Evaluating Binary Medical Data Using A Bayesian Lasso Approach

By

Vindyani Herath

A project report

Submitted to the

Department of Mathematics and Statistics

in partial fulfillment of the requirements for the

Degree of Master of Science in Statistics

2021

Department of Mathematics and Statistics

Sam Houston State University

ABSTRACT

Variable selection in high-dimensional linear models arises in many fields of scientific research. The classic subset selection method provides an interpretable model. However, it can only be applied when the design matrix is full rank. Lasso model (Tibshirani, 1996) is one of the most popular penalized regression techniques which address this issue. Bae and Mallick (2004) proposed a two-level hierarchical Bayesian model and a corresponding Gibbs sampler for probit binary regression. This paper describes a hierarchical Bayesian approach of the lasso model for binary responses by implementing sparsity promoting priors. The Laplace distribution is then proposed to retain the absolute value of the parameter, i.e., L_1 Norm. The Gibbs sampler, an MCMC method, is developed to obtain the Bayesian lasso estimates from the posterior distributions. Gibbs sampling intends to develop posterior samples by evaluating each variable to sample from its conditional distribution repeatedly with the remaining variables fixed. The performance and applicability of the proposed Bayesian lasso approach are assessed through simulation studies and analysis of a cardiovascular disease data set. Both the simulation study and the real-world data show that the proposed model provides good performance and reduces the dimension. Further study is also discussed at the end of the article.

TABLE OF CONTENTS

INTRODUCTION

Cardiovascular disease (CVD) is a general term for conditions affecting the heart or blood vessels. CVD covers a wide array of disorders including coronary heart disease, cerebrovascular disease, rheumatic heart disease and other conditions. It is usually associated with a build-up of fatty deposits inside the arteries and an increase risk of blood clots. It can also be associated with damage to arteries in organs such as the brain, heart, kidneys, and eyes. CVDs are the number one cause of death globally, taking an estimated 17.9 million lives each year.

Most cardiovascular diseases can be prevented by addressing behavioral risk factors and leading a healthy lifestyle. The main risk factors of CVDs include high blood pressure, smoking, high cholesterol, diabetes, physical inactivity, being overweight or obese, family history of CVDs and ethnic background, among others. Factors such as age, gender, diet, and excessive alcohol consumption can affect your risk of developing cardiovascular disease as well. There are several steps that can be taken to reduce the risk for heart disease such as healthy eating, exercise, avoidance of tobacco use, and limiting alcohol intake. People with cardiovascular disease or who are at high cardiovascular risk (due to the presence of one or more risk factors such as hypertension, diabetes or already established disease) need early detection and management as appropriate. Identifying those at the highest risk of CVDs and ensuring they receive appropriate treatment can prevent premature deaths.

The objective of this study is to predict whether a patient is at risk of developing cardiovascular disease by analyzing risk factors. We consider a model that can be used when the number of variables in the dataset exceeds the number of observations. That leads to developing a hierarchical Bayesian lasso model using Gibbs sampler. A multivariate Bayesian regression model is considered and the priors that favor sparseness were assigned.

Data Description

Data were obtained from the website ‘www.kaggle.com’ and it includes three types of input features: factual information, results of medical examination, and information given by the patients. The data set consists of 70,000 observations with 11 predictor variables and a binary response variable. All the dataset values were collected at the moment of medical examination.

Description of Variables

- Presence/ absence of cardiovascular disease – Binary response variable
- Age
- Height
- Weight
- Gender (1 – Female, 2 – Male)
- Systolic blood pressure
- Diastolic blood pressure
- Cholesterol level (1 – Normal, 2 – Above normal, 3 – Well above normal)
- Glucose level (1 – Normal, 2 – Above normal, 3 – Well above normal)
- Smoking
- Alcohol intake
- Physical activity

LITERATURE REVIEW

Ordinary least squares (OLS) can be considered as a common method for fitting a regression line, $Y = \mu + x'\beta + \varepsilon$. It estimates the parameters in a regression model by minimizing the sum of the squared residuals. However, the OLS estimates often have low bias and large variance which affects the prediction accuracy. This issue can be improved by shrinking or setting some coefficients to 0. One other issue with using OLS estimates is the interpretation of the model. When there is a large number of predictors in the model, interpretation cannot be easily done. Therefore, we would like to select a smaller subset of predictors which reveal the strongest effects.

Two methods to improve OLS estimates are subset selection and ridge regression. Even though subset selection provides an interpretable model, small changes in the dataset can result in very different models which results in lower prediction accuracy. Furthermore, these subset selection procedures cannot be applied when the number of predictors is greater than the number of observations (p >> n), i.e., subset selection can only be applied when the design matrix is full rank. Ridge regression can be considered as a stable process since this technique shrinks the coefficients. It creates a parsimonious model and can be used when the number of predictor variables in a set exceeds the number of observations, i.e., when the design matrix is not full rank. The ridge regression solutions can be found using,

$$\hat{\beta}^{ridge} = (X'X + \lambda I)^{-1}X'Y$$

and the additional positive constant (λI) makes a nonsingular design matrix even if $X'X$ is not full rank. However, ridge regression does not set any coefficient to zero, hence the efficiency of the model is not good.

Tibshirani (1996) proposed the lasso (least absolute shrinkage and selection operator) which is also a method that improves the OLS estimates. It uses regularization which is a technique that is used to avoid overfitting of data. Regularization is implemented by adding a penalty term to the best fit model from the trained data, to achieve a lesser variance with the tested data and it also restricts the influence of predictor variables over the output variable by compressing the coefficients. The lasso uses L_1 regularization technique and it shrinks some coefficients and sets others to zero which in turn gives greater prediction accuracy and increases model interpretability.

$$(y - x\beta)'(y - x\beta) + \lambda \sum_{j=1}^{d} |\beta|$$

λ is known as the tuning parameter and it denotes the amount of shrinkage. When λ equals zero, it implies that all features of the data are considered, and it is the same as OLS regression. When λ increases, it eliminates more and more features. $\sum_{j=1}^{p} |\beta|$ is the L_1 regularization term.

Figure 1 depicts the lasso regression when there are only two parameters. The residual sum of squares has elliptical contours, centered at the full least squares estimate.

