Bayesian Learning Computer Lab 2

bisku859 and gowku593

4/18/2021

Assignment 1.

1.(a) and (b).

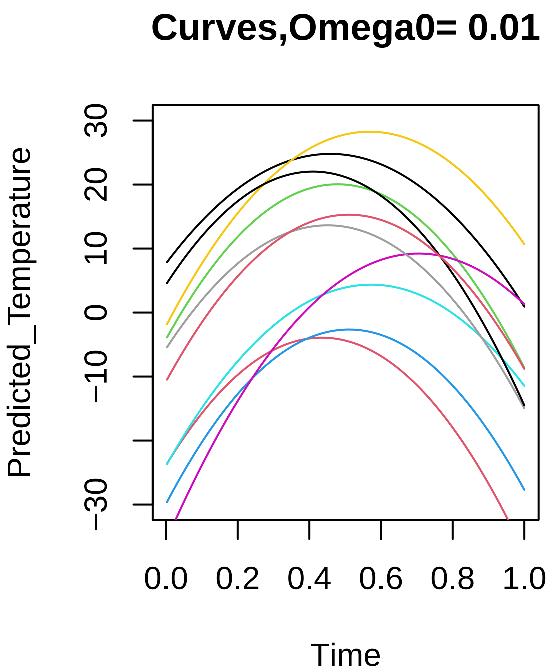
Conjugate prior for θ or β and σ^2
$\beta \mid \sigma^2 \sim N(\mu_0, \sigma^2\Omega_0^{-1})$ .. equation 1
$\sigma^2 \sim Inv - \chi^2(\nu_o, \sigma_o^2)$ ... equation 2

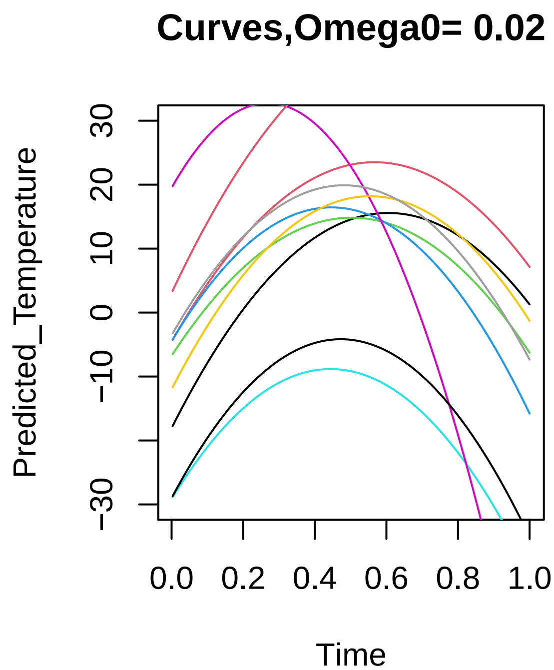
Posterior
$\beta \mid \sigma^2, y \sim N(\mu_n, \sigma^2\Omega_n^{-1})$ ...equation 3
$\sigma^2 \mid y \sim Inv - \chi^2(\nu_n, \sigma_n^2)$ ...equation 4

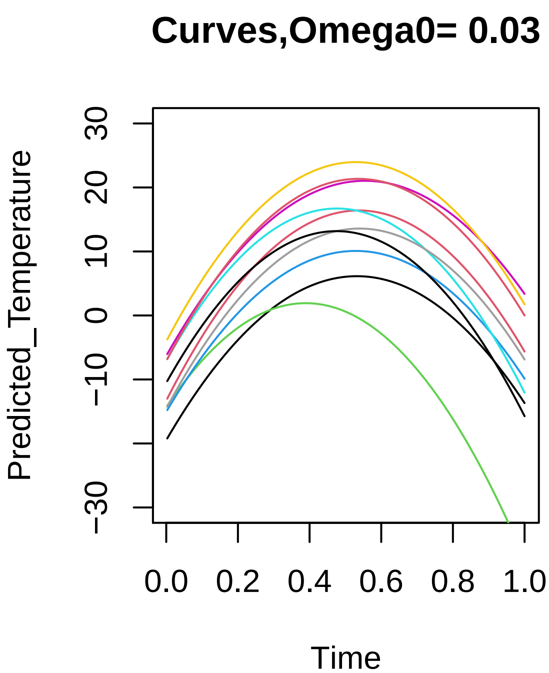
$Omega_o = 0.01 I_3 \ldots$ equation 5
$\mu_n = (X'X + \Omega_0)^{-1}(X'X\hat{\beta} + \Omega_0 + \mu_0)$
$\Omega_n = X'X + \Omega_0$
$\nu_n = \nu_0 + n$
$\nu_n\sigma^2 = \nu_0\sigma_0^2 + (y'y + \mu_0'\Omega_0\mu_0 - \mu_0'\Omega_n\mu_n) \ldots$ equation 6
$Y = X_p|beta + \epsilon$ ...equation 7
$\epsilon = N(0, \sigma^2)$ ...equation 8

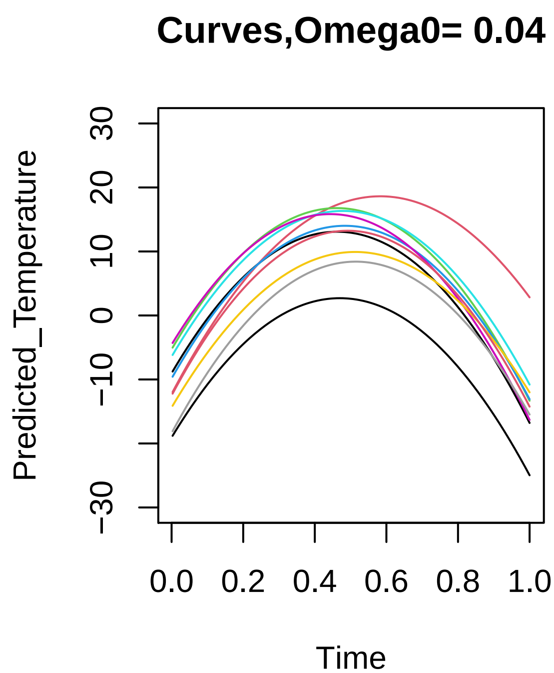
```r
set.seed(123)
#1 a & b
data<-read.table("TempLinkoping.txt",col.name=c("time","temp"),
                 stringsAsFactors = FALSE,header = TRUE)

# calculating time square as it is required to solve expression ahead
data$time_sq<-data[,1]^2

data$constant=1
#rearrange columns
data<-data[,c(4,1,3,2)]

# hyper parameters (given)
mu0<-t(c(-10,100,-100))
omega<-diag(3)
# as per quation 5 and given
omega0<-omega*0.01
nu0<-4
```

