# Breast Cancer Diagnosis

Hongjie Liu, Xicheng Xie, Jiajun Tao, Shaohan Chen, Yujia Li

2023-02-27

# Outline

# Background

**Breast Cancer Diagnosis**

- The data is the breast cancer medical data retrieved from "breast-cancer.csv", which has 569 rows and 32 columns.
- The first column `ID` labels individual breast tissue images; The second column `Diagnosis` identifies if the image is coming from cancer tissue or benign cases (M = malignant, B = benign). There are 357 benign and 212 malignant cases.
- The other 30 columns correspond to mean, standard deviation and the largest values (points on the tails) of the distributions of the following 10 features computed for the cellnuclei;

# Objectives

- The goal of the exercise is to build a predictive model based on logistic regression to facilitate cancer diagnosis.
- We will move towards the goal with the steps of task 1, 2, 3 and 4.

# Task 1

**Objective**

Build a logistic model to classify the images into malignant/benign, and write down your likelihood function, its gradient and Hessian matrix.

# Task 1 - Build a logistic model

Define the "Diagnosis" variable will be coded as 1 for malignant cases and 0 for benign cases.

Given $n$ i.i.d. observations with $p$ predictors, we consider a logistic regression model

$$P(Y_i = 1 \mid \mathbf{x}_i) = \frac{e^{\mathbf{x}_i^\top \boldsymbol{\beta}}}{1 + e^{\mathbf{x}_i^\top \boldsymbol{\beta}}}, \ i = 1, \ldots, n \tag{1}$$

where $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_p)^\top \in \mathbb{R}^{p+1}$ is the parameter vector, $\mathbf{x}_i = (1, X_{i1}, \ldots, X_{ip})^\top$ is the vector of predictors in the $i$-th observation, and $Y_i \in \{0, 1\}$ is the binary response in the $i$-th observation.

# Task 1 - Build a logistic model

Let $\mathbf{y} = (Y_1, Y_2, \ldots, Y_n)^\top$ denote the response vector, $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)^\top \in \mathbb{R}^{n\times(p+1)}$ denote the design matrix. The observed likelihood of $\{(Y_1, \mathbf{x}_1), (Y_2, \mathbf{x}_2) \ldots, (Y_n, \mathbf{x}_n)\}$ is

$$L(\beta; \mathbf{y}, \mathbf{X}) = \prod_{i=1}^{n} \left[ \left( \frac{e^{\mathbf{x}_i^\top \beta}}{1 + e^{\mathbf{x}_i^\top \beta}} \right)^{Y_i} \left( \frac{1}{1 + e^{\mathbf{x}_i^\top \beta}} \right)^{1-Y_i} \right].$$

# Task 1 - Build a logistic model

Maximizing the likelihood is equivalent to maximizing the log-likelihood function:

$$f(\beta; \mathbf{y}, \mathbf{X}) = \sum_{i=1}^{n} \left[ Y_i \mathbf{x}_i^\top \beta - \log\left(1 + e^{\mathbf{x}_i^\top \beta}\right) \right]. \tag{2}$$

The estimates of model parameters are

$$\widehat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} \; f(\beta; \mathbf{y}, \mathbf{X}),$$

and the optimization problem is

$$\max_{\boldsymbol{\beta}} \; f(\beta; \mathbf{y}, \mathbf{X}). \tag{3}$$

# Task 1 - Build a logistic model

Denote $p_i = P(Y_i = 1 \mid \mathbf{x}_i)$ as given in (1) and $\mathbf{p} = (p_1, p_2, \ldots, p_n)^\top$. The gradient of $f$ is

$$\begin{aligned}
\nabla f(\beta; \mathbf{y}, \mathbf{X}) &= \mathbf{X}^\top(\mathbf{y} - \mathbf{p}) \\
&= \sum_{i=1}^{n} (Y_i - p_i)\mathbf{x}_i \\
&= \begin{pmatrix} \sum_{i=1}^{n} (Y_i - p_i) \\ \sum_{i=1}^{n} (Y_i - p_i) X_{i1} \\ \vdots \\ \sum_{i=1}^{n} (Y_i - p_i) X_{ip} \end{pmatrix}.
\end{aligned}$$