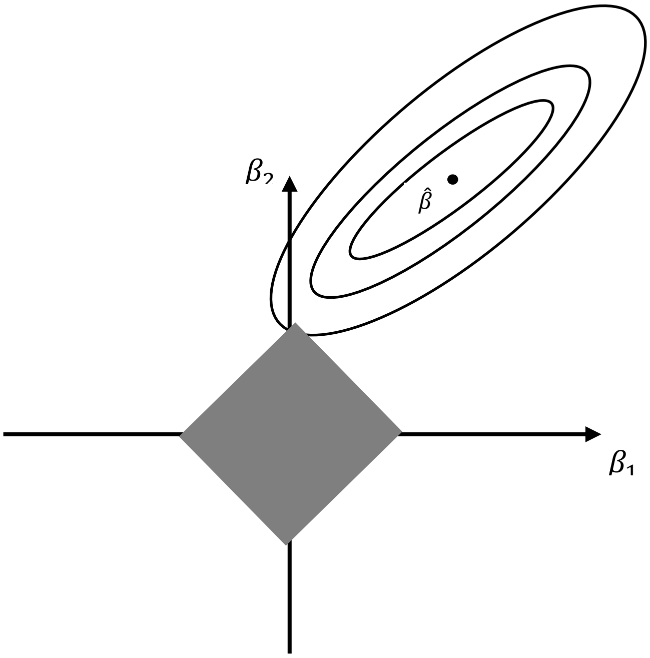

Figure 1: Two-dimensional Contour Plot of Lasso Regression

The constraint region for lasso is the diamond $|\beta_1| + |\beta_2| \leq t$. The elliptical contours are the cost function of linear regression. Lasso determines the coefficients by finding the first point where the elliptical contours hit the constraint region. Since the diamond has corners on the axes, whenever the elliptical region hits such a point, it has one parameter β_j equals to zero and one of the features completely vanishes. For higher dimensional feature space there can be many solutions on the axis with lasso regression and thus we get only the important features selected.

One major flaw of the linear probability model is that it assumes the conditional probability function to be linear. Therefore, an approach is needed that uses a nonlinear function to model the conditional probability function of a binary dependent variable. One method to fix this issue is using probit linear regression, which was first introduced by Bliss (1934). Lasso can be used to probit regression as well. In probit regression, the cumulative standard normal distribution function $\phi(.)$ is used the model the regression function where the dependent variable is binary.

Assume that Y is a binary response variable. The model

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k + \varepsilon$$

with

$$P(Y = 1|x_1, x_2, \ldots, x_k) = \phi(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k)$$

is the population probit model with multiple regressors $x_1, x_2, \ldots, x_k$ and $\phi(.)$ is the cumulative standard normal distribution function. Since ϕ is a nonlinear function of X, the estimated regression function has a stretched "S"– shape as shown, and it ensures that the predicted conditional probabilities lie between 0 and 1.

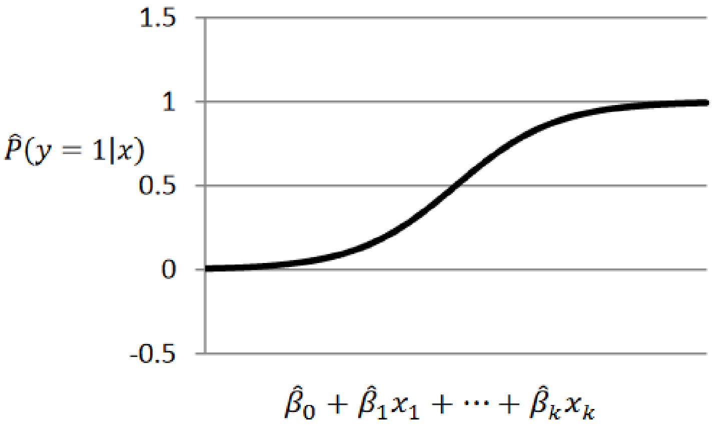

Figure 2 Plot of Probit Function

Bayesian statistics is a method to analyze data based on Bayes' theorem, in which available data about parameters (prior information) in a statistical model is updated with the help of the information in observed data. The background knowledge is used as prior distribution and combined with observational data in the form of a likelihood function to determine the posterior probability distribution for the parameter:

$$p(\theta|y) \propto p(y|\theta)p(\theta)$$

The term $p(y|\theta)$ represents the conditional probability of the data given the model parameters, (i.e., likelihood function) and the term $p(\theta)$ represents the probability of model parameter values existing in the population (i.e., prior distribution). The posterior distribution provides the basis for statistical inferences related to the parameter. The Bayesian model provides a clear approach for specifying advance hierarchical models for complex data.

The Bayesian lasso of Park and Casella (2008) provides valid standard errors for β and provides more stable point estimates by using the posterior center. According to Tibshirani (1996), the lasso estimate is equivalent to the mode of the posterior distribution under a normal likelihood and an independent Laplace (double exponential) prior:

$$\pi(\beta) = \frac{\lambda}{2} exp\left(-\lambda|\beta_j|\right)$$

The Bayesian lasso estimates appear to be a compromise between the Lasso and ridge regression estimates. Even though Bayesian lasso estimates are computationally intensive, they are easy to implement and provides interval estimates for all parameters, including error variance.

One important technique in Bayesian sampling is the Gibbs sampling method. The Gibbs sampling is a Markov Chain Monte Carlo (MCMC) method for simulating a random sample from a multivariate posterior $f(\theta)$. It constructs a Markov Chain whose values converge towards a target distribution. The idea in Gibbs sampling is to generate posterior samples by sweeping through each variable to sample from its conditional distribution with the remaining variables fixed to their current values.

Suppose that the parameter vector of interest is $\theta = \left(\theta_1, \dots, \theta_p\right)$. The joint posterior distribution of θ, which we denote by $[\theta|data]$, maybe of high dimension and difficult to summarize. Suppose we define the set of conditional distributions,

$$[\theta_1|\theta_2, \dots, \theta_p, data],$$

$$[\theta_2|\theta_1, \theta_3 \dots, \theta_p, data],$$

$$\vdots$$

$$[\theta_p|\theta_1, \dots, \theta_{p-1}, data],$$

where $[X|Y, Z]$ represents the distribution of X conditional on values of the random variables Y and Z. In Gibbs sampling we can set up a Markov chain simulation algorithm from the joint posterior distribution by successfully simulating individual parameters from the set of p conditional distributions. Simulating one value of each individual parameter from these distributions in turn is called one cycle of Gibbs sampling. Under general conditions, draws from this simulation algorithm will converge to the target distribution of interest. Gibbs sampling can be especially useful for hierarchical models.