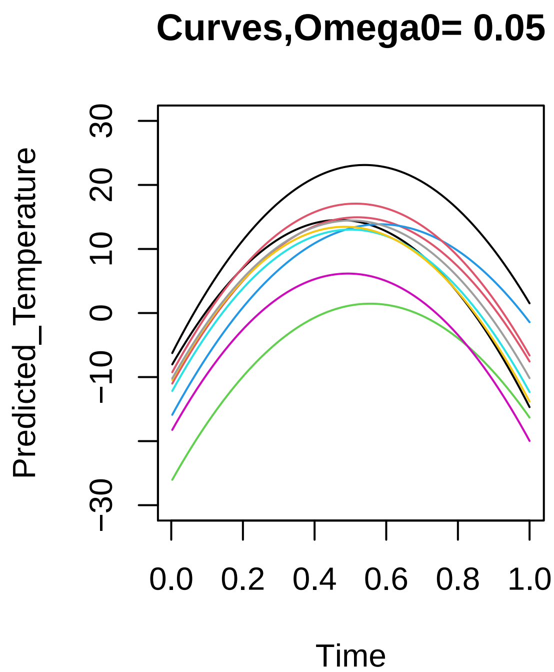
```r
k<-3 # no.of beta values
sigma0_square<-1


sigma_square<-c()
beta_draws<-matrix(nrow = 1000,ncol = 3)
quad<-function(betas,time,err){
  temp_equation<-betas[1]+ betas[2]*time+betas[3]*time^2+ err
  return(temp_equation)
}

pred<-function(Omega_knot,k){
pred_temp<-matrix(nrow = 365,ncol = 1000)
sigma_sq_total<-c()
for (i in 1:1000){
x = rchisq(n = 1, df = nu0)
# As per equation 2 and solving it for sigma square
sigma_square<-(nu0*sigma0_square)/x
sigma_sq_total<-c(sigma_sq_total,sigma_square)

# As per equation 1 , solving Beta for given sigma_square
beta_draws[i,]<-rmvnorm(n=1,mean=mu0,sigma=sigma_square*solve(Omega_knot))
pred_temp[,i]<-sapply(data$time,quad,betas=beta_draws[i,],
                      err=rnorm(n=1,mean=0,sd=sqrt(sigma_square)))

}



plot(data$time, pred_temp[,1],type="l",
     ylim=c(-30,30),main= paste("Curves,Omega0=", k),xlab="Time",
     ylab="Predicted_Temperature")
for(i in 2:10){
  lines(data$time, pred_temp[,i],col=i)
}
}
layout(matrix(c(1:2, 0, 0), nrow=1, ncol=2, byrow=TRUE))
for (k in seq(0.01,0.1,0.01)){
pred(omega*k,k)}
```

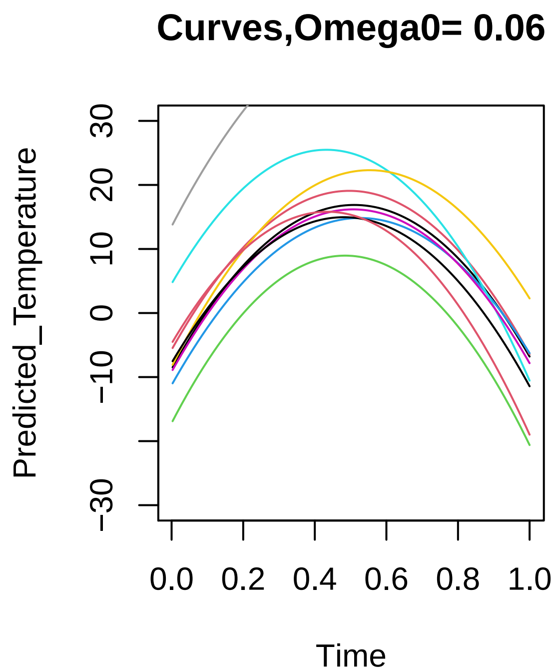

Curves,Omega0= 0.01
Predicted_Temperature
−30
−10 0 10 20 30
Time
0.0 0.2 0.4 0.6 0.8 1.0

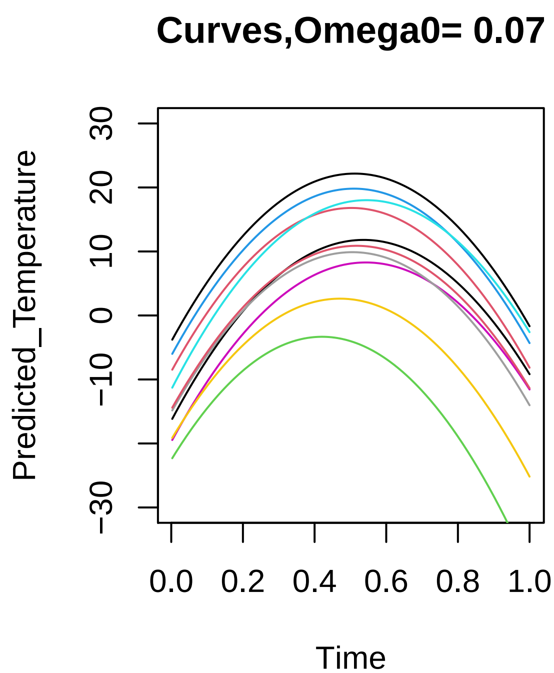

Curves,Omega0= 0.02
Predicted_Temperature
−30
−10 0 10 20 30
Time
0.0 0.2 0.4 0.6 0.8 1.0

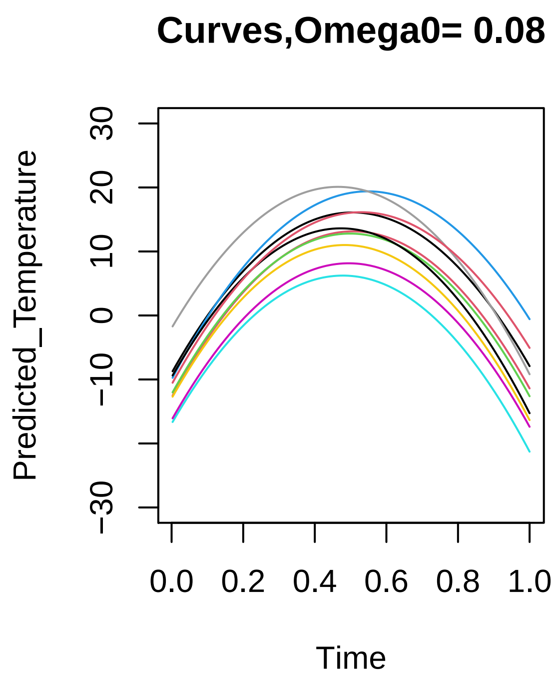

Curves,Omega0= 0.03
Predicted_Temperature
−30
−10 0 10 20 30
Time
0.0 0.2 0.4 0.6 0.8 1.0

