P8160 - Breast Cancer Data: To lasso or to not lasso

Group 1

Amy Pitts, Hun Lee, Jimmy Kelliher,
Tucker Morgan, Waveley Qiu

2022-04-01

Abstract

we will write an abstract eventually.

1. Introduction

1.1. Overview

1.2. Objectives

The major objective of this project is to used the dataset provided to fit two different models both with the goes of modeling patient diagnosis. The first model will be a full Newton-Raphson algorithm (full model). The second will be a logistic-LASSO model (optimal model) to select features in our dataset. This will be implemented using path-wise coordinate-wise optimization algorithm to and 5-fold cross validation obtain the optimal solution. The algorithms for both these methods will be discussed in our methods section and corresponding code will be found in the appendix.

2. Methods

2.1. Data Cleaning and Exploratory Analysis

The data set of interest contains 569 rows and 33 columns all related to breast tissue images. Each entry of the table represent an individual patient. Of primary interest is the column containing information about patient diagnosis of cancer taking on values either malignant or benign. One column contains information about patient ID which will be removed from our dataset. The other 30 columns correspond to numerical information about the breast imaging. The variables containing information such as radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, and fractal dimension. These variable are going to be used to help model the diagnosis.

In a quick exploration of the data it has been found that there is a high correlation between many of the variables. This is seen in **Figure 1**. The heat map correlation plot there are a lot of varaible with dark blue coloring representing a high correlation. High correlations can cause major issues with the methods we are going to use so understanding why there is high correlation in our data is important.

To explore the correlation more **Figure 2** shows the highest 25 correlation in our data. We see that the highest correlation is between radius mean and perimeter mean with a correlation values of 0.998. In the graph there are 21 combinations of variables that achieve a correlation over 0.90. This is cause for major concern. We can break these 21 pairings into equivalance classes for further inspection. The first grouping that are all mutually correlated are {`area_mean`, `area_worst`, `perimeter_mean`, `perimeter_worst`, `radius_mean`, `radius_worst`}. This grouping represents the 15 correlation pairs in **Figure 2**. Mathematically, if we consider the equivalence classes of variables that are highly correlated, these six variables would belong to the same equivalence class. To identify the best proxy for this grouping we look at the highest mean correlation which turns out to be `radius_worst`. The next grouping of correlated variables is {`radius_sd`, `perimeter_se`, `area_se`}. The best representative will be `radius_se`. Next we can group {`concavity_mean`,`concavity_worst`, `concave.point_worst`, `concave.point_mean`} together, and we find that the best variable proxy is `concave.point_worst`. Finally, {`texture_mean`, `texture_worst`} is our last grouping with `texture_mean` being the variable saved. Thus from all the grouping and saving only the best proxy we will be removing 10 variables leaving 20 variables in our dataset.

All the variables used can be seen in Table 1. In Table 1 we see that there are 357 benign (B) cases and 212 malignant (M) cases. This dataset does not contain any missing values. To implement both the full and optimal model the data set will be split into train and test sets using an 80-20 split.

