# STAT7016 Final Project

## Age Prediction for Abalone by Bayesian Analysis

Zihao Li (u7031432)

## *Introduction*

### 1.1 *Preface*

*In the recent few years, abalones are a highly valued and popular seafood in Asia. As the growth of demand increases rapidly, lots of gastronomes and seafood lovers are looking for those abalones with higher quality. One of the judgment criteria is to see the age of the abalone. The old-fashioned way of estimating the age of abalone is to count the rings on the shell and it's obviously time-consuming and complicated (must remove the shell first). So, as an enthusiastic seafood lover (especially abalone), I wish to use the knowledge of Bayesian Inference I learned to contribute on assessing the age of the abalone purely on its physical measurements.*

### 1.2 *Objective*

*The objective of this project is to* build an optimal model for *predicting the age of abalone from their physical characteristics using the 1994 original data from "The population Biology of Abalone (Haliotis species) in Tasmania. I. Blacklip Abalone (H. rubra) from the North Coast and Islands of Bass Strait." This dataset is sourced from UC Irvine Machine Learning Repository* https://archive.ics.uci.edu/ml/index.php*. The main goal of this project is to perform a model selection using Bayesian* approach and assess the relative importance of the variables which will be stated below. Moreover, *I'm interested to assess a model with all main effects* and some *two-way interactions* included*.*

## 1.3 *Data Sets*

*There are totally 4177 observations, 8 descriptive features and 1 response feature in the abalone dataset.*

### 1.3.1 *Response feature*

*'Rings' is an integer type variable range from 1 to 29. It indicates the rings on the shell of the abalone which is used to describe the age of abalone. Adding 1.5 gives the age in years.*

### 1.3.2 *Descriptive features*

*Name / Data Type / Measurement Unit / Description*

-----------------------------

*Sex / nominal / -- / M, F, and I (infant)*

*Length / continuous / mm / Longest shell measurement*

*Diameter / continuous / mm / perpendicular to length*

*Height / continuous / mm / with meat in shell*

*Whole weight / continuous / grams / whole abalone*

*Shucked weight / continuous / grams / weight of meat*

*Viscera weight / continuous / grams / gut weight (after bleeding)*

*Shell weight / continuous / grams / after being dried*

*Rings / integer / -- / +1.5 gives the age in years*

## *Methodology*

## 2.0 Packages or Libraries used

- "e1071"
- "coda"
- "regression_gprior" (existed in Lectures)

## 2.1 Variable Analysis

In the abalone dataset, we are going to use 'Rings' as the response variable to predict the age of abalone and rest of the variables as predictors. Since 'Sex' is a categorical variable with 3 levels (Female, Infant, Male), I need to transfer it to a quantitative variable with values 1,2,3 respectively. As a consideration of sex is not belong to the physical features, we will not add interaction terms involved sex into the implementation. For the response variable, 'Rings' is a discrete integer from 1 to 29, and the distribution of the values is not balance. Its values and the predictors' values are also not in the same scale. So, I decide to apply min-max normalization to the dataset. In addition, all the interaction terms (total 21 terms) will also be taken into consideration.

## 2.2 Prior Distribution and Sampling Model

In this case, I'm going to use Multivariate normal distribution as the sampling model because the normal density is often a useful approximation to the "true" population distribution because of a central limit effect. And the assumption of prior will be a weakly informative g-prior on $\beta$ with k = g$\sigma^2$ and g = n since I need parameter estimation to be invariant to changes in the scale of the covariates. Typical settings are reference in lectures. And the mathematical expression will show below. ($\beta$: regression coefficients, $\sigma^2$: residual variance, I: identity matrix, $\nu_0$: hyperparamter of $\sigma^2$)

$$y|X, \beta, \sigma^2 \sim MVN(X\beta, \sigma^2 I)$$

$$\sigma^2 \sim InvGamma(\nu_0/2 = 1/2, \nu_0\sigma_0^2/2 = \hat{\sigma}_{ols}^2/2) \quad (unit\ prior\ on\ \sigma^2)$$

The X in this case is a 1670 x 30 matrix with a first column of 1's (intercept term), columns two to nine are for the main effects for each of the 8 predictors, and the remaining columns represent the values of the 21 two-way interaction terms. The source of 1670 will be talked in the later context.

Let $z_j$ = 1 if variable j is selected to be included in the model (j = 1,...,30). Let $X_z$ denotes the columns of X for which the corresponding model selection indicator variable $z_j$ = 1. The steps of the sampling algorithm at iteration t are are:

Step 1: Update model selection indicator variable z = c($z_1$,$z_2$,...,$z_{30}$)

Step 1(a): Set z = $z^{t-1}$ Step 1(b): For j takes value of {1,...,30} in random order, replace $z_j$ with a sample from $p(z_j|z_{-j}, y, X)$ Step1(c): Set $z^{(t)}$ = z

Step 2: Update $\sigma^2$

Sample

$$\sigma_{(t)}^2 \sim InvGamma(\frac{(\nu_0 + n)}{2}, \frac{\nu_0\sigma_0^2 + SSR_g}{2})$$

Since our assumption is weakly informative priors, I set $\nu_0$ = 1 and $\sigma_0^2$ = $\sigma_{ols}^2$ and g = n, which $\sigma_{ols}^2$ = 0.005556809 and $SSR_g$ =

$$y^T(I - \frac{g}{g+1} X_z(X_z^T X_z)^{-1} X_z)y$$

Step 3: Update $\beta$

Sample $\beta^{(t)}$ ~ MVN($\beta_m$, $V_\beta$), where

$$\beta_m = V_\beta X_z^T y$$

and

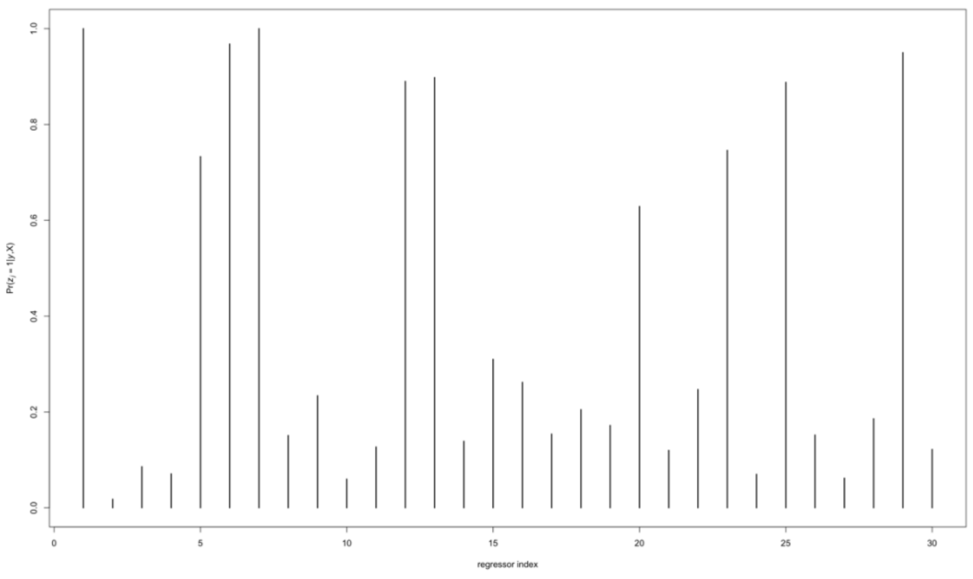
$$V_\beta = \frac{g}{g+1}\sigma_{(t)}^2 (X_z^T X_z)^{-1}$$

Then, let's implement our MCMC algorithm into the R code and see how it works.

```
BETA<-Z<-matrix(NA,S,p)
S2<-NULL
z<-rep(1,dim(X)[2] )
lpy.c<-lpy.X(Y,X[,z==1,drop=FALSE])

for(s in 1:S)
{
  for(j in sample(2:p))
  {
    zp<-z ; zp[j]<-1-zp[j]
    lpy.p<-lpy.X(Y,X[,zp==1,drop=FALSE])
    r<- (lpy.p - lpy.c)*(-1)^(zp[j]==0)
    z[j]<-rbinom(1,1,1/(1+exp(-r)))
    if(z[j]==zp[j]) {lpy.c<-lpy.p}
  }

  beta<-z
  if(sum(z)>0){
    temp<-lm.gprior(Y,X[,z==1,drop=FALSE],S=1)
    beta[z==1]<-temp$beta
    s2<-temp$s2}

  Z[s,]<-z
  BETA[s,]<-beta
  S2<-c(S2,s2)
  print(s)
}
```

Since I have up to 30 predictors features, I'm passively use cross-validation in this case, one reason is that could save up some execution times (I initially set the number of iterations to be executed are 500 times, it took about 2hr 32mins and causes my laptop over-heated), the other reason is for the calculation of prediction error as the assessing criteria for our model selection. Based on the proportion of 30% to 70%, there are 1670 rows of data in the train dataset and 2507 rows in the test dataset. In this case, it took around 46 minutes for the execution of our MCMC algorithm.

