STAT GR5206 Lab 6

Cheng Zhang cz2532

Instructions

Make sure that you upload an RMarkdown file to the canvas page (this should have a .Rmd extension) as well as the PDF output after you have knitted the file (this will have a .pdf extension). The files you upload to the Canvas page should be updated with commands you provide to answer each of the questions below. You can edit this file directly to produce your final solutions. The lab is due by the end of the semester.

Goal

The goal of Lab 6 is to estimate a logistic regression model via maximum likelihood. The optimization procedure used is Newton's method, which is the most common (and amazing) form of the gradient decent algorithm. A description of the dataset and logistic statistical model follow below:

Data Description

A university medical center urology group was interested in the association between prostate-specific antigen (PSA) and a number of prognostic clinical measurements in men with advanced prostate cancer. Data were collected on 97 men who were about to undergo radical prostatectomies. The 8 variables are:

Below we read in the dataset and name it **prostate**.

```
setwd("C:/Users/Alan_/Desktop/Semester1 Courses/STAT computing/Week 12/lab6")
prostate <- read.table("Lab6.txt")
head(prostate)
```

```
##      X1     X2     X3 X4 X5 X6 X7 Y
## 1 0.651 0.5599 15.959 50  0  0  0 6
## 2 0.852 0.3716 27.660 58  0  0  0 7
## 3 0.852 0.6005 14.732 74  0  0  0 7
## 4 0.852 0.3012 26.576 58  0  0  0 6
## 5 1.448 2.1170 30.877 62  0  0  0 6
## 6 2.160 0.3499 25.280 50  0  0  0 6
```

In our setting we create a new binary response variable Y, called high-grade cancer by letting $Y = 1$ if Gleason score equals 8, and $Y = 0$ otherwise (i.e., if Gleason score equals 6 or 7). The goal is to carry out a logistic regression analysis, where the response of interest is high-grade cancer (Y).

```
prostate$Y <- ifelse(prostate$Y==8,1,0)
head(prostate)

##      X1     X2     X3 X4 X5 X6 X7 Y
## 1 0.651 0.5599 15.959 50  0  0  0 0
## 2 0.852 0.3716 27.660 58  0  0  0 0
## 3 0.852 0.6005 14.732 74  0  0  0 0
## 4 0.852 0.3012 26.576 58  0  0  0 0
## 5 1.448 2.1170 30.877 62  0  0  0 0
## 6 2.160 0.3499 25.280 50  0  0  0 0

nrow(prostate)

## [1] 97
```

Logistic Model

Let $Y_1, Y_2, \dots, Y_{97}$ be independent Bernoulli random variables with expected values

Part 1: Fit Logistic Model Using glm()

Problem 1)

Use the base **R** function **glm()** to fit the logistic model. Run the code below:

```
model <- glm(Y~X1+X2+X3+X4+X5+X6+X7,
               data=prostate,
               family=binomial(link = "logit")
               )
model

##
## Call:  glm(formula = Y ~ X1 + X2 + X3 + X4 + X5 + X6 + X7, family = binomi
al(link = "logit"),
##     data = prostate)
##
## Coefficients:
## (Intercept)           X1           X2           X3           X4
##  -10.276610     0.055228     0.086869     0.001615     0.106633
##          X5           X6           X7
##   -0.071827    -1.154137     0.123324
##
## Degrees of Freedom: 96 Total (i.e. Null);  89 Residual
## Null Deviance:       101.4
## Residual Deviance: 61.57     AIC: 77.57
```

Problem 2)

Consider a respondent with the following characteristics:

Estimate the probability that this respondent is among the high-grade cancer group. Based on this estimated probability, do you believe that a respondent with these characteristics belongs to the high-grade cancer group? The **R** code is explicitly given below.

```
## Solution goes here
logistic_pred <- predict(model,newdata=data.frame(X1=21.3,X2=8.4,X3=48.4,X4=6
8,X5=4.7,X6=0,X7=3.2))

probs <- exp(logistic_pred)/(1+exp(logistic_pred))
probs

##         1
## 0.2720329
```

The probability is just 0.272. So,a respondent with these characteristics do not belongs to the high-grade cancer group.

Part 1: Maximum Likelihood Estimation and Newton's Method

In Model (), the response values represent high-grade cancer and the features $X_1, X_2, \ldots, X_7$ are outlined in the data description from earlier in this document. To estimate Model (), we use the method of *Maximum Likelihood*. The objective function of interest (log-likelihood) is

The above log-likelihood is the same function derived in class except we have several more parameters to estimate, i.e., $\beta_0, \beta_1, \ldots, \beta_7$. In class we only considered *simple logistic regression* (one feature).

