

# Breast Cancer Diagnosis

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

2023-02-27

# Outline

- ▶ Background
- ▶ Task Introduction
- ▶ Task 1
- ▶ Task 2
- ▶ Task 3
- ▶ Task 4
- ▶ Discussions
- ▶ Limitations and Future Work
- ▶ Reference
- ▶ Q&A

# Background

## **Breast Cancer Diagnosis:**

- ▶ In this project we study the breast cancer diagnosis problem
- ▶ The goal of the exercise is to build a predictive model based on logistic regression to facilitate cancer diagnosis
- ▶ We move towards the goal with the steps of task 1, 2, 3 and 4.

# Background

## Data Source:

- ▶ 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 cell nuclei

# Task Introduction

- ▶ Task 1: Build a logistic model to classify the images into malignant/benign, and write down your likelihood function, its gradient and Hessian matrix.
- ▶ Task 2: Develop a Newton-Raphson algorithm to estimate your model.
- ▶ Task 3: Build a logistic-LASSO model to select features, and implement a path-wise coordinate-wise optimization algorithm to obtain a path of solutions with a sequence of descending  $\lambda$ 's.
- ▶ Task 4: Use 5-fold cross-validation to select the best  $\lambda$ . Compare the prediction performance between the 'optimal' model and 'full' model

# Task 1 - Objective

## **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

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

where  $\boldsymbol{\beta} = (\beta_0, \beta_1, \dots, \beta_p)^\top \in \mathbb{R}^{p+1}$  is the parameter vector,  $\mathbf{x}_i = (1, X_{i1}, \dots, X_{ip})^\top$  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  $\mathbf{y} = (Y_1, Y_2, \dots, Y_n)^\top$  denote the response vector, and  $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)^\top \in \mathbb{R}^{n \times (p+1)}$  denote the design matrix.

The observed likelihood of  $\{(Y_1, \mathbf{x}_1), (Y_2, \mathbf{x}_2), \dots, (Y_n, \mathbf{x}_n)\}$  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:

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

The estimates of model parameters are

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

and the optimization problem is

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

## Task 1 - Build a logistic model

Denote  $p_i = P(Y_i = 1 \mid \mathbf{x}_i)$  as given in (1) and  $\mathbf{p} = (p_1, p_2, \dots, p_n)^\top$ . 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  $\mathbf{W} = \text{diag}(w_1, \dots, w_n)$ . The Hessian matrix of  $f$  is given by

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

## 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  $\alpha$ , given that  $\mathbf{X}$  has full rank,  $\mathbf{X}\alpha$  is also a nonzero vector. Since  $\mathbf{W}$  is positive-definite, we have

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

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

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

# Task 1 - Logistic model R code

```
cancer <- read.csv("breast-cancer.csv") %>%
janitor::clean_names()%>%
select(-1,-33) %>%
mutate(diagnosis = recode(diagnosis, "M" = 1, "B" = 0))
cor()
ggcorrplot(corr, type = "upper", tl.cex = 8)

set.seed(1)
trainRows <- createDataPartition(y = cancer$diagnosis, p = 0.8, list = FALSE)
train <- cancer[trainRows, ]
test <- cancer[-trainRows, ]
glm.fit <- glm(diagnosis ~ .,
               data = train,
               subset = trainRows,
               family = binomial(link = "logit"))
summary(glm.fit)
pred <- predict(glm.fit, newdata = test, type = "response")
y_test <- factor(test$diagnosis)
auc_full <- auc(y_test, pred)
auc_full
```

- ▶ Final AUC of full model reaches 0.9641.
- ▶ And if we remove correlated variables, the AUC is uplifted to 0.9962.

## Task 2 - Objective

### Objective:

Develop a Newton-Raphson algorithm to estimate your model

- Recall The target function  $f$  given in task 1 is:

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

- We develop a modified Newton-Raphson algorithm including a step-halving step to maximize the target function.