# Task 1 - Build a logistic model

Denote $w_i = p_i(1 - p_i) \in (0, 1)$ and $\mathbf{W} = \mathrm{diag}(w_1, \ldots, w_n)$. The Hessian matrix of $f$ is given by

$$
\begin{aligned}
\nabla^2 f(\beta; \mathbf{y}, \mathbf{X}) &= -\mathbf{X}^\top \mathbf{W} \mathbf{X} \\
&= -\sum_{i=1}^{n} w_i \mathbf{x}_i \mathbf{x}_i^\top \\
&= -\begin{pmatrix}
\sum_{i=1}^n w_i & \sum_{i=1}^n w_i X_{i1} & \cdots & \sum_{i=1}^n w_i X_{i1} \\
\sum_{i=1}^n w_i X_{i1} & \sum_{i=1}^n w_i X_{i1}^2 & \cdots & \sum_{i=1}^n w_i X_{i1} X_{ip} \\
\vdots & \vdots & \ddots & \vdots \\
\sum_{i=1}^n w_i X_{ip} & \sum_{i=1}^n w_i X_{in} X_{i1} & \cdots & \sum_{i=1}^n w_i X_{ip}^2
\end{pmatrix}.
\end{aligned}
$$

# Task 1 - Build a logistic model

Next, we show that the Hessian matrix $\nabla^2 f(\beta; \mathbf{y}, \mathbf{X})$ is a negative-definite matrix if $\mathbf{X}$ has full rank.

***Proof.*** For any $(p+1)$-dimensional nonzero vector $\boldsymbol{\alpha}$, given that $\mathbf{X}$ has full rank, $\mathbf{X}\boldsymbol{\alpha}$ is also a nonzero vector. Since $\mathbf{W}$ is positive-definite, we have

$$\begin{aligned}\boldsymbol{\alpha}^\top \nabla^2 f(\beta; \mathbf{y}, \mathbf{X})\boldsymbol{\alpha} &= \boldsymbol{\alpha}^\top(-\mathbf{X}^\top \mathbf{W}\mathbf{X})\boldsymbol{\alpha} \\ &= -(\mathbf{X}\boldsymbol{\alpha})^\top \mathbf{W}(\mathbf{X}\boldsymbol{\alpha}) \\ &< 0.\end{aligned}$$

Thus, $\nabla^2 f(\beta; \mathbf{y}, \mathbf{X})$ is negative-definite. $\square$

Hence, the optimization problem (3) is a well-defined problem.

# Task 2

**Objective**

Develop a Newton-Raphson algorithm to estimate your model

The target function $f$ given in task 1:

$$f(\beta; \mathbf{y}, \mathbf{X}) = \sum_{i=1}^{n} \left[ Y_i \mathbf{x}_i^\top \beta - \log\left(1 + e^{\mathbf{x}_i^\top \beta}\right) \right]. \tag{4}$$

# Task 2 - Newton-Raphson algorithm

**Algorithm 1** Newton-Raphson algorithm

**Require:** $f(\boldsymbol{\beta})$ - target function as given in (4); $\boldsymbol{\beta}_0$ - starting value

**Ensure:** $\widehat{\boldsymbol{\beta}}$ such that $\widehat{\boldsymbol{\beta}} \approx \arg\max_{\boldsymbol{\beta}} f(\boldsymbol{\beta})$

$i \leftarrow 0$, where $i$ is the current number of iterations

$f(\boldsymbol{\beta}_{-1}) \leftarrow -\infty$

**while** convergence criterion is not met **do**

$i \leftarrow i + 1$

$\mathbf{d}_i \leftarrow -[\nabla^2 f(\boldsymbol{\beta}_{i-1})]^{-1} \nabla f(\boldsymbol{\beta}_{i-1})$, where $\mathbf{d}_i$ is the direction in the $i$-th iteration

$\lambda_i \leftarrow 1$, where $\lambda_i$ is the multiplier in the $i$-th iteration