METHODOLOGY

Assume y is a binary response variable with Bernoulli (p) probability model and there is a vector of regressors X. Then the model takes the form of:

$$P(Y = 1|X) = \phi(X'\beta)$$

It is possible to extend the probit model as a latent variable model. Suppose there exist a latent variable z such that,

$$z = x'\beta + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0,1)$$

It can be observed that,

$$y_i = \begin{cases} 0, if\ z_i \leq 0 \\ 1, if\ z_i > 0 \end{cases}$$

When $z_i > 0 \rightarrow x'\beta + \varepsilon > 0, \rightarrow \varepsilon > -x'\beta$

Then,

$$\begin{aligned} P(Y = 1|X) \quad &= P(z > 0) \\ &= P(x'\beta + \varepsilon > 0) \\ &= P(\varepsilon > -x'\beta) \\ &= P(\varepsilon < x'\beta) \\ &= \phi(x'\beta) \end{aligned}$$

The likelihood of the probit linear regression is:

$$\prod_{i=1}^{n} p^{y_i}(1-p)^{(1-y_i)}$$

$$\prod_{i=1}^{n}[\phi(x'\beta)]^{y_i}[1-\phi(x'\beta)]^{(1-y_i)}$$

We can implement MLE on the probit model which maximize the log-likelihood and in turn minimizes the negative of the log-likelihood.

$$\left(-\sum_{i=1}^{n} y_i \log\,(\phi(x'\beta))\right) - \sum_{i=1}^{n}(1-y_i)\log\big(\phi(-x'\beta)\big) + \lambda||\beta||_1$$

where $||\beta||_1 = \sum_{j=1}^{d}\left|\beta_j\right|$ (L1 norm) provides the Lasso property, and λ is the tuning parameter.

The lasso constraint $\sum_{j=1}^{d}\left|\beta_j\right| \leq \lambda$ is equivalent to the addition of a penalty term $\lambda\sum\left|\beta_j\right|$ to the residual sum of squares. Tibshirani (1996) suggested that Lasso estimates can be interpreted as posterior mode estimates when the regression parameters have independent and identical Laplace (i.e., double-exponential) priors because $\left|\beta_j\right|$ is proportional to the (minus) log- density of the double exponential distribution.

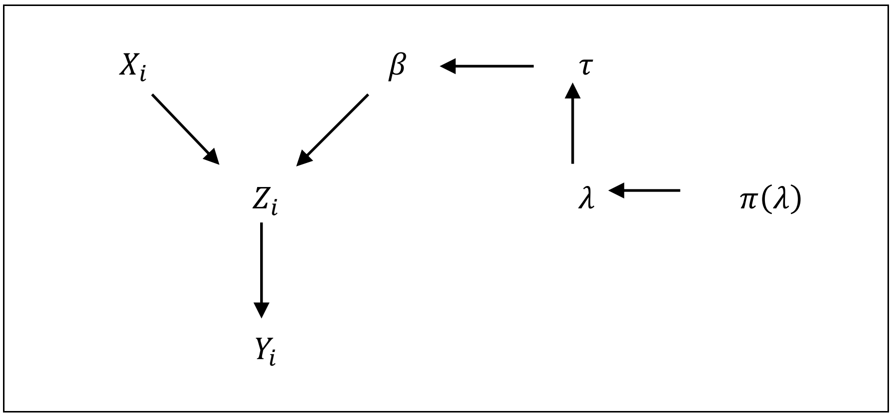

Figure 3: Graphical representation of the hierarchical model

If $\beta \sim Laplace(0, {}^{1}/_{\lambda})$, then the distribution will take the following form:

$$f(\beta_j|\lambda) = \frac{\lambda}{2}\exp\left(-\lambda|\beta|\right)$$

Then, $\prod_{i=1}^{d}\left\{\frac{\lambda}{2}\exp\left(-\lambda|\beta|\right)\right\} = \left(\frac{\lambda}{2}\right)^{d} exp\,(\lambda||\beta||_1)$

Therefore, the Bayesian hierarchical model is as follows:

$y_i = \mathbb{I}_{(0,\infty)}(Z_i)$

$Z_i|\beta \sim \mathcal{N}(X'\beta, 1)$

$\beta|\tau_1^2, \dots, \tau_d^2 \sim \mathcal{N}_d(0_d, D_\tau)$, where $D_\tau = diag(\tau_1^2, \tau_2^2, \dots, \tau_d^2)$

$\tau_1^2, \tau_2^2, \dots, \tau_d^2 \sim Exp\left(\frac{\lambda^2}{2}\right)$

$\lambda \sim \pi(\lambda) \propto C$

$\tau_1^2, \tau_2^2, \dots, \tau_d^2 > 0$

Also, figure (3) demonstrates the Bayesian hierarchical model.

The posterior distribution can be found by multiplying the prior with the following marginal distributions.

$Z_i|\tau, \beta, \lambda, y_i = [TN(x'\beta, 1, 0, \infty)]^{y_i} \times [TN(x'\beta, 1, -\infty, 0)]^{1-y_i}$

$$\beta|\tau, Z, \lambda, y_i \propto \prod_{i=1}^{N} \phi(Z_i, x'\beta, 1) \times \phi(\beta; 0, D_\tau)$$

$\beta|\tau, Z, \lambda, y_i \sim \mathcal{N}(A^{-1}x'Z, A^{-1})$ where $A = D_\tau^{-1} + X'X$

$$\tau^2|Z, \beta, \lambda, y \propto \phi(\beta, 0, D_\tau) \times \prod_{i=1}^{d} \frac{\lambda^2}{2} e^{\frac{-\lambda^2\tau_j^2}{2}}$$

$\tau_j^{-2}|\, z, \beta, \lambda, y \sim IG(\mu_j, \lambda^2)$ with $\mu_j = \lambda|\beta_j|^{-1}$

$$\lambda|\tau, \beta, z, y \ \propto \prod_{j=1}^{d} \frac{\lambda^2}{2} e^{\frac{-\lambda^2\tau_j^2}{2}} \times \pi(\lambda)$$

where $\lambda|\tau, \beta, z, y \sim Gamma(d + 1, \frac{1}{2}\sum_{j=1}^{d}\tau_j^2)$

RESULTS AND DISCUSSION

SIMULATION STUDIES

Here, Monte Carlo simulations are carried out to study the performance of the proposed Bayesian lasso method. This method is carried out in terms of variable selection and prediction. we consider 1024 observations with 15 predictors and a response variable. It is assumed that the predictor variables follow the below distributions.

$$X_1 \sim N(n, 3,\ 1)$$

$$X_2 \sim N(n,\ X_1,\ 1)$$

$$X_3 \sim N(n,\ X_2,\ 2)$$

$$X_4 \sim U(n, 5,\ 10)$$

$$X_5 \sim U(n,\ X_4,\ \ X_4 + 3)$$

$$X_6 \sim N(n, 3.5,\ 1)$$

$$X_7 \sim N(n,\ X_6,\ 1)$$

$$X_8 \sim N(n, 5.2,\ 2)$$

$$X_9 \sim U(n,\ X_8\ ,\ X_8 + 3)$$

$$X_{10} \sim U(n,\ X_9, X_9 + 1)$$

$$X_{11} \sim N(n,\ 5,\ 1)$$

$$X_{12} \sim N(n,\ X_{11}\ , 1)$$

$$X_{13} \sim N(n,\ X_{12},\ 2)$$

$$X_{14} \sim U(n\ , 5,\ 10)$$

$$X_{15} \sim U(n,\ X_{14},\ X_{14} + 3)$$

Assume a linear regression model between the response variable and the 15 covariates.
The response variable was generated according to the model,

$$y_i = 4x_{1i} - 4x_{2i} + 5x_{3i} + 7x_{4i} - 6x_{5i} + \varepsilon_i$$

Where $\varepsilon_i \sim N(0, \sigma^2)$ and the true regression coefficients are $\beta = (4, -4,\ 5,\ 7, -6, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)'$. Probabilities of success were obtained using the probit link. Training data set was generated with 70% of observations. In the simulation study, a Gibbs sequence of 270,000 iterations was generated and the first 20,000 iterations were used as burn-in. The Bayesian estimates are posterior means. The results are summarized in Table (1) and it clearly shows that the estimated regression coefficients are approximately equal to the true regression coefficients.