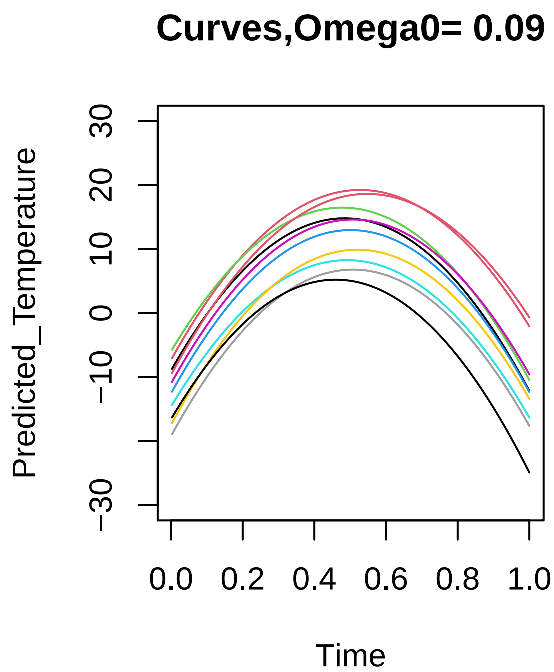

Curves,Omega0= 0.04
Predicted_Temperature
−30
−10 0 10 20 30
Time
0.0 0.2 0.4 0.6 0.8 1.0

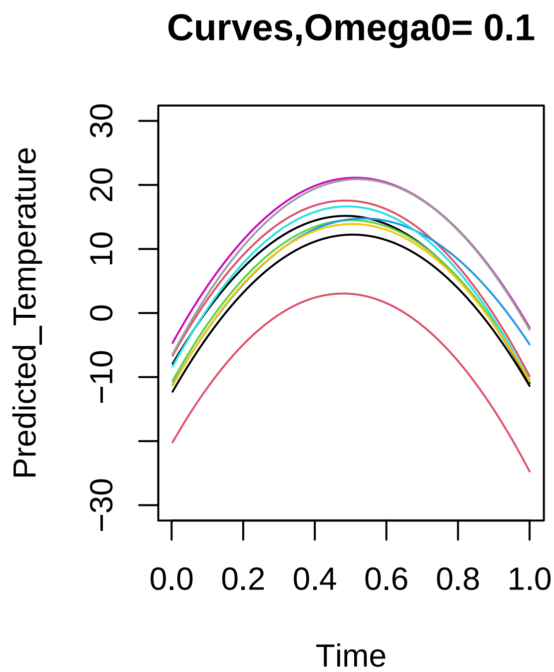

Curves,Omega0= 0.05
Predicted_Temperature
−30
−10 0 10 20 30
0.0 0.2 0.4 0.6 0.8 1.0
Time


Curves,Omega0= 0.06
Predicted_Temperature
−30
−10 0 10 20 30
0.0 0.2 0.4 0.6 0.8 1.0
Time


Curves,Omega0= 0.07
Predicted_Temperature
−30
−10 0 10 20 30
0.0 0.2 0.4 0.6 0.8 1.0
Time


Curves,Omega0= 0.08
Predicted_Temperature
−30
−10 0 10 20 30
0.0 0.2 0.4 0.6 0.8 1.0
Time





We plotted curves with varying hyperparameters and changed Ω_0 from 0.01 to 0.1 to understand its impact on the temperature prediction over time.

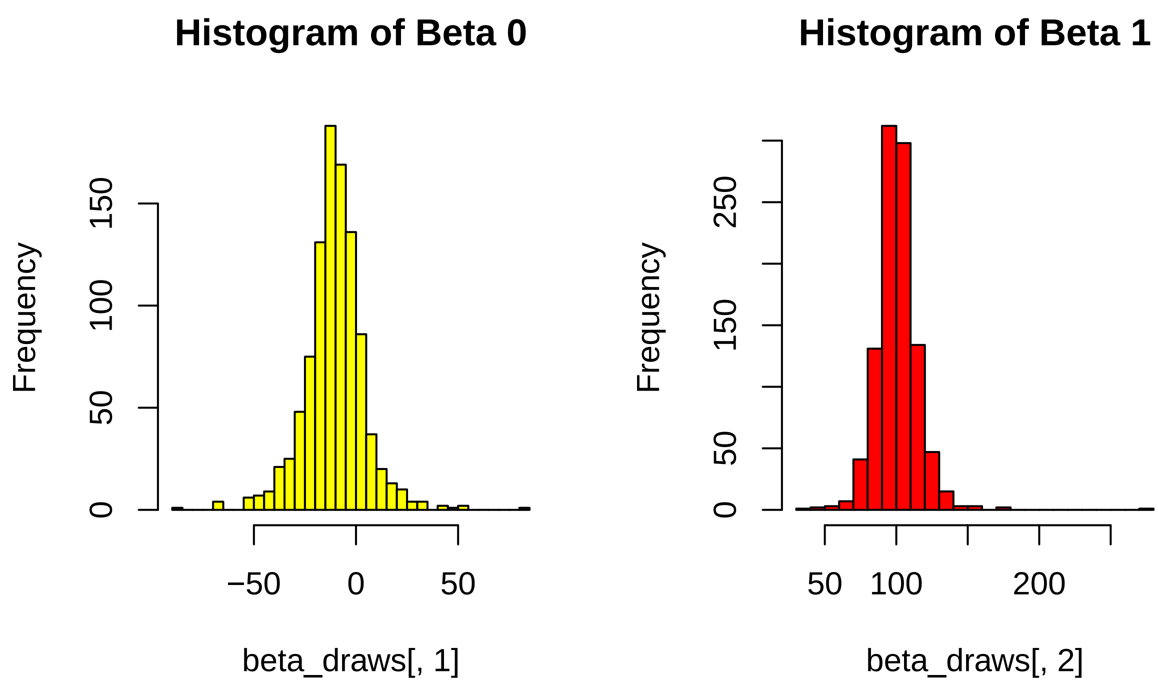
The curve didn't looked reasonable initially with $\Omega_0 = 0.01$ as the variations in temeprature around the year was too much and uneven. With increase of the Ω_0 value, the collection of curve started getting smooth and realistic ,i.e as it seems to capture the seasonal temperature variations within range. We could notice that the summer or the mid range along x-axis has higher temperature than the extremes (winter season). Looking at the plots, we could comfortably say that the temperature prediction looks good with $\Omega_0 = 0.1$ rather than with initial $\Omega_0 = 0.01$

1(b).

