Bayesian Lab - 2

Vinay Bengaluru(vinbe289), Tejashree R Mastamardi(tejma768),

May 5, 2019

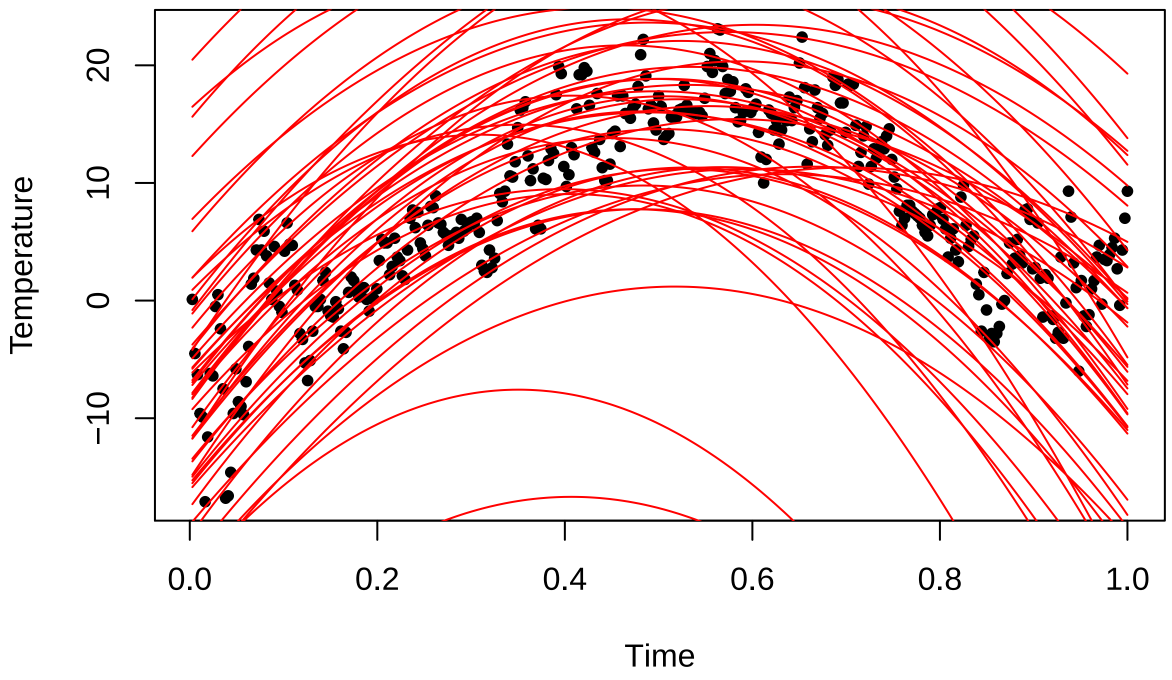
Contents

```r
library(geoR)
library(mvtnorm)
library(MASS)
```