Table 1: Comparison of true and estimated regression coefficients

Variable	True β	$\hat{\beta}$
X_1	4	3.959
X_2	-4	-4.100
X_3	5	5.184
X_4	7	6.985
X_5	-6	-5.939
X_6	0	-0.308
X_7	0	0.312
X_8	0	-0.107
X_9	0	0.773
X_{10}	0	-0.737
X_{11}	0	-0.449
X_{12}	0	0.063
X_{13}	0	0.131
X_{14}	0	0.334
X_{15}	0	-0.097

Figure (4) shows the distribution of estimated regression coefficients for 30 simulations. It can be observed that the estimated regression coefficients converge to the true regression coefficients. Therefore, the performance of the Bayesian lasso model appears quite good.

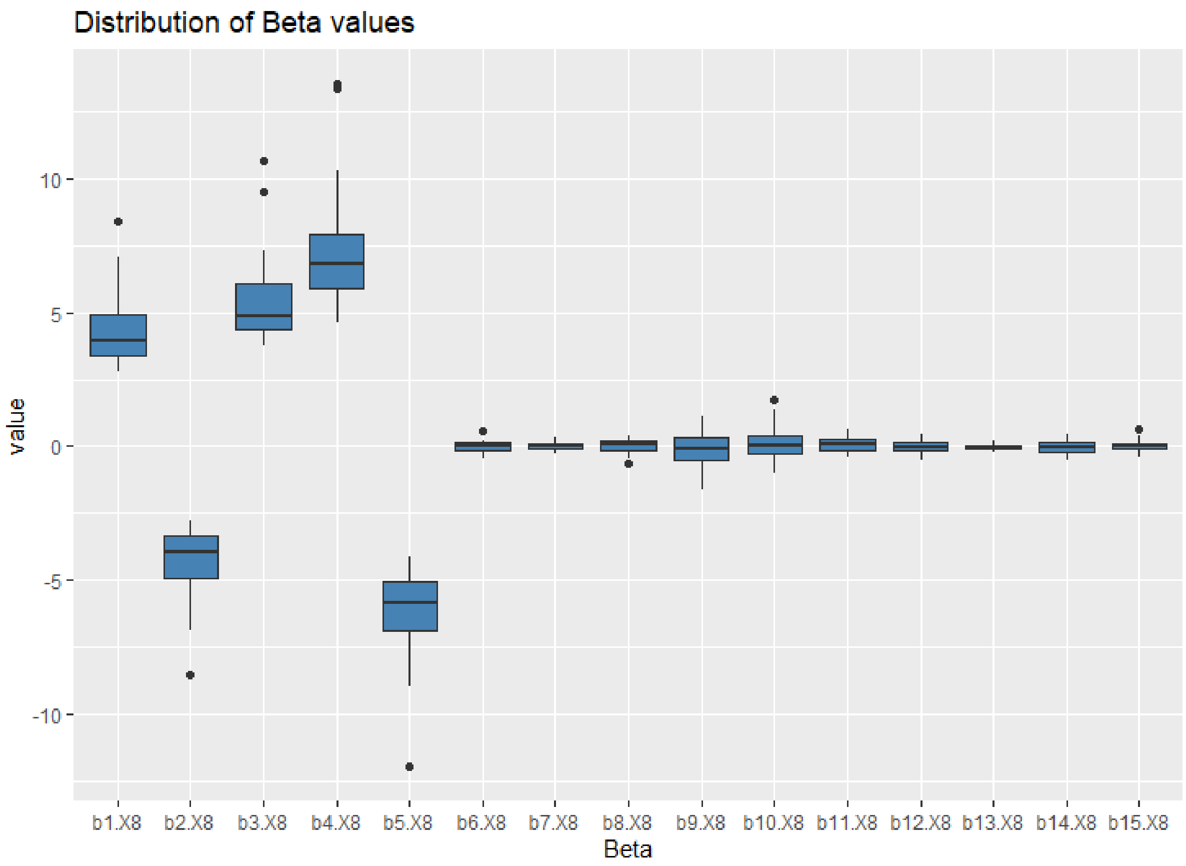

Figure 4: Distribution of estimated regression coefficients

APPLICATION OF VARIABLE SELECTION

Cardiovascular Dataset

Here, we consider the cardiovascular data which is available for public use in Kaggle. We compare this method in terms of prediction and variable selection. There are 70,000 observations, 11 predictors and a response variable. Predictors include age, height, weight, gender, systolic blood pressure, diastolic blood pressure, cholesterol, glucose, smoking, alcohol intake and physical activity. The data are split into a training set consisting of 49,000 observations of which 24,537 are patients with cardiovascular disease, and a test set of 21,000 observations of which 10,442 are patients with cardiovascular disease.

Assume a linear regression model between the response variable and the 11 covariates. To analyze the dataset, the covariates have been standardized. We run the Gibbs sampling algorithm with 270,000 iterations and 20,000 burn-in. We obtain samples from the marginal posterior distribution and obtain the estimates for β_i's. The Bayesian estimates were obtained using posterior means and results are summarized in Table (2).

Table 2 Bayesian estimates of the covariates.

Covariate	$\hat{\beta}$
Age	0.2778
Gender	-0.0861
Height	-0.0229
Weight	0.2135
Systolic blood pressure	0.1665
Diastolic blood pressure	0.0750
Cholesterol	0.3377
Glucose	-0.1318
Smoke	-0.0262
Alcohol intake	-0.0468
Physical activity	-0.1969

64.72% of accuracy was obtained using the test dataset. Table (3) shows the comparison between the response variable of test data and the predicted response from the Bayesian lasso model.

Table 3: Cardiovascular Data: Prediction of the test data

Test Data	Predicted Response
1	1
0	1
1	0
0	0
1	0
0	1
:	:
:	:
:	:
:	:
:	:

It can be said that the Bayesian lasso model provides good prediction accuracy for the cardiovascular dataset. The model can be used to predict whether a person is at risk of developing cardiovascular disease. Furthermore, the probability of the risk of developing cardiovascular disease can also be determined by the proposed model.