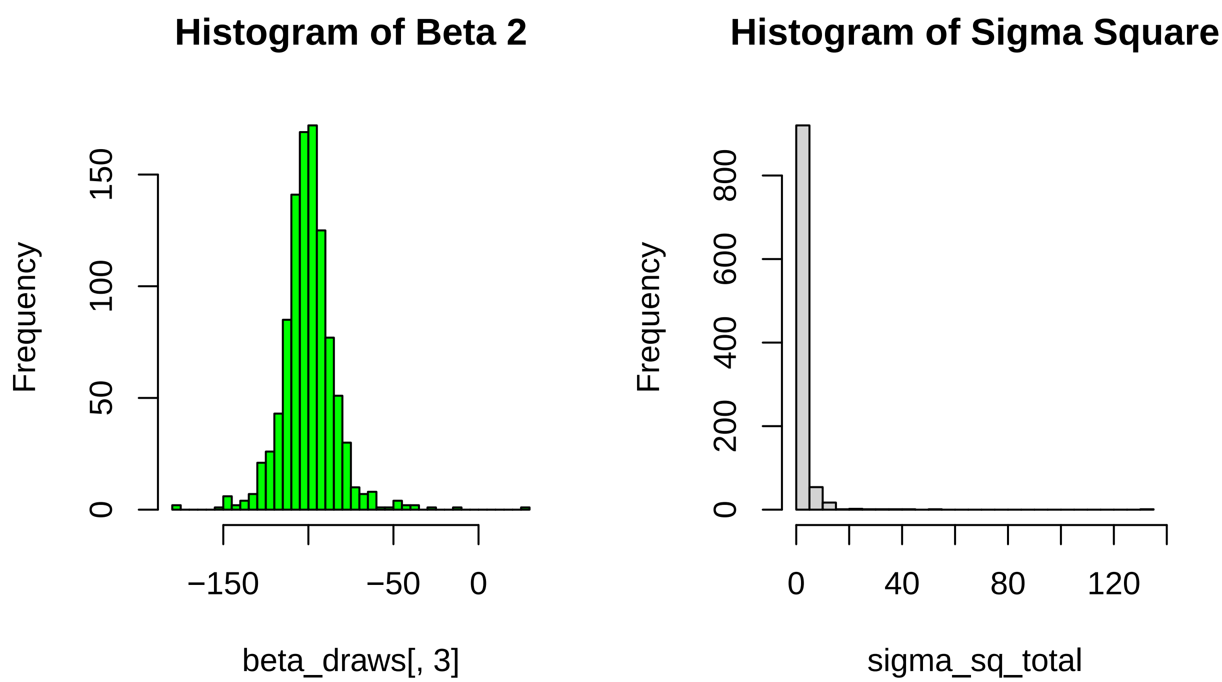
i. Plot a histogram for each marginal posterior of the parameters

```
pred_temp<-matrix(nrow = 365,ncol = 1000)
sigma_sq_total<-c()
omega0<-omega*0.01
for (i in 1:1000){
x = rchisq(n = 1, df = nu0)
# As per equation 2 and solving it for sigma square
sigma_square<-(nu0*sigma0_square)/x
sigma_sq_total<-c(sigma_sq_total,sigma_square)

# As per equation 1 , solving Beta for given sigma_square
beta_draws[i,]<-rmvnorm(n=1,mean=mu0,sigma=sigma_square*solve(omega0))
pred_temp[,i]<-sapply(data$time,quad,betas=beta_draws[i,],
                       err=rnorm(n=1,mean=0,sd=sqrt(sigma_square)))}
```

```r
# Plot histograms of the posterior draws of parameters

layout(matrix(c(1:2, 0, 0), nrow=1, ncol=2, byrow=TRUE))# Splits in 2-by-2 structure
hist(beta_draws[,1],col="yellow",main = "Histogram of Beta 0",breaks = 30)
hist(beta_draws[,2],col="red",main = "Histogram of Beta 1",breaks = 30)
```



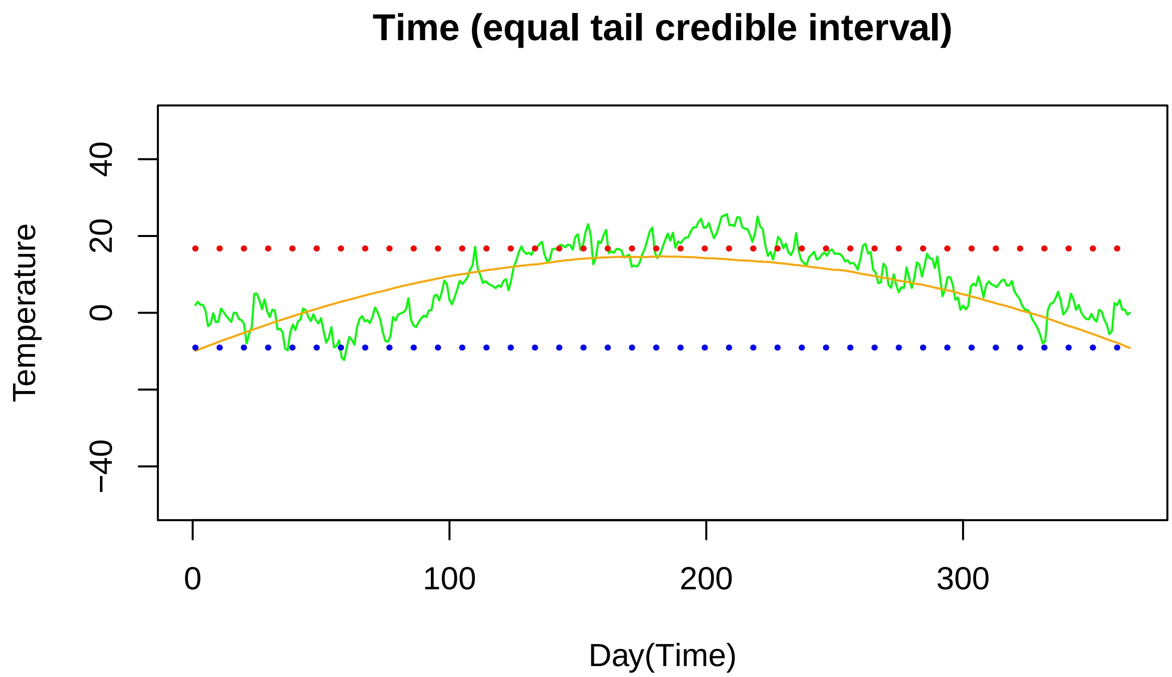

```r
hist(beta_draws[,3],col="green",main = "Histogram of Beta 2",breaks = 30)
hist(sigma_sq_total,main = "Histogram of Sigma Square",breaks = 30)
```



ii.

Make a scatter plot of the temperature data and overlay a curve for the posterior median of the regression function

```
pred_temp[,i]<-sapply(data$time,quad,betas=beta_draws[i,],
                      err=rnorm(n=1,mean=0,sd=sqrt(sigma_square)))

temp_mean<-rowMedians(pred_temp)
value_ul<-c()
value_ll<-c()
for (i in 1:365){

dense_rmv<-density(temp_mean)
normal_data<-cumsum(dense_rmv$y)/sum(dense_rmv$y)

lower_limit<-which(normal_data>=0.025)[1]
value_ll<-c(value_ll,dense_rmv$x[lower_limit])

upper_limit<-which(normal_data>=0.975)[1]
value_ul<-c(value_ul,dense_rmv$x[upper_limit])}

plot(data$temp,col="green", main = "Time (equal tail credible interval)",
     type="l",ylim=c(-50,50), xlab="Day(Time)",ylab="Temperature")
lines(temp_mean,col="orange")
lines(value_ll, col="blue", lwd=3,lty=3,ylim=c(-50,50))
lines(value_ul, col="red", lwd=3,lty=3,ylim=c(-50,50))
```



The intervals (credible) have been denoted in red and blue color. The median temperature is highlighted in orange. The intervals seems to capture most of data-sets within the band . Since, the band is denoted by 95% equal tail posterior probability intervals, it should capture most of the data sets and the same is visible by the above plot.