Figure 1: Correlation Heat Plot of all Covariates

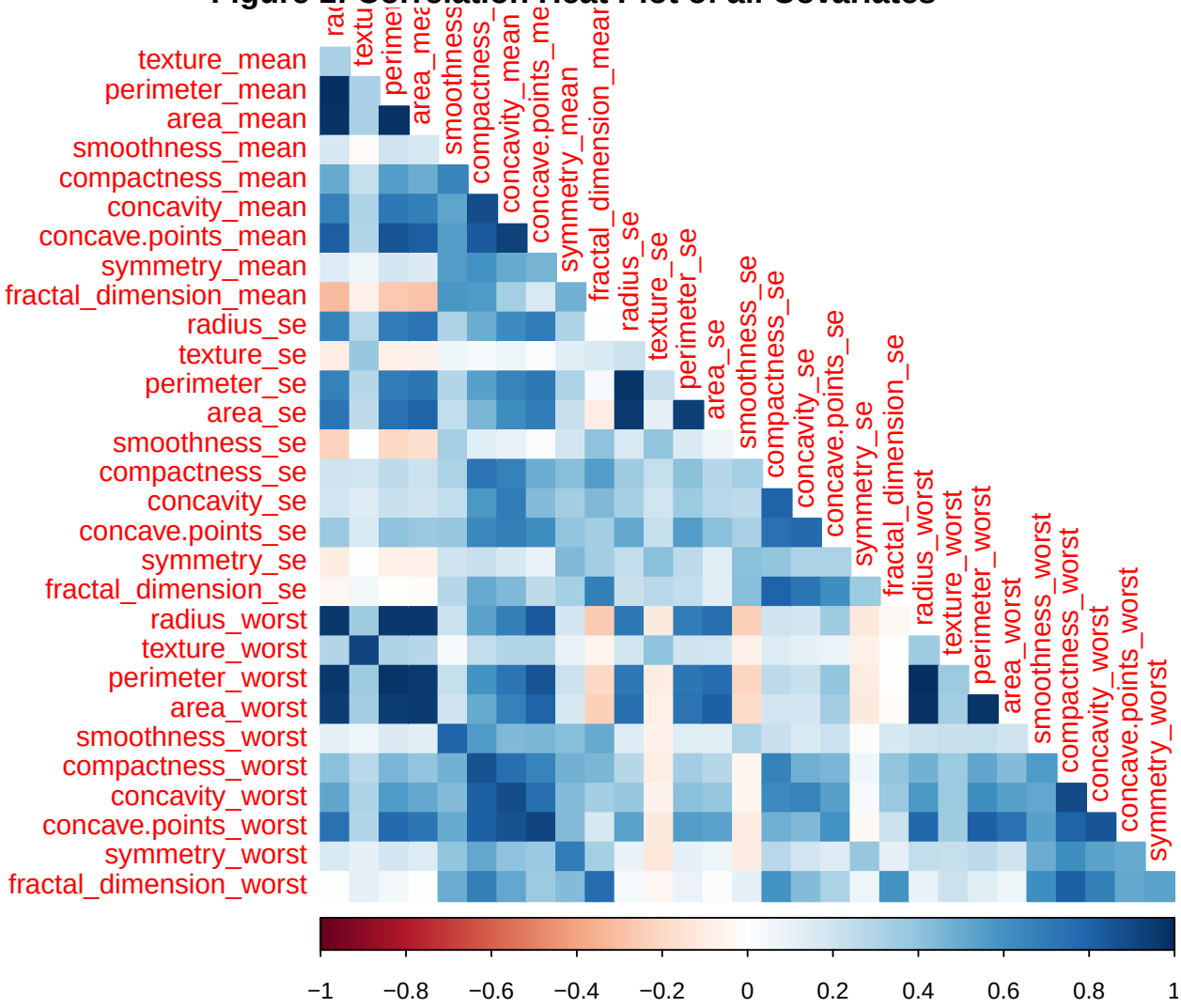

Figure 2: Ranked Cross-Correlations

25 most relevant

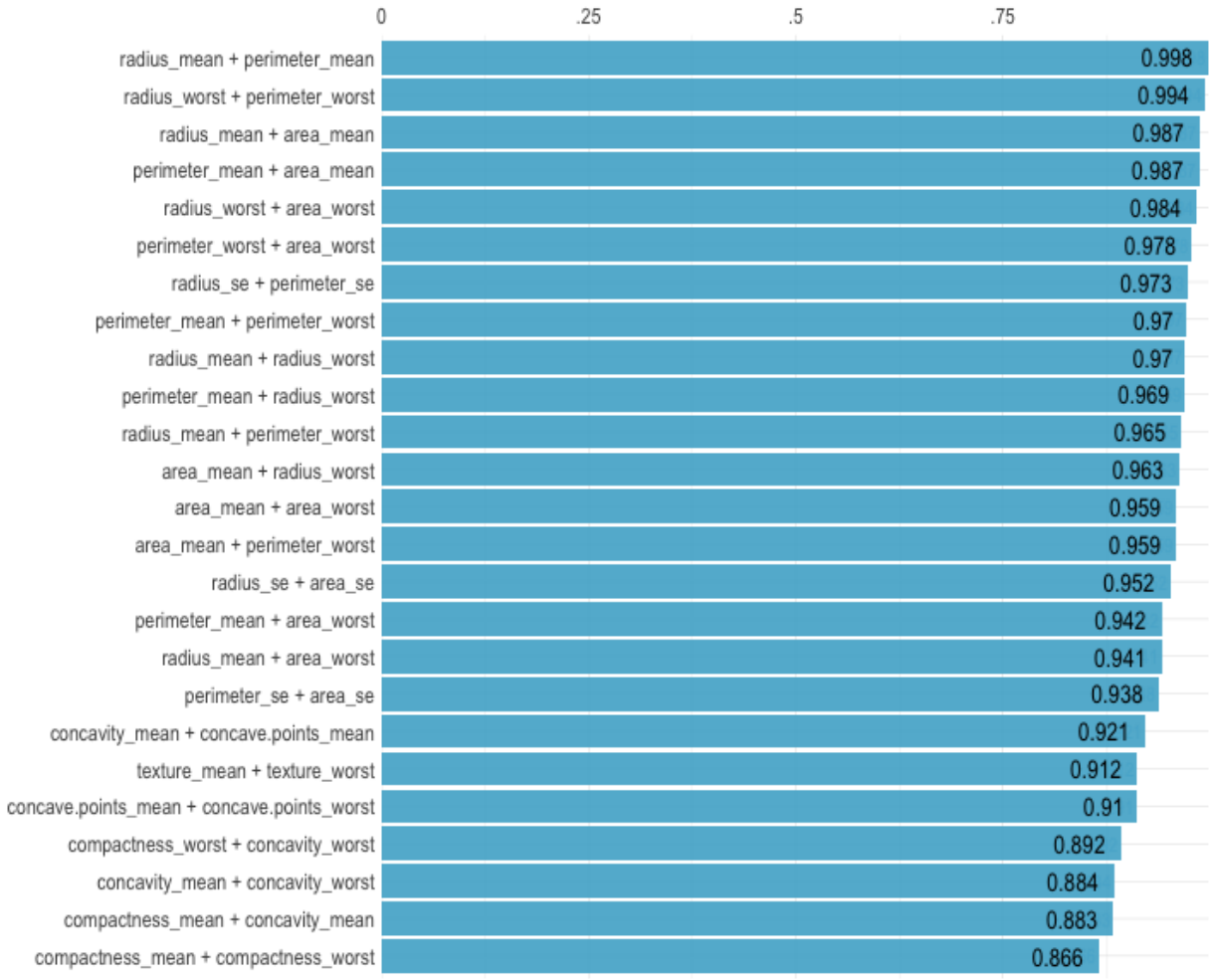

Table 1: Patient Characteristics

Variable	**B**, N = 357	**M**, N = 212	**p-value**
texture_mean	17.91 (4.00)	21.60 (3.78)	<0.001
smoothness_mean	0.09 (0.01)	0.10 (0.01)	<0.001
compactness_mean	0.08 (0.03)	0.15 (0.05)	<0.001
concave points_mean	0.03 (0.02)	0.09 (0.03)	<0.001
symmetry_mean	0.17 (0.02)	0.19 (0.03)	<0.001
fractal_dimension_mean	0.06 (0.01)	0.06 (0.01)	0.5
radius_se	0.28 (0.11)	0.61 (0.35)	<0.001
texture_se	1.22 (0.59)	1.21 (0.48)	0.6
smoothness_se	0.01 (0.00)	0.01 (0.00)	0.2
compactness_se	0.02 (0.02)	0.03 (0.02)	<0.001
concavity_se	0.03 (0.03)	0.04 (0.02)	<0.001
concave points_se	0.01 (0.01)	0.02 (0.01)	<0.001
symmetry_se	0.02 (0.01)	0.02 (0.01)	0.028
fractal_dimension_se	0.00 (0.00)	0.00 (0.00)	<0.001
radius_worst	13.38 (1.98)	21.13 (4.28)	<0.001
smoothness_worst	0.12 (0.02)	0.14 (0.02)	<0.001

Variable	**B**, N = 357	**M**, N = 212	**p-value**
compactness_worst	0.18 (0.09)	0.37 (0.17)	<0.001
concavity_worst	0.17 (0.14)	0.45 (0.18)	<0.001
symmetry_worst	0.27 (0.04)	0.32 (0.07)	<0.001
fractal_dimension_worst	0.08 (0.01)	0.09 (0.02)	<0.001

2.2 Newton-Raphson Algorithm

To implement the Newton-Raphson Algorithm we need to first examine the likelihood, and derive the gradient and Hessian matrix.