First thing first, let's look at the plot of posterior probabilities that each coefficient is non-zero.

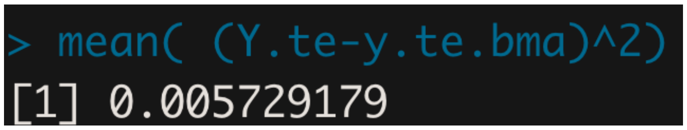


(Figure 1. Plot of Pr (z_j = 1|y, X))

The Figure 1 indicates the estimated posterior probabilities Pr ($z_j$ = 1|y, X). We see that model inclusion indicators for many coefficients are seemed important. If we take a high threshold of 0.8, the plot tells us that 'Whole Weight', 'Shucked Weight', 'Length:Whole Weight', 'Length:Shucked Weight', 'Whole Weight:Shucked Weight' and 'Shucked Weight:Shell Weight' are strongly predictive of 'Rings'. However, this might be affected by the high correlation between weight predictors. Also in the exploration of dataset not shown here, it reports the high correlation between ‘length’ and ‘diameter’, ‘length’ and ‘whole weight’ as well.

Next, I generate the posterior predictive values based our posterior regression coefficients $\beta_j$ and posterior residual variance $\sigma^2$ where j = 1,...,30. And we calculate the value of prediction error as our determining criteria to assess the data fitting.

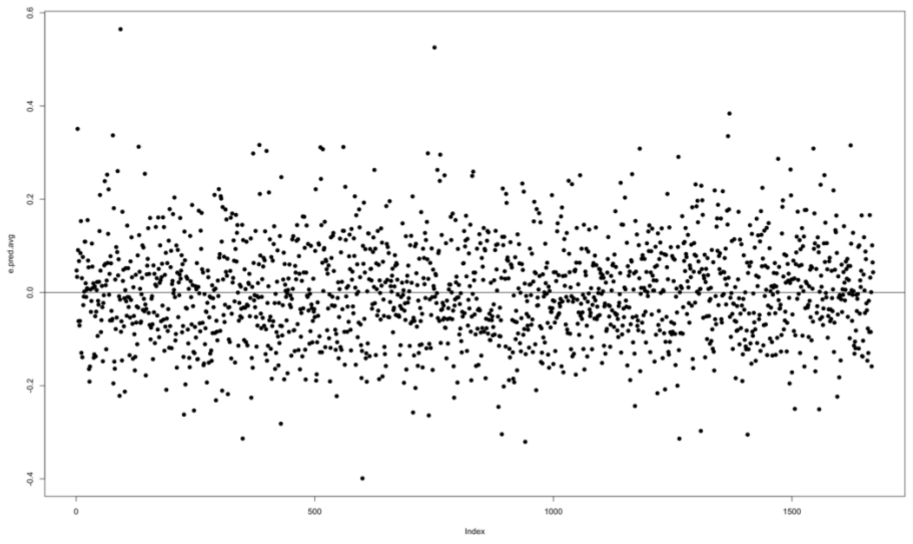
```
#Generate posterior predictive values and store them
y.pred<-NULL
for (s in 1:S){
  beta<-BETA[s,]
  s2<-S2[s]
  y.pred<-rbind(y.pred,c(X%*%beta)+rnorm(n,0,sqrt(s2)))
}
```



```
> mean( (Y.te-y.te.bma)^2)
[1] 0.005729179
```

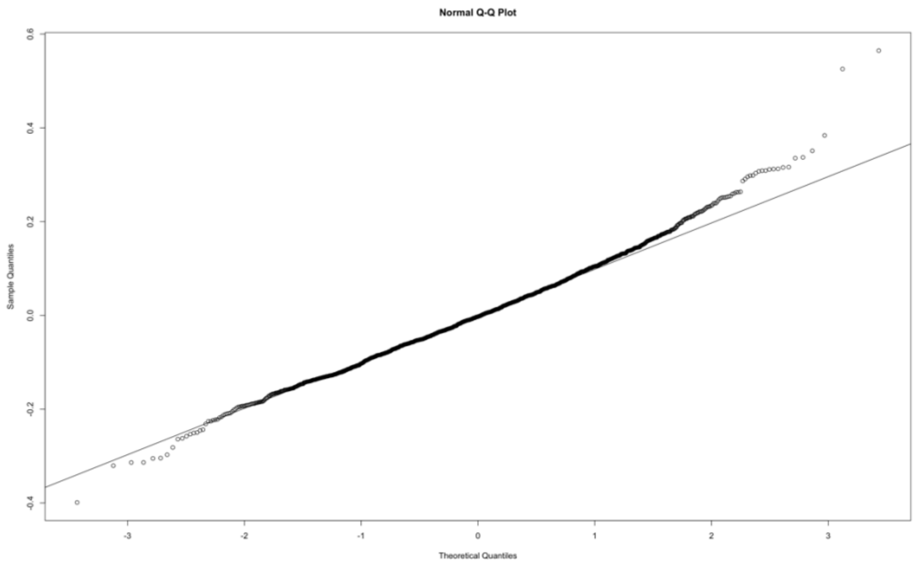
I obtain 0.005729179 as the value of prediction error, and it’s a bit higher than the prediction error generated by the classical linear regression model, which has a prediction error value of 0.005372855. That tells there still have space for improvement.

Now, Let’s check the residual plot and see if there is normality or homoscedasticity violated.



(Figure 2. Residual plot diagnostics)

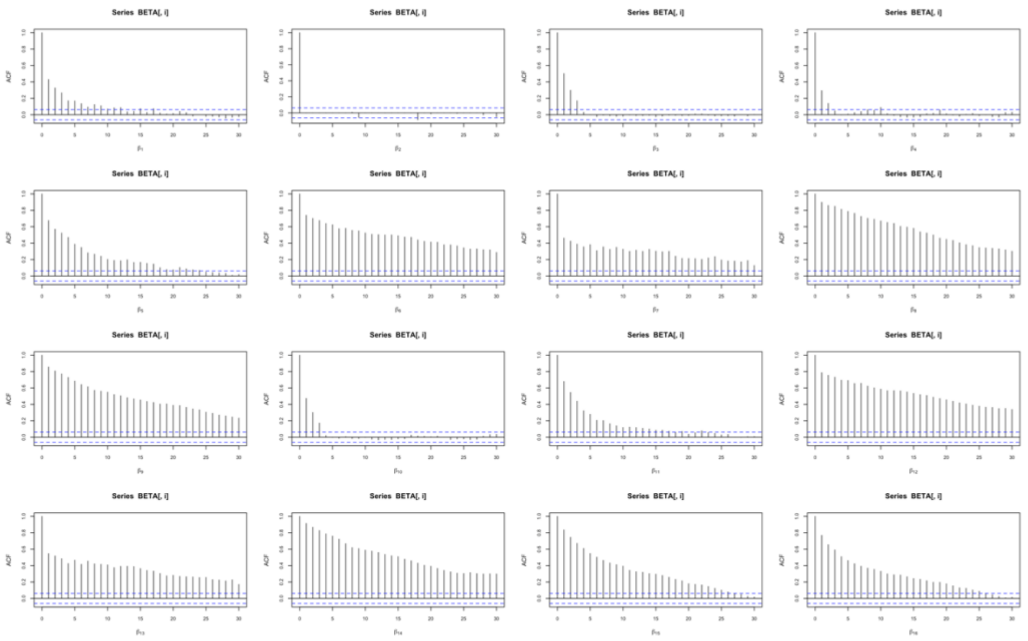
The Figure 2 shows a scatter plot of the predictive residuals which are based on the average posterior predictive residual values. And we can see there is no obvious pattern or trend in the plot. Most of the points are randomly spread around the horizontal zero line, and that tells a good fit for the linear model.

