P8160 - Bayesian Modeling of Hurricane Trajectories

Hongjie Liu, Xicheng Xie, Jiajun Tao, Zijian Xu, Shaohan Chen

5/1/2023

Contents

1 Introduction

1.1 Background

A hurricane is a powerful tropical storm characterized by high winds, heavy rain, storm surges, and flooding. Hurricanes are also known as cyclones or typhoons, depending on the region where they occur.

Hurricanes typically form over warm ocean waters and can travel for thousands of miles, causing widespread destruction and disruption to communities in their path. They are categorized on a scale of 1 to 5 based on their wind speed and potential for damage, with Category 5 being the most severe.

Hurricanes can cause significant damage to infrastructure, homes, and businesses, and can also result in loss of life. As such, it is essential to take precautions and follow instructions from emergency management officials in the event of a hurricane.

1.2 Motivation

Climate researcher are interested in modeling hurricane trajectories for early warning and preparedness, resource allocation, planning and response, and scientific research. Overall, accurate modeling of hurricane trajectories is essential for mitigating the impact of hurricanes on communities, infrastructure, and the environment, as well as for advancing our scientific understanding of these powerful storms.

In this project, we are particularly interested in forecasting the wind speed of hurricanes.

1.3 Dataset

The dataset, `hurrican703.csv`, collected the track data of 702 hurricanes in the North Atlantic area from 1950 to 2013. For all the storms, their location (longitude & latitude) and maximum wind speed were recorded every 6 hours. The data includes the following variables:

- ID: ID of the hurricanes
- Season: In which year the hurricane occurred
- Month: In which month the hurricane occurred
- Nature: Nature of the hurricane ET: Extra Tropical
  DS: Disturbance
  NR: Not Rated
  SS: Sub Tropical
  TS: Tropical Storm
- Time: dates and time of the record
- Latitude and Longitude: The location of a hurricane check point
- Wind.kt Maximum wind speed (in Knot) at each check point

2 Methods

2.1 Data pre-processing

First we need to pre-process the data. We only kept observations that occurred on 6 consecutive hour intervals. Through this step, we found that some hurricanes had the same ID but were actually different ones (eg. ALICE). Hurricanes that had fewer than 3 observations were excluded. For the purpose of seasonal comparison, we defined August, September, and October as hurricane-active season, and the rest as hurricane-inactive season. After data cleaning, there are 21691 observations across 704 unique hurricanes.

2.2 Logistic Model

Logistic model measures the probability of an event taking place by having the log-odds for the event be a linear combination of one or more independent variables, and is commonly used in classifying binary response variables.

Hereby, the variable "Diagnosis" is a binary response variable indicating if the image is coming from cancer tissue or benign cases (M = malignant, B = benign). In the following logistic regression model, 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. 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

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

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

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

The estimates of model parameters are

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

and the optimization problem is

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

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

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

Denote $w_i = p_i(1-p_i) \in (0, \frac{1}{4}]$ and $\mathbf{W} = \text{diag}(w_1, \ldots, w_n)$. The Hessian matrix of f is given by

$$\begin{aligned}\nabla^2 f(\boldsymbol{\beta}; \mathbf{y}, \mathbf{X}) &= -\mathbf{X}^\top \mathbf{W} \mathbf{X} \\ &= -\sum_{i=1}^{n} w_i \mathbf{x}_i \mathbf{x}_i^\top \\ &= -\begin{pmatrix} \sum_{i=1}^n w_i & \sum_{i=1}^n w_i X_{i1} & \cdots & \sum_{i=1}^n w_i X_{i1} \\ \sum_{i=1}^n w_i X_{i1} & \sum_{i=1}^n w_i X_{i1}^2 & \cdots & \sum_{i=1}^n w_i X_{i1} X_{ip} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^n w_i X_{ip} & \sum_{i=1}^n w_i X_{ip} X_{i1} & \cdots & \sum_{i=1}^n w_i X_{ip}^2 \end{pmatrix}.\end{aligned} \tag{4}$$

It can be shown that the Hessian matrix $\nabla^2 f(\boldsymbol{\beta}; \mathbf{y}, \mathbf{X})$ is a negative-definite matrix if $\mathbf{X}$ has full rank (Theorem). Hence, the optimization problem () is a well-defined problem.

