Untitled

Amy Pitts

3/17/2022

Data import

This dataset has 569 rows and 32 columns. The outcome variable of interst is `Diagnosis` which takes on values benign or malignant cases. There are 357 benign cases and 212 malignant cases as seen below in Table 1. One variable is `id` and the rest 30 variables are the mean, sd and largest values of the following criteria.

- radius (mean of distances from center to points on the perimeter)
- texture (standard deviation of gray-scale values)
- perimeter
- area
- smoothness (local variation in radius lengths)
- compactness (perimeter^2 / area - 1.0)
- concavity (severity of concave portions of the contour)
- concave points (number of concave portions of the contour)
- symmetry
- fractal dimension ("coastline approximation" - 1)

Using this dataset models will be compared and compared on their ability to predict cancer diagnosis.

Characteristic	**Benign**, N = 357	**Malignant**, N = 212	**p-value**
radius_mean	12.147 (1.781)	17.463 (3.204)	<0.001
texture_mean	17.915 (3.995)	21.605 (3.779)	<0.001
perimeter_mean	78.075 (11.807)	115.365 (21.855)	<0.001
area_mean	462.790 (134.287)	978.376 (367.938)	<0.001
smoothness_mean	0.092 (0.013)	0.103 (0.013)	<0.001
compactness_mean	0.080 (0.034)	0.145 (0.054)	<0.001
concavity_mean	0.046 (0.043)	0.161 (0.075)	<0.001
concave.points_mean	0.026 (0.016)	0.088 (0.034)	<0.001
symmetry_mean	0.174 (0.025)	0.193 (0.028)	<0.001
fractal_dimension_mean	0.063 (0.007)	0.063 (0.008)	0.5
radius_se	0.284 (0.113)	0.609 (0.345)	<0.001
texture_se	1.220 (0.589)	1.211 (0.483)	0.6
perimeter_se	2.000 (0.771)	4.324 (2.569)	<0.001
area_se	21.135 (8.843)	72.672 (61.355)	<0.001
smoothness_se	0.007 (0.003)	0.007 (0.003)	0.2
compactness_se	0.021 (0.016)	0.032 (0.018)	<0.001
concavity_se	0.026 (0.033)	0.042 (0.022)	<0.001
concave.points_se	0.010 (0.006)	0.015 (0.006)	<0.001
symmetry_se	0.021 (0.007)	0.020 (0.010)	0.028
fractal_dimension_se	0.004 (0.003)	0.004 (0.002)	<0.001
radius_worst	13.380 (1.981)	21.135 (4.284)	<0.001
texture_worst	23.515 (5.494)	29.318 (5.435)	<0.001
perimeter_worst	87.006 (13.527)	141.370 (29.457)	<0.001
area_worst	558.899 (163.601)	1,422.286 (597.968)	<0.001
smoothness_worst	0.125 (0.020)	0.145 (0.022)	<0.001
compactness_worst	0.183 (0.092)	0.375 (0.170)	<0.001
concavity_worst	0.166 (0.140)	0.451 (0.182)	<0.001
concave.points_worst	0.074 (0.036)	0.182 (0.046)	<0.001
symmetry_worst	0.270 (0.042)	0.323 (0.075)	<0.001
fractal_dimension_worst	0.079 (0.014)	0.092 (0.022)	<0.001

Build a logistic model to classify the images into malignant/benign

Before building the model we need to first write down the likelihood function, its gradient and Hessian matrix.

The likelihood function for our data which has a single binary response and 30 numerical explanatory variables is

$$\pi_i = P(Y_i = 1|x_{i,1}, \dots x_{i,30}) = \frac{e^{\beta_0+\beta_1 x_{i,1}+\dots\beta_{30}x_{i,30}}}{1+e^{\beta_0+\beta_1 x_{i,1}+\dots\beta_{30}x_{i,30}}} = \frac{e^{\beta_0+\sum_{j=1}^{30}\beta_i x_{i,j}}}{1+e^{\beta_0+\sum_{j=1}^{30}\beta_i x_{i,j}}}$$

Where $\mathbf{X_i}$ represents the i observation of all 30 of our predictor variables. For the data give we have the likelihood is given by

$$L(\mathbf{X}|\beta) = \prod_{i=1}^{n}\left[\pi_i^{y_i}(1-\pi_i)^{1-y_i}\right]$$

Finding the log-likelihood we have

$$l(\mathbf{X}|\vec{\beta}) = \sum_{i=1}^{n}\left[y_i\left(\beta_0+\sum_{j=1}^{30}\beta_i x_{i,j}\right) - \log\left(1+\exp\left(\beta_0+\sum_{j=1}^{30}\beta_i x_{i,j}\right)\right)\right]$$