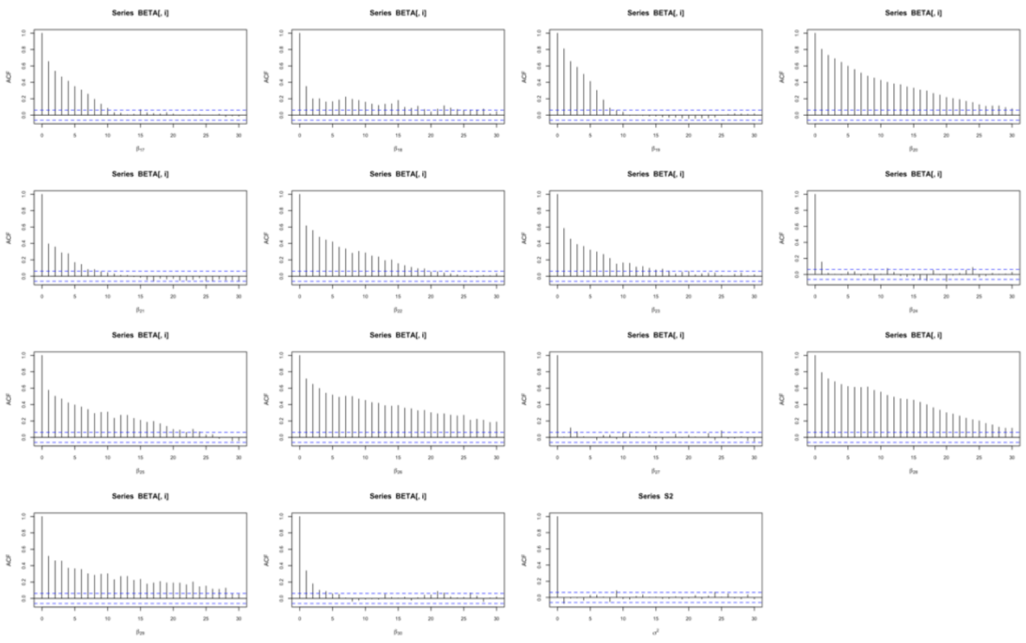


(Figure 3. QQ plot)

And the Figure 3 is a QQ plot to assess if there is any departure from the normality. We can see the overall performance of the fitting is reasonable, but there is a slight departure from the regression line. Those observations might be the suspect of the potential outliers, we might need to take a further investigation. When I look at the summary statistics of the abalone dataset, one doubt is the minimum of the height is zero that is unrealistic. And I have abstracted those two records and found their other values of weight parameters are also relatively smaller than the others. In addition, when I take a comparison between the value of ‘Whole weight’ and the sum of the values of ‘Shucked weight’, ‘Viscera weight’ and ‘Shell weight’, some of them are even negative. One wild guess for that is may due to the loss of unknown water or blood during the shucking process. All of the questionable points mentioned above could be the cause of the outliers or extreme values in the observations.

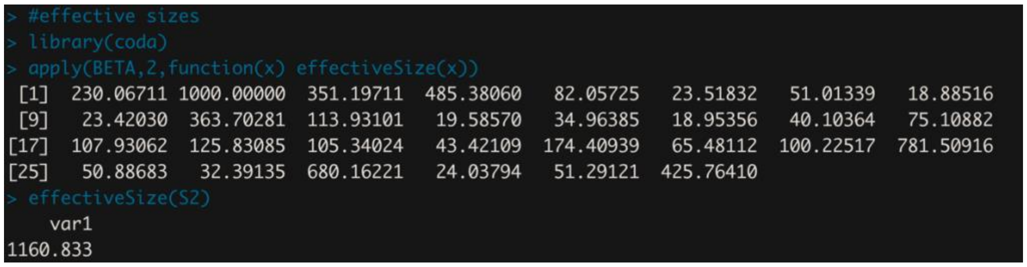
Back to the Bayesian analysis, in order to see the convergence and stationarity of the sampling algorithm implemented above. I have also generated some diagnostic plots below.





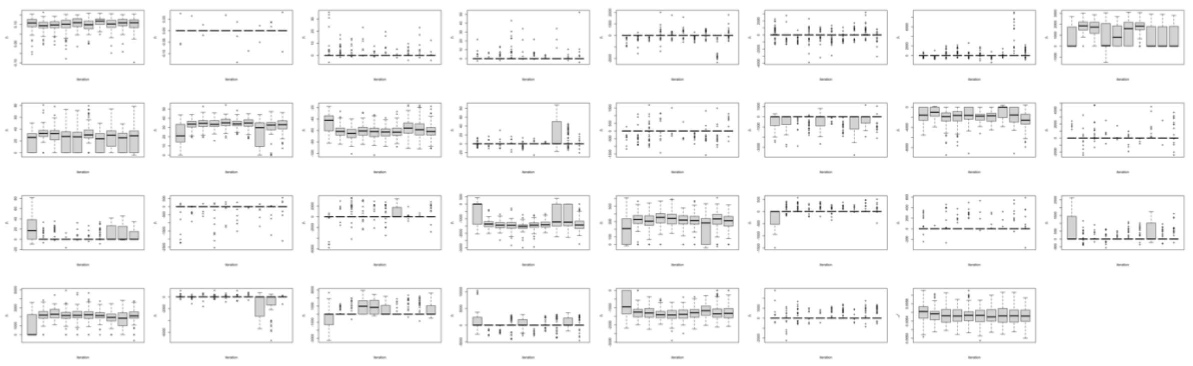
(Figure 4 includes above two, with acf plots of $\beta_1, \ldots, \beta_{30}$ and $\sigma^2$.

The Figure 4 above shows the acf plots which indicate many of them have the issue of high autocorrelation. The performance of $\sigma^2$ seems fine as the plot decays to 0 quickly. This implies that the thinning process should be applied later.



```
> #effective sizes
> library(coda)
> apply(BETA,2,function(x) effectiveSize(x))
 [1]  230.06711 1000.00000  351.19711  485.38060   82.05725   23.51832   51.01339   18.88516
 [9]   23.42030  363.70281  113.93101   19.58570   34.96385   18.95356   40.10364   75.10882
[17]  107.93062  125.83085  105.34024   43.42109  174.40939   65.48112  100.22517  781.50916
[25]   50.88683   32.39135  680.16221   24.03794   51.29121  425.76410
> effectiveSize(S2)
    var1
1160.833
```

