Bayesian Learning: Computer Lab 2

Mowniesh Asokan(mowas455), Shahrukh Iqbal(shaiq681)

May 1, 2021

1 Linear and Polynomial Regression

For the current problem, the dataset `TempLinkoping.txt` is used that consists of 365 observations of *time* and *temp*. We define the covariate *time* in **Definition 1** and the response variable *temp* in **Definition 2**.

Definition 1 (*time*)

$$time = \frac{the\ number\ of\ days\ since\ the\ beginning\ of\ the\ year}{365}$$

Definition 2 (*temp*)

$$temp = \beta_0 + \beta_1 \cdot time + \beta_2 \cdot time^2 + \epsilon, \epsilon \overset{i.i.d}{\sim} \mathrm{N}(0, \sigma^2)$$

1.1 (a)

$$\beta|\sigma^2 \sim \mathcal{N}_3(\mu_0, \sigma^2\Omega_0^{-1}) \tag{1}$$

$$\sigma^2 \sim \mathrm{Inv} - \chi^2(\nu_0, \sigma_0^2) \tag{2}$$

In the current problem, the objective is to perform a bayesian analysis of the quadratic regression model as defined in **Definition 2**. To calculate the regression model, we setup an experiment through which the values of $\beta_0, \beta_1, \beta_2$ and ϵ are sampled `nDraws` times from their respective conjugate prior distribution. For β_0, β_1 and β_2, the values are drawn from the prior distribution as defined in (1). Similarly, ϵ is drawn from a $\mathcal{N}$ distribution with 0 mean and σ^2 as sampled using (2). The hyper-parameters are initialized as follows, $\mu_0 = (-10, 100, -100)^T$, $\Omega_0 = 0.01 \cdot I_3$, $\nu_0 = 4, \sigma^2 = 1$. The experiment is implemented in `R` and defined under the function `plot_data`.

```
##### Q1 #####
library(mvtnorm)
data = read.table("TempLinkoping.txt", header = TRUE)
#a)
plot_data = function(data, v0, data_var_0, omega0, mu0, nDraws = 100){
```

```
  set.seed(2020-04-19)
  X = data.frame(c(1, data['time'], data['time']**2))
  n  = dim(X)[1]
  X_T = data.frame(t(X))
  XTX = as.matrix(X_T)%*% as.matrix(X)
  target = as.matrix(x = data['temp'], ncol=1)
  YTY = t(target) %*% target
  # Beta
  omega_o = omega0 * diag(3)
  omega_o_inv = solve(omega_o)
  omega_n = XTX + omega_o
  mu_o = matrix(data = mu0, ncol = 1)
  # variance
  vo = v0
  v_n = vo + n
  data_var_o = data_var_0
  nDraws = nDraws
  data_var = vo*data_var_o/rchisq(nDraws, df = vo)

  # Beta|sigma^2
  beta_prior = matrix(nrow = nDraws, ncol=3)
  for(draw in 1:nDraws){
    beta_prior[draw, ] = rmvnorm(1, mean =  c(-10,100,-100), sigma = data_var
  }
  ### Prior Beta
  beta = as.matrix(beta_prior, ncol = 3)
  # Y_hat
  error = rnorm(nDraws, 0, sd = sqrt(data_var))
  y_hat = as.matrix(X) %*% t(beta) + error

  # Plot
  plot(y = data$temp, x=1:365,type = "l", pch = 1,  col = "red", ylim = c(-20
       lwd = 3,
       xlab = "Time(Days)",ylab = "Temperature", main = "Time V. Temperature(
  for(i in 1:nDraws){
    lines(x = 1:365, y = y_hat[,i], type="l",
          col = rgb(0.01,0.1,0.03,0.4), lwd = 0.5)
  }
  lines(x = 1:365, y = rowMeans(y_hat), type="l", col ="blue", lwd = 3)
  legend("topleft",
         legend = c("Actual Temp", "Expected Temp", "Sample"), col = c("red",
         lty = c(1,1,1))

}
plot_data(data = data, v0 = 4,data_var_0 = 1, omega0 = 0.01, mu0 = c(-10,100,
```