CONCLUSION

In this paper, we have developed a hierarchical Bayesian version of the Lasso model for binary responses by implementing sparsity promoting priors. The main advantage of this approach is that the method can identify the significant predictor variables even when the column size of the design matrix is too large (high-dimensional data). The study uses L_1 regularization technique since it shrinks some coefficients and sets others to zero which in turn gives greater prediction accuracy and increases model interpretability. The Bayesian approach provides a Laplace prior distribution for the regression parameters. Then a Gibbs sampling algorithm is developed to obtain the Bayesian lasso estimates from the posterior distributions.

The applicability of the proposed methodology was shown on both simulated data and cardiovascular disease data. The results showed that the Monte Carlo experiments perform well in terms of prediction accuracy and variable selection. The proposed hierarchical Bayesian lasso model provides good prediction to identify patients with cardiovascular disease.

REFERENCES

01. Benoit, D.F., Alhamzawi, R. & Yu, K. Bayesian lasso binary quantile regression. Comput Stat 28, 2861–2873 (2013). https://doi.org/10.1007/s00180-013-0439-0

02. Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani. An Introduction to Statistical Learning : with Applications in R. New York :Springer, 2013.

03. Hanck C., Arnold M., Gerber A, Schmelzer M, Introduction to Econometrics with R

04. Hastie, T., Hastie, T., Tibshirani, R., & Friedman, J. H. (2001). The elements of statistical learning: Data mining, inference, and prediction. New York: Springer

05. Kyounghwa Bae, Bani K. Mallick, Gene selection using a two-level hierarchical Bayesian model, Bioinformatics, Volume 20, Issue 18, 12 December 2004, Pages 3423–3430, https://doi.org/10.1093/bioinformatics/bth419

06. Mendis S, Puska P, Norrving B (2011). Global Atlas on Cardiovascular Disease Prevention and Control (PDF). World Health Organization in collaboration with the World Heart Federation and the World Stroke Organization. pp. 3–18. ISBN 978-92-4-156437-3. Archived (PDF) from the original on 2014-08-17.

07. National Health Service (2018, September 17). “Cardiovascular Disease”. Retrieved from https://www.nhs.uk/conditions/cardiovascular-disease/

08. Park, Trevor & Casella, George, 2008. "The Bayesian Lasso," Journal of the American Statistical Association, American Statistical Association, vol. 103, pages 681-686, June.

09. Rahim Alhamzawi & Haithem Taha Mohammad Ali (2020) A new Gibbs sampler for Bayesian lasso, Communications in Statistics - Simulation and Computation, 49:7, 1855-1871, DOI: 10.1080/03610918.2018.1508699

10. Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological), 267-288.

11. World Health Organization (2017, May 17). “Cardiovascular Diseases (CVDs)” . Retrieved from https://www.who.int/en/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)