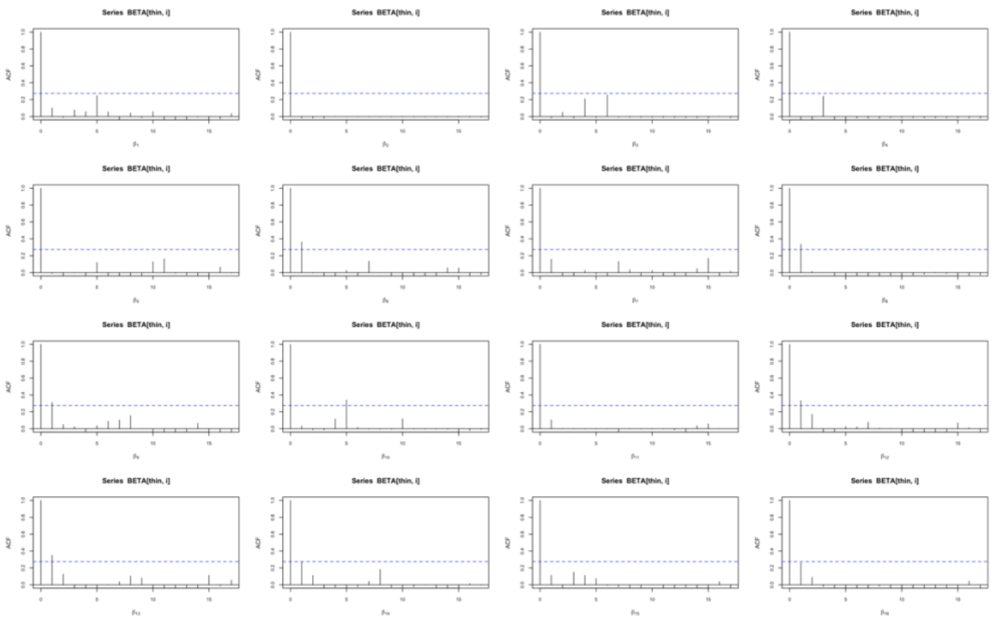
The output of effective size above includes many small sizes which are less than 1000. This could be explained by the high correlation between some weight predictors and physical characteristics (e.g. length, height, diameter). We could possibly solve this either by dropping the highly correlated variables or increasing the number of iterations of running the chain.

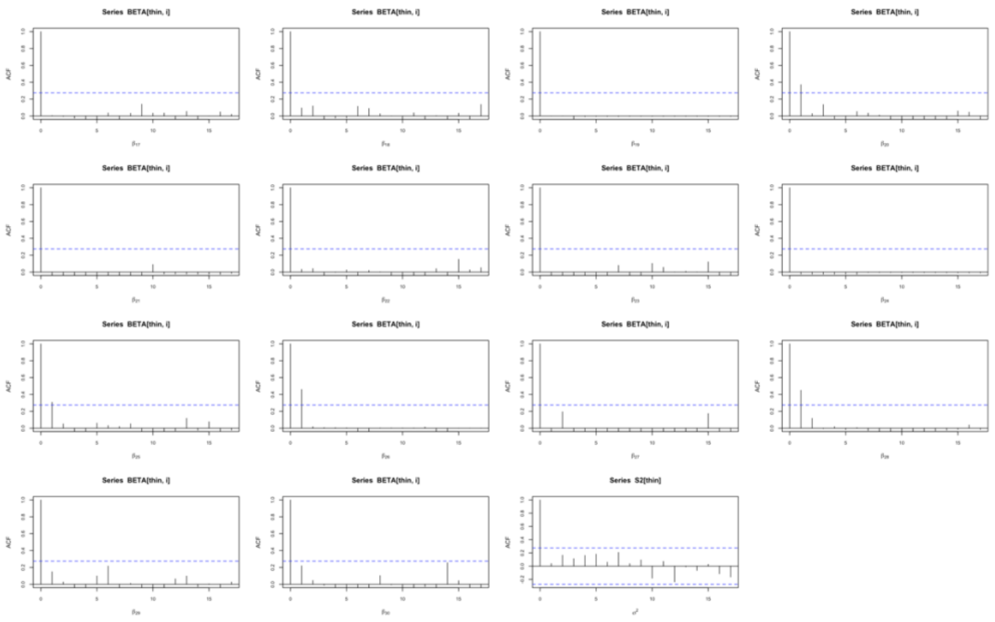


(Figure 5. Stationarity plots of $\beta_1$, …, $\beta_{30}$ and $\sigma^2$)

On the side, the Figure 5 here is the stationarity plots which show some sequences of draws of the $\beta_j$ have not reached convergence. For example, 20th and 21st in the plots. These are not big concerns as they are directly sample from p($\beta$, $\sigma^2$|y, X).

In the previous acf plots, we can see the lags do have significant effect as many of them have broken the bounds. And I use the sequence of draws taken every 20th iteration and see the difference of acf plots after thinned:



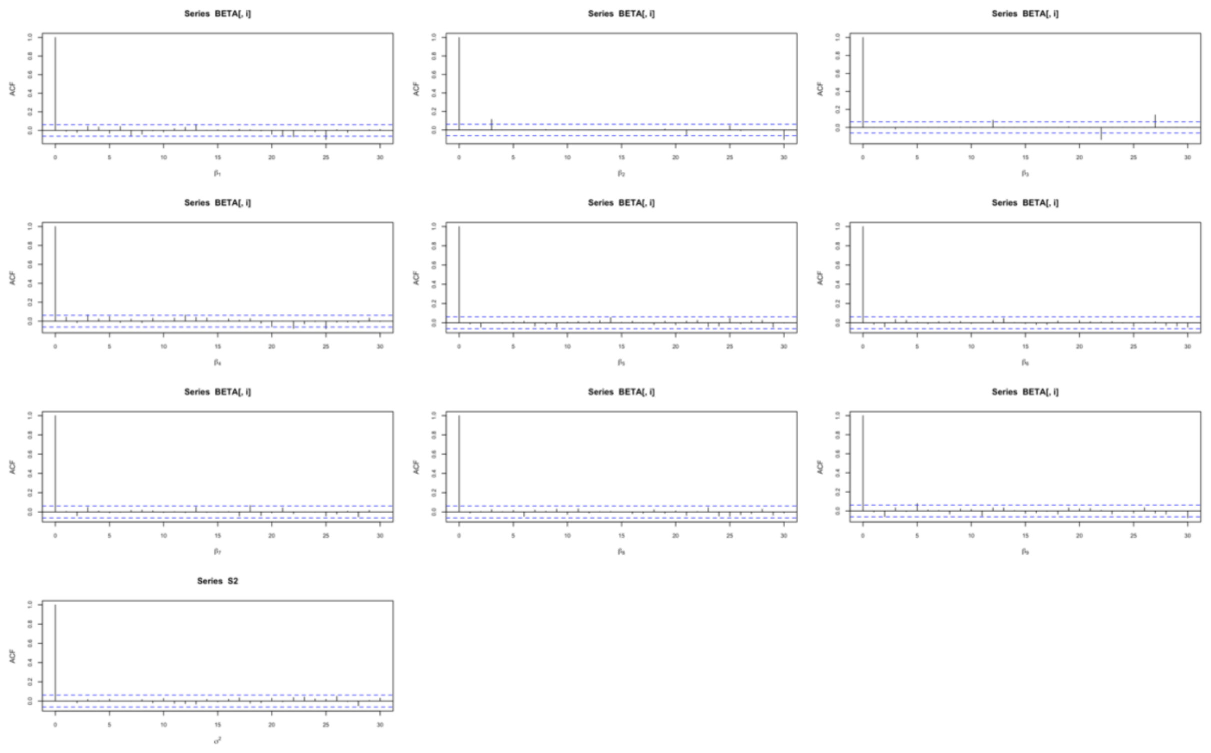


(Figure 6. Acf plots after ‘thinned’)

We can see from the Figure 6 that most of them have no problems of high autocorrelation as they rapidly decay to 0.

