Computer Lab2_Bayesian Learning

zahra jalilpour

2021-04-30

Question1: Linear and polynomial regression

The dataset TempLinkoping.txt contains daily average temperatures (in degree Celcius) at Malmslätt, Linköping over the course of the year 2018. The response variable is temp and the covariate is

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

A Bayesian analysis of the following quadratic regression model is to be performed:

$$temp = \beta_0 + \beta_1 \cdot\ time + \beta_2 \cdot\ time^2 + \varepsilon \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$$

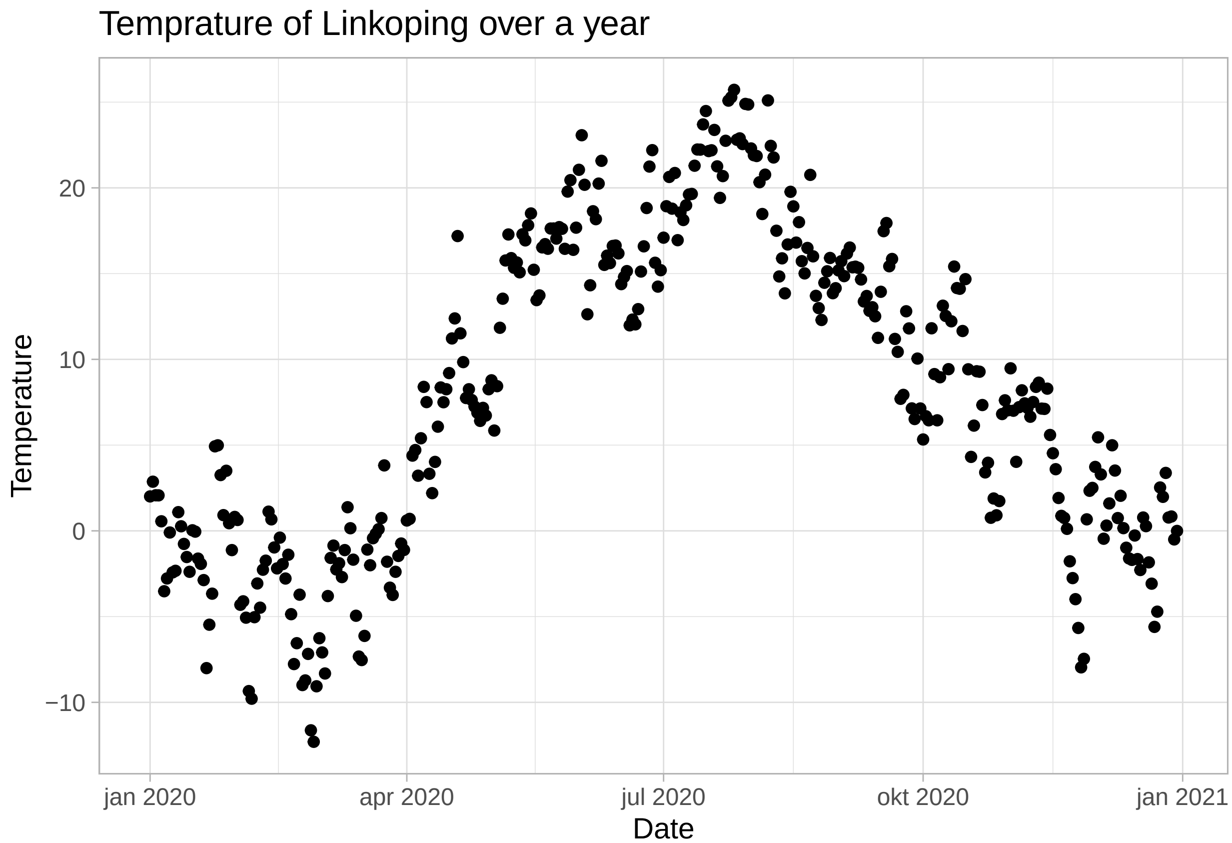
a)

Use the conjugate prior for the linear regression model. The prior hyperparameters μ_0, Ω_0, ν_0 and σ_0^2 shall be set to sensible values. Start with $\mu_0 = (-10, 100, -100)^T$, $\Omega_0 = 0.01 \cdot I_3$, $\nu_0 = 4$ and $\sigma_0^2 = 1$. 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. Does the collection of curves look reasonable? If not, change the prior hyperparameters until the collection of prior regression curves agrees with your prior beliefs about the regression curve. [Hint: the R package mvtnorm will be handy. And use your $Inv - \chi^2$ simulator from Lab 1.]

At first we take a look at our data, create a new column named date and convert the time to date by presumption that the data is related to 2020. From the plot we can see temperature condition over the year.

```r
setwd("D:/Linkoping university/second semester/second/bayesian learning/lab/lab2")
temp_data<-read.table("TempLinkoping.txt", header = TRUE)
temp_data$date<- as.Date(x=round(temp_data$time*365), origin = "2019-12-31")
df = data.frame(temperature=temp_data$temp, date=temp_data$date)
```

```r
ggplot(temp_data)+
  geom_point(aes(x=date,y=temp), color="black",fill="#dedede")+
  labs(title="Temprature of Linkoping over a year", y="Temperature", x="Date",color="Legend")+
  theme_light()
```



Now for this part we should use conjugate priors for linear regression model: Joint prior for β and σ^2:

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

\

$$\sigma^2 \sim Inv - \chi^2(\nu_o, \sigma_0^2)$$

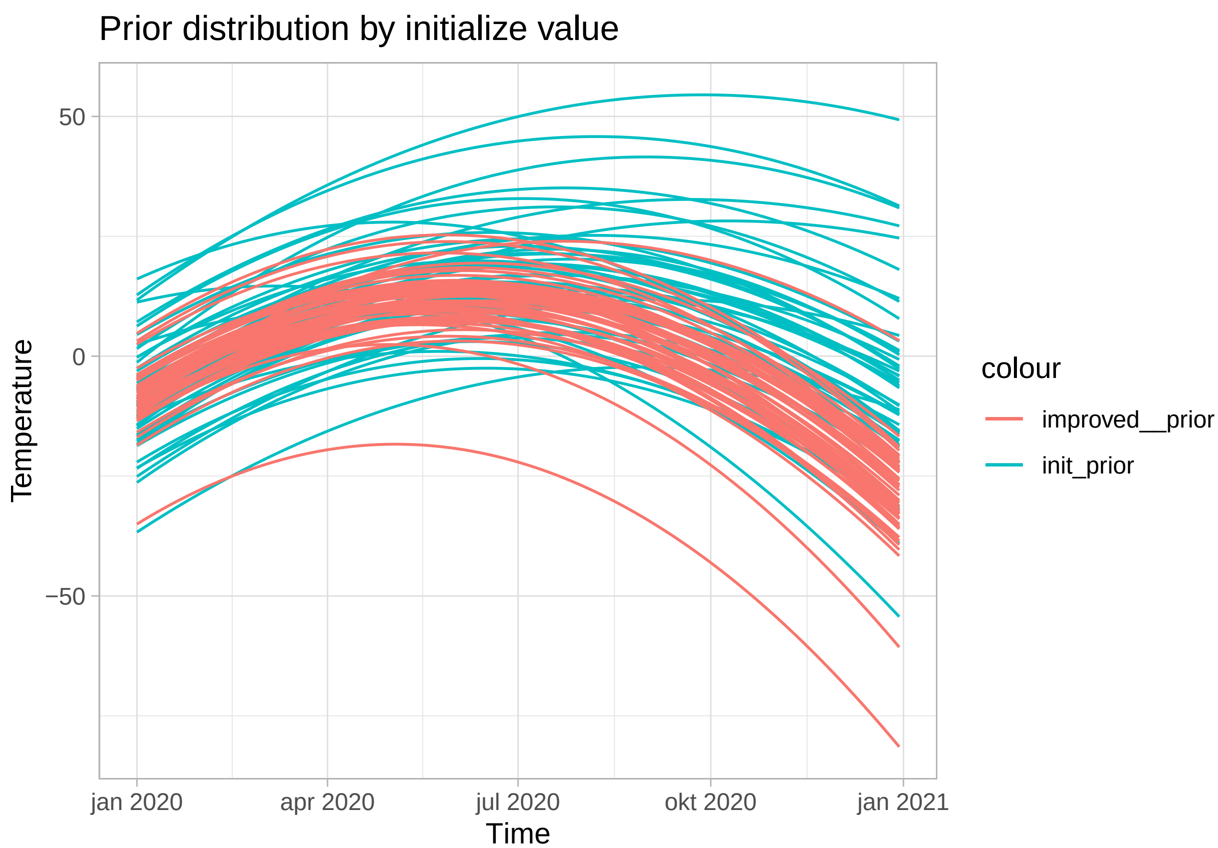
And the posterior distribution is :

$$\beta|\sigma^2, y \sim N(\mu_n, \sigma^2 \Omega_n^{-1})$$
$$\sigma^2|y \sim Inv - \chi^2(\nu_n, \sigma_n^2)$$
$$\Omega_0 = \lambda I_n$$
$$\mu_n = (X^T X + \Omega_0)^{-1}(X^T X\hat{B} + \Omega_0 \mu_0)$$
$$\Omega_n = X^T X + \Omega_0$$
$$\hat{\beta} = (X^T X)^{-1} X^T Y$$
$$\nu_0 = \nu_0 + n$$
$$\sigma_n^2 = \frac{\nu_0 \sigma_0^2 + (YY^T + \mu_0^T \Omega_0 \mu_0 - \mu_n^T \Omega_n \mu_n)}{\nu_n}$$

At first we should draw σ^2 from $Inv - \chi^2(\nu_0, \sigma_0^2)$ Then draw β from $\mathcal{N}(\mu_0, \sigma^2 \Omega_0^{-1})$ Finally Calculate the prior according to $\hat{y} = X\beta$ where $X = (1, \text{temp}, \text{temp}^2)$. The first 1 is the intercept.

```r
# initial values:
mu_0 <- c(-10, 100,-100)
omega_0 <-0.01 * diag(3)
nu_0 <- 4
sigma_sq_0 <- 1


time = temp_data$time
temp = temp_data$temp
X = matrix(c(rep(1, length(time)), time, time^2), ncol = 3)
Y = matrix(temp)
n = length(time)
prior_init <- data.frame("Date"=time, "y_hat"=0)
sigma_sq <- LaplacesDemon::rinvchisq(1, df=nu_0, scale= sigma_sq_0 )
beta <-mvrnorm(n=1, mu=mu_0, Sigma = sigma_sq* solve(omega_0))
prior_init$y_hat <-X %*% matrix(beta)
prior_init$Date<- as.Date(x=round(prior_init$Date*365), origin = "2019-12-31")
plot <- ggplot(data=prior_init, aes(x=Date, y= y_hat, col="init_prior"))+
  geom_line()+
  ggtitle("Prior distribution by initialize value") + ylab("Temperature") + xlab("Time")+
  theme_light()
```

```r
for( i in 1: 50){
  sigma_sq <- LaplacesDemon::rinvchisq(1, df=nu_0, scale= sigma_sq_0 )
  beta <-mvrnorm(n=1, mu=mu_0, Sigma = sigma_sq* solve(omega_0))
  prior_df <- data.frame("Date"=time, "prior"=0)
  prior_df$Date<- as.Date(x=round(prior_df$Date*365), origin = "2019-12-31")
  prior_df$prior <-X %*% matrix(beta)
  plot <- plot+  geom_line(data= prior_df, aes(x=Date, y=prior, col="init_prior"))


}


mu_0 <-c(-8, 100,-120)
omega_0 <-0.5 * diag(3)
nu_0 <- 3
sigma_sq_0 <- 5

for( i in 1: 50){
  sigma_sq <- LaplacesDemon::rinvchisq(1, df=nu_0, scale= sigma_sq_0 )
  beta <-mvrnorm(n=1, mu=mu_0, Sigma = sigma_sq* solve(omega_0))
  df <- data.frame("Date"=time, "improved_prior"=0)
  df$Date<- as.Date(x=round(df$Date*365), origin = "2019-12-31")
  df$improved_prior <-X %*% matrix(beta)
  plot <-plot+  geom_line(data= df, aes(x=Date, y=improved_prior, col="improved__prior"))


}
plot
```



At first, we change the value of μ_0, and we can see in some cases , temperature in January is high. the we change the value of ν_0 and σ_0^2, the best value in our experiment is $\mu_0 = (-8, 100, -120)$, $\nu_0 = 3$, $\Omega_0 = 0.5 \cdot I_3$ and $\sigma_0^2 = 5$, here we see the variance is low and temperature is logically related to the months.

b)

Write a function that simulate draws from the joint posterior distribution of $\beta_0, \beta_1, \beta_2$ and σ^2.