2.3 Newton-Raphson Algorithm

2.3.1 Algorithm Design

Given that the derivative of the log-likelihood function with respect to each parameter is nonlinear and difficult to solve analytically for maximum likelihood, we use the Newton-Raphson algorithm to solve the optimization problem () numerically.

We develop a modified Newton-Raphson algorithm including a step-halving step. Given that the Hessian matrix is negative-definite in this case, we don't need to ensure that the direction of the step is an ascent direction.

Algorithm 1 Newton-Raphson algorithm including a step-halving step

Require: $f(\boldsymbol{\beta})$ - target function as given in (); $\boldsymbol{\beta}_0$ - 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 λ_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
  $\widehat{\boldsymbol{\beta}} \leftarrow \boldsymbol{\beta}_i$

We implement the algorithm in R (see Code 1). The starting values of $\boldsymbol{\beta}$ is set to all 0's. Table 1 displays a comparison between the outcomes obtained from the application of the Newton-Raphson method and the `glm` function.

2.3.2 Complete Separation Problem

However, the algorithm does not converge when a complete separation occurs. A complete separation in a logistic regression, also referred to as perfect prediction, occurs whenever there exists some vector of

coefficients $\hat{\boldsymbol{\beta}}$ such that $Y_i = 1$ whenever $\mathbf{x}_i^\top \hat{\boldsymbol{\beta}} > 0$ and $Y_i = 0$ whenever $\mathbf{x}_i^\top \hat{\boldsymbol{\beta}} \leq 0$. In other words, complete separation occurs whenever a linear function of predictors can generate perfect predictions of response.

To further explain this problem, we prove that: if there exists a vector of coefficients $\hat{\boldsymbol{\beta}}$ that can generate perfect predictions, there does not exist $\boldsymbol{\beta}^* \in \mathbb{R}^{p+1}$ such that $\boldsymbol{\beta}^* = \arg\max_{\boldsymbol{\beta}} f(\boldsymbol{\beta})$, where f is given in () (see Theorem). Thus our Newton-Raphson algorithm does not converge.

2.4 Logistic-LASSO Model

Regularization is the common approach for variable selection, in which LASSO is to add L-1 penalty to the objective function. In the context of logistic regression, LASSO estimates the model parameters $\boldsymbol{\beta}$ by optimizing a penalized loss function:

$$\min_{\boldsymbol{\beta}} \; -\frac{1}{n} f(\boldsymbol{\beta}) + \lambda \sum_{k=1}^{p} |\beta_k|. \tag{5}$$

where $\lambda \geq 0$ is the tuning parameter and f is the log-likelihood function given in (). Note that the intercept is not penalized and all predictors are standardized.

Then we could develop a path-wise coordinate descent algorithm to solve the optimization problem () with a sequence of nested loops:

Outer Loop. In the outer loop, we compute the solutions of the optimization problem () for a decreasing sequence of values for λ: $\{\lambda_1, \ldots, \lambda_m\}$, starting at the smallest value $\lambda_1 = \lambda_{max}$ for which the estimates of all coefficients $\hat{\beta}_j = 0,\ j = 1, 2, \ldots, p$, which is

$$\lambda_{max} = \max_{j \in \{1,\ldots,p\}} \left| \frac{1}{n} \sum_{i=1}^{n} X_{ij}(Y_i - \bar{Y}) \right|, \tag{6}$$

where $\bar{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i$. For tuning parameter value λ_{k+1}, we initialize coordinate descent algorithm at the computed solution for λ_k (warm start). Apart from giving us a path of solutions, this scheme exploits warm starts, and leads to a more stable algorithm.