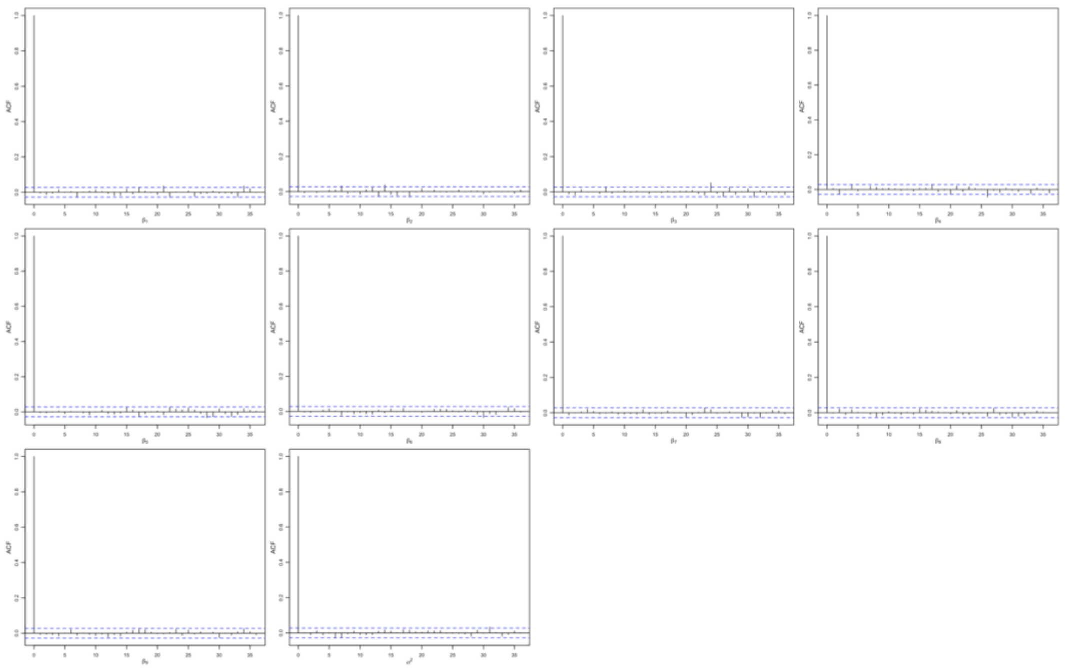
Based on the previous result, I decide to make some modifications to the choice of covariates. The change is to remove the interaction terms since the suggested predictors are having high correlation with other variables. That might mislead the output and possibly cause the problem of overfitting.

The same algorithm and routine as above have been applied on our covariates matrix. However, there is some problem occur in our acf plot.



(Figure 7. acf plot with only main effects considered)

The Figure 7 is the acf plot with only main effects considered with 1000 iterations executed. We can see $\beta_2$,$\beta_3$, and $\beta_9$ still have some troughs and peaks exceed the bounds. We might consider increasing the number of iterations and take 1000 as burn-in period. Therefore, I set the value of S (number of iterations) to 5000, and we obtain the value of prediction error as 0.005276393 which is less than that of the initial model includes the interaction terms.

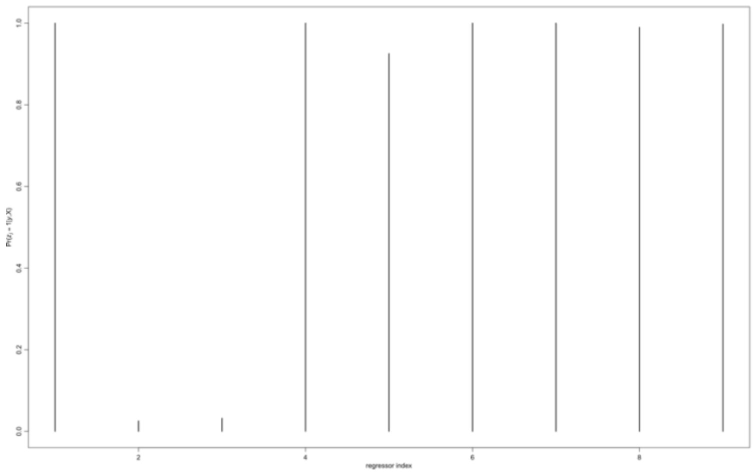


(Figure 8. acf plots under 5000 iterations)

Again, the Figure 8 above shows that there are no issues of the high autocorrelation except for the plot of intercept.

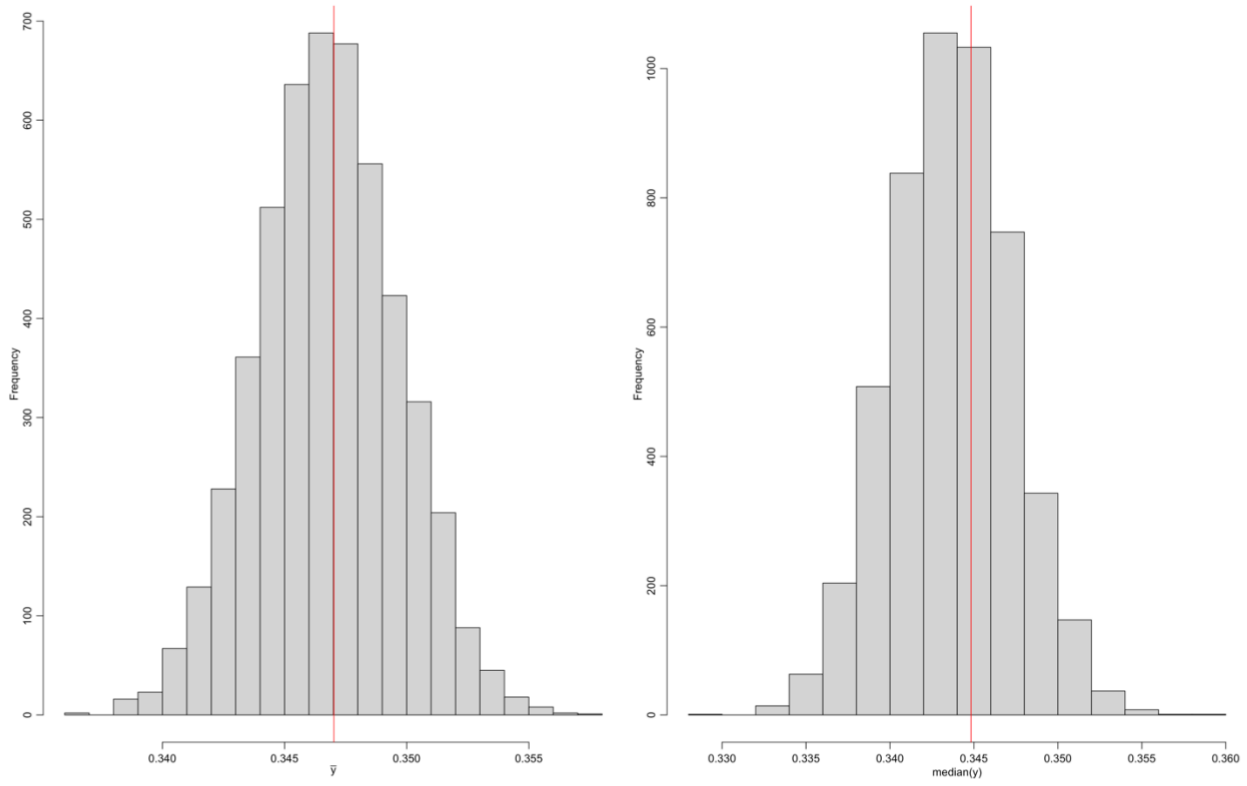
## *Results*

Overall, we can see that Figure 9 below suggests us ‘diameter’, ‘height’, ‘whole weight’, ‘shucked weight’, ‘viscera weight’ and ‘shell weight’ are strongly predictive of rings. As the model inclusion indicator for all of them have posterior probability which really close to 1.



