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

$$\nabla l(\mathbf{X}|\vec{\beta}) = \left[\sum_{i=1}^{n} y_i - \pi_i \quad \sum_{i=1}^{n} x_{i,1}(y_i - \pi) \quad \dots \quad \sum_{i=1}^{n} x_{i,30}(y_i - \pi)\right]^T_{(1\times 31)}$$

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()

#training_df <- training_df[c(1:455),]
```

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)))
```

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

  # Hessian
  W = matrix(0, nrow = dim(dat)[1], ncol = dim(dat)[1])
  diag(W)= pi*(1-pi)
  hess = -t(dat_x)%*% W %*% (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-8, maxiter = 200){
  i = 0
  cur = start
  stuff = loglike_func(dat, cur)
  res <- c(i=0, "loglik" = stuff$loglik,  "step" = 1, cur)
  prevloglik <- -Inf # To make sure it iterates
  while(i < maxiter && abs(stuff$loglik - prevloglik) > tol) {
    step = 1
    i <- i + 1
    prevloglik <- stuff$loglik

    # checking negative definite
    eigen_vals = eigen(stuff$hess)
    if(max(eigen_vals$values) <= 0 ){ # check neg def, if not change
      hess = stuff$hess
    } else{ # if it is pos def then need to adjust
      hess = stuff$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$loglik < prevloglik){
      stuff <- loglike_func(dat, prev)
      step  <- step / 2 # this is where half steping happens
      cur   <- prev - step*(solve(stuff$hess) %*% stuff$grad)
      stuff <- loglike_func(dat, cur)
    }
    # Add current values to results matrix
    res <- rbind(res, c(i, stuff$loglik, step, cur))
  }

  colnames(res) <- c("i", "loglik",  "step", "intercept", names(dat[,-1]))
  return(res)
}
#Running the algorithm with random starting values
```

```r
betavec = c(rep(0.01, 31))
ans <- NewtonRaphson(training_df, betavec)
```

Results:

```
##     i         step      loglik
## 1   0 1.000000e+00 -4572.62022
## 2   1 3.814697e-06 -4305.72478
## 3   2 3.125000e-02 -4106.78557
## 4   3 2.441406e-04 -1765.78820
## 5   4 3.125000e-02 -1500.82384
## 6   5 1.250000e-01  -570.56646
## 7   6 2.500000e-01  -186.64985
## 8   7 1.000000e+00   -95.43908
## 9   8 1.000000e+00   -58.73991
## 10  9 1.000000e+00   -39.93848
## 11 10 1.000000e+00   -29.51427
## 12 11 1.000000e+00   -22.63002
## 13 12 1.000000e+00   -18.46245
## 14 13 1.000000e+00   -15.99269
## 15 14 1.000000e+00   -14.12667
## 16 15 1.000000e+00   -11.71059
## 17 16 1.250000e-01   -11.28851
## 18 17 1.562500e-02   -11.23519
## 19 18 7.812500e-03   -11.20859
## 20 19 3.906250e-03   -11.19531
## 21 20 1.953125e-03   -11.18867
## 22 21 9.765625e-04   -11.18536
## 23 22 4.882812e-04   -11.18370
## 24 23 1.220703e-04   -11.18328
## 25 24 6.103516e-05   -11.18308
## 26 25 3.051758e-05   -11.18297
## 27 26 9.536743e-07   -11.18297
## 28 27 1.192093e-07   -11.18297
## 29 28 2.980232e-08   -11.18297
## 30 29 1.490116e-08   -11.18297
## 31 30 7.450581e-09   -11.18297
## 32 31 1.862645e-09   -11.18297
```

term	amy_est
intercept	-30.139
radius_mean	-44.734
texture_mean	0.553
perimeter_mean	0.618
area_mean	0.378
smoothness_mean	607.372
compactness_mean	-428.700
concavity_mean	170.327
concave.points_mean	307.008
symmetry_mean	-154.434
fractal_dimension_mean	254.243
radius_se	-15.974
texture_se	-5.739
perimeter_se	-6.772
area_se	1.201
smoothness_se	-110.513
compactness_se	810.326
concavity_se	-655.396
concave.points_se	3058.293
symmetry_se	-716.255
fractal_dimension_se	-8714.163
radius_worst	16.337
texture_worst	1.128
perimeter_worst	0.452
area_worst	-0.118
smoothness_worst	-206.369
compactness_worst	-70.968
concavity_worst	81.319
concave.points_worst	-50.233
symmetry_worst	151.576
fractal_dimension_worst	820.719

Comparing to glm model

Model Implementation

```
# training_df <- training_df[c(1:455),]
glm_fit <- glm(y~., data=training_df, family = "binomial")
result_glm <- summary(glm_fit)

glm_est <- glm_fit %>% broom::tidy() %>%
  select(term, estimate) %>%
  mutate(glm_est = round(estimate, 3)) %>%
  select(-estimate) %>%
  mutate(term = ifelse(term == "(Intercept)", "intercept", term))
```

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.0-2
```

```
xdat = as.matrix(dat %>% select(-y))
glmnet_fit <- glmnet(x = xdat, y = dat$y, family="binomial", lambda = 0)
glmnet_est <- as.vector(coef(glmnet_fit)) %>% round(3)
```

combining

```
combine_res <- amy_est %>%
  full_join(glm_est) %>%
  mutate(glmnet_est= glmnet_est)
```

```
## Joining, by = "term"
```

```
combine_res %>%
  mutate(
    glm_est = round(glm_est,2)
  ) %>%
  knitr::kable()
```

term	amy_est	glm_est	glmnet_est
intercept	-30.139	-2.881303e+06	-63.001
radius_mean	-44.734	2.427012e+06	-199.921
texture_mean	0.553	1.957831e+05	4.230
perimeter_mean	0.618	1.473188e+06	3.056
area_mean	0.378	-1.301181e+05	1.656
smoothness_mean	607.372	-1.524522e+08	1694.708
compactness_mean	-428.700	-6.428386e+06	-2296.582
concavity_mean	170.327	1.041553e+06	1427.085
concave.points_mean	307.008	-1.715686e+07	1217.363
symmetry_mean	-154.434	4.048592e+07	-821.444
fractal_dimension_mean	254.243	-4.232918e+07	2560.066
radius_se	-15.974	3.328481e+07	214.003
texture_se	-5.739	6.368392e+06	-18.584
perimeter_se	-6.772	1.700716e+06	-45.148
area_se	1.201	-6.393464e+05	3.775
smoothness_se	-110.513	7.491721e+08	-3656.025
compactness_se	810.326	-1.773076e+08	3535.532
concavity_se	-655.396	1.528648e+08	-3157.434
concave.points_se	3058.293	-1.259854e+09	13110.563
symmetry_se	-716.255	2.890109e+08	-4185.736
fractal_dimension_se	-8714.163	1.512103e+09	-30097.121
radius_worst	16.337	-6.130231e+06	46.664
texture_worst	1.128	-5.832457e+05	3.953
perimeter_worst	0.452	-3.538204e+05	2.667
area_worst	-0.118	8.950446e+04	-0.269
smoothness_worst	-206.369	-2.161128e+07	-287.861
compactness_worst	-70.968	8.986336e+06	-268.719
concavity_worst	81.319	-3.027927e+07	190.525
concave.points_worst	-50.233	1.431304e+08	365.168
symmetry_worst	151.576	-2.473590e+07	897.403
fractal_dimension_worst	820.719	-3.698324e+07	2691.665