Question 1: Linear and Polynomial Regression

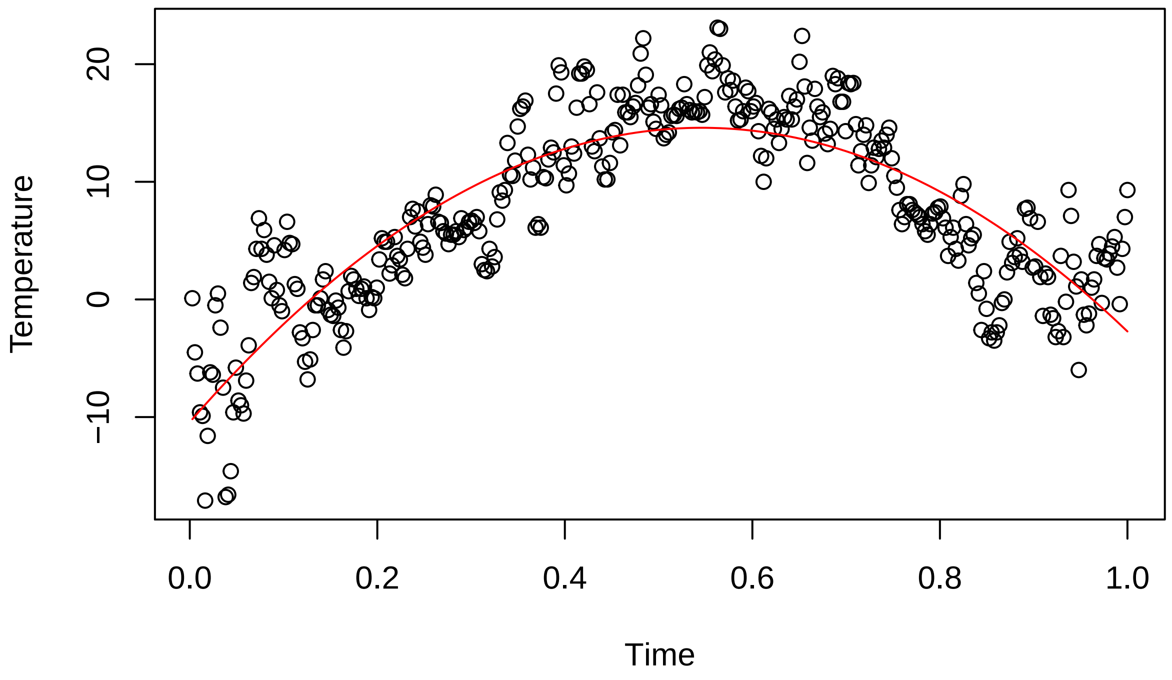
1(a) Determining the prior distribution of the model parameters. Use the conjugate prior for the linear regression model.Check if this prior agrees with your prior opinions by simulating draws from the joint prior of all parameters and for every draw compute the regression curve. This gives a collection of regression curves, one for each draw from the prior. Do the collection of curves look reasonable? If not, change the prior hyperparameters until the collection of prior regression curves do agree with your prior beliefs about the regression curve.

```r
data <- read.table("TempLinkoping.txt",header = T)

mu0 <- c(-10,100,-100)
omega0 <-  0.01*diag(3)
nu0 <- 4
sigma0sq <- 1
hyperparameter <- list(mu0=mu0,omega0=omega0,nu0=nu0,sigma0sq=sigma0sq)


time_df <- data.frame(rep(1,nrow(data)),data$time,data$time^2)
time_mat<- as.matrix(time_df)
temp_mat <- matrix(data$temp,ncol = 1)

prior <- function(data2 , hyperparameter){
  #Prior equations in Lecture 5 slide 7
  sigmasq <- rinvchisq(n = 1,df = hyperparameter$nu0,scale = hyperparameter$sigma0sq)
  beta <- rmvnorm(n = 1,mean = hyperparameter$mu0,sigma = sigmasq * solve(hyperparameter$omega0))
  data2 %*% t(beta)
}

plot(data$time, data$temp, pch = 20, xlab = "Time",ylab = "Temperature" )
X <- sort(data$time)
for(i in 1:50){
  temp <- prior(data2 = time_mat,hyperparameter = hyperparameter)
  lines(x = X,y = temp,col = 'red',type = 'l')
}
```

