Bayesian Learning Lab 2

Qinyuan Qi(qinqi464)   Satya Sai Naga Jaya Koushik Pilla (satpi345)

2024-05-24

1. Linear and polynomial regression

We have a dataset Linkoping2022.xlsx that contains daily average temperatures (in degrees Celcius) in Linköping over the year 2022.

The response variable is temp and covariate time, the original data is a date-time string, it also needs to be converted to a numeric value using the following formula.

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

A Bayesian analysis of the following quadratic regression model will be performed as follows.

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

1.a. Use the conjugate prior for the linear regression model

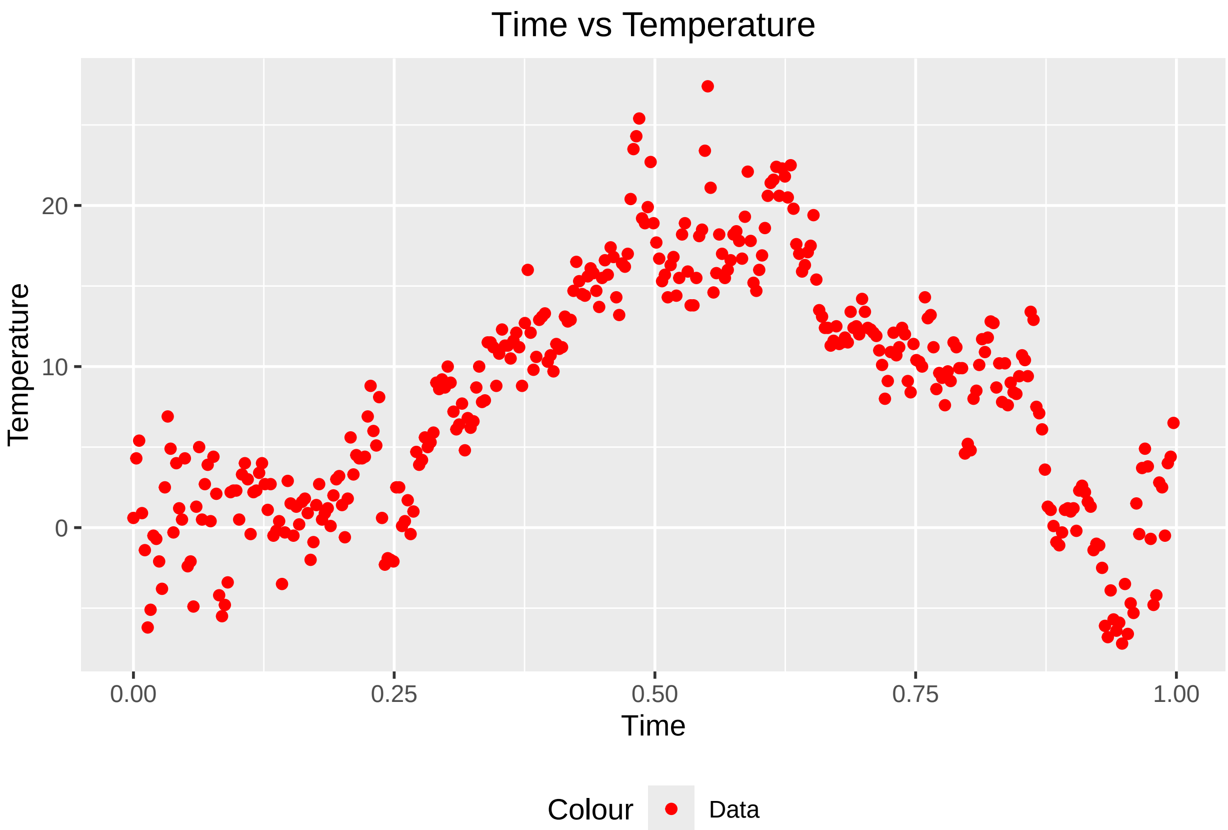
According to the question, we have the following init value.

```
#------------------------------------------------------------------
# Code for question 1a
#------------------------------------------------------------------
# given values
mu_0 <- c(0, 100, -100)
omega_0 <- 0.01 * diag(3)
v_0 <- 1
sigma2_0 <- 0.1
days_in_year <- 365
first_day_of_year <- as.Date("2022-01-01")
```

We calculate the parameter time,time_square according to the formula mentioned in the question.

```
#------------------------------------------------------------------
# load data and plot
#------------------------------------------------------------------
# load the data
temperature_data <- read_xlsx(path="Linkoping2022.xlsx")
temperature_data$time <- as.numeric(as.Date(temperature_data$datetime) - first_day_of_year) /
                         days_in_year
temperature_data$time_square <- temperature_data$time ** 2
```

Then plot the original data to visualize the relationship between time and temperature.