First we will look at the likelihood function for our data which has a single binary response and 20 numerical explanatory variables. First we know that

$$\pi_i = P(Y_i = 1|x_{i,1}, \dots x_{i,20}) = \frac{e^{\beta_0+\beta_1 x_{i,1}+\dots\beta_{20}x_{i,20}}}{1+e^{\beta_0+\beta_1 x_{i,1}+\dots\beta_{20}x_{i,20}}} = \frac{e^{\beta_0+\sum_{j=1}^{20}\beta_j x_{i,j}}}{1+e^{\beta_0+\sum_{j=1}^{20}\beta_j x_{i,j}}}$$

where $\mathbf{X_i}$ represents the i observation of all 20 of our predictor variables. For the data likelihood funciton is is given by

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

where y_i represents out binary outcome for the i individual. Finding the log-likelihood we have

$$l(\mathbf{X}|\vec{\beta}) = \sum_{i=1}^{n}\left[y_i\left(\beta_0+\sum_{j=1}^{20}\beta_j x_{i,j}\right) - \log\left(1+\exp\left(\beta_0+\sum_{j=1}^{20}\beta_j x_{i,j}\right)\right)\right]$$

The gradient is the partial derivative of the log-likelihood with respect to each β_j variable. Observe

$$\nabla l(\mathbf{X}|\vec{\beta}) = \begin{bmatrix}\sum_{i=1}^{n} y_i-\pi_i & \sum_{i=1}^{n} x_{i,1}(y_i-\pi_i) & \dots & \sum_{i=1}^{n} x_{i,20}(y_i-\pi_i)\end{bmatrix}^T_{(1\times 21)}$$

Finally, using the gradient we can derive our hessian matrix. Note that due to the 20 predictor variables the hessian will be a 21 by 21 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 $X = (x_{i,1}, \dots, x_{i,20})$. Note that this matrix will always be negative definite at all parameters making the this a well behaved problem. Using the likelihood, gradient and hessian the Newton-Raphson algorithm was implement in R and can be seen in **Appendix #**. There are two modification that have been made to the algorithm. The first is to control assent direction by checking the eigen values are negative representing a negative definite matrix. The second modification is to include half stepping to increase the speed of the algorithm. Note that major problems arise when highly correlated variables are included in the model. Without excluding the 1 variables the Newton-Raphson method would not converge due to multicollinearity issues.

2.3 Logistic LASSO Algorithm

Lemma 1. Consider the optimization problem

$$\min_{x\in\mathbb{R}}\left\{\frac{1}{2}(x-b)^2+c|x|\right\}$$

for $b\in\mathbb{R}$ and $c\in\mathbb{R}_{++}$. It follows that the minimizer is given by

$$\hat{x}=S(b,c),$$

where S is the soft-thresholding operator.

Lemma 2. Consider the optimization problem

$$\min_{\beta_k\in\mathbb{R}}\left\{\frac{1}{2n}\sum_{i=1}^{n}w_i\left(z_i-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2\right\}$$

for some $k\in\{1,\dots,p\}$. It follows that the minimizer is given by

$$\hat{\beta}_k=\left(\sum_{i=1}^{n}w_i x_{ik}^2\right)^{-1}\sum_{i=1}^{n}w_i x_{ik}\left(z_i-\sum_{j\neq k}\beta_j x_{ij}\right).$$

Lemma 3. With $\hat{\beta}_k$ defined as above,

$$\begin{aligned}&\min_{\beta_k\in\mathbb{R}}\left\{\frac{1}{2n}\sum_{i=1}^{n}w_i\left(z_i-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}|\beta_j|\right\}\\&\quad=\min_{\beta_k\in\mathbb{R}}\left\{\frac{1}{2}(\beta_k-\hat{\beta}_k)^2+\left(\frac{1}{n}\sum_{i=1}^{n}w_i x_{ik}^2\right)^{-1}\lambda|\beta_k|\right\}.\end{aligned}$$

Proposition. By Lemma 1 and Lemma 3,

$$\begin{aligned}&\arg\min_{\beta_k\in\mathbb{R}}\left\{\frac{1}{2n}\sum_{i=1}^{n}w_i\left(z_i-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}|\beta_j|\right\}\\&\quad=S\left(\hat{\beta}_k,\left(\frac{1}{n}\sum_{i=1}^{n}w_i x_{ik}^2\right)^{-1}\lambda\right)\end{aligned}$$

2.4 5-fold Cross Validation

To properly implement the logistic LASSO model an optimal λ must be selected. First, we need to define our range of possible λ values. To start we define the biggest value as λ_{max}. This is defined as the smallest penalty for which $\beta_k=0$ for all $k\in\{1,\dots,p\}$. The smallest value in our range is $\lambda_{min}=\lambda_{max}/1000$. This formula was recommended in a paper **Find paper**, however, we found it does not achieve our derived minimum property. Therefore we imperially found λ_{min} by starting with the suggested formula and then selecting a value a quite a bit smaller (a fourth of the size). Thus, our range of values we are going to try for λ is $(\lambda_{max},\lambda_{min})$ with step sizes between values on a log scale result is 100 values. This process will decrement through the λ values.