1(c)

It is of interest to locate the time with the highest expected temperature (i.e. the time where f(time) is maximal).

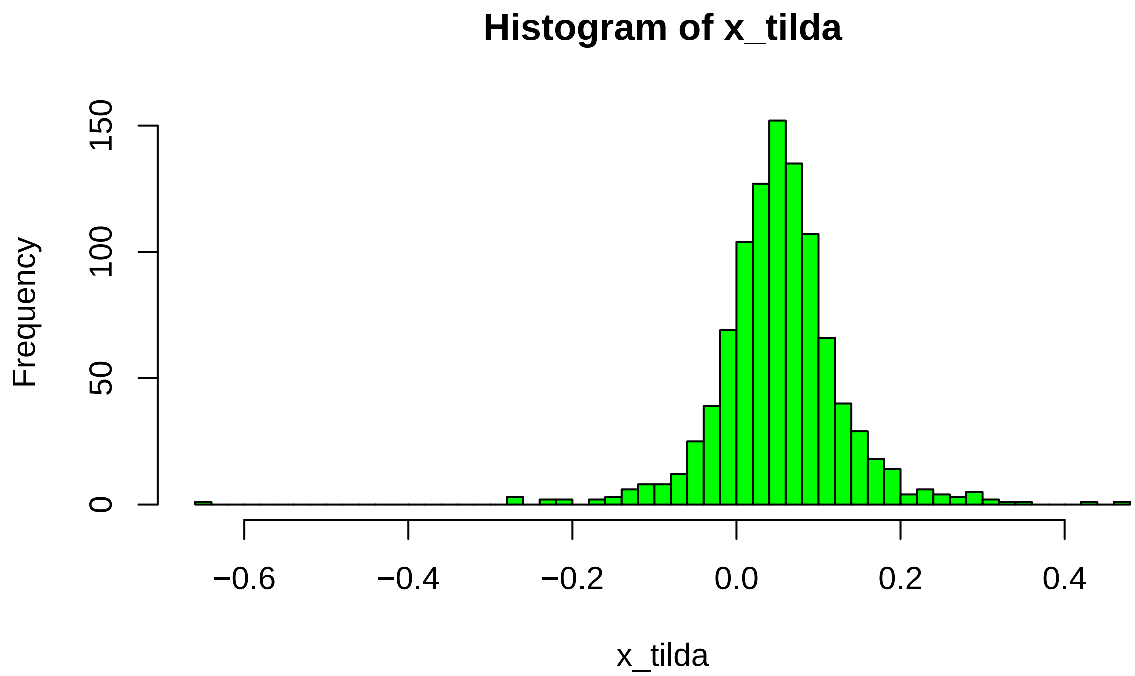
$$temp = \beta_0 + \beta_1.time + \beta_2.time^2 + err$$

We take derivative wrt time and equate it to zero to get time at maximum.

On solving we get,

$$\tilde{x} = \frac{-\beta_1}{2\beta_2}$$

```
x_tilda<- (-beta_draws[,1]/(2*beta_draws[,2]))
hist(x_tilda,col="green",breaks = 50)
```



1 d.

Since, the higher order terms may not be needed for 7th order polynomial model, we can eliminate the variables for the same. Lasso regression comes very handy when we need to eliminate the unwanted features/variables.

The Lasso is equivalent to the posterior mode under Laplace prior and is given by :

$$\beta_i \mid \sigma^2 \approx Laplace(0, \frac{sigma^2}{\lambda})$$

2. Posterior approximation for classification with logistic regression

2(a)

Logistic regression when y=1

$$Pr(y=1 \mid x) = \frac{exp(X'\beta)}{1+exp(X'\beta)})$$

Likelihood is given by :

$$P(y \mid X, \beta) = \prod_{i=1}^{n} \frac{(exp(X_i'\beta))^y i}{1+exp(X_i'\beta)}$$

when y=0: $p(y=0 \mid X) = \frac{1}{1+exp(X'\beta)}$ when women does not work
as $p(y=1 \mid X) + p(y=0 \mid X) = 1$ Total probability

Likelihood,
$Pr(y \mid X,\beta) = \sum_{i=1}^{n} y_i log(\frac{exp(X'\beta)}{1+exp(X'\beta)}) + (1-y_i)log(\frac{1}{1+exp(X'\beta)})$

$Pr(y \mid X,\beta) = \sum_{i=1}^{n} y_i log(\frac{1}{1+exp(-X'\beta)}) + (1)log(\frac{1}{1+exp(X'\beta)}) - y_i log(\frac{1}{1+exp(X'\beta)})$

$Pr(y \mid X,\beta) = \sum_{i=1}^{n} y_i [log(\frac{\frac{1}{1+exp(-X'\beta)}}{\frac{1}{1+exp(X'\beta)}}) + log(\frac{1}{1+exp(X'\beta)})$

$Pr(y \mid X,\beta) = \sum_{i=1}^{n} y_i [log(exp(-X'\beta)) - log(1+exp(X'\beta))$