A linear regression model with the formula mentioned above was built and the summary of the model as follows.

```r
#-------------------------------------------------------------------
# make a linear regression model
#-------------------------------------------------------------------
linRegModel <- lm(temp ~ time + time_square, data = temperature_data)
summary(linRegModel)
```

```
##
## Call:
## lm(formula = temp ~ time + time_square, data = temperature_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.660  -2.851  -0.184   2.509  12.659
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -7.3041     0.6584  -11.09   <2e-16 ***
## time         83.1499     3.0498   27.26   <2e-16 ***
## time_square -78.2985     2.9605  -26.45   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.216 on 362 degrees of freedom
## Multiple R-squared:  0.6725, Adjusted R-squared:  0.6707
## F-statistic: 371.7 on 2 and 362 DF,  p-value: < 2.2e-16
```

According to the conjugate prior for the linear regression in slide 5 of Lecture 5, we have the following formula to simulating draws from the joint prior distribution.

Joint prior for β and σ^2 are as follows.

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

$$\sigma^2 \sim Inv - \chi^2(v_0, \sigma_0^2)$$

The posterior are as follows.

$$\beta|\sigma^2, y \sim N(\mu_n, \sigma^2\Omega_n^{-1})$$

$$\sigma^2|y \sim Inv - \chi^2(v_n, \sigma_n^2)$$

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

$$\Omega_n = X'X + \Omega_0$$

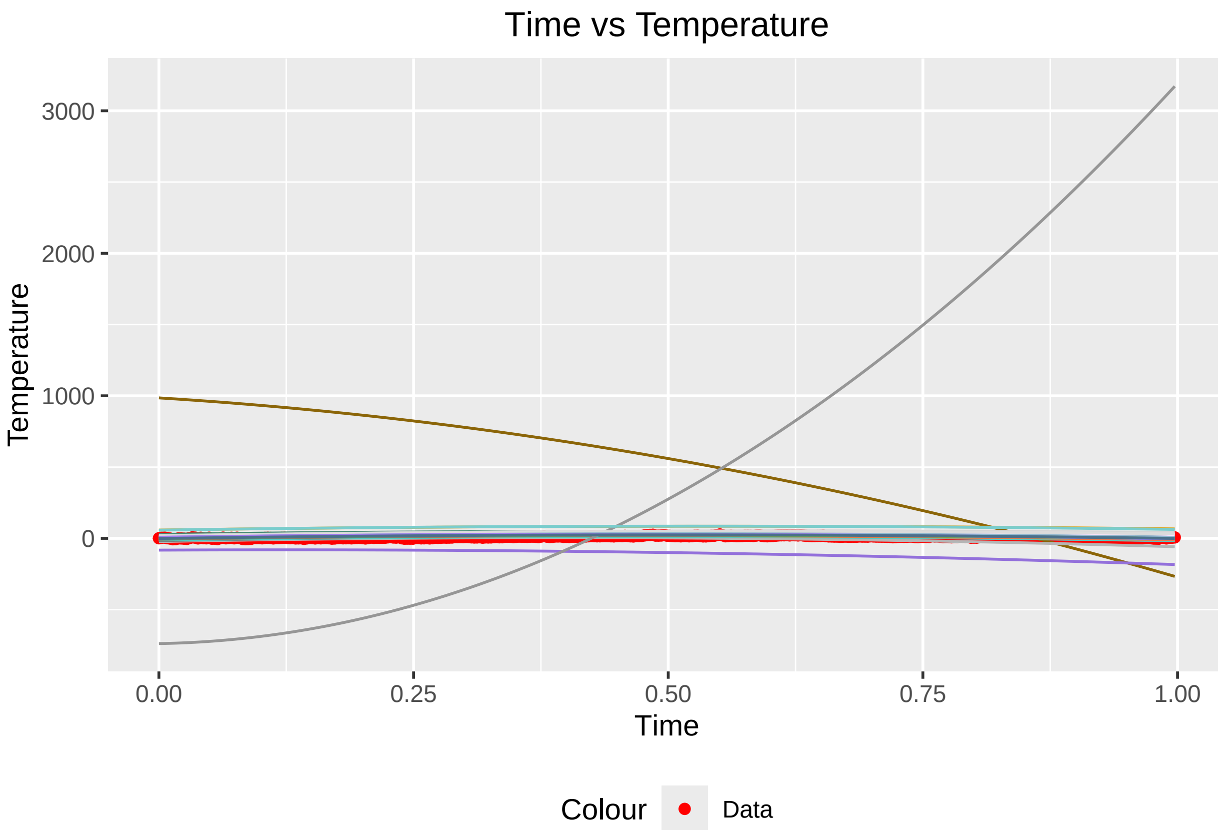
$$v_n = v_0 + n$$

From slide 4 of Lecture 5, we have the $\hat{\beta}$ formula as follows.

$$\hat{\beta} = (X'X)^{-1}X'y$$

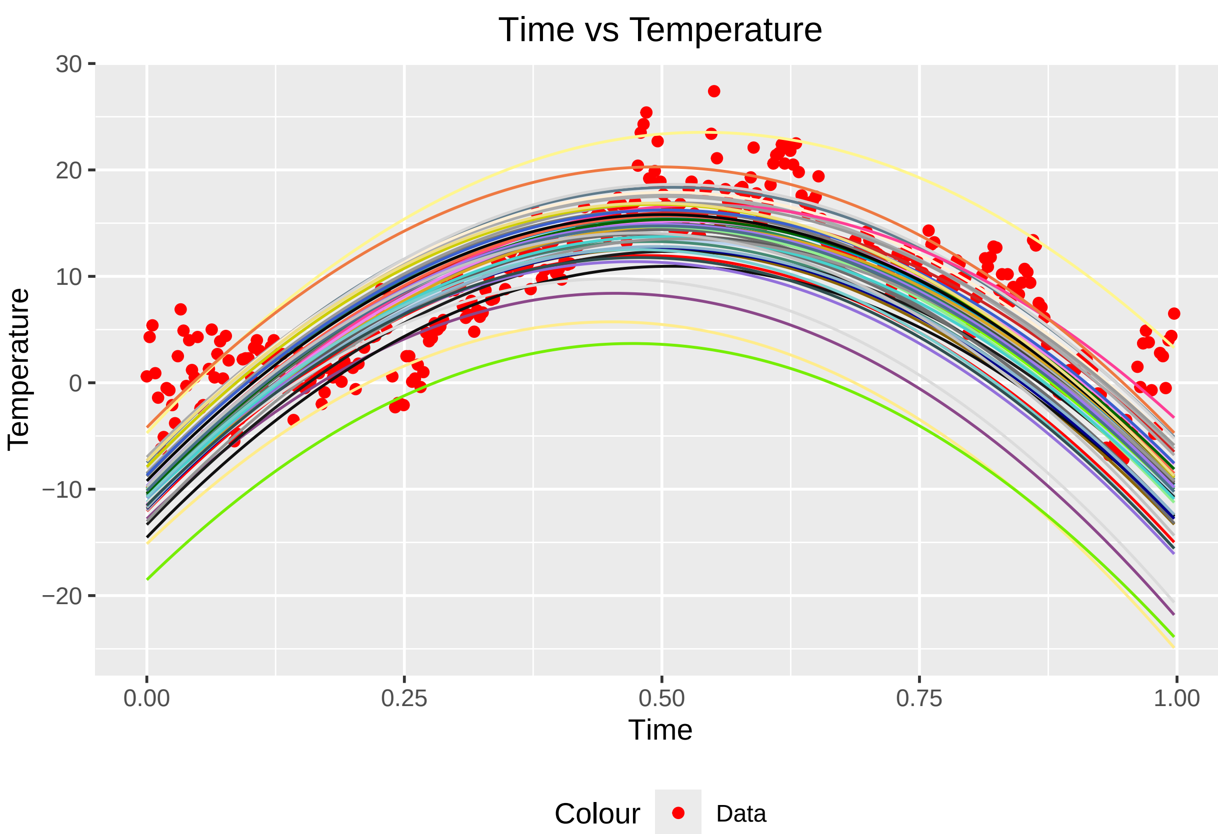
$$s^2 = \frac{1}{n-k}(y - X\hat{\beta})'(y - X\hat{\beta})$$

Based on the above formula, we can simulate draws from the joint prior distribution of $\beta_0, \beta_1, \beta_2$ and σ^2 as follows. We will simulate 50 draws this time.