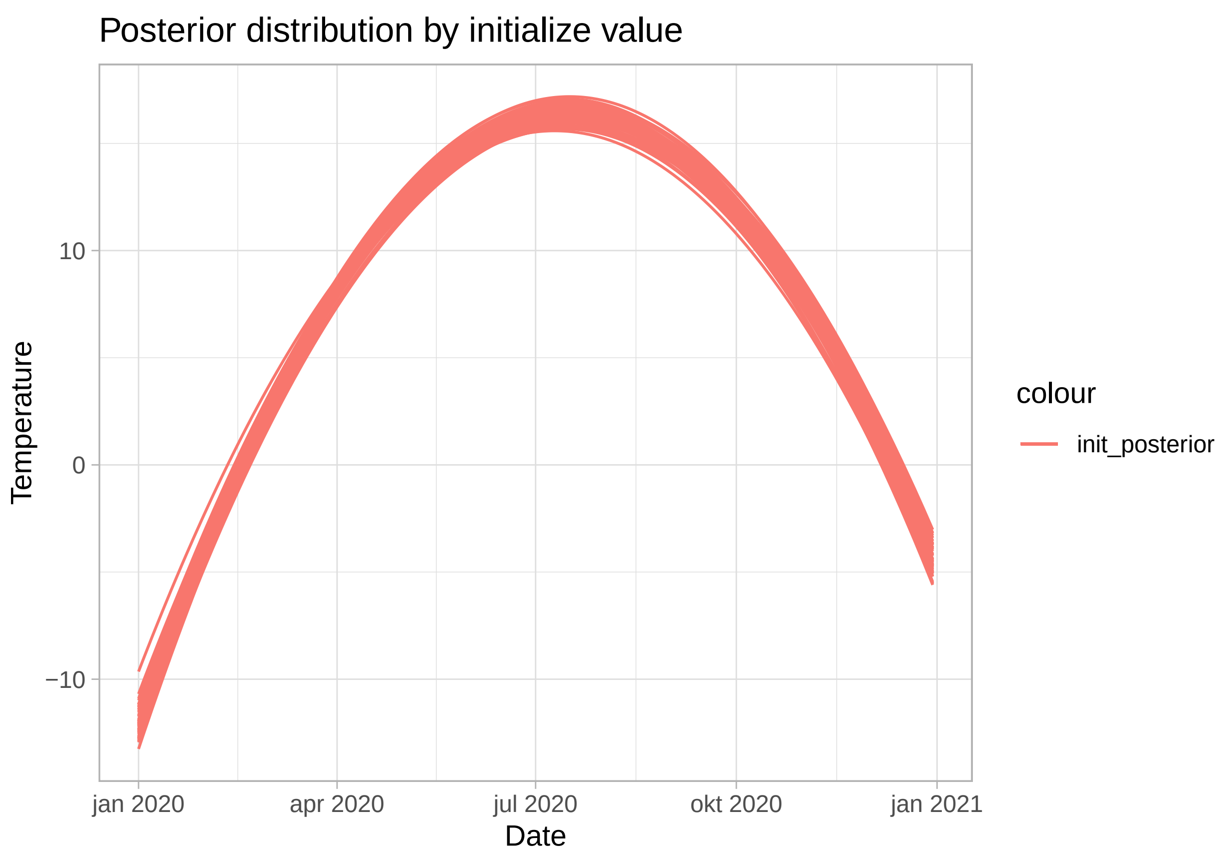
i)

Plot a histogram for each marginal posteriors of the parameters .

Here at first we should find $\hat{B}$ to put it's value in joint posterior distribution to draw from it.

```
beta_hat <- (solve(t(X)%*%X))%*% t(X)%*% Y
mu_n <- solve(t(X)%*%X + omega_0) %*% (t(X)%*%X %*% beta_hat + omega_0 %*% mu_0)
omega_n <- t(X)%*%X + omega_0
nu_n <- nu_0 + n
nu_sigma2_n <-  nu_0 * sigma_sq_0 + (t(Y)%*% Y + t(mu_0) %*%omega_0 %*% mu_0  - t(mu_n)%*% omega_n %*% r
sigma_sq_n <- as.numeric(nu_sigma2_n/ nu_n)

sigma_sq <- LaplacesDemon::rinvchisq(n=1, df=nu_n, scale= sigma_sq_n )

betas <-mvtnorm::rmvnorm(n=1, mean=mu_n, sigma = sigma_sq* solve(omega_n))

posterior_df <- data.frame("Date"=time, "init_posterior"=0)
```