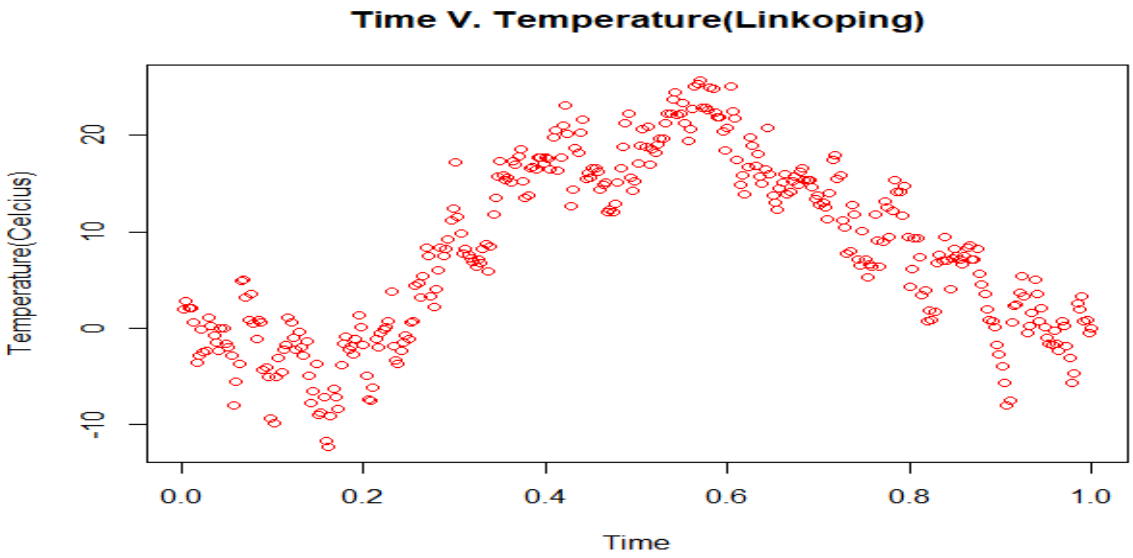
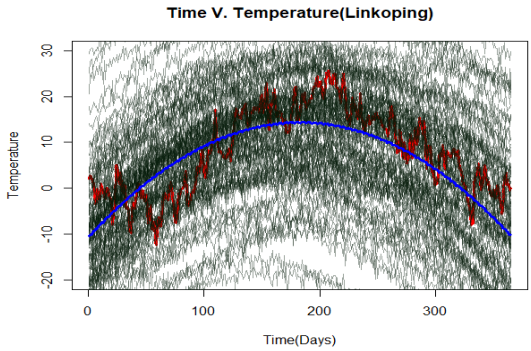
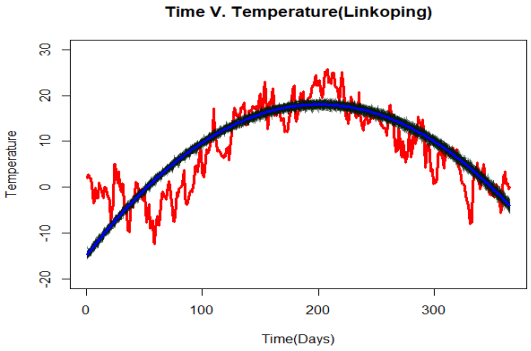


Figure 1: Daily average temperatures (in degree Celcius) at Malmslätt, Linköping over the course of the year 2018



(a) Default Hyper-Parameters



(b) $\mu_0 = (-15, 120, -109)^T$, $\Omega_0 = 0.1 \cdot I_3$, $\nu_0 = 10, \sigma^2 = 0.1$

Figure 2: Plot of Linear Regression Model using conjugate priors

On visual inspection of **Figure 1**, we postulate a prior belief that the temperature in Linköping is highest during June-July and continues to decrease till Feb. Then, there is again a trend of Temperature rising from Feb to June. In addition, we plot the resultant *temp* by simulating draws from the joint prior of all the parameters and for every draw compute the regression curve in **Figure 2(a)**. In **Figure 2(a)** it is observed that the resultant regression curve draws have a high variation and are not in accord with the established prior belief especially during summer season. Thus, we try several manual configurations of the hyper-parameters to update the regression curve closest to out prior belief. So, after manual tuning of hyper-parameters to $\mu_0 = (-15, 120, -109)^T$, $\Omega_0 = 0.1 \cdot I_3$, $\nu_0 = 10, \sigma^2 = 0.1$, in **Figure 2(b)** we are able to create regression curves that are visually able to much better

represent the prior beliefs.