From the graph above, we found that at least 2 curves are not fitting the data well, which reach 3000 Celcius degree and 1000 Celcius degree. So we change the parameter to make the prior more suitable for the data. We change parameters and simulate 50 draws from new joint prior distribution and plot it. The following is the new parameters.

```
#--------------------------------------------------------------------
# draw from joint_conjugate_prior and plot(new parameters)
#--------------------------------------------------------------------
# new given parameters
mu_0 <- c(-10, 100, -100)
omega_0 <- 0.03 * diag(3)
v_0 <- 2
sigma2_0 <- 0.1
```



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

We write a function that simulates draws from the joint posterior distribution of $\beta_0, \beta_1, \beta_2$ and σ^2 first.

```
#----------------------------------------------------------------------
# function to simulate draws from the joint posterior distribution
#----------------------------------------------------------------------
# function to simulate draws from the joint posterior distribution
joint_conjugate_posterior <- function(nr_of_iterations, v_0, sigma2_0, mu_0, omega_0, days_in_year=365)
  # data frame to save prior coef data
  prior <- data.frame(matrix(ncol = 3, nrow = nr_of_iterations)) # for every beta one column

  # data frame to save linear regression line
  regline <- data.frame(matrix(nrow = days_in_year, ncol = nr_of_iterations)) # every col for one day

  # joint conjugate prior
  for (i in 1:nr_of_iterations) {

    #  calculate var
    var <- LaplacesDemon::rinvchisq(1,v_0,sigma2_0)

    # calculate beta
    beta <- MASS::mvrnorm(1, mu_0, var*solve(omega_0))

    # set values
    prior[i,1:3] <- beta
```

```
    regline[,i] <- beta[1] + beta[2] * temperature_data$time + beta[3] * temperature_data$time_square
  }

  # add meaningful column and row names
  for (i in 1:nr_of_iterations) {
    colnames(regline)[i] <- paste("pred","", i)
    rownames(regline)[i] <- paste("day","",i)
  }

  # return the prior and regline
  return(list(p = prior, r = regline))
}
```

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

According to the formulas mentioned in 1.a, we can calculate the posterior distribution of the parameters μ_n, Ω_n, v_n.

```
#-----------------------------------------------------------------------
# calculate the posterior distribution parameters
#-----------------------------------------------------------------------
k <- 3
X <- cbind(1, temperature_data$time, temperature_data$time_square)
y <- temperature_data$temp
n <- nrow(temperature_data)
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
v_n <- v_0 + n
sigma2_n <- (v_0 * sigma2_0 + t(y) %*% y + t(mu_0) %*% omega_0 %*%
            mu_0 - t(mu_n) %*% omega_n %*% mu_n) / v_n
```