Middle Loop. In the middle loop, we find the estimates of $\boldsymbol{\beta}$ by solving the optimization problem () for a fixed λ. For each iteration of the middle loop, based on the current parameter estimates $\tilde{\boldsymbol{\beta}}$, we form a quadratic approximation to the log-likelihood f using a Taylor expansion:

$$\begin{aligned}
f(\boldsymbol{\beta}) \approx \ell(\boldsymbol{\beta}) &= f(\tilde{\boldsymbol{\beta}}) + (\boldsymbol{\beta} - \tilde{\boldsymbol{\beta}})^\top \nabla f(\tilde{\boldsymbol{\beta}}) + \frac{1}{2}(\boldsymbol{\beta} - \tilde{\boldsymbol{\beta}})^\top \nabla^2 f(\tilde{\boldsymbol{\beta}})(\boldsymbol{\beta} - \tilde{\boldsymbol{\beta}}) \\
&= f(\tilde{\boldsymbol{\beta}}) + [\mathbf{X}(\boldsymbol{\beta} - \tilde{\boldsymbol{\beta}})]^\top (\mathbf{y} - \tilde{\mathbf{p}}) - \frac{1}{2}[\mathbf{X}(\boldsymbol{\beta} - \tilde{\boldsymbol{\beta}})]^\top \tilde{\mathbf{W}} \mathbf{X}(\boldsymbol{\beta} - \tilde{\boldsymbol{\beta}}) \\
&= f(\tilde{\boldsymbol{\beta}}) + \sum_{i=1}^{n} (Y_i - \tilde{p}_i)\mathbf{x}_i^\top (\boldsymbol{\beta} - \tilde{\boldsymbol{\beta}}) - \frac{1}{2}\sum_{i=1}^{n} \tilde{w}_i \left[\mathbf{x}_i^\top (\boldsymbol{\beta} - \tilde{\boldsymbol{\beta}})\right]^2 \\
&= -\frac{1}{2}\sum_{i=1}^{n} \tilde{w}_i \left\{ \left[\mathbf{x}_i^\top (\tilde{\boldsymbol{\beta}} - \boldsymbol{\beta})\right]^2 + 2\frac{Y_i - \tilde{p}_i}{\tilde{w}_i} \left[\mathbf{x}_i^\top (\tilde{\boldsymbol{\beta}} - \boldsymbol{\beta})\right] \right\} + f(\tilde{\boldsymbol{\beta}}) \\
&= -\frac{1}{2}\sum_{i=1}^{n} \tilde{w}_i \left[\mathbf{x}_i^\top (\tilde{\boldsymbol{\beta}} - \boldsymbol{\beta}) + \frac{Y_i - \tilde{p}_i}{\tilde{w}_i} \right] + \frac{1}{2}\sum_{i=1}^{n} \tilde{w}_i \left(\frac{Y_i - \tilde{p}_i}{\tilde{w}_i} \right)^2 + f(\tilde{\boldsymbol{\beta}}),
\end{aligned}$$

where $\tilde{\mathbf{p}} = (\tilde{p}_1, \ldots, \tilde{p}_n)^\top$ and $\tilde{\mathbf{W}} = \mathrm{diag}(\tilde{w}_1, \ldots, \tilde{w}_n)$ are the estimates of $\mathbf{p}$ and $\mathbf{W}$ based on $\tilde{\boldsymbol{\beta}}$.
We rewrite the function $\ell(\boldsymbol{\beta})$ as follows:

$$\ell(\boldsymbol{\beta}) = -\frac{1}{2}\sum_{i=1}^{n} \tilde{w}_i (\tilde{z}_i - \mathbf{x}_i^\top \boldsymbol{\beta})^2 + C(\tilde{\boldsymbol{\beta}}), \tag{7}$$

Algorithm 2 Path-wise coordinate-wise optimization algorithm

Require: $g(\boldsymbol{\beta}, \lambda) = -\frac{1}{n} f(\boldsymbol{\beta}) + \lambda \sum_{k=1}^{p} |\beta_k|$ - target function, where $f(\boldsymbol{\beta})$ is given in (); $\boldsymbol{\beta}_0$ - starting value; $\{\lambda_1, \ldots, \lambda_m\}$ - a sequence of descending λ's, where $\lambda_1 = \lambda_{max}$ is given in (); ϵ - tolerance; N_s, N_t - maximum number of iterations of the middle and inner loops