$\boldsymbol{\beta}_i \leftarrow \boldsymbol{\beta}_{i-1} + \lambda_i \mathbf{d}_i$

**while** $f(\boldsymbol{\beta}_i) \leq f(\boldsymbol{\beta}_{i-1})$ **do**

$\lambda_i \leftarrow \lambda_i / 2$

$\boldsymbol{\beta}_i \leftarrow \boldsymbol{\beta}_{i-1} + \lambda_i \mathbf{d}_i$

**end while**

**end while**

$\widehat{\boldsymbol{\beta}} \leftarrow \boldsymbol{\beta}_i$

# Task 2 - Newton-Raphson algorithm

```
NewtonRaphson <- function(dat, func, start, tol = 1e-8, maxiter = 200) {
  i <- 0
  cur <- start
  stuff <- func(dat, cur)
  res <- c(0, stuff$f, cur)
  prevf <- -Inf
  X <- cbind(rep(1, nrow(dat)), as.matrix(dat[, -1]))
  y <- dat[, 1]
  warned <- 0
  while (abs(stuff$f - prevf) > tol && i < maxiter) {
    i <- i + 1
    prevf <- stuff$f
    prev <- cur
    d <- -solve(stuff$Hess) %*% stuff$grad
    cur <- prev + d
    lambda <- 1
    maxhalv <- 0
    while (func(dat, cur)$f < prevf && maxhalv < 50) {
      maxhalv <- maxhalv + 1
      lambda <- lambda / 2
      cur <- prev + lambda * d
    }
    stuff <- func(dat, cur)
    res <- rbind(res, c(i, stuff$f, cur))
    y_hat <- ifelse(X %*% cur > 0, 1, 0)
    if (warned == 0 && sum(y - y_hat) == 0) {
      warning("Complete separation occurs. Algorithm does not converge.")
      warned <- 1
    }
  }
  colnames(res) <- c("iter", "target_function", "(Intercept)", names(dat)[-1])
  return(res)
}
```

# Task 2 - Newton-Raphson algorithm

## Data preprocessing and data partition.

```
bc_df <- read.csv("breast-cancer.csv")[-c(1, 33)] %>% # remove variable ID and an NA column
  mutate(diagnosis = ifelse(diagnosis == "M", 1, 0)) # code malignant cases as 1
bc_df[, -1] <- scale(bc_df[, -1]) # predictors are standardized for the model in task 3

set.seed(1)
indexTrain <- createDataPartition(y = bc_df$diagnosis, p = 0.8, list = FALSE)
Training <- bc_df[indexTrain, ]
Test <- bc_df[-indexTrain, ]

glm(diagnosis ~ ., family = binomial(link = "logit"), data = Training)

logisticstuff <- function(dat, betavec) {
  dat <- as.matrix(dat)
  n <- nrow(dat)
  p <- ncol(dat) - 1
  X <- cbind(rep(1, n), dat[, -1]) # design matrix
  y <- dat[, 1] # response vector
  u <- X %*% betavec # x_i^T beta, i=1,...,n
  f <- sum(y * u - log1pexp(u)) # function `log1pexp` to compute log(1 + exp(x)))
  p_vec <- sigmoid(u) # function `sigmoid` to compute exp(x)/(1 + exp(x))
  grad <- t(X) %*% (y - p_vec)
  Hess <- -t(X) %*% diag(c(p_vec * (1 - p_vec))) %*% X
  return(list(f = f, grad = grad, Hess = Hess))
}
```

# Task 3

# Task 4

# Discussions

- There is much freedom when designing the simulations.
- In our algorithm, we have 5 parameters. n, p, ratio, c, corr
- More parameters can be adjusted.

# Limitations and Future Work

- Limitation: We reproduced high-dimensional scenarios, but we still don't know the solution.
- Future Work: We may adjust other parameters to investigate further.

# Reference

1. Li Y, Hong HG, Ahmed SE, Li Y. Weak signals in high-dimensional regression: Detection, estimation and prediction. Appl Stochastic Models Bus Ind. 2018;1–16. https://doi.org/10.1002/asmb.2340

# Q&A

- Thanks for listening!
- Any questions?