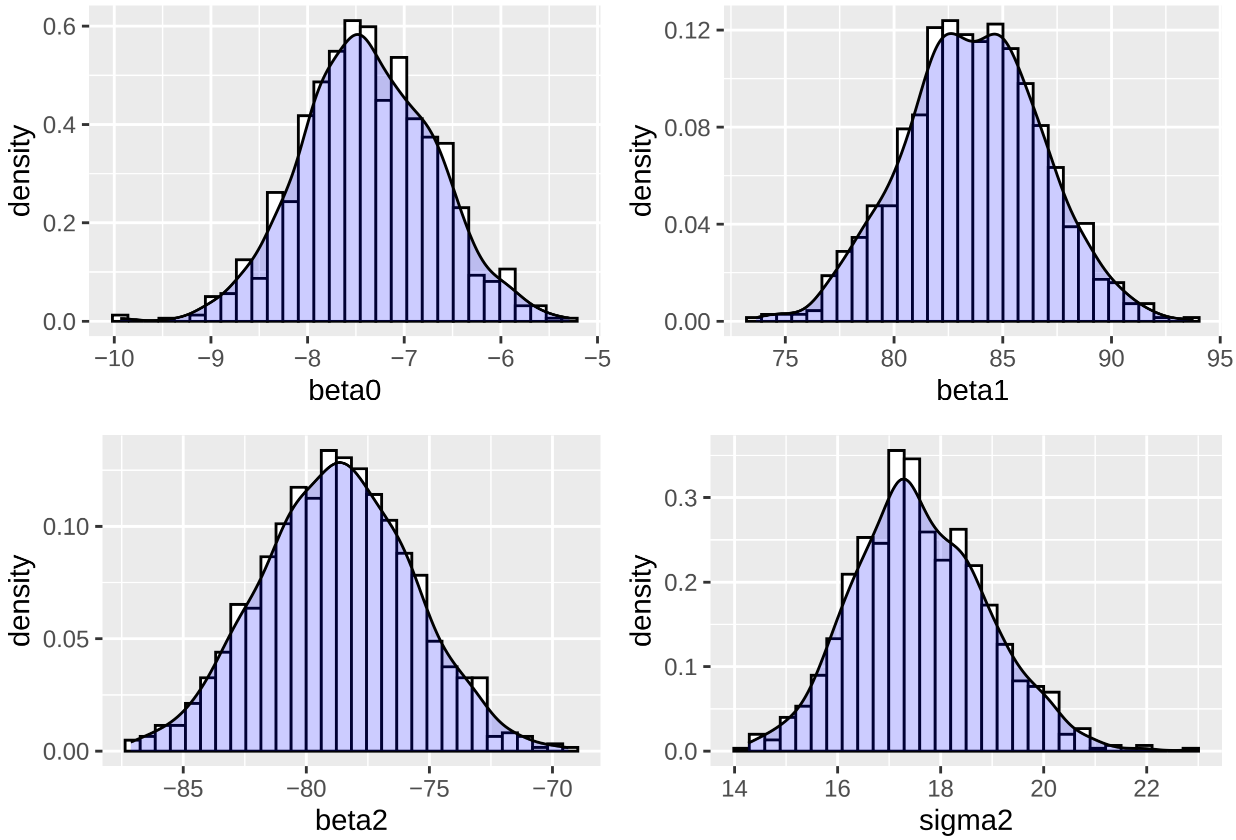
Marginal posterior of β is a t-distribution with n-k freedom.

$$\beta|y \sim t_{n-k}[\mu_n, s^2(X'X)^{-1}]$$

```
#-----------------------------------------------------------------------
# simulate the beta parameter and draw
#-----------------------------------------------------------------------

# simulate the beta parameter
data_hist <- as.data.frame(
  mvtnorm::rmvt(n = 1000, delta = mu_n, df = n-k,
                sigma = as.numeric(sigma2_n) * solve(t(X) %*% X)))

# simulate the sigma^2 parameter
var_hist <- LaplacesDemon::rinvchisq(n = 1000, v_n, sigma2_n)
```



1.b.ii Make a plot, and overlay curves for the 90% equal tail posterior probability intervals of f(time), then comment

We calculate median of the β and f(time) for every time first.