## Task 2 - Newton-Raphson algorithm design

---

### Algorithm 1 Newton-Raphson algorithm

---

**Require:**  $f(\beta)$  - target function as given in (2);  $\beta_0$  - starting value

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

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

$f(\beta_{-1}) \leftarrow -\infty$

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

$i \leftarrow i + 1$

$\mathbf{d}_i \leftarrow -[\nabla^2 f(\beta_{i-1})]^{-1} \nabla f(\beta_{i-1})$ , 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

$\beta_i \leftarrow \beta_{i-1} + \lambda_i \mathbf{d}_i$

**while**  $f(\beta_i) \leq f(\beta_{i-1})$  **do**

$\lambda_i \leftarrow \lambda_i / 2$

$\beta_i \leftarrow \beta_{i-1} + \lambda_i \mathbf{d}_i$

**end while**

**end while**

$\hat{\beta} \leftarrow \beta_i$

---

## Task 2 - Newton-Raphson algorithm R code

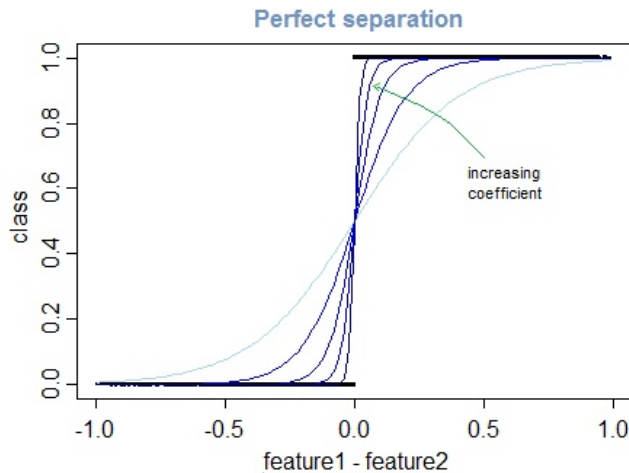
```
NewtonRaphson <- function(dat, func, start, tol = 1e-8, maxiter = 200) {  
  i <- 0  
  cur <- start  
  stuff <- func(dat, cur)  
  res <- c(0, stuff$f, cur)  
  prevf <- -Inf  
  X <- cbind(rep(1, nrow(dat)), as.matrix(dat[, -1]))  
  y <- dat[, 1]  
  warned <- 0  
  while (abs(stuff$f - prevf) > 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 - Complete separation

- ▶ Sometimes our algorithm does not converge because of the complete separation.
- ▶ A complete separation in a logistic regression, sometimes also referred as perfect prediction, occurs whenever there exists some vector of coefficients  $\beta$  such that  $Y_i = 1$  whenever  $x_i^\top \beta > 0$  and  $Y_i = 0$  whenever  $x_i^\top \beta \leq 0$ .
- ▶ Complete separation occur when a linear function of predictors can perfectly classify the response.

## Task 2 - Complete separation



## Task 2 - Complete separation

- ▶ We have proved that: when there exists a vector of coefficients  $\hat{\beta}$  such that  $Y_i = 1$  whenever  $x_i^\top \hat{\beta} > 0$  and  $Y_i = 0$  whenever  $x_i^\top \hat{\beta} \leq 0$ , there does not exist  $\beta^* \in \mathbb{R}^{(p+1)}$  such that  $\beta^* = \arg \max_{\beta} f(\beta)$ , where  $f$  is given in (2). Thus our algorithm does not converge. (proof is attached in report appendix)
- ▶ If there is no complete separation, the parameters output by `glm` function and our algorithm are demonstrated to be the same
- ▶ In practice, if complete separation occurs, we randomly pick the parameters which satisfy complete separation as our full model

## Task 2 - Comparison

Comparison of using glm function and our algorithm (part of)

predictor	ours	glm
(Intercept)	111.7230206	90.9690365
radius_mean	-3646.8235387	-2560.3938902
texture_mean	-2.8949199	0.8812037
perimeter_mean	1257.9481173	789.2724398
area_mean	2091.5001210	1539.8345650
smoothness_mean	180.4591375	128.8762247
compactness_mean	-471.3749334	-346.6691873
concavity_mean	-0.3063034	9.0810082
concave.points_mean	287.3555092	215.4808031
symmetry_mean	10.8459784	10.4772590
fractal_dimension_mean	-40.6413497	-28.6916549
radius_se	854.4290933	617.5687358
texture_se	-105.8920782	-79.9788638
perimeter_se	-1231.6514585	-917.3916527
area_se	789.0550146	628.6222313
smoothness_se	-83.3449730	-63.7660367
compactness_se	473.1970464	344.3180183
concavity_se	-440.6629325	-323.8533730
concave.points_se	503.3728692	374.7610875
symmetry_se	-144.4580558	-108.6211792

## 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?