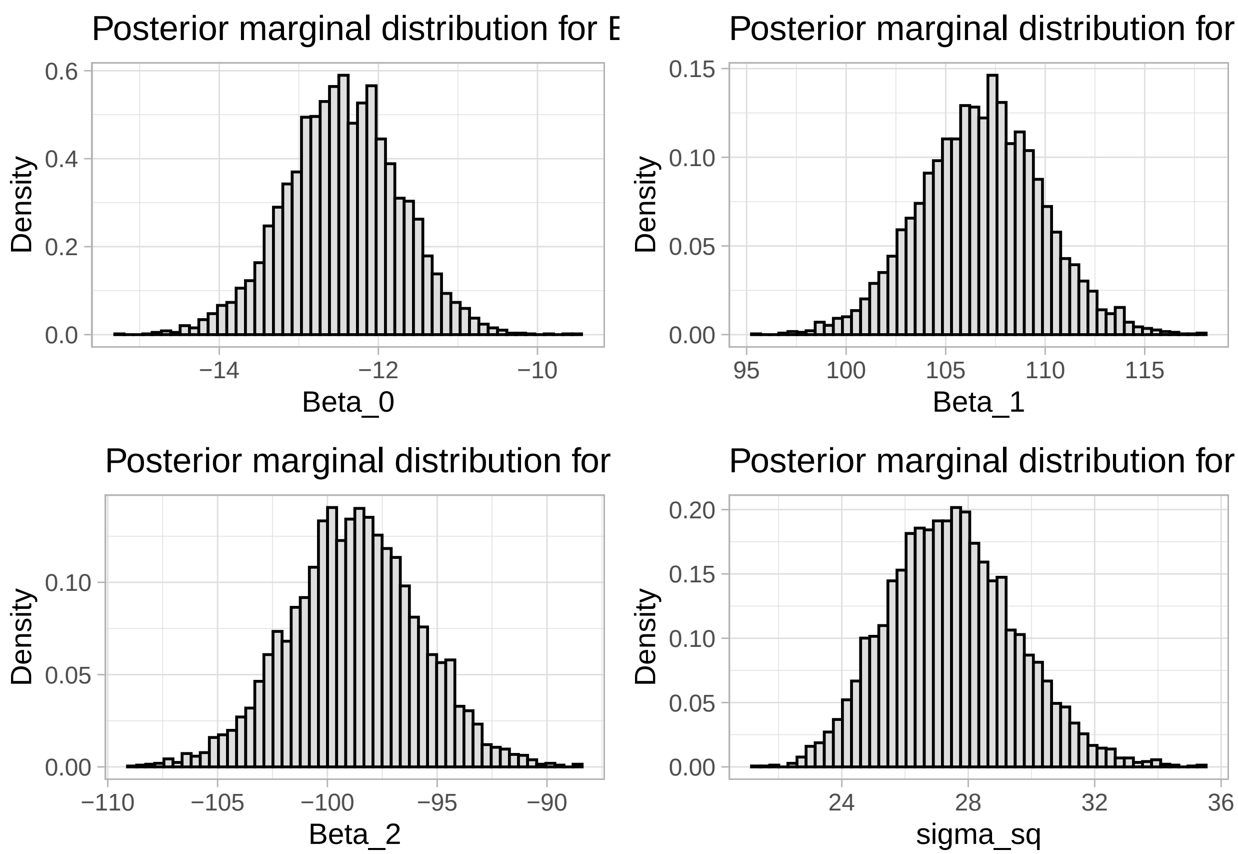
```r
posterior_df$Date<- as.Date(x=round(posterior_df$Date*365), origin = "2019-12-31")
posterior_df$init_posterior <- X %*% matrix(betas)

plot_posterior <- ggplot(data=posterior_df, aes(x=Date, y= init_posterior, col="init_posterior"))+
  geom_line()+
  ggtitle("Posterior distribution by initialize value") + ylab("Temperature") + xlab("Date")+
  theme_light()

for(i in 1:50){
  sigma_sq <- LaplacesDemon::rinvchisq(n=1, df=nu_n, scale= sigma_sq_n )
  betas <-mvtnorm::rmvnorm(n=1, mean=mu_n, sigma = sigma_sq* solve(omega_n))
  df <- data.frame("Date"=time, "posterior"=0)
  df$Date<- as.Date(x=round(df$Date*365), origin = "2019-12-31")
  df$posterior <- X %*% matrix(betas)
  plot_posterior <- plot_posterior + geom_line(data= df, aes(x=Date, y=posterior, col="init_posterior"))

}
plot_posterior
```



ii)