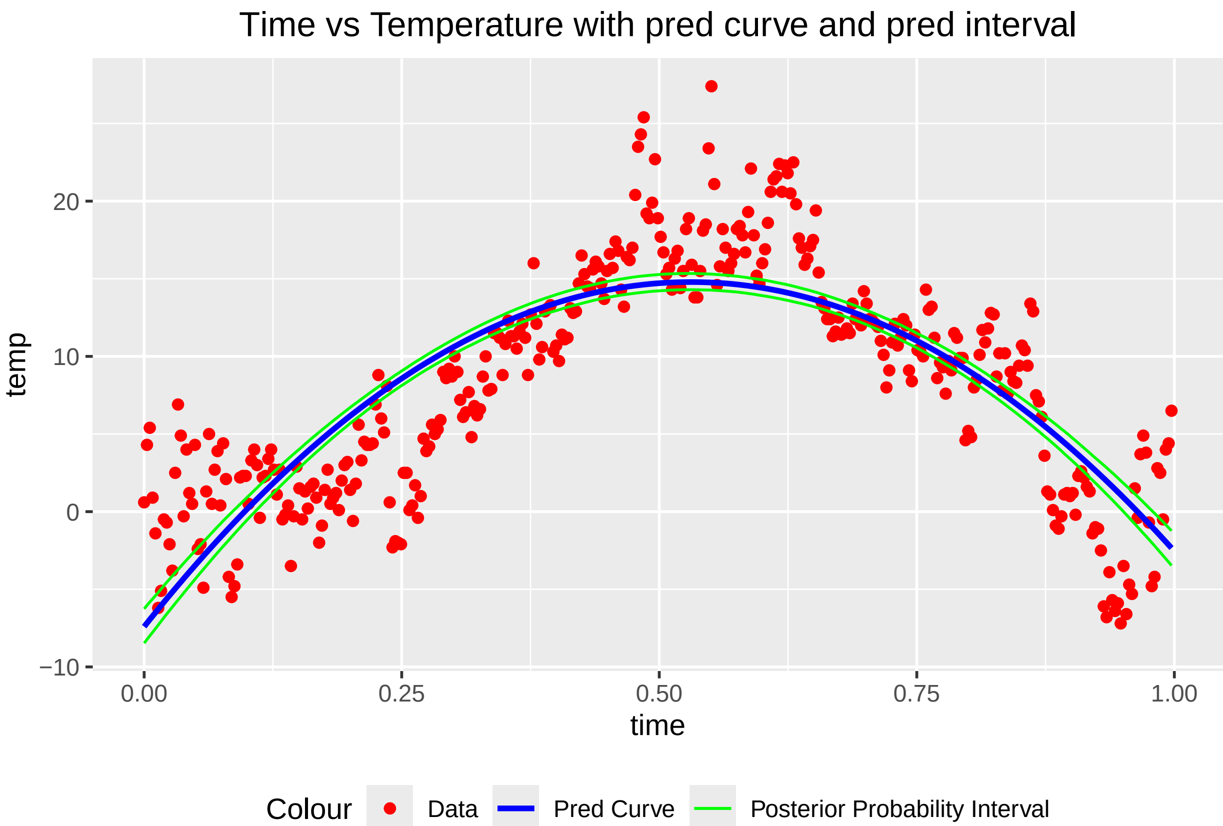
```
#-----------------------------------------------------------------------
# calculate the median of beta and f(time)
#-----------------------------------------------------------------------
data_hist = data_hist[,1:3]

beta_median = matrixStats::colMedians(as.matrix(data_hist))
pred.1b <- beta_median %*% t(X)
```

Then we calculate the equal tail posterior probability intervals of f(time) and plot them on the graph below.

```
#-----------------------------------------------------------------------
# calculate the 90% credible interval
#-----------------------------------------------------------------------
preds <- as.matrix(data_hist) %*% t(X) # regression function
pred_interval <- data.frame(nrow = n, nrow = 2)
colnames(pred_interval) <- c("lower","upper")

for(i in 1:days_in_year){
  data_t <- preds[,i]
  pred_interval[i,] <- quantile(data_t, probs = c(0.05,0.95))
}
```