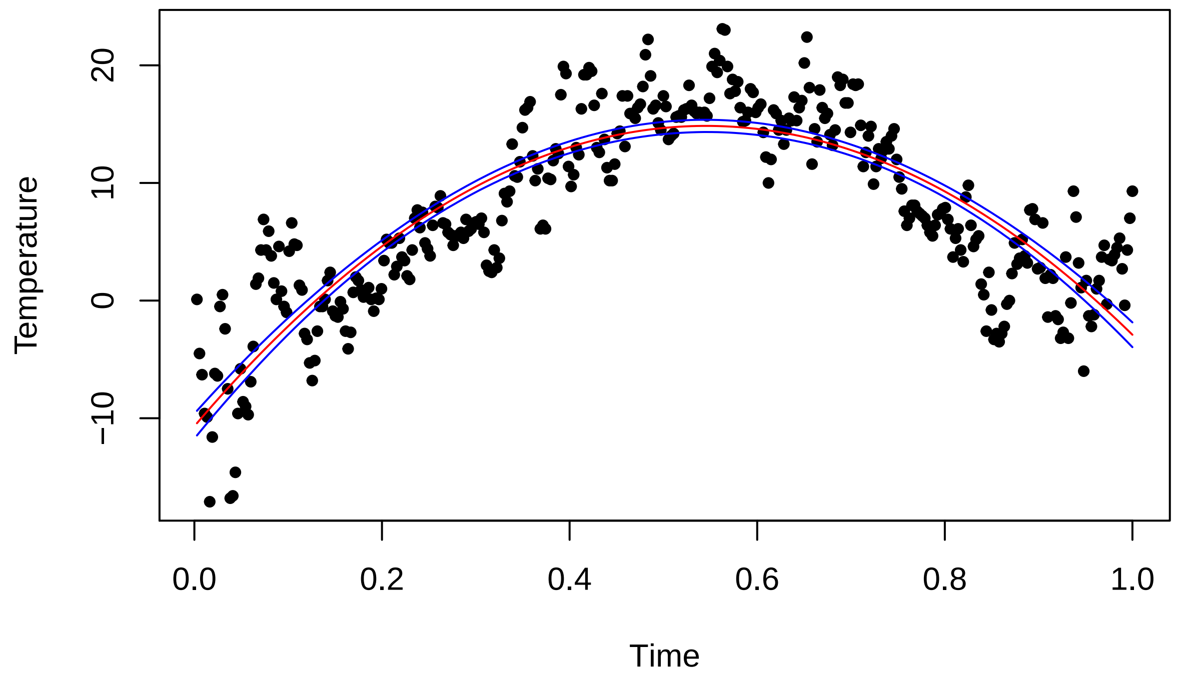


1(b) compute the 95% equal tail posterior probability intervals for every value of and then connect the lower and upper limits of the interval by curves. Does the interval bands contain most of the data points? Should they?

```r
library(mvtnorm)
library(MASS)
set.seed(12345)
jointPost <- function(data,hyperparameter){
  #Equations in Lecture 5 slide 7
  n <- nrow(data)
  nun <- hyperparameter$nu0 + n
  y <- as.matrix(data$temp)
  X <- time_mat
  X_t_X <- t(X)%*%X
  betacap <- solve(X_t_X)%*%t(X)%*%y
  mun <- solve(X_t_X + hyperparameter$omega0)%*%(X_t_X%*%betacap + hyperparameter$omega0%*%hyperparamete
  omegan <- X_t_X + hyperparameter$omega0
  sigmansq <- (hyperparameter$nu0*hyperparameter$sigma0sq + (t(y)%*%y + t(hyperparameter$mu0)%*%hyperpar
  #From posterior equations in Lecture 5 slide 7
  sigmasq2 <- rinvchisq(n = 1,df = nun,scale = sigmansq)
  beta2 <- mvrnorm(n = 1,mu = mun,Sigma = as.numeric(sigmasq2)*solve(omegan))
  betaEstimate <- list(beta2=beta2, sigmasq2=sigmasq2)

  # #Marginal posterior equation from lecture 5 slide 6
  # betaMarginal <-
}

#betaEstimate <- jointPost(data = data,hyperparameter = hyperparameter)
postEstimate <- function(data,hyperparameter){
  sample <- jointPost(data, hyperparameter)
  time_mat %*% sample$beta
}


plot(x = data$time,y = data$temp,xlab = "Time",ylab = "Temperature")

ids <- order(data$time)
x <- data$time[ids]
y <- postEstimate(data,hyperparameter)[ids]
lines(x,y,col='red',type='l')
```



```
estimates <- sapply(1:10000, FUN = function(x) postEstimate(data,hyperparameter))
credInterval <- apply(estimates, MARGIN = 1, quantile, probs = c(0.05, 0.95)) ###ERROR:Not able to get

y1 <- rowMeans(estimates)[ids]
y2 <- credInterval[1,][ids]
y3 <- credInterval[2,][ids]
plot(data$time, data$temp, pch = 20 ,xlab = "Time",ylab = "Temperature")
lines(x, y1, col='red', type='l', lwd=1)
lines(x, y2, col='blue', type='l', lwd=1)
lines(x, y3, col='blue', type='l', lwd=1)
```



The interval bands does not contain most of the data points.