1.2 (b)

1.2.1 i.

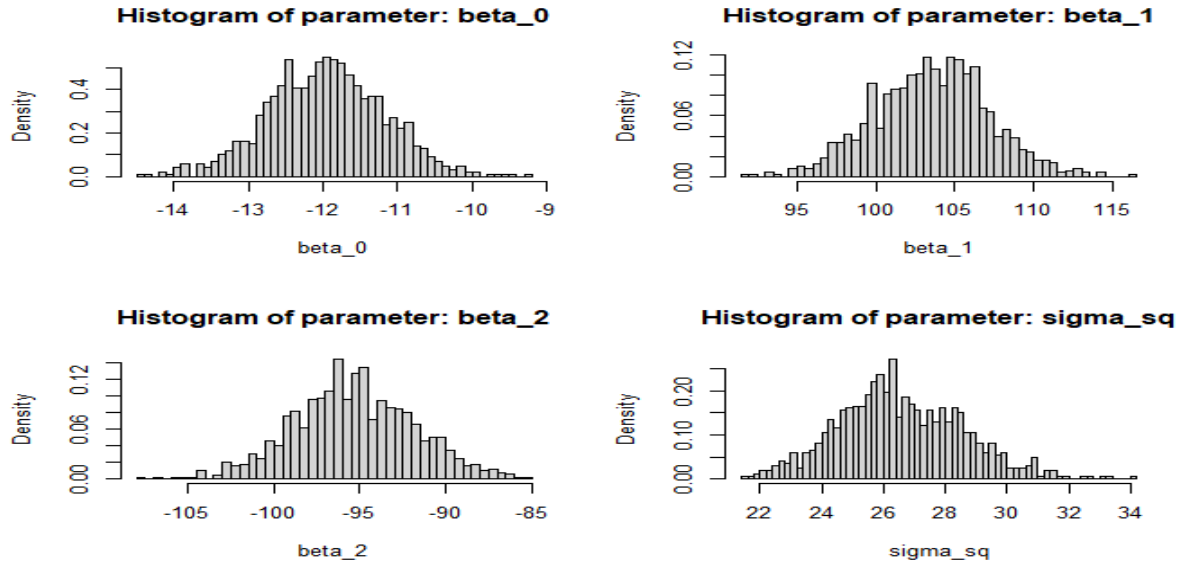


Figure 3: Histogram of $\beta_0, \beta_1, \beta_2$ and σ^2

Using (1) and (2) as the joint prior for β and σ^2, the posterior is given as (3) and (4). Using the results from (3) and (4), we simulate `nDraws` samples using the `R` function `plot_posterior_data`. The results of the histogram are presented in **Figure 3**.

$$\beta|\sigma^2 \sim \mathcal{N}_3(\mu_{\mathrm{n}}, \sigma^2\Omega_{\mathrm{n}}^{-1}) \tag{3}$$

$$\sigma^2 \sim \mathrm{Inv}-\chi^2(\nu_{\mathrm{n}}, \sigma_{\mathrm{n}}^2) \tag{4}$$

where,

$$\mu_n = (X'X + \Omega_0)^{-1}\left(X'X\hat{\beta} + \Omega_0\mu_0\right) \tag{5}$$

