)
correctOnesLogit <- mean(valuesLogit$Predicted[logitOneLocations] == 1)
correctOnesLogit
```

```
## [1] 0.3157895
```

```r
##correct proportion of 0's
logitZeroLocations <- which(valuesLogit$Actual==0)
correctZerosLogit <- mean(valuesLogit$Predicted[logitZeroLocations]==0)
correctZerosLogit
```

```
## [1] 0.7954545
```

A 1 is correctly classified 31.58% of the time. A 0 is correctly classified 79.55% of the time.

Let's revisit the success rates in classifying the test datasets for both types of regression:

```r
df <- data.frame(LinearRegression = c(successProb, correctOnesLin, correctZerosLin),
                 LogisticRegression = c(logitTest, correctOnesLogit, correctZerosLogit))
rownames(df) <- c("Success Rate", "Value 1 Success Rate", "Value 0 Success Rate")
df
```

```
##                      LinearRegression LogisticRegression
## Success Rate                0.7301587          0.6507937
## Value 1 Success Rate        0.1578947          0.3157895
## Value 0 Success Rate        0.9772727          0.7954545
```

We can see that linear regression is slightly better at classifying the data than logistic regression as a whole. However, logistic regression has a higher success rate at correctly classifying a woman with a low birth weight child, which is the main focus of the project.

It should be noted that if I were to spend more time on this project, I would analyze each contributing variable and build a better model that only consists of significant variables.