Ensure: $\widehat{\boldsymbol{\beta}}(\lambda_r)$ such that $\widehat{\boldsymbol{\beta}}(\lambda_r) \approx \arg\min_{\boldsymbol{\beta}} \; g(\boldsymbol{\beta}, \lambda_r),\ r = 1, \ldots, m$

$\tilde{\boldsymbol{\beta}}_0(\lambda_1) \leftarrow \boldsymbol{\beta}_0$

OUTER LOOP

for $r \in \{1, \ldots, m\}$, where r is the current number of iterations of the outer loop, **do**

$s \leftarrow 0$, where s is the current number of iterations of the middle loop

$g(\tilde{\boldsymbol{\beta}}_{-1}(\lambda_r), \lambda_r) \leftarrow \infty$

MIDDLE LOOP

while $t \geq 2$ and $s < N_s$ **do**

$s \leftarrow s + 1$

Update $\tilde{w}_i^{(s)}$, $\tilde{z}_i^{(s)}$ $(i = 1, \ldots, n)$, and thus $\ell_s(\boldsymbol{\beta})$ as given in () based on $\tilde{\boldsymbol{\beta}}_{s-1}(\lambda_r)$

$t \leftarrow 0$, where t is the current number of iterations of the inner loop

$\tilde{\boldsymbol{\beta}}_s^{(0)}(\lambda_r) \leftarrow \tilde{\boldsymbol{\beta}}_{s-1}(\lambda_r)$

$h_s(\tilde{\boldsymbol{\beta}}_s^{(-1)}(\lambda_r), \lambda_r) \leftarrow \infty$, where $h_s(\boldsymbol{\beta}, \lambda) = -\frac{1}{n}\ell_s(\boldsymbol{\beta}) + \lambda \sum_{k=1}^{p} |\beta_k|$

INNER LOOP

while $\left| h_s(\tilde{\boldsymbol{\beta}}_s^{(t)}(\lambda_r), \lambda_r) - h_s(\tilde{\boldsymbol{\beta}}_s^{(t-1)}(\lambda_r), \lambda_r) \right| > \epsilon$ and $t < N_t$ **do**

$t \leftarrow t + 1$

$\tilde{\beta}_0^{(t)}(\lambda_r) \leftarrow \sum_{i=1}^{n} \tilde{w}_i^{(s)} \left(\tilde{z}_i^{(s)} - \sum_{k=1}^{p} X_{ik} \tilde{\beta}_k^{(t-1)}(\lambda_r) \right) \Big/ \sum_{i=1}^{n} \tilde{w}_i^{(s)}$

for $j \in \{1, \ldots, p\}$ **do**

$\tilde{\beta}_j^{(t)}(\lambda_r) \leftarrow S\left(\frac{1}{n} \sum_{i=1}^{n} \tilde{w}_i^{(s)} X_{ij} \left(\tilde{z}_i^{(s)} - \sum_{k<j} X_{ik} \tilde{\beta}_k^{(t)}(\lambda_r) - \sum_{k>j} X_{ik} \tilde{\beta}_k^{(t-1)}(\lambda_r) \right), \lambda_r \right) \Big/ \frac{1}{n} \sum_{i=1}^{n} \tilde{w}_i^{(s)} X_{ij}^2$

end for

end while

$\tilde{\boldsymbol{\beta}}_s(\lambda_r) \leftarrow \tilde{\boldsymbol{\beta}}_s^{(t)}(\lambda_r)$

end while

$\widehat{\boldsymbol{\beta}}(\lambda_r) \leftarrow \tilde{\boldsymbol{\beta}}_s(\lambda_r)$

$\tilde{\boldsymbol{\beta}}_0(\lambda_{r+1}) \leftarrow \widehat{\boldsymbol{\beta}}(\lambda_r)$

end for

where

$$\tilde{z}_i = \mathbf{x}_i^\top \tilde{\boldsymbol{\beta}} + \frac{Y_i - \tilde{p}_i}{\tilde{w}_i}$$

is the working response, $\tilde{w}_i$ is the working weight, and C is a function that does not depend on $\boldsymbol{\beta}$.