Problem 3)

Create a function in **R** called **logistic.NLL** with inputs **beta** and **data**, where **beta** is a vector of β coefficients and **data** is a dataframe defaulted by **data=prostate**. The function **logistic.NLL** should output the negative of the log-likelihood, i.e.,

$$-\ell(\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6, \beta_7).$$

Recall that maximizing the log-likelihood is equivalent to minimizing the negative log-likelihood. Also evaluate the function using the vector **beta=rep(0,8)**, i.e., run the code **logistic.NLL(beta=rep(0,8))**.

```
logistic.NLL <- function(beta,data=prostate) {
  b0 <- beta[1]
  bi <- beta[2:8]
  x <- data[,1:7]
  l = 0
  for (i in 1:dim(data)[1]){
    l = l + data[i,8]*(b0 + sum(bi*x[i,])) - log(1+exp(b0 + sum(bi*x[i,])))
  }

  neg.l = -l
  return(neg.l)

}
logistic.NLL(beta=rep(0,8))

## [1] 67.23528
```

Problem 4)

Write a **R** function called **Newtons.Method** that performs *Newton's Optimization Method* on a generic function **f.** The function should have inputs **f**, **x0**, **max.iter**, **stopping.deriv** and **….** The input **f** is the generic function we wish to minimize, **x0** is the starting point in the Newton's Method algorithm, **max.iter** is the maximum number of iterations (defaulted at 200), **stopping.deriv** is the gradient's threshold at which the algorithm terminates (defaulted at 0.001) and **…** allows you to pass additional arguments, based on **f**, into the **Newtons.Method** function. The output of **Newtons.Method** should be a list giving all updates of our minimizer and the number of iterations required to perform the procedure. You are welcome to add additional outputs if you would like. Hint: just copy and paste the **grad.descent** function from class and change the update step.

```
## Solution goes here
library("numDeriv")


Newtons.Method <- function(f, x0, max.iter = 200, stopping.deriv = 0.01, ...)
 {

  n <- length(x0)
  xmat <- matrix(0, nrow = n, ncol = max.iter)
  xmat[,1] <- x0
  for (k in 2:max.iter) {
    # Calculate the gradient
    grad.cur <- grad(f, xmat[ ,k-1], ...)
    # the hessian    use hessian to be the step_length of each step
    hess.cur <- hessian(f, xmat[, k-1], ...)
    # Should we stop?
    if (all(abs(grad.cur) < stopping.deriv)) {
      k <- k-1
      break
    }
    # Move in the opposite direction of the grad    solve():get inverse of ma
trix
    xmat[ ,k] <- xmat[ ,k-1] - solve(hess.cur) %*% grad.cur
  }

  xmat <- xmat[ ,1:k] # Trim
  return(list(x = xmat[,k], xmat = xmat, k = k, minimum=f(xmat[,k],...)))
}
```

Problem 5)

Run the function **Newtons.Method** to minimize the function **logistic.NLL** using initial value **x0=rep(0,8)**, maximum iterations **max.iter=200** and **stopping.deriv=.001**. Display the estimated parameters for $\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6, \beta_7$. How many iterations did the algorithm take to converge?

```
## Solution goes here

gd_newtons <- Newtons.Method(logistic.NLL,x0=rep(0,8),max.iter = 2000, stoppi
ng.deriv = 0.001,data=prostate)

gd_newtons$x

## [1] -10.276609184   0.055227995   0.086868612   0.001614891   0.106632573
## [6]  -0.071827474  -1.154137141   0.123324153

gd_newtons$k

## [1] 7
```

Problem 6)

Use the base **R** function **nlm()** to minimize the negative log-likelihood using initial value **x0=rep(0,8)**. Display the estimated parameters for $\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6, \beta_7$. How many iterations did the algorithm take to converge?

```
## Solution goes here
logistic.nlm <- nlm(logistic.NLL,p=rep(0,8),data=prostate)
logistic.nlm$estimate

## [1] -10.276279061   0.055227384   0.086867770   0.001614479   0.106627782
## [6]  -0.071821442  -1.154123413   0.123325485

logistic.nlm$iterations

## [1] 82
```

Problem 7)

Check that the parameter estimates from the logistic model are reasonably close to the estimates coming from the **glm()** function.

```
gd_newtons$x-logistic.nlm$estimate

## [1] -3.301225e-04  6.103921e-07  8.422251e-07  4.121231e-07  4.791076e-06
## [6] -6.031254e-06 -1.372729e-05 -1.332026e-06
```

The difference of them is nearly 0. It’s obvious that they are close to each other.