APPENDIX

R Code

```
###################################
## Title: Bayesian Lasso Approach ######
## Name: Vindyani Herath ############
## Start Date: August 17, 2020 #########
## Last Updated: March 27, 2021 #######
###################################
library(mvtnorm)
library(MASS)
library(truncnorm)
library(statmod)
library(VGAM)
library(LearnBayes)
## Simulation Study ##

set.seed(100)
n=1024
d=15
x1<-rnorm(n,3,1)
x2<-rnorm(n,x1,1)
x3<-rnorm(n,x2,2)
x4<-runif(n,5,10)
x5<-runif(n,x4,x4+3)
x6<-rnorm(n,3.5,1)
x7<-rnorm(n,x6,1)
x8<-rnorm(n,5.2,2)
x9<-runif(n,x8,x8+3)
x10<-runif(n,x9,x9+1)
x11<-rnorm(n,5,1)
x12<-rnorm(n,x11,1)
x13<-rnorm(n,x12,2)
x14<-runif(n,5,10)
x15<-runif(n,x14,x14+3)
X<-cbind(x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,x11,x12,x13,x14,x15)
X[1:5,]
summary(x8)
summary(x15)
apply(X,2,mean)
beta<-c(4,-4,5,7,-6,rep(0,10));beta
length(beta)
Xb<-X%*%beta
Xb

# Obtain the vector with probabilities of success p using the probit link
p<-pnorm(Xb)
```

```
p
y.data<-rbinom(1024,1,p)
y.data
length(y.data)
## find some initial betas
fit<-glm(y.data~0+X,family=binomial(link=probit))
summary(fit)
z<-rep(NA,n)
N.sim<-270000
N1<-sum(y.data);N1
N0<-n-N1;N0
## beta
beta<-matrix(NA,nrow=N.sim+1,ncol=d)
beta[1,]<-c(fit$coefficients)
beta
## tau
tau.2<-matrix(NA,nrow=N.sim+1,ncol=d)
tau.2[1,]<-c(rep(1,d))
tau.2
## lamda
lamda<-rep(NA,n)
lamda[1]<-2
lamda
## sampling
for (j in 1:N.sim){
  mu.z<-X%*%beta[j,]
  z[y.data==0]<-rtruncnorm(N0,mean=mu.z[y.data==0],sd=1,a=-Inf,b=0)
  z[y.data==1]<-rtruncnorm(N1,mean=mu.z[y.data==1],sd=1,a=0,b=Inf)
  tau<-diag(tau.2[j,])
  E<-solve(solve(tau)+t(X)%*%X)
  beta[j+1,]<-rmvnorm(1,E%*%t(X)%*%z,E)
  tau.2[j+1,]<-1/rinv.gaussian(d,(lamda[j]^2/beta[j+1,]^2)^0.5,lamda[j]^2)
  lamda[j+1]<-(rgamma(1,d+1,0.5*sum(tau.2[j+1,])))^0.5
}
## estimation
summary(lamda)
plot(lamda)
beta[,1]
tau.2[,1]
ind<-seq(from=20000,to=270000,by=125)
summary(lamda[ind])##estimate lambda
sd(lamda[ind])
ts.plot(lamda[ind]);acf(lamda[ind])
beta.x8Changed <- colMeans(beta[ind,])##estimate beta
colMeans(tau.2[ind,])##estimate tau
newb<-beta[ind,]
sd.beta<-rep(NA,15)
for (k in 1:15){
  sd.beta[k]<-sd(newb[,k])
}
sd.beta##sd of beta estimates
quantile(beta[ind,1],c(0.025,0.5,0.975))
quantile(beta[ind,2],c(0.025,0.5,0.975))
quantile(beta[ind,3],c(0.025,0.5,0.975))
quantile(beta[ind,4],c(0.025,0.5,0.975))
quantile(beta[ind,5],c(0.025,0.5,0.975))
quantile(beta[ind,6],c(0.025,0.5,0.975))
quantile(beta[ind,7],c(0.025,0.5,0.975))
quantile(beta[ind,8],c(0.025,0.5,0.975))
quantile(beta[ind,9],c(0.025,0.5,0.975))
quantile(beta[ind,10],c(0.025,0.5,0.975))
quantile(beta[ind,11],c(0.025,0.5,0.975))
quantile(beta[ind,12],c(0.025,0.5,0.975))
quantile(beta[ind,13],c(0.025,0.5,0.975))
quantile(beta[ind,14],c(0.025,0.5,0.975))
quantile(beta[ind,15],c(0.025,0.5,0.975))
## plots check
par(mfrow=c(2,2))
ts.plot(lamda);acf(lamda,lag=5000)
ts.plot(beta[,1]);acf(beta[,1],lag=500)
ts.plot(beta[,5]);acf(beta[,5])
ts.plot(tau.2[,1]);acf(tau.2[,1])
ts.plot(tau.2[,5]);acf(tau.2[,5])
## after burn in and slicing plots
ts.plot(beta[ind,1]);acf(beta[ind,1],lag=100)

## Boxplot of distribution of beta values ##
set.seed(0204)
A.X8 <- replicate(40,{
  n=1024
  d=15
  x1<-rnorm(n,3,1)
  x2<-rnorm(n,x1,1)
  x3<-rnorm(n,x2,2)
  x4<-runif(n,5,10)
  x5<-runif(n,x4,x4+3)
  x6<-rnorm(n,3.5,1)
  x7<-rnorm(n,x6,1)
  x8<-rnorm(n,5.2,2)
  x9<-runif(n,x8,x8+3)
  x10<-runif(n,x9,x9+1)
  x11<-rnorm(n,5,1)
  x12<-rnorm(n,x11,1)
  x13<-rnorm(n,x12,2)
  x14<-runif(n,5,10)
  x15<-runif(n,x14,x14+3)
  X<-cbind(x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,x11,x12,x13,x14,x15)
  #X[1:5,]
  #summary(x8)
  #summary(x15)
  apply(X,2,mean)
  beta<-c(4,-4,5,7,-6,rep(0,10));beta
  #length(beta)
  Xb<-X%*%beta
  #Xb

  # Obtain the vector with probabilities of success p using the probit link
  p<-pnorm(Xb)
  y.data<-rbinom(1024,1,p)
  fit<-glm(y.data~0+X,family=binomial(link=logit))
  z<-rep(NA,n)
  N.sim<-270000
  N1<-sum(y.data);N1
  N0<-n-N1;N0
  ## beta
  beta<-matrix(NA,nrow=N.sim+1,ncol=d)
  (beta[1,]<-c(fit$coefficients))
  tau.2<-matrix(NA,nrow=N.sim+1,ncol=d)
  tau.2[1,]<-c(rep(1,d))
  lamda<-rep(NA,n)
  lamda[1]<-2
  ## sampling
  for (j in 1:N.sim){
    mu.z<-X%*%beta[j,]
    z[y.data==0]<-rtruncnorm(N0,mean=mu.z[y.data==0],sd=1,a=-Inf,b=0)
    z[y.data==1]<-rtruncnorm(N1,mean=mu.z[y.data==1],sd=1,a=0,b=Inf)
    tau<-diag(tau.2[j,])
    E<-solve(solve(tau)+t(X)%*%X)
    beta[j+1,]<-rmvnorm(1,E%*%t(X)%*%z,E)
    tau.2[j+1,]<-1/rinv.gaussian(d,(lamda[j]^2/beta[j+1,]^2)^0.5,lamda[j]^2)
    lamda[j+1]<-(rgamma(1,d+1,0.5*sum(tau.2[j+1,])))^0.5
  }
ind<-seq(from=20000,to=270000,by=125)
bb <- colMeans(beta[ind,])##estimate beta
})
b1.X8<-
c(A.X8[1,26],A.X8[1,2],A.X8[1,30],A.X8[1,4],A.X8[1,5],A.X8[1,6],A.X8[1,7],A.X8[1,31],A.X8[1,9