$$\Omega_n = X^{'}X + \Omega_0 \tag{6}$$

$$\nu_n = \nu_0 + n \tag{7}$$

$$\nu_n\sigma^2 = \nu_0\sigma_0^2 + (y'y + \mu_0^{'}\Omega_0\mu_0 - \mu_n^{'}\Omega_n\mu_n) \tag{8}$$

```
plot_posterior_data = function(data, v0,data_var_0,
omega0, mu0,nDraws = 1000){
  X = data.frame(c(1, data['time'], data['time']**2))
  n  = dim(X)[1] # No of observations
```

```
D = dim(X)[2] # No of covariates
X_T = data.frame(t(X))
XTX = as.matrix(X_T)%*% as.matrix(X)
target = as.matrix(x = data['temp'], ncol=1)
YTY = t(target) %*% target
# Beta
omega_o = omega0 * diag(3)
omega_o_inv = solve(omega_o)
omega_n = XTX + omega_o
mu_o = matrix(data = mu0, ncol = 1)
# variance
vo = v0
v_n = vo + n
data_var_o = data_var_0
nDraws = nDraws
data_var = vo*data_var_o/rchisq(nDraws, df = vo)

# Beta|sigma^2
beta_prior = matrix(nrow = nDraws, ncol=3)
for(draw in 1:nDraws){
  beta_prior[draw, ] = rmvnorm(1, mean =
  c(-10,100,-100), sigma = data_var[draw]* omega_o_inv)
}
### Prior Beta
beta = as.matrix(beta_prior, ncol = 3)
# Y_hat
error = rnorm(nDraws, 0, sd = sqrt(data_var))
y_hat = as.matrix(X) %*% t(beta) + error

# Calculating Parameters
XTX_beta_hat =  as.matrix(X_T)%*%target
var1 = solve(XTX + omega_o)
var2 = XTX_beta_hat+(omega_o%*% mu_o)
mu_n = var1%*%var2

omega_n = XTX + omega_o
omega_ni = solve(omega_n)

v_n = vo + n

sigma_n2 = ((vo*data_var_o) + t(target)%*%target +
(t(mu_o)%*%omega_o%*%mu_o) - (t(mu_n)%*%omega_n%*%mu_n))/v_n
sigma_n2 = sigma_n2[1]

# sigma_2 posterior
```

```
  sigma2_post = matrix((v_n*sigma_n2)/rchisq(nDraws, df = v_n),
  ncol = 1)
  beta_dist = matrix(nrow = nDraws, ncol = D)

  # Beta Posterior
  for(i in 1:nDraws){
    beta_dist[i,] = rmvnorm(1,
    mean = mu_n, sigma = sigma2_post[1]*omega_ni)
  }

  dist_data = as.matrix(cbind(beta_dist, sigma2_post))

  par(mfrow = c(2,2))
  names = c("beta_0", "beta_1", "beta_2", "sigma_sq")
  for(i in 1:4){
    hist(x = dist_data[,i], xlab = names[i],
    breaks = 50,
    main = paste0("Histogram of parameter: ", names[i]), freq=F)
  }

  return(dist_data)
}

posterior_params = plot_posterior_data(data = data, v0 = 4,
data_var_0 = 1, omega0 = 0.01, mu0 = c(-10,100,-100), nDraws = 1000)
```

1.2.2 ii.

Using the generated samples of $\beta_0, \beta_1, \beta_2$ stored in `posterior_beta_draws`, we calculate the regression function *f(time)*=$\beta_0 + \beta_1 \cdot time + \beta_2 \cdot time^2$. Then, using the simulated values of *f(time)* for each time-stamp in the variable `f_time`, its 95% credible interval i.e., interval region from 2.5% to 7.5% is calculated along with the median. The analytical results are graphically presented in **Figure 4**. The `R` implementation of the same is stated below:

```
## (b)(ii)
X = data.frame(c(1, data['time'], data['time']**2))
XT = t(X)
posterior_beta_draws = posterior_params[,1:3] # Draws of Betas
#Calculate f(time)
f_time  = posterior_beta_draws %*% XT
# Calculate median for each time-stamp's draw
median = apply(f_time, 2, median)
# Calculate 95% credible region for each time-stamp
prediction_intervals = apply(f_time, 2 , quantile, probs = c(0.025,0.975))
par(mfrow = c(1,1))
```