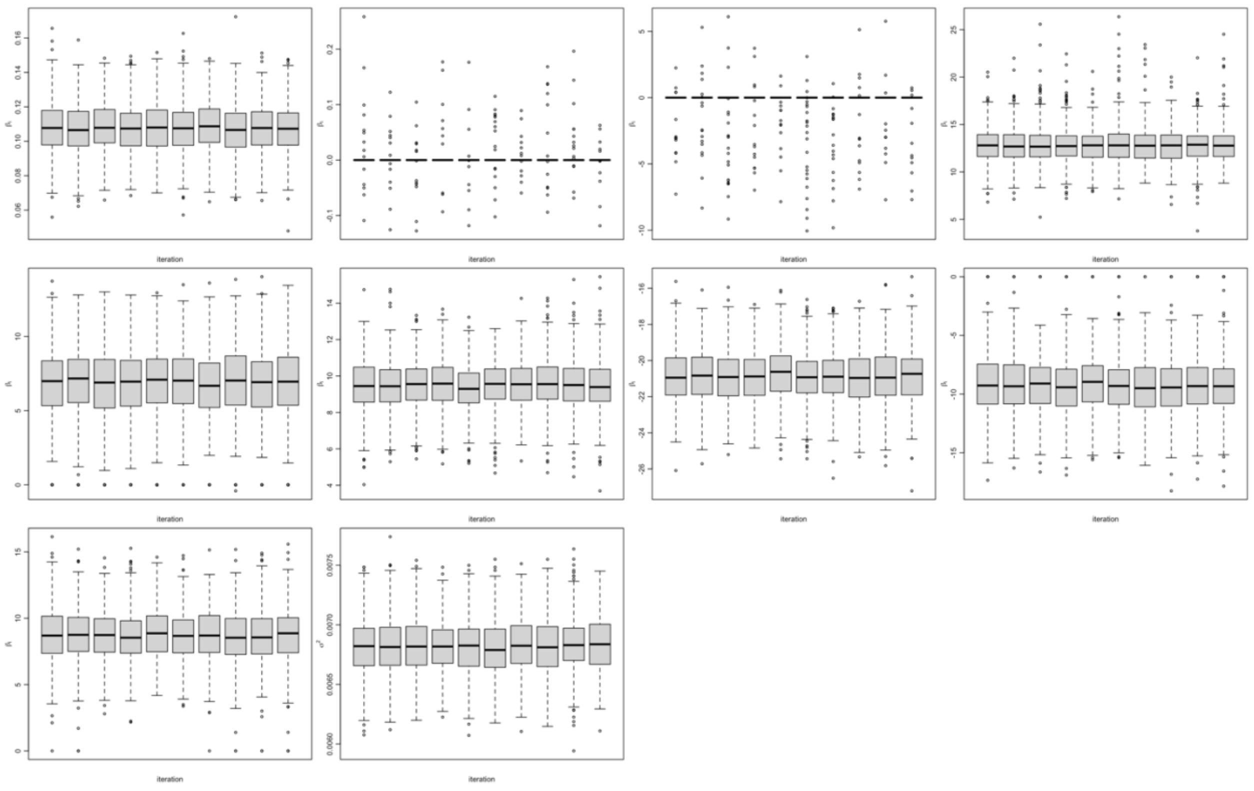
(Figure 9. Plot of Pr (z_j = 1|y, X) with main effects only)

Lastly, I did a posterior predictive checking to check our sampling model assumption based on any discrepancy of mean and median. The Figure 10 below shows the distribution of posterior mean with replicated data and red vertical line is the mean of observed data.



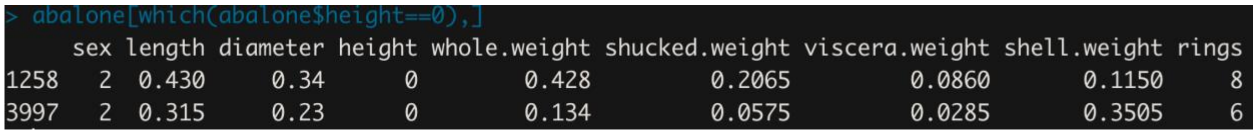
(Figure 10. Plot of posterior predictive checking)

We can see from the Figure 10 that there are not many discrepancies between empirical distribution and posterior predictive distribution. The stationarity diagnostic plots are also provided below. The convergence seems has been achieved here.


(Figure 11. Stationarity plot of $\beta_1, \ldots, \beta_9$ and $\sigma^2$)

Based on the output above we are achieving the goal mentioned in Section 1.2 (Objective) even closer. The optimal model is suggested including 'diameter', 'height', and weight related variables. The prediction error of this model is 0.005276 which is acceptable since it's still lower than that of previous model with interaction terms included (0.005729). For my personal thoughts, I don't think all the 4 weight variables (whole weight, shucked weight, viscera weight and shell weight) are all share importance of predicting the rings of abalone. The reason is that the data quality is full of doubt,

- Two observations with the height of abalone are equal to 0, but weight of that abalone is measured. These are quite suspicious for incorrectly measurement.



```
> abalone[which(abalone$height==0),]
     sex length diameter height whole.weight shucked.weight viscera.weight shell.weight rings
1258   2  0.430     0.34      0        0.428         0.2065         0.0860       0.1150     8
3997   2  0.315     0.23      0        0.134         0.0575         0.0285       0.3505     6
```

- A group of observations does not follow the common logic properly. When I saw the descriptions of variables, I think the whole weight should be a linear combination of shucked weight, viscera weight and shell weight. Whereas not all of them can have a difference of 0 between the whole weight and sum of those three weights left.

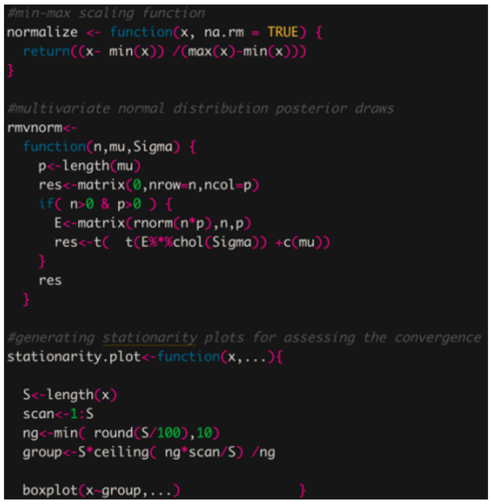
I'm not trying to blame the data collector here; they are just some flaws that I spot during the exploration of the abalone dataset. They might come from those unique individuals in the nature or some human mistakes.

## *Conclusion*

In this Bayesian Inference project, I have learned many useful Bayesian methods and the logic of Bayesian techniques which is quite different from the classical statistics. The Bayesian Statistics taught me the flow of solving a problem is mostly based on the bayes theorem. We need to use prior and sampling model to derive the posterior distribution. Some useful methods like Gibbs sampling, MCMC sampling algorithm, Metropolis-Hasting algorithm (not used in this project) and model selection procedure based on the bayes factor are all practical in my opinion. A take-way point from this project is to consider the model selection procedure is not always relying on the g-prior. Semi-conjugate prior, unit prior and g-prior with different values setting for g can lead to the different output. In my case, the g-prior with g setting to the sample size is a classical choice mentioned in the lectures. So, there is something to refer to, which is an advantage of my analysis. The drawback might be the treatment to the high collinearity is still not finalized yet. Most of the variables have high correlation with other variables except for the 'rings', 'sex', and 'height'. If I only include those two covariates and 'rings' as response variable, that will eliminate lots of potential information as I dropped rest of them. One thing I will extend my project is probably tackling more difficult questions and making use of more Bayesian techniques. However, I'm satisfied as I can proudly finish this Bayesian project now. ^_^