$Pr(y \mid X,\beta) = \sum_{i=1}^{n} y_i X'\beta - log(1+exp(X'\beta)$

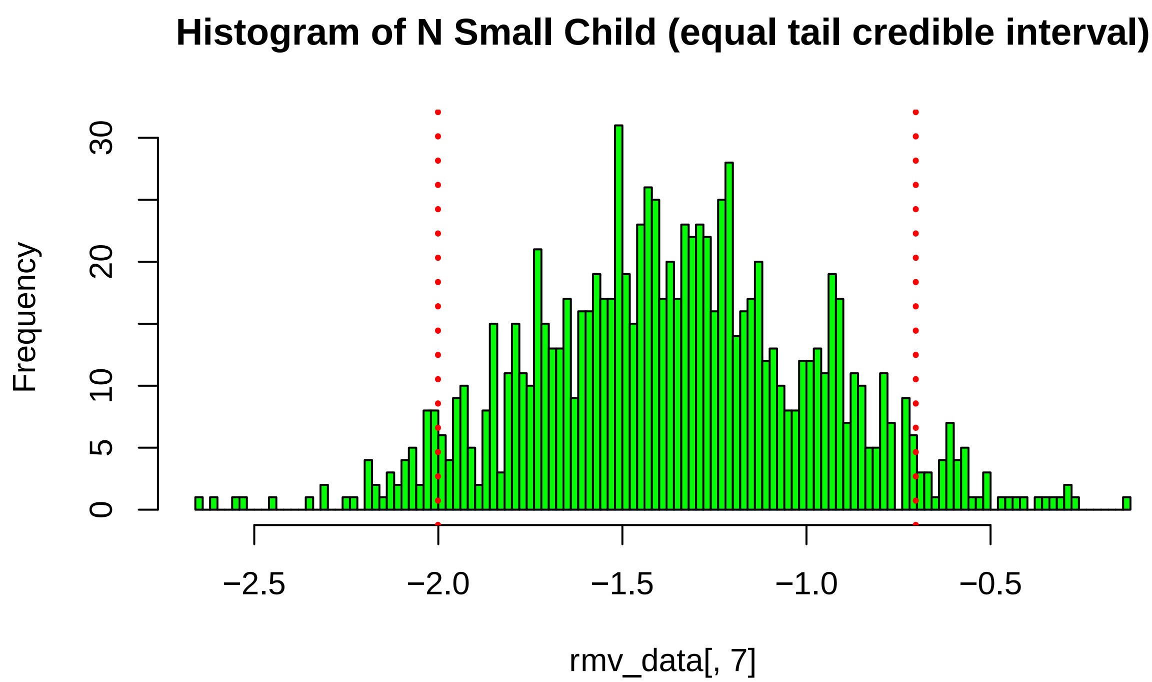
```
Data<-read.table("WomenWork.dat",header=TRUE)
#length(data_women)
chooseCov <- c(1:8)  # covariates other than target
tau <- 10      # given

# Loading data
y <- as.vector(Data[,1])
X <- as.matrix(Data[,2:9])
covNames <- names(Data)[2:length(names(Data))]
X <- X[,chooseCov]
covNames <- covNames[chooseCov]
nPara <- dim(X)[2]

# Setting up the prior
mu <- as.vector(rep(0,nPara)) # Prior mean vector
Sigma <- tau^2*diag(nPara) # as per the given prior

LogPostLogistic <- function(betaVect,y,X,mu,Sigma){
  nPara <- length(betaVect);
  linPred <- X%*%betaVect;
  # evaluating the log-likelihood
  logLik <- sum( linPred*y -log(1 + exp(linPred)));
  if (abs(logLik) == Inf) logLik = -20000
  logPrior <- dmvnorm(betaVect, matrix(0,nPara,1), Sigma, log=TRUE)
  return(logLik + logPrior)
}

initVal <- as.vector(rep(0,dim(X)[2]))
# logistic regression
logPost = LogPostLogistic

OptimResults<-optim(initVal,logPost,gr=NULL,y,X,mu,Sigma,
                    method=c("BFGS"),control=list(fnscale=-1),hessian=TRUE)
postMode<-OptimResults$par
postCov<--solve(OptimResults$hessian)

#posterior covariance matrix
print("The posterior Covariance Matrix :")
```

```
## [1] "The posterior Covariance Matrix :"
```

```r
postCov
```

```
##              [,1]          [,2]          [,3]          [,4]          [,5]
## [1,]  2.266022568  3.338861e-03 -6.545121e-02 -1.179140e-02  0.0457807243
## [2,]  0.003338861  2.528045e-04 -5.610225e-04 -3.125413e-05  0.0001414915
## [3,] -0.065451206 -5.610225e-04  6.218199e-03 -3.558209e-04  0.0018962893
## [4,] -0.011791404 -3.125413e-05 -3.558209e-04  4.351716e-03 -0.0142490853
## [5,]  0.045780724  1.414915e-04  1.896289e-03 -1.424909e-02  0.0555786706
## [6,] -0.030293450 -3.588562e-05 -3.240448e-06 -1.340888e-04 -0.0003299398
## [7,] -0.188748354  5.066847e-04 -6.134564e-03 -1.468951e-03  0.0032082535
## [8,] -0.098023929 -1.444223e-04  1.752732e-03  5.437105e-04  0.0005120144
##               [,6]          [,7]          [,8]
## [1,] -3.029345e-02 -0.1887483542 -0.0980239285
## [2,] -3.588562e-05  0.0005066847 -0.0001444223
## [3,] -3.240448e-06 -0.0061345645  0.0017527317
## [4,] -1.340888e-04 -0.0014689508  0.0005437105
## [5,] -3.299398e-04  0.0032082535  0.0005120144
## [6,]  7.184611e-04  0.0051841611  0.0010952903
## [7,]  5.184161e-03  0.1512621814  0.0067688739
## [8,]  1.095290e-03  0.0067688739  0.0199722657
```

```r
rmv_data<-rmvnorm(n=1000,mean=postMode,sigma =postCov)

dense_rmv<-density(rmv_data[,7])
normal_data<-cumsum(dense_rmv$y)/sum(dense_rmv$y)
lower_limit<-which(normal_data>=0.05)[1]
value_ll<-dense_rmv$x[lower_limit]
upper_limit<-which(normal_data>=0.95)[1]
value_ul<-dense_rmv$x[upper_limit]
hist(rmv_data[,7],col="green",breaks = 100,
     main = "Histogram of N Small Child (equal tail credible interval)") # N small child column
abline(v=value_ll, col="red", lwd=3,lty=3)
abline(v=value_ul, col="red", lwd=3,lty=3)
```




```r
#verification using glm
model<-glm(Data$Work~0+., data=Data,family = binomial)
summary(model)
```

```
##
## Call:
## glm(formula = Data$Work ~ 0 + ., family = binomial, data = Data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1662  -0.9299   0.4391   0.9494   2.0582
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## Constant     0.64430    1.52307   0.423 0.672274
## HusbandInc  -0.01977    0.01590  -1.243 0.213752
## EducYears    0.17988    0.07914   2.273 0.023024 *
## ExpYears     0.16751    0.06600   2.538 0.011144 *
## ExpYears2   -0.14436    0.23585  -0.612 0.540489
## Age         -0.08234    0.02699  -3.050 0.002285 **
## NSmallChild -1.36250    0.38996  -3.494 0.000476 ***
## NBigChild   -0.02543    0.14172  -0.179 0.857592
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 277.26  on 200  degrees of freedom
```

```
## Residual deviance: 222.73  on 192  degrees of freedom
## AIC: 238.73
##
## Number of Fisher Scoring iterations: 4
```

```r
# Beta values that maximizes log posterior
print(OptimResults$par)
```

```
## [1]  0.62672884 -0.01979113  0.18021897  0.16756670 -0.14459669 -0.08206561
## [7] -1.35913317 -0.02468351
```