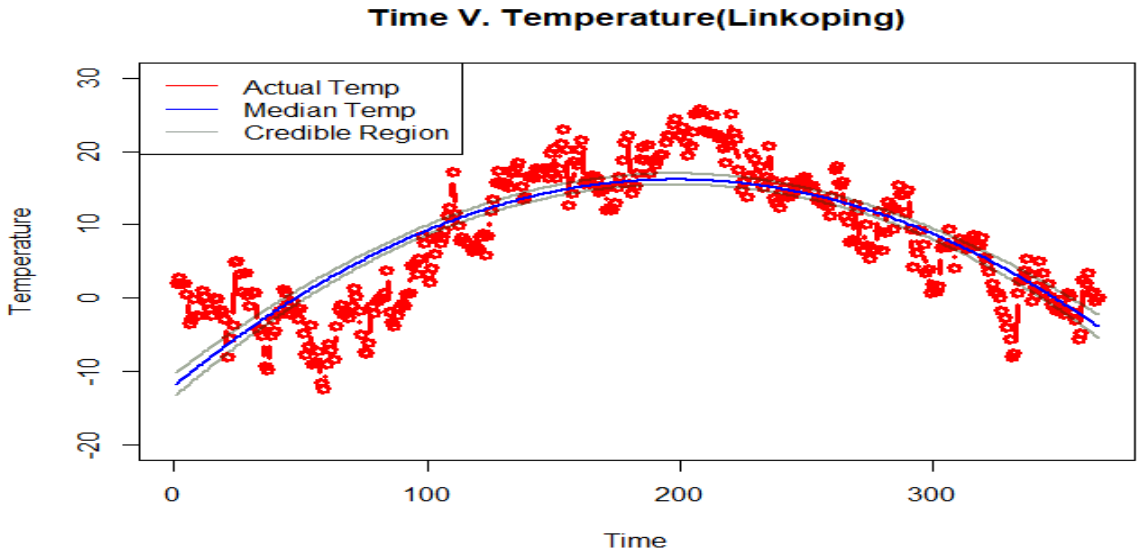


Figure 4: Scatter plot of the temperature data and overlay curve for the posterior median and credible interval of the regression function $f(time)=\beta_0+\beta_1\cdot time+\beta_2\cdot time^2$

```
plot(y = data$temp, x=1:365 ,type = "b", pch = 1,  col = "red", ylim = c(-20
     lwd = 3,
     xlab = "Time",ylab = "Temperature",
     main = "Time V. Temperature(Linkoping)")
for(i in 1:2){
  lines(x = 1:365, y = prediction_intervals[i,], type="l",
        col = rgb(0.1,0.2,0.03,0.4), lwd = 2)
}
lines(x = 1:365, y = median, col="blue", lwd=2)
legend("topleft",
       legend = c("Actual Temp", "Median Temp",
       "Credible Region"), col = c("red", "blue", rgb(0.01,0.1,0.03,0.4)),
       lty = c(1,1,1))
```

We observe that most of the data points are not contained in the posterior probability interval. It is because the posterior distribution assumes *temp* to have a quadratic polynomial distribution along the covariates and the assumed distribution does not generalize on the data. Therefore, we can say that the posterior intervals should not necessarily contain all the data points.

1.3 (c)

We devise a sampling process to find the time with the highest expected temperature. Now, having drawn `nDraws=1000` samples of *f(time)* for independent draws of β_0, β_1 and β_2, we use a sample based process of finding the maxima. For that, we sample the time-stamp(Day) for which the *temp* is maximum for each draw of 365 days. We plot a histogram of the sampled

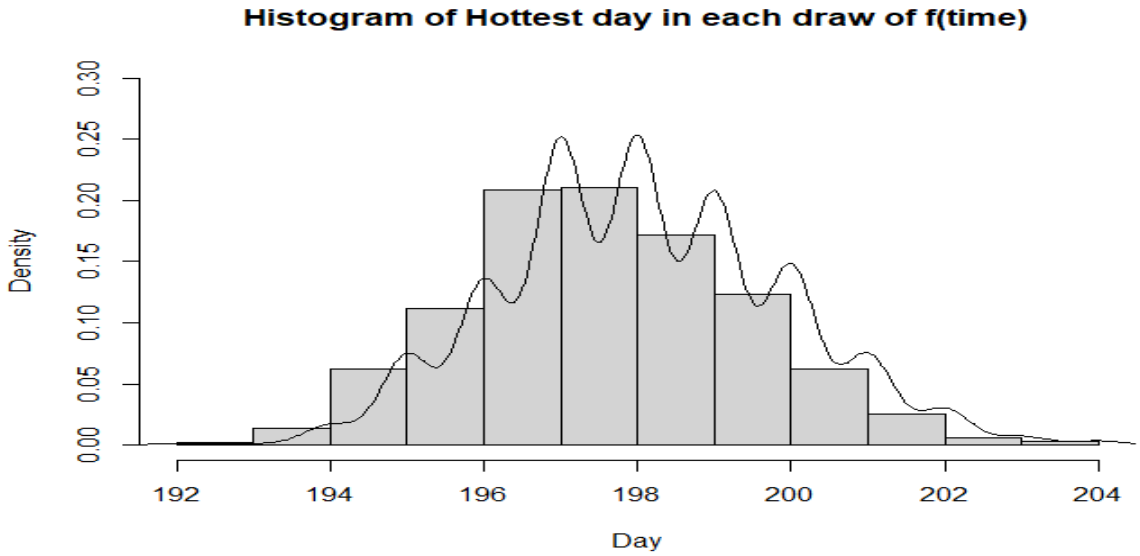


Figure 5: Hottest day in each draw of *f(time)*

dates and find that 198^{th} day i.e., July 16 or July 17 is the hottest day.