Inner Loop. In the inner loop, with fixed $\tilde{w}_i$'s, $\tilde{z}_i$'s, and thus a fixed form of ℓ based on the estimates of $\boldsymbol{\beta}$ in the previous iteration of the middle loop, we update the estimates of $\boldsymbol{\beta}$ by solving a modified optimization problem of ():

$$\min_{\boldsymbol{\beta}} \ -\frac{1}{n}\ell(\boldsymbol{\beta}) + \lambda \sum_{k=1}^{p} |\beta_k|,$$

which is equivalent to the following penalized weighted least-squares problem

$$\min_{\boldsymbol{\beta}} \ \frac{1}{2n} \sum_{i=1}^{n} \tilde{w}_i (\tilde{z}_i - \mathbf{x}_i^\top \boldsymbol{\beta})^2 + \lambda \sum_{k=1}^{p} |\beta_k|. \tag{8}$$

We use coordinate descent to update the estimates of $\boldsymbol{\beta}$. For each iteration of the inner loop, suppose we have the current estimates $\tilde{\beta}_k$ for $k \neq j$ and we wish to partially optimize with respect to β_j:

$$\min_{\beta_j} \ \frac{1}{2n} \sum_{i=1}^{n} \tilde{w}_i \left(\tilde{z}_i - X_{ij}\beta_j - \sum_{k \neq j} X_{ik}\tilde{\beta}_k \right)^2 + \lambda |\beta_j| + \lambda \sum_{k \neq j} |\tilde{\beta}_k|.$$

Thus the coordinate descent has updates

$$\tilde{\beta}_0 \leftarrow \frac{\sum_{i=1}^{n} \tilde{w}_i (\tilde{z}_i - \sum_{k=1}^{p} X_{ik}\tilde{\beta}_k)}{\sum_{i=1}^{n} \tilde{w}_i},$$

$$\tilde{\beta}_j \leftarrow \frac{S\left(\frac{1}{n} \sum_{i=1}^{n} \tilde{w}_i X_{ij} (\tilde{z}_i - \sum_{k \neq j} X_{ik}\tilde{\beta}_k), \lambda \right)}{\frac{1}{n} \sum_{i=1}^{n} \tilde{w}_i X_{ij}^2}, \ j = 1, \ldots, p$$

where $S(z, \gamma)$ is the soft-thresholding operator with value

$$S(z, \gamma) = \text{sign}(z)(|z| - \gamma)_+ = \begin{cases} z - \gamma, & \text{if } z > 0 \text{ and } \gamma < |z| \\ z + \gamma, & \text{if } z < 0 \text{ and } \gamma < |z| \\ 0, & \text{if } \gamma \geq |z| \end{cases}$$

We can then update estimates of β_j's repeatedly for $j = 0, 1, 2, ..., p, 0, 1, 2, ...$ until convergence.

We implement the algorithm in R (see Code 2). Here are some important details regarding the implementation of the algorithm:

- In the outer loop, the starting values of $\boldsymbol{\beta}$ is set to all 0's. Note that the value of λ_{max} is different from the form given in Friedman et al. (2010), which is $\lambda_{max} = \max_j |\langle \mathbf{x}_{\cdot j}, \mathbf{y} \rangle|$, where $\mathbf{x}_{\cdot j}$ is the j-th column of the design matrix $\mathbf{X}$, for $j = 1, \ldots, p$. This is because we also take the update of $\tilde{\beta}_0$ into consideration in the inner loop. (see Theorem in the appendices) In practice, we add a very small value (e.g., 10^{-10}) to λ_{max} to avoid computational errors that may cause one slope coefficient estimate not equal to zero.
- In the middle loop, care is taken to avoid coefficients diverging in order to achieve fitted probabilities of 0 or 1. When $\tilde{p}_i$ is within $\delta = 10^{-5}$ of 1, we set it to 1, and set the corresponding weight $\tilde{w}_i$ to δ. 0 is treated similarly.
- The stopping criteria of the middle loop is either the number of iterations of the inner loop is exactly 1 or it reaches a maximum number of iterations 100. The stopping criteria of the inner loop is either the difference of the penalized loss function given in () between the two iterations is less than the tolerance $\epsilon = 10^{-10}$ or it reaches a maximum number of iterations 1000.