],A.X8[1,10],A.X8[1,11],A.X8[1,37],A.X8[1,38],A.X8[1,15],A.X8[1,16],A.X8[1,17],A.X8[1,18],A.
X8[1,33],A.X8[1,20],A.X8[1,21],A.X8[1,22],A.X8[1,23],A.X8[1,24],A.X8[1,25],A.X8[1,27],A.X8[1
,28],A.X8[1,29],A.X8[1,34],A.X8[1,35],A.X8[1,36])


b2.X8<-
c(A.X8[2,26],A.X8[2,2],A.X8[2,30],A.X8[2,4],A.X8[2,5],A.X8[2,6],A.X8[2,7],A.X8[2,31],A.X8[2,9
],A.X8[2,10],A.X8[2,11],A.X8[2,37],A.X8[2,38],A.X8[2,15],A.X8[2,16],A.X8[2,17],A.X8[2,18],A.
X8[2,33],A.X8[2,20],A.X8[2,21],A.X8[2,22],A.X8[2,23],A.X8[2,24],A.X8[2,25],A.X8[2,27],A.X8[2
,28],A.X8[2,29],A.X8[2,34],A.X8[2,35],A.X8[2,36])

b3.X8<-
c(A.X8[3,26],A.X8[3,2],A.X8[3,30],A.X8[3,4],A.X8[3,5],A.X8[3,6],A.X8[3,7],A.X8[3,31],A.X8[3,9
],A.X8[3,10],A.X8[3,11],A.X8[3,37],A.X8[3,38],A.X8[3,15],A.X8[3,16],A.X8[3,17],A.X8[3,18],A.
X8[3,33],A.X8[3,20],A.X8[3,21],A.X8[3,22],A.X8[3,23],A.X8[3,24],A.X8[3,25],A.X8[3,27],A.X8[3
,28],A.X8[3,29],A.X8[3,34],A.X8[3,35],A.X8[3,36])

b4.X8<-
c(A.X8[4,26],A.X8[4,2],A.X8[4,30],A.X8[4,4],A.X8[4,5],A.X8[4,6],A.X8[4,7],A.X8[4,31],A.X8[4,9
],A.X8[4,10],A.X8[4,11],A.X8[4,37],A.X8[4,38],A.X8[4,15],A.X8[4,16],A.X8[4,17],A.X8[4,18],A.
X8[4,33],A.X8[4,20],A.X8[4,21],A.X8[4,22],A.X8[4,23],A.X8[4,24],A.X8[4,25],A.X8[4,27],A.X8[4
,28],A.X8[4,29],A.X8[4,34],A.X8[4,35],A.X8[4,36])

b5.X8<-
c(A.X8[5,26],A.X8[5,2],A.X8[5,30],A.X8[5,4],A.X8[5,5],A.X8[5,6],A.X8[5,7],A.X8[5,31],A.X8[5,9
],A.X8[5,10],A.X8[5,11],A.X8[5,37],A.X8[5,38],A.X8[5,15],A.X8[5,16],A.X8[5,17],A.X8[5,18],A.
X8[5,33],A.X8[5,20],A.X8[5,21],A.X8[5,22],A.X8[5,23],A.X8[5,24],A.X8[5,25],A.X8[5,27],A.X8[5
,28],A.X8[5,29],A.X8[5,34],A.X8[5,35],A.X8[5,36])

b6.X8<-
c(A.X8[6,26],A.X8[6,2],A.X8[6,30],A.X8[6,4],A.X8[6,5],A.X8[6,6],A.X8[6,7],A.X8[6,31],A.X8[6,9
],A.X8[6,10],A.X8[6,11],A.X8[6,37],A.X8[6,38],A.X8[6,15],A.X8[6,16],A.X8[6,17],A.X8[6,18],A.
X8[6,33],A.X8[6,20],A.X8[6,21],A.X8[6,22],A.X8[6,23],A.X8[6,24],A.X8[6,25],A.X8[6,27],A.X8[6
,28],A.X8[6,29],A.X8[6,34],A.X8[6,35],A.X8[6,36])

b7.X8<-
c(A.X8[7,26],A.X8[7,2],A.X8[7,30],A.X8[7,4],A.X8[7,5],A.X8[7,6],A.X8[7,7],A.X8[7,31],A.X8[7,9
],A.X8[7,10],A.X8[7,11],A.X8[7,37],A.X8[7,38],A.X8[7,15],A.X8[7,16],A.X8[7,17],A.X8[7,18],A.
X8[7,33],A.X8[7,20],A.X8[7,21],A.X8[7,22],A.X8[7,23],A.X8[7,24],A.X8[7,25],A.X8[7,27],A.X8[7
,28],A.X8[7,29],A.X8[7,34],A.X8[7,35],A.X8[7,36])

b8.X8<-
c(A.X8[8,26],A.X8[8,2],A.X8[8,30],A.X8[8,4],A.X8[8,5],A.X8[8,6],A.X8[8,7],A.X8[8,31],A.X8[8,9
],A.X8[8,10],A.X8[8,11],A.X8[8,37],A.X8[8,38],A.X8[8,15],A.X8[8,16],A.X8[8,17],A.X8[8,18],A.
X8[8,33],A.X8[8,20],A.X8[8,21],A.X8[8,22],A.X8[8,23],A.X8[8,24],A.X8[8,25],A.X8[8,27],A.X8[8
,28],A.X8[8,29],A.X8[8,34],A.X8[8,35],A.X8[8,36])

b9.X8<-
c(A.X8[9,26],A.X8[9,2],A.X8[9,30],A.X8[9,4],A.X8[9,5],A.X8[9,6],A.X8[9,7],A.X8[9,31],A.X8[9,9
],A.X8[9,10],A.X8[9,11],A.X8[9,37],A.X8[9,38],A.X8[9,15],A.X8[9,16],A.X8[9,17],A.X8[9,18],A.
X8[9,33],A.X8[9,20],A.X8[9,21],A.X8[9,22],A.X8[9,23],A.X8[9,24],A.X8[9,25],A.X8[9,27],A.X8[9
,28],A.X8[9,29],A.X8[9,34],A.X8[9,35],A.X8[9,36])

b10.X8<-
c(A.X8[10,26],A.X8[10,2],A.X8[10,30],A.X8[10,4],A.X8[10,5],A.X8[10,6],A.X8[10,7],A.X8[10,31]
,A.X8[10,9],A.X8[10,10],A.X8[10,11],A.X8[10,37],A.X8[10,38],A.X8[10,15],A.X8[10,16],A.X8[10
,17],A.X8[10,18],A.X8[10,33],A.X8[10,20],A.X8[10,21],A.X8[10,22],A.X8[10,23],A.X8[10,24],A.
X8[10,25],A.X8[10,27],A.X8[10,28],A.X8[10,29],A.X8[10,34],A.X8[10,35],A.X8[10,36])

b11.X8<-
c(A.X8[11,26],A.X8[11,2],A.X8[11,30],A.X8[11,4],A.X8[11,5],A.X8[11,6],A.X8[11,7],A.X8[11,31]
,A.X8[11,9],A.X8[11,10],A.X8[11,11],A.X8[11,37],A.X8[11,38],A.X8[11,15],A.X8[11,16],A.X8[11
,17],A.X8[11,18],A.X8[11,33],A.X8[11,20],A.X8[11,21],A.X8[11,22],A.X8[11,23],A.X8[11,24],A.
X8[11,25],A.X8[11,27],A.X8[11,28],A.X8[11,29],A.X8[11,34],A.X8[11,35],A.X8[11,36])

b12.X8<-
c(A.X8[12,26],A.X8[12,2],A.X8[12,30],A.X8[12,4],A.X8[12,5],A.X8[12,6],A.X8[12,7],A.X8[12,31]
,A.X8[12,9],A.X8[12,10],A.X8[12,11],A.X8[12,37],A.X8[12,38],A.X8[12,15],A.X8[12,16],A.X8[12
,17],A.X8[12,18],A.X8[12,33],A.X8[12,20],A.X8[12,21],A.X8[12,22],A.X8[12,23],A.X8[12,24],A.
X8[12,25],A.X8[12,27],A.X8[12,28],A.X8[12,29],A.X8[12,34],A.X8[12,35],A.X8[12,36])

b13.X8<-
c(A.X8[13,26],A.X8[13,2],A.X8[13,30],A.X8[13,4],A.X8[13,5],A.X8[13,6],A.X8[13,7],A.X8[13,31]
,A.X8[13,9],A.X8[13,10],A.X8[13,11],A.X8[13,37],A.X8[13,38],A.X8[13,15],A.X8[13,16],A.X8[13
,17],A.X8[13,18],A.X8[13,33],A.X8[13,20],A.X8[13,21],A.X8[13,22],A.X8[13,23],A.X8[13,24],A.
X8[13,25],A.X8[13,27],A.X8[13,28],A.X8[13,29],A.X8[13,34],A.X8[13,35],A.X8[13,36])

b14.X8<-
c(A.X8[14,26],A.X8[14,2],A.X8[14,30],A.X8[14,4],A.X8[14,5],A.X8[14,6],A.X8[14,7],A.X8[14,31]
,A.X8[14,9],A.X8[14,10],A.X8[14,11],A.X8[14,37],A.X8[14,38],A.X8[14,15],A.X8[14,16],A.X8[14
,17],A.X8[14,18],A.X8[14,33],A.X8[14,20],A.X8[14,21],A.X8[14,22],A.X8[14,23],A.X8[14,24],A.
X8[14,25],A.X8[14,27],A.X8[14,28],A.X8[14,29],A.X8[14,34],A.X8[14,35],A.X8[14,36])