Also, we can verify the same using results from (9). It is to be noted that *time* is maximum for the given *f(time)* only if $2 \cdot \beta_2 < 0$. So, in R, we calculate the value of $-\frac{\beta_1}{2 \cdot \beta_2}$ for each posterior draw of β values and take a mean of all the values to get an expected value of *time* at which *f(time)* is maximum. Our analytical analysis reveals that *time = 198*. Thus corroborating with the findings using the sampling technique.

$$
\begin{aligned}
& f(time) = \beta_0 + \beta_1 \cdot time + \beta_2 \cdot time^2 + \epsilon \\
& \Rightarrow \frac{df}{d(time)} = 0 \ when \ time = \frac{-\beta_1}{2 \cdot \beta_2} \\
& Taking \ derivative \ of \ second \ order, \ \frac{d^2 f}{d^2(time)} = 2 \cdot \beta_2
\end{aligned} \tag{9}
$$

```
#(c)
# Drawing the samples of hottest days
d = density(apply(f_time, 1, which.max))
hist(apply(f_time, 1, which.max), freq = F, ylim = c(0,0.3),
     xlab="Day",
     main = "Histogram of Hottest day in each draw of f(time)") # Close to Ju
lines(x = d$x, y  = d$y)

# Analytically calculating the hottest days from f(time)
b1 = posterior_beta_draws[,2]
b2 = posterior_beta_draws[,3]
mean((-b1)/(2*b2))
```

1.4 (d)

In the current problem, the objective is to estimate a polynomial regression of order 7. Given that higher order terms are not needed, we want to eliminate the terms that lead to overfitting. To resolve this issue we propose a *Laplace* prior for β_i i.e. $\beta_i|\sigma^2$ *Laplace* $\left(0, \frac{\sigma^2}{\lambda}\right)$. In doing so, the coefficients for many predictors will become zero. The rationale behind choosing the Laplacian prior is that 1) We have many predictors and 2) Data size is small to moderate.[1, p.369]

2 Posterior approximation for classication with logistic regression

We are provided with the `WomenWork.dat` dataset that contains $n = 200$ observation of `Work` response variable and eight features(including 1 as a model intercept).

2.1 (a)

Definition 3 (*Logistic Regression Model*)

$$\Pr(y = 1|\mathbf{x}) = \frac{exp(\mathbf{x}^T\beta)}{1 + exp(\mathbf{x}^T\beta)}$$

$$\beta|\mathbf{y}, \mathbf{x} \sim \mathcal{N}\left(\tilde{\beta}, \mathrm{J}_{\mathbf{y}}^{-1}(\tilde{\beta})\right) \tag{10}$$

We define the logistic regression model where the event y=1 represents a working woman and 0 if she does not. $\mathbf{x}$ is an 8-dimension vector of covariates. The posterior distribution of the parameter vector β is sampled from the multivariate normal distribution using (10) where $\tilde{\beta}$ is the posterior mode and $\mathrm{J}(\tilde{\beta}) = -\dfrac{\partial^2 lnp(\beta|\mathbf{y})}{\partial\beta\partial\beta^T}$ is the negative of the hessian evaluated at the posterior mode i.e., at $\beta = \tilde{\beta}$. We generated β values using the prior distribution of β as defined in (12) and Log-likelihood in (11), we find the distribution of Log-posterior β using Bayes Theorem as implemented in `R` function `logPostLogistic`.