2.5 Five-fold Cross-validation for LASSO

Since the Logistic-Lasso model depends on penalty term for variable selection, we further develop 5-fold cross-validation to select the tuning parameter λ to obtain the optimized result.

For the initial λ range, we use a sequence of descending λ's, starting with the largest λ_1, which is the maximum value of λ_{max} in each training set as given in (). We set $\lambda_{min} = \lambda_1/e^6$, and create 30 λ candidates on a log scale. The coefficients of predictors would shrink as λ gets larger, as shown in Figure.

To select the optimal tuning parameter λ, the original data is randomly shuffled and split into five equally sized groups. One of the groups is used as the validation set, while the remaining groups are used as the training set. The path-wise coordinate-wise optimization algorithm is then applied to the training set, and AUC scores are calculated for each λ using the validation set. This process is repeated until each of the five groups has been used as the validation set, and the mean AUC for each λ is computed.

We select the best λ in two ways. One way is that we wrote a function to do the 5-fold cross-validation (see Code 3) and the other way is to use the `cv.glmnet` function from the popular package `caret`.

Here we take the λ with the greatest mean AUC as the best λ. Further, we look at the predictors that the best λ chooses and take them to re-fit the logistic regression model on the training data which is the 'optimal' model. We also fit a logsitical regression model on the training data using our Newton-Raphson algorithm to get the 'full' model. After that, we compare the two models' prediction performance on the test data in specificity, sensitivity, and AUC.

3 Results

3.1 Posterior Summaries and 95% Credible Intervals of γ

We select the best λ using our own function and `cv.glmnet` as shown in Figure . Both our cross-validation function and `cv.glmnet` function reach the greatest mean AUC when $\lambda = 0.0099322$ as shown in Figure , thus this is the best λ we choose. As for the predictors, when $\lambda = \lambda_{best}$, both methods select ten same predictors, which are `texture_mean`, `concave.points_mean`, `radius_se`, `fractal_dimension_se`, `radius_worst`, `texture_worst`, `smoothness_worst`, `concavity_worst`, `concave.points_worst`, and `symmetry_worst`, as shown in Figure.

3.2 Model Comparison

The ten selected predictors are used to re-fit the logistic regression model as the 'optimal' model, which is then compared with the 'full' model generated by the Newton-Raphson algorithm. The result turns out that the two models differ in prediction performance. The 'optimal' model outperformed the 'full' model with higher specificity, sensitivity, larger AUC, and fewer predictors as shown in Figure and Figure .

4 Discussion

The main conclusion is that the logistic Lasso model performed better than the full logistic model in this case. Recall that our goal is to build a predictive model to classify each patient's cancer diagnosis accurately according to the information extracted from the images. The specificity and sensitivity should be as high as possible.

For future work, now that we have selected the ten predictors, we want to know why do these ten matter. What's the interpretation of this model? Do these predictors have any clinical significance? These questions may need to be answered by a cancer expert.

Group Contributions

References

Freer, Timothy W., and Michael J. Ulissey. "Screening mammography with computer-aided detection: prospective study of 12,860 patients in a community breast center." Radiology 220.3 (2001): 781-786.

Friedman J, Hastie T, Tibshirani R. Regularization Paths for Generalized Linear Models via Coordinate Descent. J Stat Softw. 2010;33(1):1-22. PMID: 20808728; PMCID: PMC2929880.

Appendices

Figures and Tables