1(c) It is of interest to locate the time with the highest expected temperature(that is the time where f(time) is maximal). Use the simulations in b) to simulate from the posterior distribution of x.

```
betas <- sapply(1:10000, FUN = function(x) jointPost(data,hyperparameter)$beta2)
hottest <- mean(-betas[2,] / (2 * betas[3,]))
hottest * 365
```

```
## [1] 199.0099
```

1(d) Say now that you want to estimate a polynomial model of order 7, but you suspect that higher order terms may not be needed, and you worry about overfitting. Suggest a suitable prior that mitigates this potential problem. You do not need to compute the posterior, just write down your prior.

We can set mu values to zero and few omega values to lower values and the other omega to higher values.This way we will be more certain about the latter 4 to 5 values.

Question 2 : Posterior approximation for clasiification with Logistic Regression

2(a) Fit a Logistic Regression using Maximum Likelihood.

```
my_data <- read.table("WomenWork.dat", header = TRUE)
glmModel <- glm(Work ~ 0 + ., data = my_data, family = binomial)
```

2(b) Compute an approximate 95% credible interval for the variable NSmallChild. Would you say that this feature is an important determinant of the probability that a women works?

```
log_prior <- function(beta, mean, sigma){
dmvnorm(beta, mean = mean, sigma = sigma, log = TRUE)
}
log_likelihood <- function(beta, X, Y){
linear_pred <- t(X) %*% beta
probabs <- (Y * linear_pred) - log(1 + exp(linear_pred))
log_like <- sum(probabs)
log_like
}
log_posterior <- function(beta, X, Y, mean, sigma){
log_likelihood(beta, X, Y) + log_prior(beta, mean, sigma)
}
tau <- 10
Mu <- rep(0,8)
Sigma <- tau^2 * diag(8)
X_Women <- as.matrix(my_data[,2:ncol(my_data)])
Y_Women <- as.matrix(my_data[,1])
Res <- optim(par = matrix(rep(0, 8), ncol = 1),
fn = log_posterior, method = "BFGS", hessian = TRUE,
X = t(X_Women), Y = Y_Women,
mean = Mu, sigma = Sigma,
control=list(fnscale=-1))
beta_posterior <- function(n, Mu, Sigma){
rmvnorm(n, mean = Mu, sigma = Sigma)
}
Mu <- Res$par
Sigma <- -solve(Res$hessian)
set.seed(12345)
betas <- beta_posterior(n=1000, Mu=Mu, Sigma=Sigma)
credible_interval <- apply(betas, 2, quantile, prob=c(0.025, 0.975))
colnames(credible_interval) <- colnames(my_data)[-1]
credible_interval
```

```
##        Constant  HusbandInc  EducYears   ExpYears  ExpYears2         Age
## 2.5%  -2.313434 -0.05091541 0.03095675 0.04723847 -0.6143409 -0.13476231
## 97.5%  3.533276  0.00780642 0.32739377 0.29656573  0.2943841 -0.03095853
##       NSmallChild  NBigChild
## 2.5%   -2.1296887 -0.3023168
## 97.5%  -0.5759195  0.2692414
```

We can say that NSmallChilds is an important feature of our model as its credibility interval is less than 0.