$$\mathrm{Log}\left(\mathrm{p}(\mathrm{y}|\mathbf{x}, \beta)\right) = \sum_{n=1}^{N} \left(x_i^T\beta\right)^{y_i} - \left(1 + exp(x_i^T\beta)\right) \tag{11}$$

$$\beta \sim \mathcal{N}_8(\mathbf{0}, \tau^2 I) \tag{12}$$

With reference to the code as presented in Lecture-6, we implement an optimizer based program to analytically calculate the values of $\tilde{\beta}$ and $\mathrm{J}_{\mathbf{y}}^{-1\tilde{\beta}}$ for the `WomenWork` data. The results are listed in **Table 1** and **Table 2**.

```
##### Q 2 #####
## (a)
library(mvtnorm)
WomenWorkData =read.table(file = "WomenWork.dat", header = 1)
```

	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7
$\tilde{\beta}_i$	0.626	-0.019	0.18	0.167	-0.14	-0.08	-1.36	-0.02

Table 1: Numerical Values of Posterior Mode $\tilde{\beta}$

	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7
β_0	2.26	0.0033	-0.065	-0.01	0.045	-0.03	-0.18	-0.09
β_1		0.00025	-0.00056	-0.00003	0.00014	-0.00004	0.00051	-0.00014
β_2			0.00622	-0.00036	0.0019	0	-0.0061	0.00175
β_3				0.00435	-0.0142	-0.00013	-0.00147	0.00054
β_4					0.055	-0.00033	0.0032	0.00051
β_5						0.00072	0.00518	0.001
β_6							0.15	0.0067
β_7								0.019

Table 2: Matrix of $\mathrm{J}_{\mathbf{y}}^{-1}(\tilde{\beta})$

```
covs= c(2:9) # Select which covariates/features to include
tao = 10 # scaling factor for the prior of beta
Nobs = dim(WomenWorkData)[1]
y = WomenWorkData$Work # Target Vector
X = as.matrix(WomenWorkData[,covs]) # Co-variate Matrix
Xnames=colnames(X)
Npar = dim(X)[2] # No. of Parameters
# Setting up Prior
mu = as.matrix(rep(0,Npar))  # Prior mean vector
Sigma = 100*diag(Npar) # Prior covariance matrix
# Functions that returns the log posterior for the logistic regression.
logPostLogistic <- function(betas, y,X, mu,Sigma){
  set.seed(2020-04-19)
  linPred = X%*%betas
  logLik = sum( linPred*y - log(1+exp(linPred) ) )
  logPrior = dmvnorm(betas, mean = mu, Sigma, log = TRUE)
  return(logLik + logPrior)
}
# Initialize Betas
initVal = matrix(0,Npar,1)
OptimRes <- optim(initVal,
logPostLogistic,gr=NULL,y,X,mu,Sigma,method=c("BFGS"),
control=lis(fnscale=-1),hessian=TRUE)

#Printing the results to the screen
names(OptimRes$par) <- Xnames # Naming the coefficient by covariates
 # Computing approximate standard deviations.
```

```
approxPostStd <- sqrt(diag(-solve(OptimRes$hessian)))
names(approxPostStd) <- Xnames # Naming the coefficient by covariates
print('The posterior mode is:')
##### Numerical Values of Beta @ Mode
print(OptimRes$par)
print('The approximate posterior standard deviation is:')
approxPostStd <- sqrt(diag(-solve(OptimRes$hessian)))
print(approxPostStd)
# Hessian Matrix
hessian = OptimRes$hessian
#solve(-hessian) == -solve(hessian)
# Neg Inv Hessian
posterior_var_mat = -solve(hessian)
# Beta draws
betas = rmvnorm(1000, mean = OptimRes$par, sigma = posterior_var_mat)
# 95% Equal Tail Posterior Probability for 'NSmalChild' variable
quantile(betas[,7], probs = c(0.025,0.975))
# Compare Beta Vals with MLE
glm_model = glm(Work ~ 0 + ., data = WomenWorkData, family = binomial)
round(glm_model$coefficients - OptimRes$par, digits = 2)
```

2.2 (b)

We use the normal approximation to the posterior as defined in (a) to simulate 1000 draws from distribution $\Pr(y = 1|\mathbf{x})$ The `R` function `sim_draws` implements the same and the plot of the event $\Pr(y = 1|\mathbf{x})$ for the provided values of $\mathbf{x}$ covariates is presented in **Figure 6**. We observe that the distribution is highly skewed towards 0. Thus, suggesting that it is improbable for a 37 year old woman with two children, 8 years of education, 11 years of experience and a husband with an income of 13 to work.