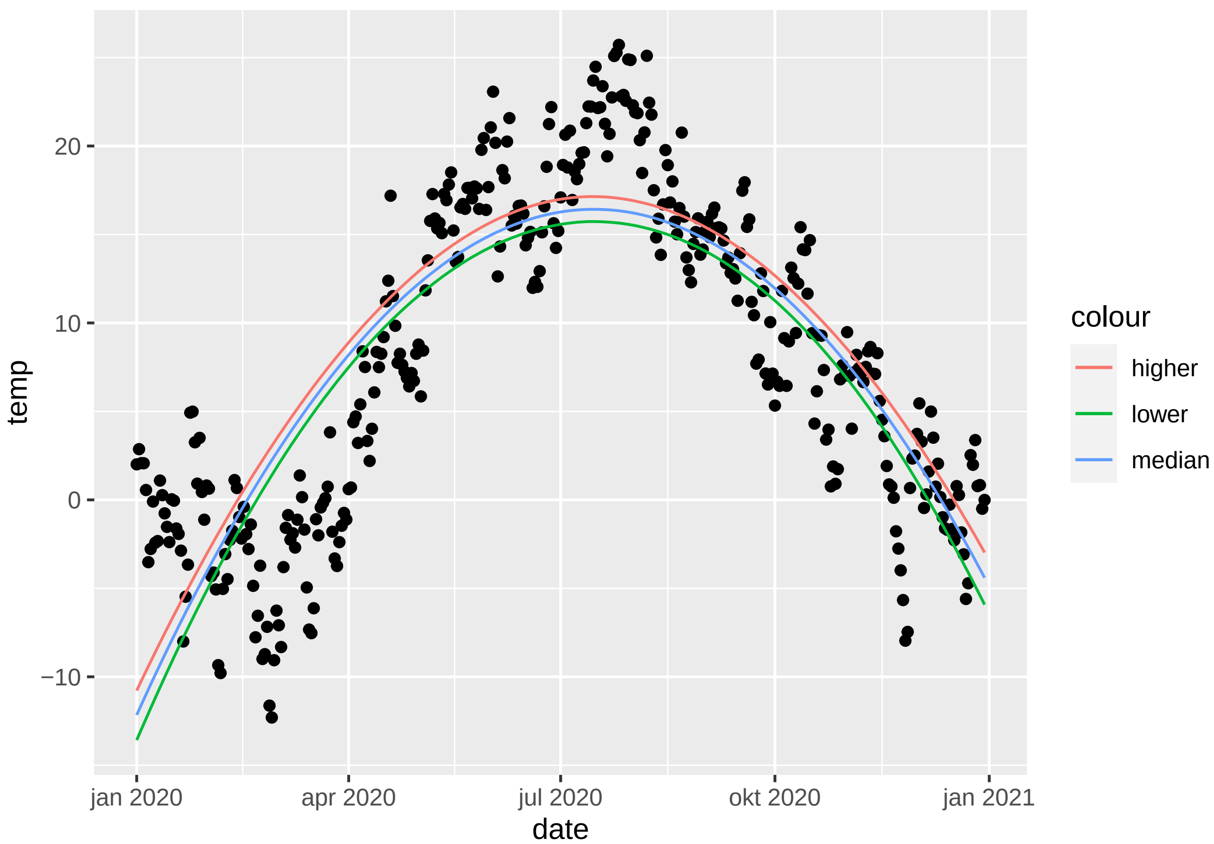
Make a scatter plot of the temperature data and overlay a curve for the posterior median of the regression function $f(\ time) = \beta_0 + \beta_1 \cdot \ time + \beta_2 \cdot \ time^2$, i.e. median is computed for every value of *time*. In addition overlay curves for the 95% equal tail posterior probability interval for $f(\ time)$. i.e. the 2.5 and 97.5 posterior percentiles is computed for every value of time. Does the posterior probability intervals contain most of the data points? Should they?

```
library(gridExtra)
new_df <- data.frame("Beta_0"=0, "Beta_1"=0, "Beta_2"=0, "Sigma_2"=0)
for(i in 1:5000){
  sigma_sq_p<- LaplacesDemon::rinvchisq(n=1, df=nu_n, scale= sigma_sq_n )
  betas_p<-mvtnorm::rmvnorm(n=1, mean=mu_n, sigma = sigma_sq* solve(omega_n))
  new_df[i,]<-cbind(betas_p, sigma_sq_p)
}

p0= ggplot(new_df) +
  geom_histogram(aes(x = Beta_0, y=..density..),
                 bins = 50, color = "black",
                 fill = "#DEDEDE") +
  labs(title = "Posterior marginal distribution for Beta_0",
       y = "Density", x = "Beta_0") +
  scale_color_manual("Legend", values = c("#0039C7", "#000000")) +
  theme_light()

p1= ggplot(new_df) +
  geom_histogram(aes(x = Beta_1, y=..density..),
                 bins = 50, color = "black",
                 fill = "#DEDEDE") +
  labs(title = "Posterior marginal distribution for Beta_1",
       y = "Density", x = "Beta_1") +
  scale_color_manual("Legend", values = c("#0039C7", "#000000")) +
  theme_light()

p2= ggplot(new_df) +
  geom_histogram(aes(x = Beta_2, y=..density..),
                 bins = 50, color = "black",
                 fill = "#DEDEDE") +
  labs(title = "Posterior marginal distribution for Beta_2",
       y = "Density", x = "Beta_2") +
  scale_color_manual("Legend", values = c("#0039C7", "#000000")) +
  theme_light()

p3= ggplot(new_df) +
  geom_histogram(aes(x = Sigma_2, y=..density..),
                 bins = 50, color = "black",
                 fill = "#DEDEDE") +
  labs(title = "Posterior marginal distribution for sigma_sq",
       y = "Density", x = "sigma_sq") +
  scale_color_manual("Legend", values = c("#0039C7", "#000000")) +
  theme_light()

grid.arrange(p0, p1, p2, p3, nrow=2)
```




```
new_temp <- temp_data
beta_matrix <- as.matrix(new_df[,1:3])


temp_pred <- X %*% t(beta_matrix)
new_temp$median <- 0
new_temp$lower <- 0
new_temp$higher <- 0
for(i in 1:nrow(temp_data)){
  new_temp$median[i]<- median(temp_pred[i,])
  new_temp$lower[i]<- quantile(temp_pred[i,], probs = 0.025)
  new_temp$higher[i]<-quantile(temp_pred[i,], probs = 0.975)
}
```

```
plot2 <- ggplot(new_temp, aes(x=date, y=temp))+  geom_point()
plot2 <- plot2 + geom_line(aes(x=date, y=median, col="median"))
plot2 <- plot2 + geom_line(aes(x=date, y=lower, col="lower"))
plot2 <- plot2 + geom_line(aes(x=date, y=higher, col="higher"))
plot2
```



c)

It is of interest to locate the *time* with the highest expected temperature (i.e., the *time* where $f(time)$ is maximal). Let's call this value $\tilde{x}$. Use the simulations in b) to simulate from the posterior distribution of $\tilde{x}$. [Hint: the regression curve is a quadratic polynomial. Given each posterior of β_0, β_1 and β_2.you can find a simple formula for $\tilde{x}$]

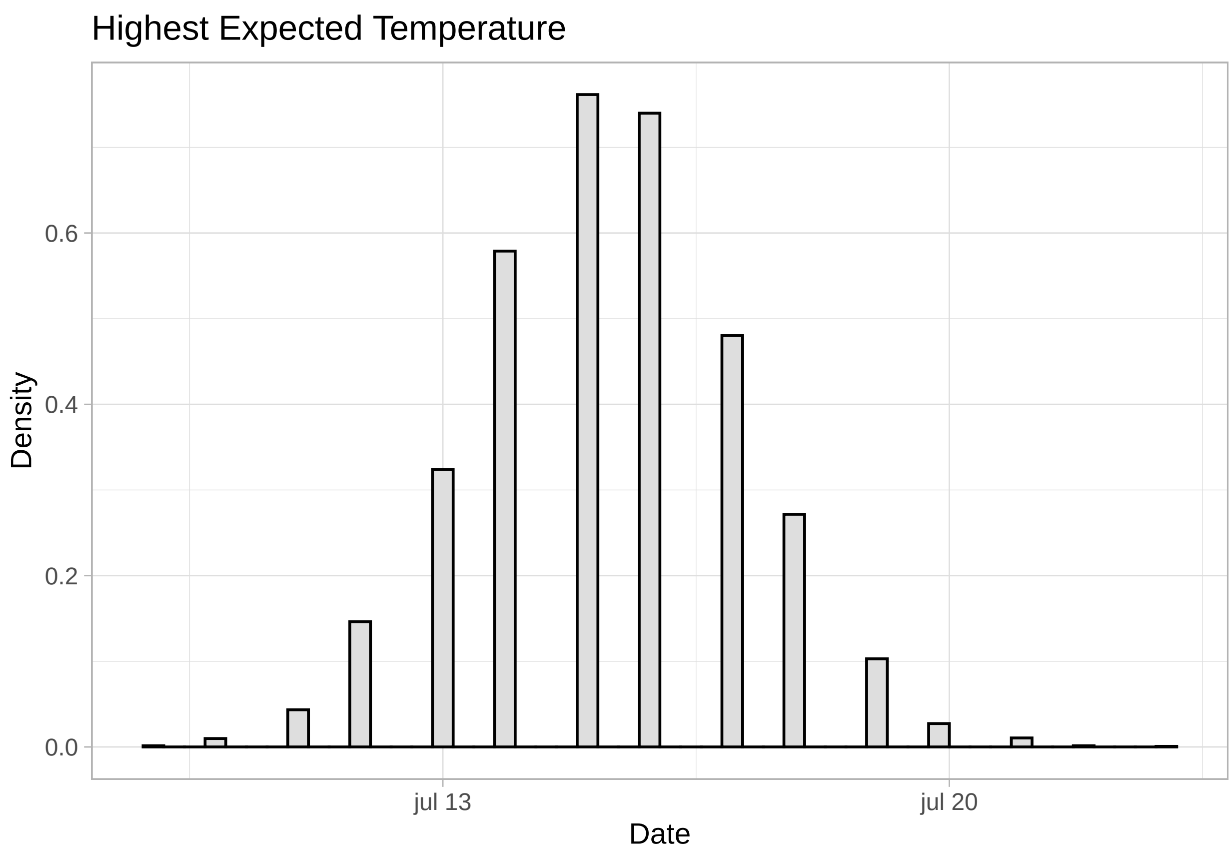
Since the given expression is quadratic the first derivate will be zero that is:

$$y = \beta_0 + \beta_1 time + \beta_2 time^2$$
$$0 = \beta_1 + 2\beta_2 time$$
$$time = -0.5 \cdot \frac{\beta_1}{\beta_2}$$

From above equation and based on posterior distribution from part b, we can obtain highest expected temperature. As we can see in the plot, Highest temperature is in(11 july to 20 july), and max temperature is on july15.

```
#max_df <- data.frame("Date"=time, )
t_hat<- -(new_df$Beta_1)/(2*new_df$Beta_2)
df <- data.frame("Date"=t_hat)
df$Date<- as.Date(x=round(df$Date*365), origin = "2019-12-31")
ggplot(df)+
  geom_histogram(aes(x = Date, y=..density..),
```

```r
                bins = 50, color = "black",
                fill = "#DEDEDE") +
  labs(title = "Highest Expected Temperature",
       y = "Density", x = "Date") +
  scale_color_manual("Legend", values = c("#0039C7", "#000000")) +
  theme_light()
```



d)

Say now that you want to estimate a polynomial regression of order 7,but you suspect that higher order terms may not be needed, and you worry about over fitting the data. Suggest a suitable prior that mitigates this potential problem. You do not need to compute the posterior. Just write down your prior. [Hint: the task is to specify μ_0 and ω_0 in a suitable way.]

As it is written in lecture 5, we use shrinkage to avoid overfitting. So we use ridge regression. In this regard we consider $\mu = \vec{0}$ and covariance matrix $\sigma^2\Omega_0^{-1}$ in which $\Omega_0 = \lambda I$ For large value of λ, beta values will be close to zero. Hence for avoiding over fitting, large value of lambda and $\mu_0 = 0$, will decrease the spread of beta.

Question1: Posterior approximation for classification with logistic regression

Variable	Data Type	Meaning	Role
Work	Binary	Whether or not the woman works	Response Y
Constant	1	Constant to the intercept	Feature
HusbandInc	Numeric	Husbands income	Feature
EducYears	Counts	Years of education	Feature
ExpYears	Counts	Years of experience	Feature
ExpYears2	Numeric	(Years of experience/10)^2	Feature
Age	Counts	Age	Feature
NSmallChildren	Counts	Number of child <= 6 years in household	Feature
NBigChildren	Counts	Number of child > 6 years in household	Feature

a)

Consider the logistic regression model:

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

where y equals 1 if the woman works and 0 if she does not. x is an 8-dimensional vector containing the eight features (including a 1 to model the intercept). The goal is to approximate the posterior distribution of the parameter vector $\beta$with a multivariate normal distribution:

$$\beta|y, X \sim \mathcal{N}(\tilde{\beta}, J_y^{(-1)}(\tilde{\beta})),$$

where $\tilde{\beta}$ is the posterior mode and $J(\tilde{\beta}) = -\frac{\partial^2 lnp(\beta|y)}{\partial\beta\partial\beta^T}|_{\beta=\tilde{\beta}}$ is the negative of the observed Hessian evaluated at the posterior mode Note that $\frac{\partial^2 lnp(\beta|y)}{\partial\beta\partial\beta^T}$ is an 8*8 matrix with second derivatives on he diagonal and cross-derivatives $\frac{\partial^2 lnp(\beta|y)}{\partial\beta_i\partial\beta_i^T}$ on the off-diagonal. It is actually not hard to compute this derivative by hand, but don???t worry, we will let the computer do it numerically for you. Now, both $\tilde{\beta}$ and $J(\tilde{\beta})$ are computed by the optim function in R.[Hint: You may use code snippets from my demo of logistic regression in Lecture 6.] Use the prior $\beta \sim \mathcal{N}(0, \tau^2 I)$, with $\tau = 10$. Present the numerical values for $\tilde{\beta}$ and $J_y^{(-1)}\tilde{\beta}$ for the `WomanWork` data. Compute an approximate 95% equal tail posterior probability interval for the regression coefficient to the variable NSmallChild. Would you say that this feature is of importance for the probability that a women works?[Hint: To verify that your results are reasonable, you can compare to you get by estimating the parameters using maximum likelihood.