The gradient can then can be solved for. Observe

$$\nabla l(\mathbf{X}|\vec{\beta}) = \begin{bmatrix} \sum_{i=1}^{n} y_i - \pi_i & \sum_{i=1}^{n} x_{i,1}(y_i - \pi) & \dots & \sum_{i=1}^{n} x_{i,30}(y_i - \pi) \end{bmatrix}^T_{(1\times 31)}$$

Finally, with the gradient we can derive our hessian. Note that due to the 30 predictor variables the hessian will be a 31 by 31 matrix.

$$\begin{aligned} \nabla^2 l(\mathbf{X}|\vec{\beta}) &= -\sum_{i=1}^{n} \begin{pmatrix} 1 \\ X \end{pmatrix} \begin{pmatrix} 1 & X \end{pmatrix} \pi_i(1-\pi_i) \\ &= -\begin{pmatrix} 1 & X \end{pmatrix} diag(\pi_i(1-\pi_i)) \begin{pmatrix} 1 \\ X \end{pmatrix} \end{aligned}$$

Where $X = (x_{i,1}, \dots, x_{i,30})$. Note that this matrix will always be negative definite at all parameters making the this a well behaved problem.

Develop a Newton-Raphson algorithm to estimate your model

Modifications:

- I include half stepping in the Newton-Raphson method.
- Assent direction

Data Preprocess

```
set.seed(7777)
dat <- bc %>% select(-id) %>%
  rename(y = diagnosis) %>%
  mutate(y = ifelse(y=="B", 0, 1))


#split <- initial_split(dat, prop = 0.8)

training_df <- dat # split %>% training()

#testing_df <- split %>% testing()
```

Functions

```
loglike_func <- function(dat, betavec){
  # setting up an intercept
  dat_temp = dat %>%
    rename(intercept = y) %>%
    mutate(intercept = rep(1, nrow(dat) ))
  dat_x = unname(as.matrix(dat_temp)) # creating the x matrix

  # finding the pi values
  u = dat_x%*%betavec
  pi <- exp(u) / (1+exp(u))

  # loglikelihood
  loglik <- sum(dat$y*u - log(1 + exp(u)))

  #gradient
```

```r
  grad <- t(dat_x)%*%(dat$y - pi)

  # Hessian
  mat_temp = matrix(0, nrow = dim(dat)[1], ncol = dim(dat)[1])
  diag(mat_temp)= pi*(1-pi)
  hess = -t(dat_x)%*% mat_temp %*%(dat_x)

  return(list(loglik = loglik, grad = grad, hess = hess))

}
#loglike_func(dat, betavec = c( rep(0.03, 31)))  # test!


NewtonRaphson <- function(dat,  start, tol = 1e-11, maxiter = 200){
  i = 0
  cur = start
  stuff = loglike_func(dat, cur)
  res <- c(i=0, "loglik" = stuff$loglik,  cur, "step" = 1)
  prevloglik <- -Inf # To make sure it iterates
  while(i < maxiter && abs(stuff$loglik - prevloglik) > tol) {
    step = 1
    i <- i + 1
    prevloglik <- stuff$loglik

    # ascent direct check
    eigen_vals = eigen(stuff$hess)
    if(max(eigen_vals$values) <= 0 ){ # we don't want neg def
      hess = stuff$hess
    } else{ # if it is neg def then need to adjust
      hess = stuff$hess - (max(eigen_vals$values) + 0.1)*diag(nrow(stuff$hess))
    }
    prev  <- cur
    cur   <- prev - rep(step, length(prev))*(solve(stuff$hess) %*% stuff$grad)
    stuff <- loglike_func(dat, cur) # log-lik, gradient, Hessian

    # doing half stepping
    while(stuff$loglik < prevloglik){
      stuff <- loglike_func(dat, prev)
      step  <- step / 2 # this is where half steping happens
      cur   <- prev - rep(step, length(prev))*(solve(stuff$hess) %*% stuff$grad)
      stuff <- loglike_func(dat, cur)
    }
    # Add current values to results matrix
    res <- rbind(res, c(i, stuff$loglik, cur, step))
  }

  colnames(res) <- c("intercept", names(dat[,-1]), "i", "loglik", "step")
  return(res)
}
#Running the algorithm with random starting values
betavec = c(rep(0.01, 31))
ans <- NewtonRaphson(training_df, betavec)
```