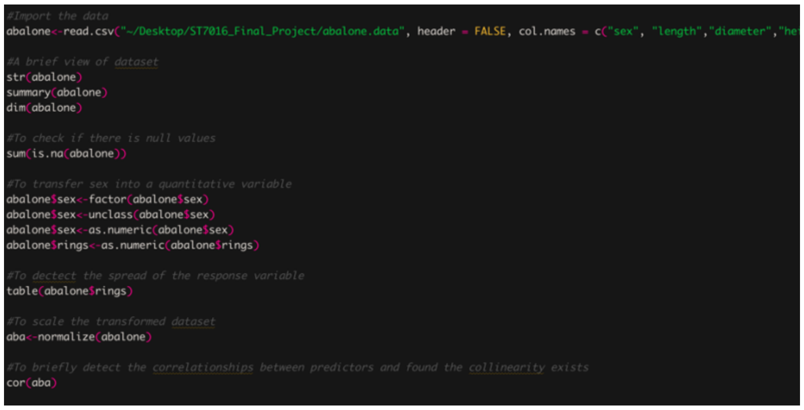
## *Appendix:*

- Some functions I used inside and outside of the class



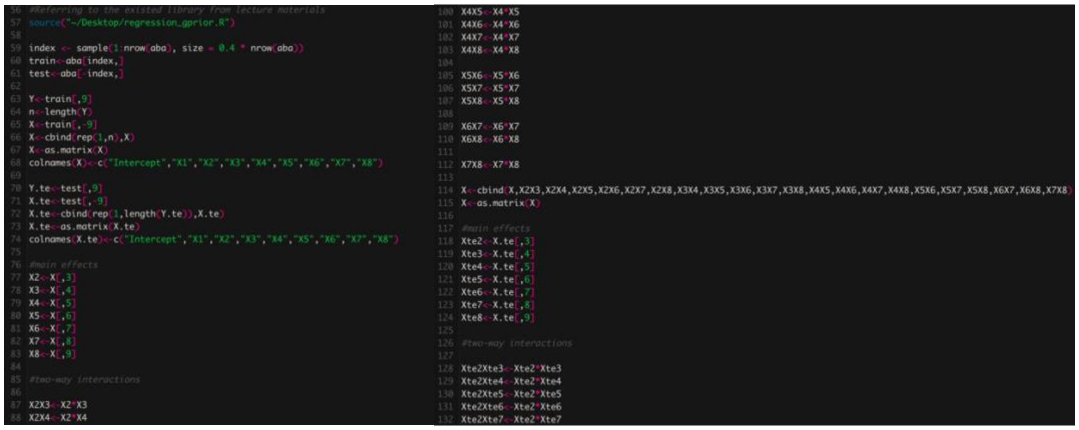
```r
#min-max scaling function
normalize <- function(x, na.rm = TRUE) {
  return((x- min(x)) /(max(x)-min(x)))
}

#multivariate normal distribution posterior draws
rmvnorm<-
  function(n,mu,Sigma) {
    p<-length(mu)
    res<-matrix(0,nrow=n,ncol=p)
    if( n>0 & p>0 ) {
      E<-matrix(rnorm(n*p),n,p)
      res<-t(  t(E%*%chol(Sigma)) +c(mu))
    }
    res
  }

#generating stationarity plots for assessing the convergence
stationarity.plot<-function(x,...){

  S<-length(x)
  scan<-1:S
  ng<-min( round(S/100),10)
  group<-S*ceiling( ng*scan/S) /ng

  boxplot(x~group,...)              }
```

- Data preparation and some exploration of dataset



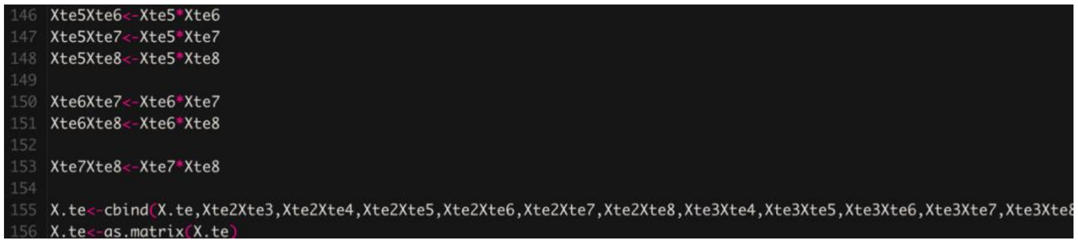
```r
#Import the data
abalone<-read.csv("~/Desktop/ST7016_Final_Project/abalone.data", header = FALSE, col.names = c("sex", "length","diameter","he

#A brief view of dataset
str(abalone)
summary(abalone)
dim(abalone)

#To check if there is null values
sum(is.na(abalone))

#To transfer sex into a quantitative variable
abalone$sex<-factor(abalone$sex)
abalone$sex<-unclass(abalone$sex)
abalone$sex<-as.numeric(abalone$sex)
abalone$rings<-as.numeric(abalone$rings)

#To dectect the spread of the response variable
table(abalone$rings)

#To scale the transformed dataset
aba<-normalize(abalone)

#To briefly detect the correlationships between predictors and found the collinearity exists
cor(aba)
```

- Data processing and ready for the later implementation



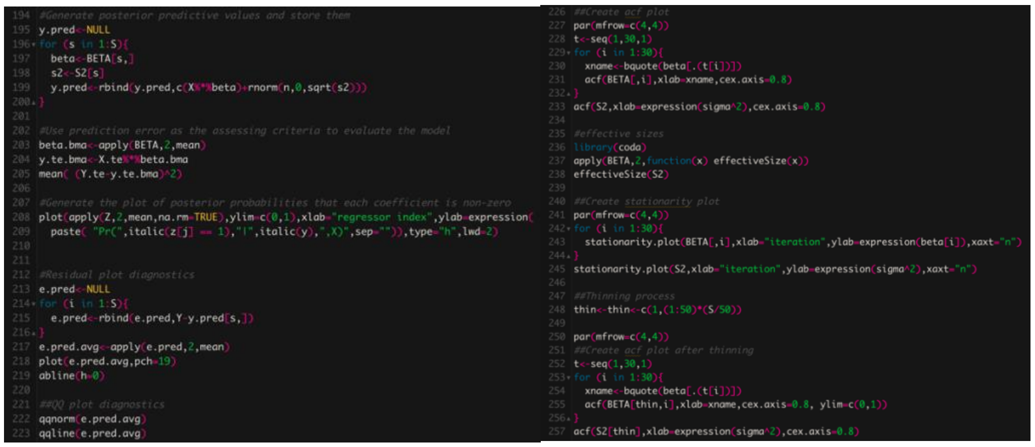
```r
#Referring to the existed library from lecture materials
source("~/Desktop/regression_gprior.R")

index <- sample(1:nrow(aba), size = 0.4 * nrow(aba))
train<-aba[index,]
test<-aba[-index,]

Y<-train[,9]
n<-length(Y)
X<-train[,-9]
X<-cbind(rep(1,n),X)
X<-as.matrix(X)
colnames(X)<-c("Intercept","X1","X2","X3","X4","X5","X6","X7","X8")

Y.te<-test[,9]
X.te<-test[,-9]
X.te<-cbind(rep(1,length(Y.te)),X.te)
X.te<-as.matrix(X.te)
colnames(X.te)<-c("Intercept","X1","X2","X3","X4","X5","X6","X7","X8")

#main effects
X2<-X[,3]
X3<-X[,4]
X4<-X[,5]
X5<-X[,6]
X6<-X[,7]
X7<-X[,8]
X8<-X[,9]

#two-way interactions

X2X3<-X2*X3
X2X4<-X2*X4
```

```r
X4X5<-X4*X5
X4X6<-X4*X6
X4X7<-X4*X7
X4X8<-X4*X8

X5X6<-X5*X6
X5X7<-X5*X7
X5X8<-X5*X8

X6X7<-X6*X7
X6X8<-X6*X8

X7X8<-X7*X8

X<-cbind(X,X2X3,X2X4,X2X5,X2X6,X2X7,X2X8,X3X4,X3X5,X3X6,X3X7,X3X8,X4X5,X4X6,X4X7,X4X8,X5X6,X5X7,X5X8,X6X7,X6X8,X7X8)
X<-as.matrix(X)

#main effects
Xte2<-X.te[,3]
Xte3<-X.te[,4]
Xte4<-X.te[,5]
Xte5<-X.te[,6]
Xte6<-X.te[,7]
Xte7<-X.te[,8]
Xte8<-X.te[,9]

#two-way interactions

Xte2Xte3<-Xte2*Xte3
Xte2Xte4<-Xte2*Xte4
Xte2Xte5<-Xte2*Xte5
Xte2Xte6<-Xte2*Xte6
Xte2Xte7<-Xte2*Xte7
```