glmmodel = glm(Work~0+., data=WomenWork, family=binomial)

```r
WomenWork <- read.table("WomenWork.dat", header=TRUE)

head(WomenWork)
```

```
##   Work Constant HusbandInc EducYears ExpYears ExpYears2 Age NSmallChild
## 1    1        1   22.39494        12        7      0.49  43           0
## 2    0        1    7.23200         8       10      1.00  34           0
## 3    1        1   18.27199        12        4      0.16  41           1
## 4    0        1   28.06900        14        2      0.04  43           0
## 5    1        1   23.80000        12       24      5.76  45           0
## 6    0        1   96.00000        17        1      0.01  34           1
##   NBigChild
## 1         3
## 2         7
## 3         5
```

```
## 4          2
## 5          1
## 6          2
```

```
y <- as.matrix(WomenWork[,1])
x <- as.matrix(WomenWork[,2:ncol(WomenWork)])
# Feature names
feature_names =  colnames(WomenWork[,2:ncol(WomenWork)])
colnames(x) = feature_names
Npar = dim(x)[2]
tu_prior <- 10

# Setting up the prior
mu <- as.matrix(rep(0,Npar)) # Prior mean vector
Sigma <- (tu_prior^2)*diag(Npar) # Prior covariance matrix

# Functions that returns the log posterior for the logistic and probit regression.
# First input argument of this function must be the parameters we optimize on,
# i.e. the regression coefficients beta.

LogPostLogistic <- function(betas,y,x,mu,Sigma){
  linPred <- x%*%betas;
  logLik <- sum( linPred*y - log(1 + exp(linPred)) );
  #if (abs(logLik) == Inf) logLik = -20000; # Likelihood is not finite, stear the optimizer away from h
  logPrior <- dmvnorm(betas, mu, Sigma, log=TRUE);

  return(logLik + logPrior)
}

# Select the initial values for beta
initVal <- matrix(0,Npar,1)

# The argument control is a list of options to the optimizer optim, where fnscale=-1 means that we mini
# the negative log posterior. Hence, we maximize the log posterior.
OptimRes <- optim(initVal,LogPostLogistic,gr=NULL,y,x,mu,Sigma,method=c("BFGS"),control=list(fnscale=-1

# Printing the results to the screen
posterior_mode = as.vector(OptimRes$par)
names(posterior_mode) = feature_names
#names(OptimRes$par) <- Xnames # Naming the coefficient by covariates
posterior_covariance = - solve(OptimRes$hessian)
posterior_sd = sqrt(diag(posterior_covariance))
names(posterior_sd) = feature_names # Naming the coefficient by covariates
```

Posterior Mode ($\tilde{\beta}$) for the optim approach:

```
posterior_mode
```

```
##    Constant  HusbandInc   EducYears    ExpYears   ExpYears2         Age
##  0.62672884 -0.01979113  0.18021897  0.16756670 -0.14459669 -0.08206561
## NSmallChild   NBigChild
## -1.35913317 -0.02468351
```

Approximated Standard Deviation:

```
posterior_sd
```

```
##    Constant  HusbandInc   EducYears    ExpYears   ExpYears2         Age
##  1.50533138  0.01589983  0.07885556  0.06596754  0.23575129  0.02680412
## NSmallChild   NBigChild
##  0.38892439  0.14132327
```

To compute an approximate 95% equal tail posterior probability interval for the regression coefficient to the variable NSmallChild we can use qnorm function by this way:

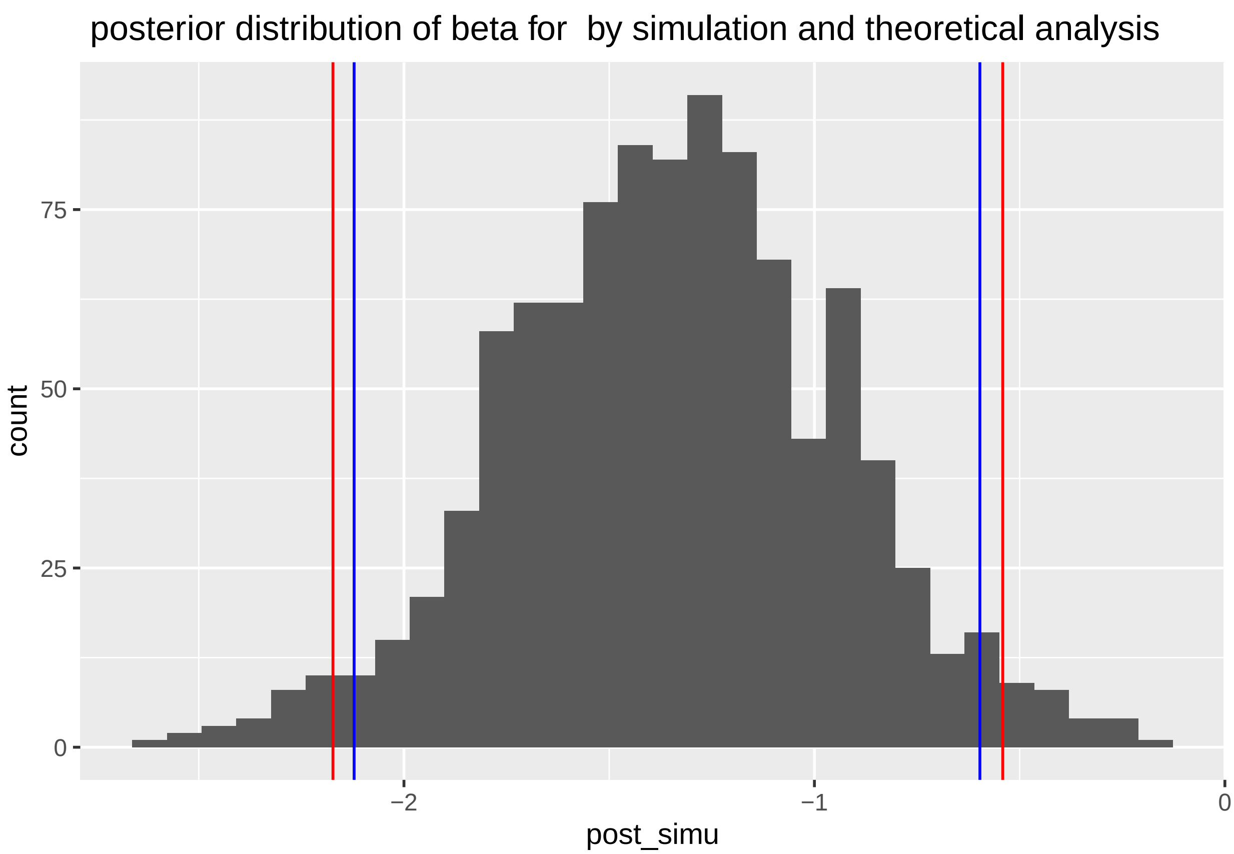
```
NSmallChild_mode = as.numeric(posterior_mode["NSmallChild"])
NSmallChild_std = as.numeric(posterior_sd["NSmallChild"])
ci = qnorm(p=c(0.025, 0.975), mean=NSmallChild_mode, sd=NSmallChild_std)
paste("The lower bound of the 95 % credible interval for the feature NSmallChild is",
            round(ci[1], 4), "and the upper bound is",
            round(ci[2], 6))
```

```
## [1] "The lower bound of the 95 % credible interval for the feature NSmallChild is -2.1214 and the upp
```

```
# Control that the calculations have been made correctly
glmModel = glm(Work ~ 0+., data=WomenWork, family=binomial)

ps_df <- data.frame("Coefficient"=glmModel$coefficients, "posterior_mode"=posterior_mode)
ps_df
```

```
##             Coefficient posterior_mode
## Constant     0.64430363     0.62672884
## HusbandInc  -0.01977457    -0.01979113
## EducYears    0.17988062     0.18021897
## ExpYears     0.16751274     0.16756670
## ExpYears2   -0.14435946    -0.14459669
## Age         -0.08234033    -0.08206561
## NSmallChild -1.36250239    -1.35913317
## NBigChild   -0.02542986    -0.02468351
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

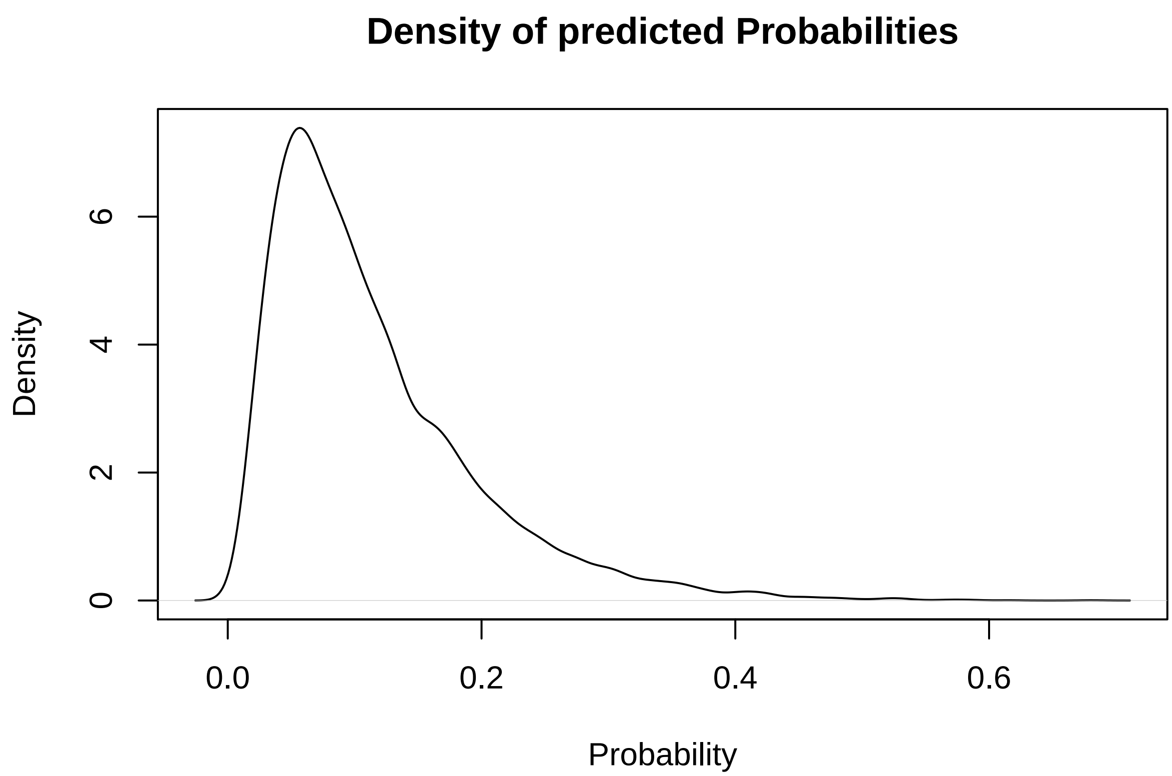


b)

Use your normal approximation to the posterior from (a).Write a function that simulate draws from the posterior predictive distribution of $Pr(y = 1|x)$ where the values of x corresponds to a 37-year-old woman, with two children (3 and 6 years old), 8 years of education, 11 years of experience, and a husband with an income of 13. Plot the posterior predictive distribution of $Pr(y = 1|x)$ for this woman. [Hints: The R package mvtnorm will be useful. Remember that $Pr(y = 1|x)$ can be calculated for each posterior draw of $beta$].

```
X_woman = matrix(c(1, 13, 8, 11, (11/10)^2, 37, 2, 0))
nDraws=10000
sigmoid = function(value) {
  return (exp(value)/(1+exp(value)))
}
simulate_posterior = function(data, mean, sigma, nDraws) {
  beta_Pred = rmvnorm(nDraws, mean=mean, sigma=sigma)
  Pred = beta_Pred %*% data
  logit_Pred = sigmoid(Pred)
  return(logit_Pred)
}

simulated_probabilities = simulate_posterior(X_woman,
                                             mean = posterior_mode,
                                             sigma = posterior_covariance,
                                             nDraws = 10000)
```

```r
# Estimate density
simulated_density = density(simulated_probabilities)
plot(simulated_density , main="Density of predicted Probabilities", xlab="Probability", ylab="Density")
```




```r
probability_working =
  mean(rbinom(length(simulated_probabilities), 1, simulated_probabilities))
cat(paste("Therefore the women is working with a probability of "),probability_working )
```

```
## Therefore the women is working with a probability of  0.1213
```

c)

Now, consider 8 women which all have the same features as the woman in (b). Rewrite your function and plot the posterior predictive distribution for the number of women, out of these 8, that are working. [Hint: Simulate from the binomial distribution, which is the distribution for a sum of Bernoulli random variables.]

```r
posterior_predict = function(data, mean, sigma, nDraws, n) {
  multiple_Pred=c()
  for (i in 1:nDraws) {
    beta_Draw = simulate_posterior(data, mean, sigma, 1)
    multiple_Pred=c(multiple_Pred, rbinom(1, n, beta_Draw))
  }
  barplot(table(multiple_Pred), main=paste("Posterior predictive distribution for", n, "women"),
```

```
        xlab="Number of women")
}

posterior_predict(X_woman, posterior_mode, posterior_covariance, 10000, 8)
```