2.3 (c)

From the `nDraws=1000` sampled draws of the event $Pr(y = 1|\mathbf{x})$ for the provided values of $\mathbf{x}$, we assess the expected value of the event $Pr(y = 1|\mathbf{x})$ by treating each result as an independent draw. Using the resultant $E[Pr(y = 1|\mathbf{x})] = 0.046$, we present the plot of Binomial distribution of no. of women likely to work out of 8 women. The results are showd in **Figure 7**. From the figure we can comment that either no, one or two women out of 8 are likely to work.

```
### c)
# Expected P(y=1|x)
theta = mean(p_y)
hist(rbinom(1000, 8, p=theta), freq=F,
     main="", xlab="No of working women(Out of 8)")
```

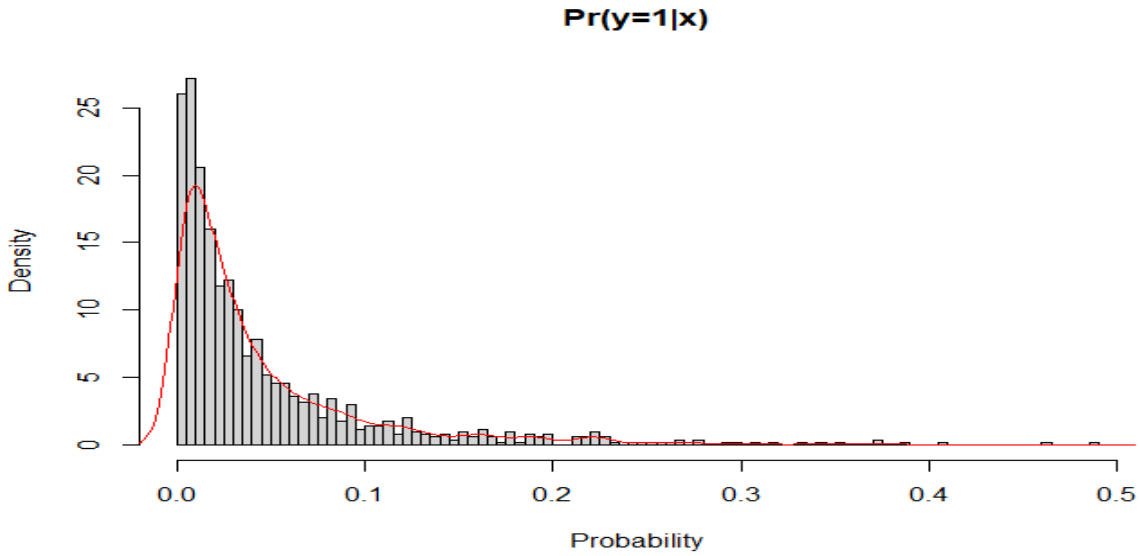


Figure 6: Distribution of event $\Pr(y = 1|\mathbf{x})$ on 1000 draws

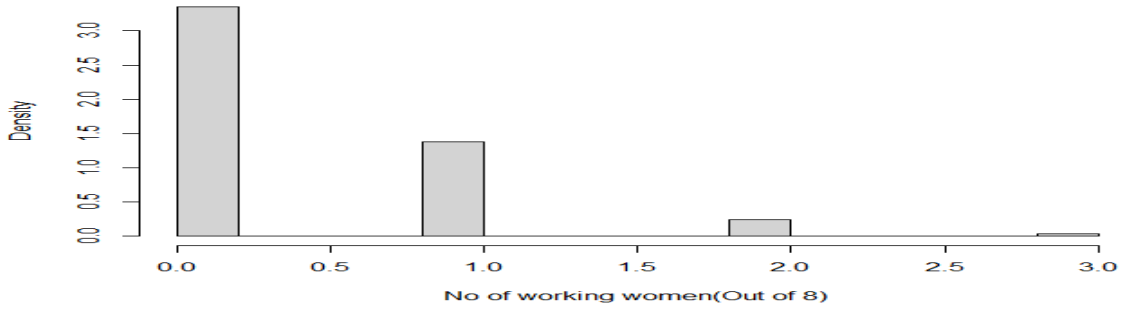


Figure 7: Histogram of Working Women(Out of 8)

References

[1] Andrew Gelman et al. *Bayesian data analysis*. CRC press, 2013.