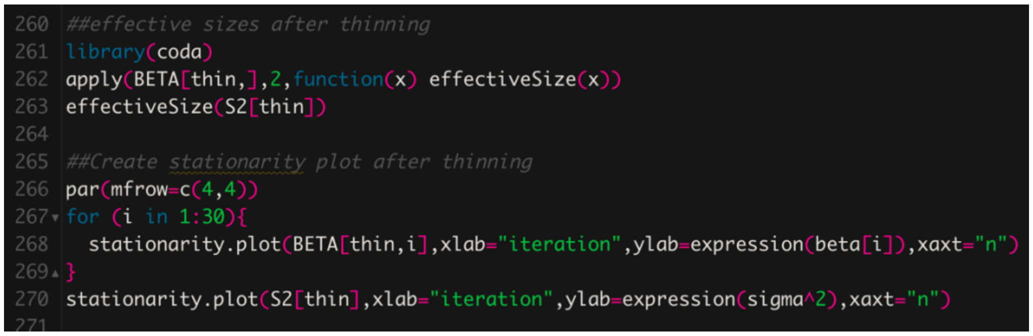
```r
Xte5Xte6<-Xte5*Xte6
Xte5Xte7<-Xte5*Xte7
Xte5Xte8<-Xte5*Xte8

Xte6Xte7<-Xte6*Xte7
Xte6Xte8<-Xte6*Xte8

Xte7Xte8<-Xte7*Xte8

X.te<-cbind(X.te,Xte2Xte3,Xte2Xte4,Xte2Xte5,Xte2Xte6,Xte2Xte7,Xte2Xte8,Xte3Xte4,Xte3Xte5,Xte3Xte6,Xte3Xte7,Xte3Xte8
X.te<-as.matrix(X.te)
```

- Parts used in Methodology

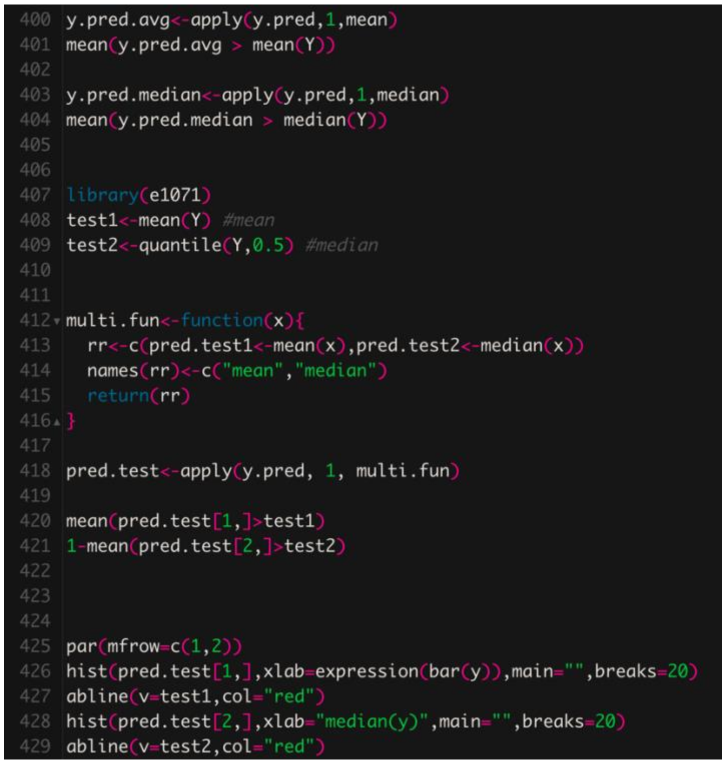


```r
#Generate posterior predictive values and store them
y.pred<-NULL
for (s in 1:S){
  beta<-BETA[s,]
  s2<-S2[s]
  y.pred<-rbind(y.pred,c(X%*%beta)+rnorm(n,0,sqrt(s2)))
}

#Use prediction error as the assessing criteria to evaluate the model
beta.bma<-apply(BETA,2,mean)
y.te.bma<-X.te%*%beta.bma
mean( (Y.te-y.te.bma)^2)

#Generate the plot of posterior probabilities that each coefficient is non-zero
plot(apply(Z,2,mean,na.rm=TRUE),ylim=c(0,1),xlab="regressor index",ylab=expression(
  paste( "Pr(",italic(z[j] == 1),"|",italic(y),",X)",sep="")),type="h",lwd=2)


#Residual plot diagnostics
e.pred<-NULL
for (i in 1:S){
  e.pred<-rbind(e.pred,Y-y.pred[s,])
}
e.pred.avg<-apply(e.pred,2,mean)
plot(e.pred.avg,pch=19)
abline(h=0)

##QQ plot diagnostics
qqnorm(e.pred.avg)
qqline(e.pred.avg)
```

```r
##Create acf plot
par(mfrow=c(4,4))
t<-seq(1,30,1)
for (i in 1:30){
  xname<-bquote(beta[.(t[i])])
  acf(BETA[,i],xlab=xname,cex.axis=0.8)
}
acf(S2,xlab=expression(sigma^2),cex.axis=0.8)

#effective sizes
library(coda)
apply(BETA,2,function(x) effectiveSize(x))
effectiveSize(S2)

##Create stationarity plot
par(mfrow=c(4,4))
for (i in 1:30){
  stationarity.plot(BETA[,i],xlab="iteration",ylab=expression(beta[i]),xaxt="n")
}
stationarity.plot(S2,xlab="iteration",ylab=expression(sigma^2),xaxt="n")

##Thinning process
thin<-thin<-c(1,(1:50)*(S/50))

par(mfrow=c(4,4))
##Create acf plot after thinning
t<-seq(1,30,1)
for (i in 1:30){
  xname<-bquote(beta[.(t[i])])
  acf(BETA[thin,i],xlab=xname,cex.axis=0.8, ylim=c(0,1))
}
acf(S2[thin],xlab=expression(sigma^2),cex.axis=0.8)
```



```r
##effective sizes after thinning
library(coda)
apply(BETA[thin,],2,function(x) effectiveSize(x))
effectiveSize(S2[thin])

##Create stationarity plot after thinning
par(mfrow=c(4,4))
for (i in 1:30){
  stationarity.plot(BETA[thin,i],xlab="iteration",ylab=expression(beta[i]),xaxt="n")
}
stationarity.plot(S2[thin],xlab="iteration",ylab=expression(sigma^2),xaxt="n")
```



```r
y.pred.avg<-apply(y.pred,1,mean)
mean(y.pred.avg > mean(Y))

y.pred.median<-apply(y.pred,1,median)
mean(y.pred.median > median(Y))


library(e1071)
test1<-mean(Y) #mean
test2<-quantile(Y,0.5) #median


multi.fun<-function(x){
  rr<-c(pred.test1<-mean(x),pred.test2<-median(x))
  names(rr)<-c("mean","median")
  return(rr)
}

pred.test<-apply(y.pred, 1, multi.fun)

mean(pred.test[1,]>test1)
1-mean(pred.test[2,]>test2)



par(mfrow=c(1,2))
hist(pred.test[1,],xlab=expression(bar(y)),main="",breaks=20)
abline(v=test1,col="red")
hist(pred.test[2,],xlab="median(y)",main="",breaks=20)
abline(v=test2,col="red")
```