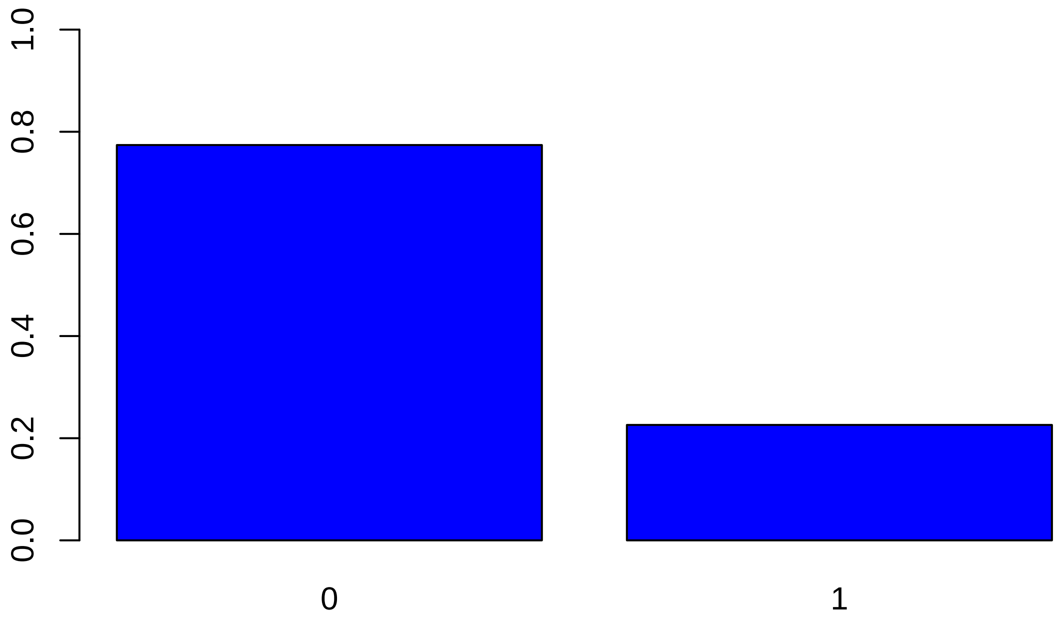
2(c) Write a function that simulates from the predictive distribution of the response variable in a logistic regression. Use your normal approximation from 2(b).

```
predictive_disttribution <- function(X, beta){
linear_pred <- t(X) %*% beta
prob <- exp(linear_pred) / (1 + exp(linear_pred))
rbinom(n=1, size=1, prob=prob)
}
x <- matrix(c(1, 10, 8, 10, (10 / 10)^2, 40, 1, 1), ncol=1)
```

```
pred_samples <- apply(betas, 1, function(beta) predictive_disttribution(x, beta))
T1 <- table(pred_samples)
T1
```

```
## pred_samples
##   0   1
## 774 226
```

```
barplot(T1 / sum(T1), ylim=c(0, 1), col = "blue")
```



From the above barplot, and also considering the given data it is evident that the women are not working under our model.