Our goal is to select an optimal λ value. To select the best value we will implement a 5-fold Cross Validation system. In **Figure 3** the first step of the process is to split the data into train and test data as described above. Using just the training data the cross validation procedure will implement a 5 fold process. In each fold the data will be split into 5 chunks. During the first fold the first split of data will be used as a test set while the 4 other splits will be used as the training data. The second fold will use the second split as the test and the rest as train set. This process will continue for each fold. Each fold will produce an AUC value for each λ in our range values. For example, in on Fold we have 100 λ values all with one corresponding AUC value. Thus, for 5 folds each λ will have 5 AUC values.

To select our optimal we will use two processes greatest mean AUC or Minimax AUC

The greatest mean AUC takes the average AUC value for each 100 λ in our range. Then is selects the largest AUC value as the optimal model.

The Minimax AUC selects the worst AUC value for each 100 λ in our range. Then the maximum worst AUC is selected as our optional model.

The greatest mean AUC and Minimax AUC optimal models will be compared against the full model.

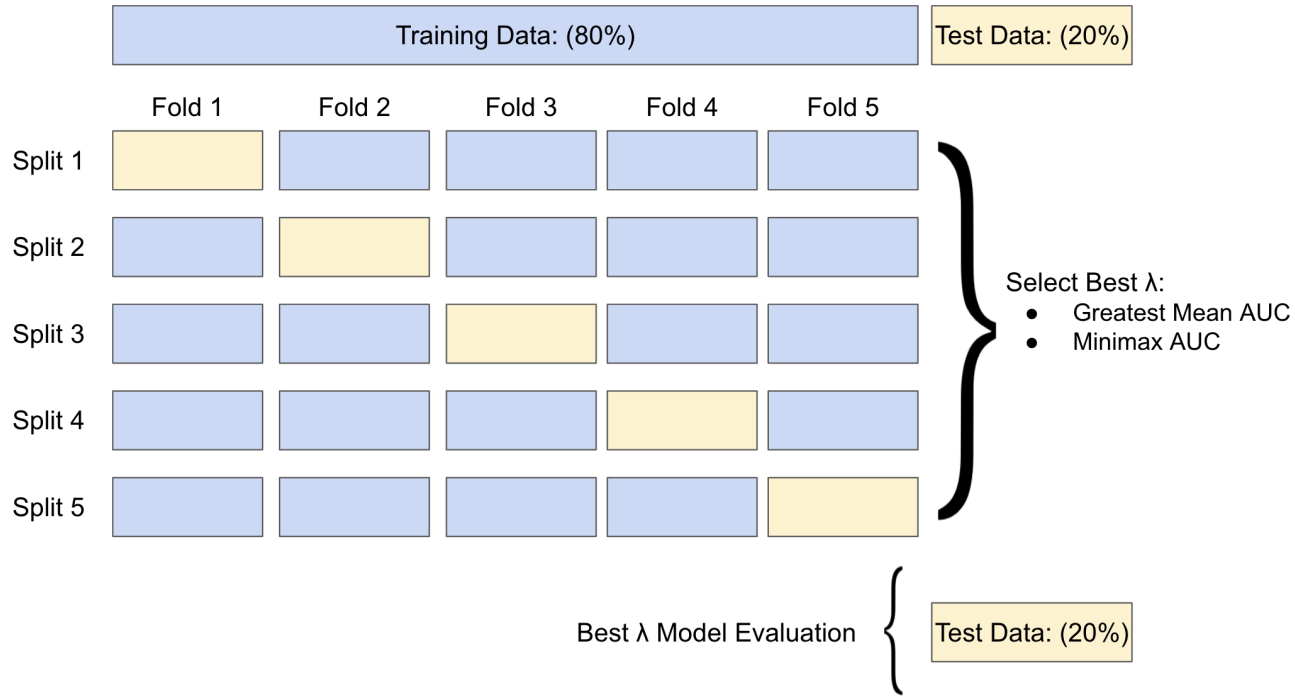

Figure 1: Figure 3

3.0 Results

3.1 Cross-Validation Results

3.2 Coefficient Model Comparison

3.3 Model Validation Results

4.0 Dissucsion

4.1. Summary of Findings

4.2. Limitations

4.3. Next Steps and Future Research

4.4. Group Contributions

References

Appendix