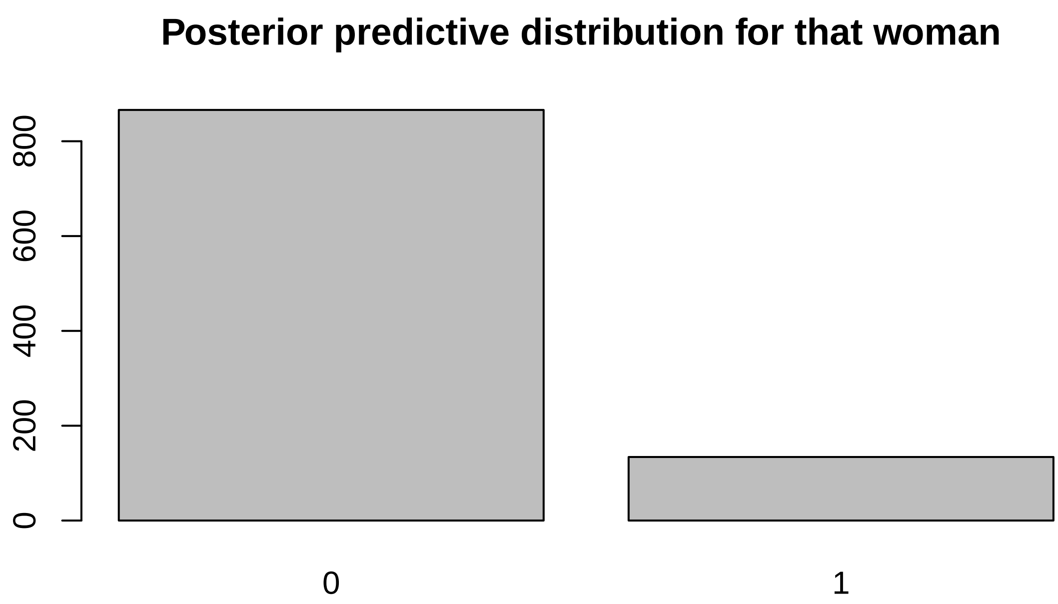
```r
# Other way
new_sigma<--solve(OptimResults$hessian)
beta_value<-rmvnorm(n=1000,mean=OptimResults$par,sigma=new_sigma)
```

The above plotted credible interval suggests that it does take 95% into account and the distribution looks normal or bell-shaped. We think that it is a major feature for the probability that a womens works as small child (i.e less than or equal to 6 years in age) needs special care and is therefore a major decision maker for women . This feature looks significant.

Also, Looking at the summary of the GLM model, the coeff NSmallChild is significant determinant of the probability that a woman works.It verifies the samne.

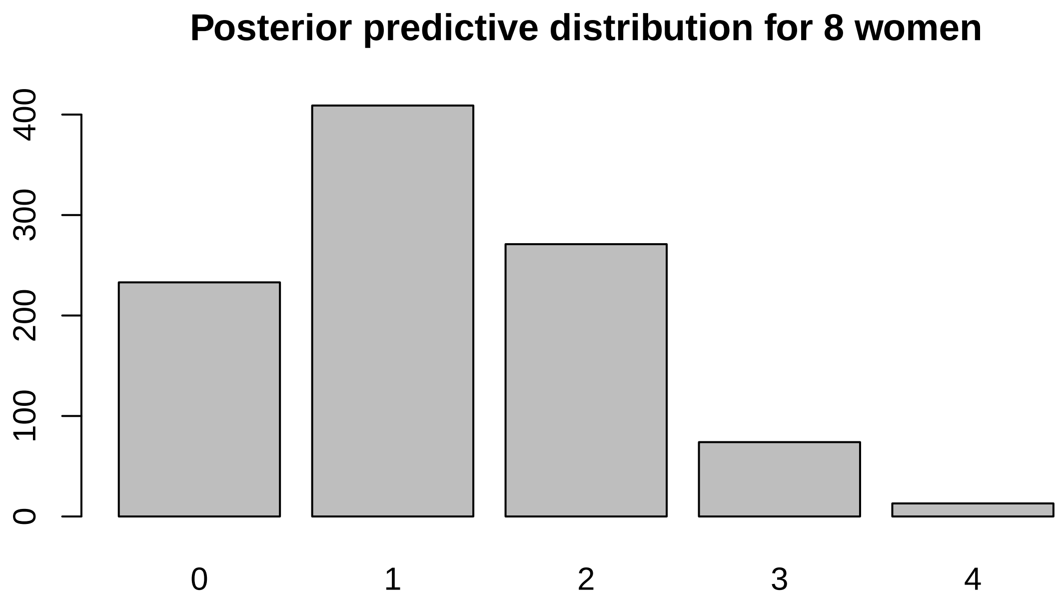
2(b)

```r
# given values stored in a vector
vector<-c(1,13,8,11,1,37,2,0)

# creating required function
beta_function<- function(vector,beta_value){
  pred_dist<-c()
  for(i in 1:dim(beta_value)[1]){
  pred_dist[i]<-(exp(t(vector)%*%beta_value[i,])/(1+exp(t(vector)%*%beta_value[i,])))
}
  return(pred_dist)
}


set.seed(123)
res<-c()
for(i in 1:1000){
res[i]<-sum(rbinom(n=1,1,prob=beta_function(vector,beta_value)))}

barplot(table(res),main = "Posterior predictive distribution for that woman ")
```



2(c)

```
# 2c
set.seed(123)
# 8 women which all have the same features
res<-c()
for(i in 1:1000){
res[i]<-sum(rbinom(n=8,1,prob=beta_function(vector,beta_value)))}

barplot(table(res),main = "Posterior predictive distribution for 8 women")
```



Contribution :

Biswas contributed majorly with Assignment # 1, report writing and overall trouble-shooting. Gowtham contributed majorly with Assignment #2. Both team members discussed on solution approach and expected outcomes of all the assignments.

Note : We have referred lecture notes, our group's previous submission and R -documentation

