Hwk#8

Euchie Jn Pierre

2022-11-22

This homework is to practice finding the maximum likelihood estimates for a logistic regression.

High-dimensional NewtonRaphson (HDNR)function

```
newtonraphson <- function(ftn, x0, tol = 1e-9, max.iter = 100) {
  x <- x0 # x0: the initial value
  fx <- ftn(x)
  iter <- 0
  while ((max(abs(fx[[1]])) > tol) & (iter < max.iter)) {
    x <- x - solve(fx[[2]]) %*% fx[[1]]
    fx <- ftn(x)
    iter <- iter + 1
  }
  if (max(abs(fx[[1]])) > tol) {
    cat('Algorithm failed to converge\n')
    return(NULL)
  } else { # max(abs(fx[[1]])) <= tol
    cat("Algorithm converged\n")
    return(x)
  }
}
```

#Ex20-1 preparing data

```
#Preparing data
resp<- read.csv("Data/resp.csv", header = T)
head(resp)
```

```
##   center id treatment gender age baseline visit outcome
## 1      1  1         P      M  46        0     1       0
## 2      1  1         P      M  46        0     2       0
## 3      1  1         P      M  46        0     3       0
## 4      1  1         P      M  46        0     4       0
## 5      1  2         P      M  28        0     1       0
## 6      1  2         P      M  28        0     2       0
```

#Ex20-1 (Part 1) using Newton-Raphson method to find the MLE of the regression coefficients of the logistic regression

```
#constructing design matrix for X
X <- cbind(rep(1, length(resp$outcome)), ifelse(resp$treatment=='P', 1,0), resp$age, resp$baseline)
head(X)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    1   46    0
## [2,]    1    1   46    0
## [3,]    1    1   46    0
## [4,]    1    1   46    0
## [5,]    1    1   28    0
## [6,]    1    1   28    0
```

```
dim(X)
```

```
## [1] 444   4
```

```
Y <- resp$outcome # preparing column vector for Y

ftn <- function(betacoeff) {
  pi1 <- exp(X%*%betacoeff)/ (1+exp(X%*%betacoeff))
  gradient <- t(X)%*%(Y-pi1)
  hessian <- -t(X)%*%diag(c(pi1*(1-pi1)), length(resp$outcome))%*%X
  return(list(gradient, hessian)) #preparing function for high-dimensionalNR
}

newtonraphson(ftn,c(0,0,0,0)) # running HDNR to find intercept and first 3 regression coeffs
```

```
## Algorithm converged
```

```
##             [,1]
## [1,]  0.43670552
## [2,] -1.23475884
## [3,] -0.01140389
## [4,]  1.98241179
```

```
glm(outcome~treatment+age+baseline, family = binomial, data = resp) #using glm to check answers.
```

```
##
## Call:  glm(formula = outcome ~ treatment + age + baseline, family = binomial,
##     data = resp)
##
## Coefficients:
## (Intercept)   treatmentP          age     baseline
##      0.4367      -1.2348      -0.0114       1.9824
##
## Degrees of Freedom: 443 Total (i.e. Null);  440 Residual
## Null Deviance:       609.4
## Residual Deviance: 495.9     AIC: 503.9
```

#Ex20-1 (Part 2) finding the variance-covariance (VCOV) matrix for the beta coefficients

```
beta<-newtonraphson(ftn,c(0,0,0,0)) #saving the four betacoeffs from the HDNR as an object 'beta'
```

```
## Algorithm converged
```

```
head(beta)
```

```
##             [,1]
## [1,]  0.43670552
## [2,] -1.23475884
## [3,] -0.01140389
## [4,]  1.98241179
```

```
model<-glm(outcome~treatment+age+baseline, family = binomial, data = resp)#saving answer from glm
solve(-ftn(beta)[[2]])# finding variance-covariance (VCOV)matrix for mle.
```

```
##             [,1]          [,2]          [,3]          [,4]
## [1,]  0.09951118 -2.070235e-02 -2.168930e-03 -1.289550e-02
## [2,] -0.02070235  5.133126e-02 -3.292548e-05 -1.351023e-02
## [3,] -0.00216893 -3.292548e-05  6.630252e-05 -2.084783e-05
## [4,] -0.01289550 -1.351023e-02 -2.084783e-05  5.426992e-02
```

```
vcov(model) #to check if our calculation for VCOV above is correct
```

```
##             (Intercept)    treatmentP           age      baseline
## (Intercept)  0.09951118 -2.070235e-02 -2.168930e-03 -1.289551e-02
## treatmentP  -0.02070235  5.133126e-02 -3.292548e-05 -1.351023e-02
## age         -0.00216893 -3.292548e-05  6.630252e-05 -2.084783e-05
## baseline    -0.01289551 -1.351023e-02 -2.084783e-05  5.426992e-02
```

#Ex20-1 (Part 3) finding the log likelihood at the beta coefficients

```
ftn1 <- function(betacoeff) {
  pi1 <- exp(X%*%betacoeff)/ (1+exp(X%*%betacoeff))
  gradient <- t(X)%*%(Y-pi1)
  hessian <- -t(X)%*%diag(c(pi1*(1-pi1)), length(resp$outcome))%*%X
  loglike <- sum(Y*log(pi1/(1-pi1))+log(1-pi1))
  return(list(gradient, hessian, loglike)) #finding loglikelihood of the betacoeffs
}

ftn1(beta) [[3]] # retrieving the 'loglike' from the list.
```

```
## [1] -247.9434
```

```
logLik(model) #using base R function 'loglike' to check answers.
```

```
## 'log Lik.' -247.9434 (df=4)
```