Appendix

```
knitr::opts_chunk$set(echo = TRUE)
library(geoR)
library(mvtnorm)
library(MASS)
data <- read.table("TempLinkoping.txt",header = T)

mu0 <- c(-10,100,-100)
omega0 <-  0.01*diag(3)
nu0 <- 4
sigma0sq <- 1
hyperparameter <- list(mu0=mu0,omega0=omega0,nu0=nu0,sigma0sq=sigma0sq)
```

```r
time_df <- data.frame(rep(1,nrow(data)),data$time,data$time^2)
time_mat<- as.matrix(time_df)
temp_mat <- matrix(data$temp,ncol = 1)

prior <- function(data2 , hyperparameter){
  #Prior equations in Lecture 5 slide 7
  sigmasq <- rinvchisq(n = 1,df = hyperparameter$nu0,scale = hyperparameter$sigma0sq)
  beta <- rmvnorm(n = 1,mean = hyperparameter$mu0,sigma = sigmasq * solve(hyperparameter$omega0))
  data2 %*% t(beta)
}

plot(data$time, data$temp, pch = 20, xlab = "Time",ylab = "Temperature" )
X <- sort(data$time)
for(i in 1:50){
  temp <- prior(data2 = time_mat,hyperparameter = hyperparameter)
  lines(x = X,y = temp,col = 'red',type = 'l')
}
library(mvtnorm)
library(MASS)
set.seed(12345)
jointPost <- function(data,hyperparameter){
  #Equations in Lecture 5 slide 7
  n <- nrow(data)
  nun <- hyperparameter$nu0 + n
  y <- as.matrix(data$temp)
  X <- time_mat
  X_t_X <- t(X)%*%X
  betacap <- solve(X_t_X)%*%t(X)%*%y
  mun <- solve(X_t_X + hyperparameter$omega0)%*%(X_t_X%*%betacap + hyperparameter$omega0%*%hyperparamete
  omegan <- X_t_X + hyperparameter$omega0
  sigmansq <- (hyperparameter$nu0*hyperparameter$sigma0sq + (t(y)%*%y + t(hyperparameter$mu0)%*%hyperpa
  #From posterior equations in Lecture 5 slide 7
  sigmasq2 <- rinvchisq(n = 1,df = nun,scale = sigmansq)
  beta2 <- mvrnorm(n = 1,mu = mun,Sigma = as.numeric(sigmasq2)*solve(omegan))
  betaEstimate <- list(beta2=beta2, sigmasq2=sigmasq2)

  # #Marginal posterior equation from lecture 5 slide 6
  # betaMarginal <-
}

#betaEstimate <- jointPost(data = data,hyperparameter = hyperparameter)
postEstimate <- function(data,hyperparameter){
  sample <- jointPost(data, hyperparameter)
  time_mat %*% sample$beta
}


plot(x = data$time,y = data$temp,xlab = "Time",ylab = "Temperature")

ids <- order(data$time)
x <- data$time[ids]
y <- postEstimate(data,hyperparameter)[ids]
lines(x,y,col='red',type='l')
```

```
estimates <- sapply(1:10000, FUN = function(x) postEstimate(data,hyperparameter))
credInterval <- apply(estimates, MARGIN = 1, quantile, probs = c(0.05, 0.95)) ###ERROR:Not able to get

y1 <- rowMeans(estimates)[ids]
y2 <- credInterval[1,][ids]
y3 <- credInterval[2,][ids]
plot(data$time, data$temp, pch = 20 ,xlab = "Time",ylab = "Temperature")
lines(x, y1, col='red', type='l', lwd=1)
lines(x, y2, col='blue', type='l', lwd=1)
lines(x, y3, col='blue', type='l', lwd=1)
betas <- sapply(1:10000, FUN = function(x) jointPost(data,hyperparameter)$beta2)
hottest <- mean(-betas[2,] / (2 * betas[3,]))
hottest * 365
my_data <- read.table("WomenWork.dat", header = TRUE)
glmModel <- glm(Work ~ 0 + ., data = my_data, family = binomial)
log_prior <- function(beta, mean, sigma){
dmvnorm(beta, mean = mean, sigma = sigma, log = TRUE)
}
log_likelihood <- function(beta, X, Y){
linear_pred <- t(X) %*% beta
probabs <- (Y * linear_pred) - log(1 + exp(linear_pred))
log_like <- sum(probabs)
log_like
}
log_posterior <- function(beta, X, Y, mean, sigma){
log_likelihood(beta, X, Y) + log_prior(beta, mean, sigma)
}
tau <- 10
Mu <- rep(0,8)
Sigma <- tau^2 * diag(8)
X_Women <- as.matrix(my_data[,2:ncol(my_data)])
Y_Women <- as.matrix(my_data[,1])
Res <- optim(par = matrix(rep(0, 8), ncol = 1),
fn = log_posterior, method = "BFGS", hessian = TRUE,
X = t(X_Women), Y = Y_Women,
mean = Mu, sigma = Sigma,
control=list(fnscale=-1))
beta_posterior <- function(n, Mu, Sigma){
rmvnorm(n, mean = Mu, sigma = Sigma)
}
Mu <- Res$par
Sigma <- -solve(Res$hessian)
set.seed(12345)
betas <- beta_posterior(n=1000, Mu=Mu, Sigma=Sigma)
credible_interval <- apply(betas, 2, quantile, prob=c(0.025, 0.975))
colnames(credible_interval) <- colnames(my_data)[-1]
credible_interval
predictive_disttribution <- function(X, beta){
linear_pred <- t(X) %*% beta
prob <- exp(linear_pred) / (1 + exp(linear_pred))
rbinom(n=1, size=1, prob=prob)
}
x <- matrix(c(1, 10, 8, 10, (10 / 10)^2, 40, 1, 1), ncol=1)
```

```r
pred_samples <- apply(betas, 1, function(beta) predictive_disttribution(x, beta))
T1 <- table(pred_samples)
T1
barplot(T1 / sum(T1), ylim=c(0, 1), col = "blue")
```