b15.X8<-
c(A.X8[15,26],A.X8[15,2],A.X8[15,30],A.X8[15,4],A.X8[15,5],A.X8[15,6],A.X8[15,7],A.X8[15,31]
,A.X8[15,9],A.X8[15,10],A.X8[15,11],A.X8[15,37],A.X8[15,38],A.X8[15,15],A.X8[15,16],A.X8[15
,17],A.X8[15,18],A.X8[15,33],A.X8[15,20],A.X8[15,21],A.X8[15,22],A.X8[15,23],A.X8[15,24],A.
X8[15,25],A.X8[15,27],A.X8[15,28],A.X8[15,29],A.X8[15,34],A.X8[15,35],A.X8[15,36])

length(b1.X8)
max(b1.X8)
beta.df.X8<-
data.frame(b1.X8,b2.X8,b3.X8,b4.X8,b5.X8,b6.X8,b7.X8,b8.X8,b9.X8,b10.X8,b11.X8,b12.X8,b13.
X8,b14.X8,b15.X8)
library(ggplot2)
library(reshape)
beta_long.X8 <- melt(beta.df.X8)
ggplot(beta_long.X8, aes(x = variable, y = value)) +           # Applying ggplot function
  geom_boxplot(fill="steelblue") +
  labs(title = "Distribution of Beta values", x = "Beta")

## Application using real world data: Cardiovascular Disease Data ##
attach(cardio_train)
```

```
length(cardio_train)
nrow(cardio_train)
cardio <- cardio_train
nrow(cardio)
library(mvtnorm)
library(MASS)
library(truncnorm)
library(statmod)
library(VGAM)
library(LearnBayes)
n = 70000
str(cardio_train)
d = 11
X1 <- scale(cardio$age)
X2 <- cardio$gender
X3 <- scale(cardio$height)
X4 <- scale(cardio$weight)
X5 <- scale(cardio$ap_hi)
X6 <- scale(cardio$ap_lo)
X7 <- cardio$cholesterol
X8 <- cardio$gluc
X9 <- cardio$smoke
X10 <- cardio$alco
X11 <- cardio$active
head(X1)
X.data.scaled <- cbind(X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11) #Combine predictor variables by columns
colnames(X.data.scaled) <- c("X1", "X2", "X3", "X4", "X5", "X6", "X7", "X8", "X9", "X10", "X11")
X.data.scaled[1:5,] #Display first 5 observations.
summary(X3) #Summary of height variable
summary(X4) #summary of weight variable
summary(X7) #summary of cholesterol level
apply(X.data,2,mean) #Mean of predictor variables.
y.data <- cardio$cardio
length(y.data)
cardio <- data.frame(X.data.scaled,y.data)
head(cardio)
split.size <- 0.70
n1 = n*split.size ; n1
n2 = n*(1- split.size) ;n2
sample.size <- floor(split.size*nrow(cardio)) #training data size = 49000
set.seed(1234)
train_indices <- sample(seq_len(nrow(cardio)), size = sample.size)
cardio.train <- cardio[train_indices, ]    # 70% of data
cardio.test <- cardio[-train_indices, ]    # 30% of data.
#X matrix of training data
```

```
X.train.data<-
cbind(cardio.train$X1,cardio.train$X2,cardio.train$X3,cardio.train$X4,cardio.train$X5,
cardio.train$X6,cardio.train$X7,cardio.train$X8,cardio.train$X9,cardio.train$X10, cardio.train$X11)
head(X.train.data)
Y.train.data <- cardio.train$y.data
length(Y.train.data)
model.01 <- glm(Y.train.data~0+X.train.data, family=binomial(link=probit), data = cardio.train)
summary(model.01) #summary of the model
model.01$coefficients
z<-rep(NA,n1)
N.sim<-270000
N1<-sum(Y.train.data);N1
N0<-n1-N1;N0
beta.train<-matrix(NA,nrow=N.sim+1,ncol=d) #beta matrix
beta.train[1,]<-c(model.01$coefficients)
beta.train
tau.2<-matrix(NA,nrow=N.sim+1,ncol=d)
tau.2[1,]<-c(rep(1,d))
tau.2
lamda<-rep(NA,n1)
lamda[1]<-2
lamda
## sampling
for (j in 1:N.sim){
  mu.z<-X.train.data%*%beta.train[j,]
  z[Y.train.data==0]<-rtruncnorm(N0,mean=mu.z[Y.train.data==0],sd=1,a=-Inf,b=0)
  z[Y.train.data==1]<-rtruncnorm(N1,mean=mu.z[Y.train.data==1],sd=1,a=0,b=Inf)
  tau<-diag(tau.2[j,])
  E<-solve(solve(tau)+t(X.train.data)%*%X.train.data)
  beta.train[j+1,]<-rmvnorm(1,E%*%t(X.train.data)%*%z,E)
  tau.2[j+1,]<-1/rinv.gaussian(d,(lamda[j]^2/beta.train[j+1,]^2)^0.5,lamda[j]^2)
  lamda[j+1]<-(rgamma(1,d+1,0.5*sum(tau.2[j+1,])))^0.5
}
summary(lamda)
plot(lamda)
z
beta.train[,1]
tau.2[,1]
ind<-seq(from=20000,to=270000,by=125)
summary(lamda[ind])##estimate lambda
sd(lamda[ind])
ts.plot(lamda[ind]);acf(lamda[ind])
beta.means <-colMeans(beta.train[ind,])##estimate beta.train
tau.means <- colMeans(tau.2[ind,])##estimate tau
newb<-beta.train[ind,]
sd.beta.train<-rep(NA,11)
for (k in 1:11){
  sd.beta.train[k]<-sd(newb[,k])
}
sd.beta.train##sd of beta.train estimates
cardio.test
X1.test <- cardio.test$X1
X2.test <- cardio.test$X2
X3.test <- cardio.test$X3
X4.test <- cardio.test$X4
X5.test <- cardio.test$X5
X6.test <- cardio.test$X6
X7.test <- cardio.test$X7
X8.test <- cardio.test$X8
X9.test <- cardio.test$X9
X10.test <- cardio.test$X10
X11.test <- cardio.test$X11
y.test.data <- cardio.test$y.data
#X matrix of testing data
X.test.data<-
cbind(X1.test,X2.test,X3.test,X4.test,X5.test,X6.test,X7.test,X8.test,X9.test,X10.test,X11.test)
head(X.test.data)
#y data of testing data
y.test.data <- cardio.test$y.data
beta.means <- colMeans(beta.train[ind,])
xb.train <- X.test.data%*%beta.means
y.prob <- pnorm(xb.train)
y.pred <- ifelse(y.prob > 0.5, 1,0)
head(cardio.test$y.data)
head(y.pred)
accuracy_01 <- (sum(y.pred==y.test.data)) / length(y.test.data) ;accuracy_01
mean(y.pred == y.test.data)
```