Results:

```
##              i         step      loglik
## 1    0.0100000 1.000000e+00    0.010000
## 2   -8.7417254 3.814697e-06   -1.179555
## 3   77.5461663 3.125000e-02  488.328508
## 4  -37.2945192 2.441406e-04 -109.021004
## 5   -0.5525171 3.125000e-02  110.172111
## 6    7.9084537 1.250000e-01  -73.777844
## 7    0.6217574 2.500000e-01   43.286394
## 8    2.8144108 1.000000e+00   49.370660
## 9    6.1210033 1.000000e+00   69.073972
## 10   9.3973114 1.000000e+00  108.752603
## 11  12.7591621 1.000000e+00  165.900414
## 12  18.5983341 1.000000e+00  232.811890
## 13  28.2379507 1.000000e+00  315.879945
## 14  40.9114174 1.000000e+00  410.630445
## 15  61.8581625 1.000000e+00  529.187768
## 16 131.6888689 1.000000e+00  757.177885
## 17 147.4077605 1.250000e-01  807.047155
## 18 149.4996063 1.562500e-02  813.889267
## 19 150.5536412 7.812500e-03  817.351665
## 20 151.0826931 3.906250e-03  819.093312
## 21 151.3477290 1.953125e-03  819.966761
## 22 151.4803745 9.765625e-04  820.404145
## 23 151.5467293 4.882812e-04  820.623002
## 24 151.5633219 1.220703e-04  820.677736
## 25 151.5716188 6.103516e-05  820.705106
## 26 151.5757673 3.051758e-05  820.718792
## 27 151.5758970 9.536743e-07  820.719220
## 28 151.5759132 1.192093e-07  820.719273
## 29 151.5759172 2.980232e-08  820.719286
## 30 151.5759193 1.490116e-08  820.719293
## 31 151.5759203 7.450581e-09  820.719297
## 32 151.5759205 1.862645e-09  820.719297
## 33 151.5759206 4.656613e-10  820.719298
## 34 151.5759206 2.328306e-10  820.719298
## 35 151.5759206 1.164153e-10  820.719298
## 36 151.5759206 5.820766e-11  820.719298
## 37 151.5759206 2.910383e-11  820.719298
## 38 151.5759206 1.455192e-11  820.719298
## 39 151.5759206 7.275958e-12  820.719298
## 40 151.5759206 3.637979e-12  820.719298
## 41 151.5759206 1.818989e-12  820.719298
```

Variables Name	Beta Coef
intercept	40.000
radius_mean	-11.183
texture_mean	-30.139
perimeter_mean	-44.734
area_mean	0.553
smoothness_mean	0.618
compactness_mean	0.378
concavity_mean	607.372
concave.points_mean	-428.700
symmetry_mean	170.327
fractal_dimension_mean	307.008
radius_se	-154.434
texture_se	254.243
perimeter_se	-15.974
area_se	-5.739
smoothness_se	-6.772
compactness_se	1.201
concavity_se	-110.513
concave.points_se	810.326
symmetry_se	-655.396
fractal_dimension_se	3058.293
radius_worst	-716.255
texture_worst	-8714.163
perimeter_worst	16.337
area_worst	1.128
smoothness_worst	0.452
compactness_worst	-0.118
concavity_worst	-206.369
concave.points_worst	-70.968
symmetry_worst	81.319
fractal_dimension_worst	-50.233

Comparing to glm model

Model Implementation

```
glm_fit <- glm(y~., data=training_df, family = "binomial")
result_glm <- summary(glm_fit)

glm_fit %>% broom::tidy() %>%
  select(term, estimate) %>%
  mutate(estimate = round(estimate, 3)) %>%
  knitr::kable()
```

term	estimate
(Intercept)	-2.881303e+06
radius_mean	2.427012e+06
texture_mean	1.957831e+05
perimeter_mean	1.473188e+06
area_mean	-1.301181e+05
smoothness_mean	-1.524522e+08
compactness_mean	-6.428386e+06
concavity_mean	1.041553e+06
concave.points_mean	-1.715686e+07
symmetry_mean	4.048592e+07
fractal_dimension_mean	-4.232918e+07
radius_se	3.328481e+07
texture_se	6.368392e+06
perimeter_se	1.700716e+06
area_se	-6.393464e+05
smoothness_se	7.491721e+08
compactness_se	-1.773076e+08
concavity_se	1.528648e+08
concave.points_se	-1.259854e+09
symmetry_se	2.890109e+08
fractal_dimension_se	1.512103e+09
radius_worst	-6.130231e+06
texture_worst	-5.832457e+05
perimeter_worst	-3.538204e+05
area_worst	8.950446e+04
smoothness_worst	-2.161128e+07
compactness_worst	8.986336e+06
concavity_worst	-3.027927e+07
concave.points_worst	1.431304e+08
symmetry_worst	-2.473590e+07
fractal_dimension_worst	-3.698324e+07