Code Appendix

```
knitr::opts_chunk$set(echo = TRUE)
library("mvtnorm")
library("robustbase")

set.seed(123)
#1 a & b
data<-read.table("TempLinkoping.txt",col.name=c("time","temp"),
                 stringsAsFactors = FALSE,header = TRUE)

# calculating time square as it is required to solve expression ahead
data$time_sq<-data[,1]^2

data$constant=1
```

```r
#rearrange columns
data<-data[,c(4,1,3,2)]

# hyper parameters (given)
mu0<-t(c(-10,100,-100))
omega<-diag(3)
# as per quation 5 and given
omega0<-omega*0.01
nu0<-4
k<-3 # no.of beta values
sigma0_square<-1


sigma_square<-c()
beta_draws<-matrix(nrow = 1000,ncol = 3)
quad<-function(betas,time,err){
  temp_equation<-betas[1]+ betas[2]*time+betas[3]*time^2+ err
  return(temp_equation)
}

pred<-function(Omega_knot,k){
pred_temp<-matrix(nrow = 365,ncol = 1000)
sigma_sq_total<-c()
for (i in 1:1000){
x = rchisq(n = 1, df = nu0)
# As per equation 2 and solving it for sigma square
sigma_square<-(nu0*sigma0_square)/x
sigma_sq_total<-c(sigma_sq_total,sigma_square)

# As per equation 1 , solving Beta for given sigma_square
beta_draws[i,]<-rmvnorm(n=1,mean=mu0,sigma=sigma_square*solve(Omega_knot))
pred_temp[,i]<-sapply(data$time,quad,betas=beta_draws[i,],
                      err=rnorm(n=1,mean=0,sd=sqrt(sigma_square)))

}



plot(data$time, pred_temp[,1],type="l",
     ylim=c(-30,30),main= paste("Curves,Omega0=", k),xlab="Time",
     ylab="Predicted_Temperature")
for(i in 2:10){
  lines(data$time, pred_temp[,i],col=i)
}
}
layout(matrix(c(1:2, 0, 0), nrow=1, ncol=2, byrow=TRUE))
for (k in seq(0.01,0.1,0.01)){
pred(omega*k,k)}


pred_temp<-matrix(nrow = 365,ncol = 1000)
sigma_sq_total<-c()
omega0<-omega*0.01
```

```r
for (i in 1:1000){
x = rchisq(n = 1, df = nu0)
# As per equation 2 and solving it for sigma square
sigma_square<-(nu0*sigma0_square)/x
sigma_sq_total<-c(sigma_sq_total,sigma_square)

# As per equation 1 , solving Beta for given sigma_square
beta_draws[i,]<-rmvnorm(n=1,mean=mu0,sigma=sigma_square*solve(omega0))
pred_temp[,i]<-sapply(data$time,quad,betas=beta_draws[i,],
                      err=rnorm(n=1,mean=0,sd=sqrt(sigma_square)))}

# Plot histograms of the posterior draws of parameters

layout(matrix(c(1:2, 0, 0), nrow=1, ncol=2, byrow=TRUE))# Splits in 2-by-2 structure
hist(beta_draws[,1],col="yellow",main = "Histogram of Beta 0",breaks = 30)
hist(beta_draws[,2],col="red",main = "Histogram of Beta 1",breaks = 30)
hist(beta_draws[,3],col="green",main = "Histogram of Beta 2",breaks = 30)
hist(sigma_sq_total,main = "Histogram of Sigma Square",breaks = 30)



pred_temp[,i]<-sapply(data$time,quad,betas=beta_draws[i,],
                      err=rnorm(n=1,mean=0,sd=sqrt(sigma_square)))

temp_mean<-rowMedians(pred_temp)
value_ul<-c()
value_ll<-c()
for (i in 1:365){

dense_rmv<-density(temp_mean)
normal_data<-cumsum(dense_rmv$y)/sum(dense_rmv$y)

lower_limit<-which(normal_data>=0.025)[1]
value_ll<-c(value_ll,dense_rmv$x[lower_limit])

upper_limit<-which(normal_data>=0.975)[1]
value_ul<-c(value_ul,dense_rmv$x[upper_limit])}

plot(data$temp,col="green", main = "Time (equal tail credible interval)",
     type="l",ylim=c(-50,50), xlab="Day(Time)",ylab="Temperature")
lines(temp_mean,col="orange")
lines(value_ll, col="blue", lwd=3,lty=3,ylim=c(-50,50))
lines(value_ul, col="red", lwd=3,lty=3,ylim=c(-50,50))


x_tilda<- (-beta_draws[,1]/(2*beta_draws[,2]))
hist(x_tilda,col="green",breaks = 50)

Data<-read.table("WomenWork.dat",header=TRUE)
#length(data_women)
chooseCov <- c(1:8)  # covariates other than target
tau <- 10      # given
```

```r
# Loading data
y <- as.vector(Data[,1])
X <- as.matrix(Data[,2:9])
covNames <- names(Data)[2:length(names(Data))]
X <- X[,chooseCov]
covNames <- covNames[chooseCov]
nPara <- dim(X)[2]

# Setting up the prior
mu <- as.vector(rep(0,nPara)) # Prior mean vector
Sigma <- tau^2*diag(nPara) # as per the given prior

LogPostLogistic <- function(betaVect,y,X,mu,Sigma){
  nPara <- length(betaVect);
  linPred <- X%*%betaVect;
  # evaluating the log-likelihood
  logLik <- sum( linPred*y -log(1 + exp(linPred)));
  if (abs(logLik) == Inf) logLik = -20000
  logPrior <- dmvnorm(betaVect, matrix(0,nPara,1), Sigma, log=TRUE)
  return(logLik + logPrior)
}

initVal <- as.vector(rep(0,dim(X)[2]))
# logistic regression
logPost = LogPostLogistic

OptimResults<-optim(initVal,logPost,gr=NULL,y,X,mu,Sigma,
                    method=c("BFGS"),control=list(fnscale=-1),hessian=TRUE)
postMode<-OptimResults$par
postCov<--solve(OptimResults$hessian)

#posterior covariance matrix
print("The posterior Covariance Matrix :")
postCov

rmv_data<-rmvnorm(n=1000,mean=postMode,sigma =postCov)

dense_rmv<-density(rmv_data[,7])
normal_data<-cumsum(dense_rmv$y)/sum(dense_rmv$y)
lower_limit<-which(normal_data>=0.05)[1]
value_ll<-dense_rmv$x[lower_limit]
upper_limit<-which(normal_data>=0.95)[1]
value_ul<-dense_rmv$x[upper_limit]
hist(rmv_data[,7],col="green",breaks = 100,
     main = "Histogram of N Small Child (equal tail credible interval)") # N small child column
abline(v=value_ll, col="red", lwd=3,lty=3)
abline(v=value_ul, col="red", lwd=3,lty=3)


#verification using glm
model<-glm(Data$Work~0+., data=Data,family = binomial)
summary(model)
```

```r
# Beta values that maximizes log posterior
print(OptimResults$par)

# Other way
new_sigma<--solve(OptimResults$hessian)
beta_value<-rmvnorm(n=1000,mean=OptimResults$par,sigma=new_sigma)


# given values stored in a vector
vector<-c(1,13,8,11,1,37,2,0)

# creating required function
beta_function<- function(vector,beta_value){
  pred_dist<-c()
  for(i in 1:dim(beta_value)[1]){
  pred_dist[i]<-(exp(t(vector)%*%beta_value[i,])/(1+exp(t(vector)%*%beta_value[i,])))
}
  return(pred_dist)
}


set.seed(123)
res<-c()
for(i in 1:1000){
res[i]<-sum(rbinom(n=1,1,prob=beta_function(vector,beta_value)))}

barplot(table(res),main = "Posterior predictive distribution for that woman ")
# 2c
set.seed(123)
# 8 women which all have the same features
res<-c()
for(i in 1:1000){
res[i]<-sum(rbinom(n=8,1,prob=beta_function(vector,beta_value)))}

barplot(table(res),main = "Posterior predictive distribution for 8 women")
```