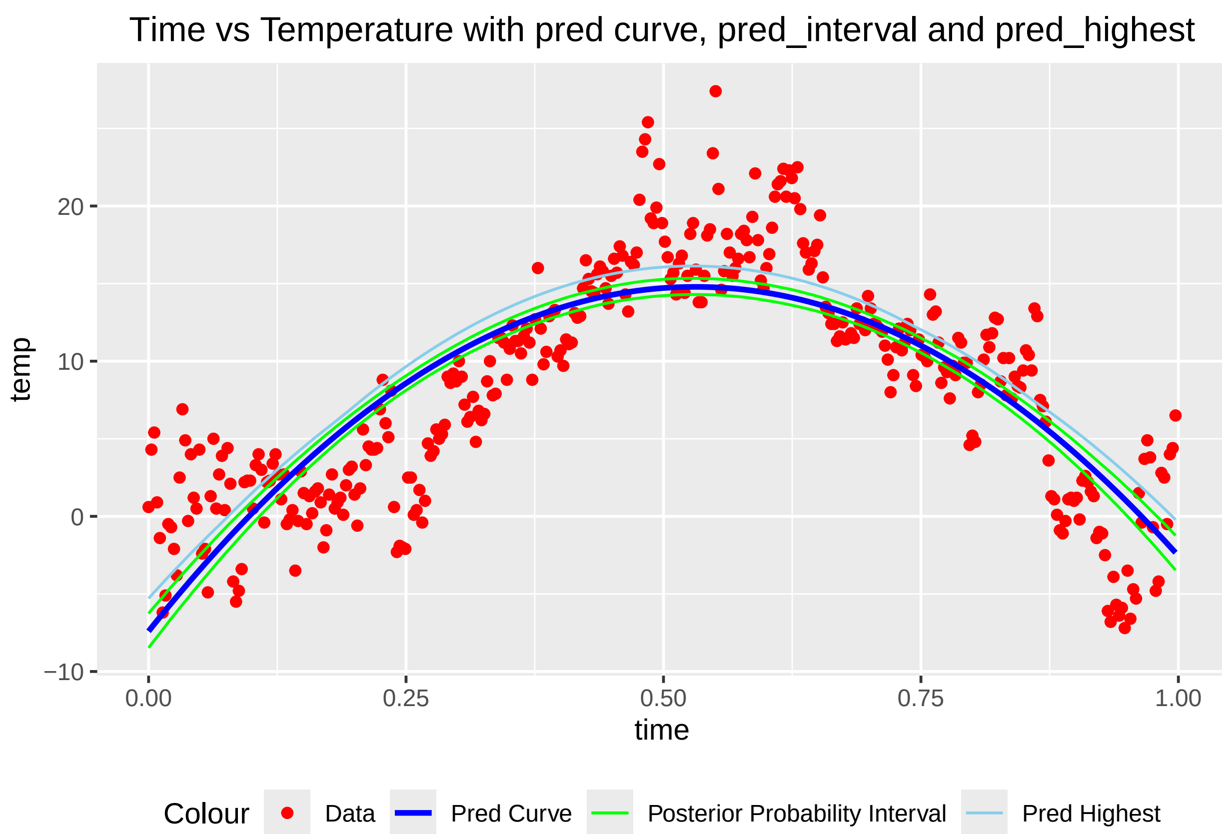
A posterior probability interval(the space between 2 green lines above) is an interval where the posterior probability of the parameter within that interval exceeds a specified threshold. It is an interval to evaluate the robustness of the parameter. It does not and should not contain most of the data points in our case.

1.c. Use the simulated draws in (b) to simulate the posterior distribution of $\tilde{x}$

We calculate the highest prediction for every day from the data in 1.b. Then we add this curve to the plot above and get the following graph.

```
#-----------------------------------------------------------------
# calculate the highest prediction for every day
#-----------------------------------------------------------------

pred_highest <- c()
for (i in 1:days_in_year) {
  pred_highest[i] <- max(preds[,i])
}
```



1.d. suitable prior that mitigates this potential problem

From slide 9 of Lecture 5, we know that too many knots lead to overfitting. We can use regularization prior to avoid the problem mentioned in the question. A suitable prior that mitigates this potential problem is as follows.

$$\beta_i \mid \sigma^2 \overset{\text{iid}}{\sim} N(\mu_0, \frac{\sigma^2}{\lambda}) \quad where \quad \Omega_0 = \lambda \cdot I$$

2. Posterior approximation for classification with logistic regression

We have a dataset WomenAtWrok.dat that contains n=132 observations, and the 8 variables related to women are listed in the following table.

Variable	Data Type	Meaning	Role
Work	Binary	Whether or not the woman works	Response y
Constant	1	Constant to the intercept	Feature
HusbandInc	Numeric	Husband's income	Feature
EducYears	Counts	Years of education	Feature
ExpYears	Counts	Years of experience	Feature
Age	Counts	Age	Feature
NSmallChild	Counts	Number of child $\leq$ 6 years in household	Feature
NBigChild	Counts	Number of child $>$ 6 years in household	Feature

2.a. Approximate the posterior distribution of the parameter vector β with a multivariate normal distribution and comment. compute 95% equal tail posterior probability interval and comment.

We have the following logistic regression model, if y=1 means woman works and 0 means woman not work.

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

Posterior distribution of the parameter vectoor β with a multivariate normal distribution as follows.

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

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.

Also we use prior $\beta \sim N(\mu, \tau^2 I)$, where $\mu = 0$ and $\tau = 2$, so we have the following init values.

```r
#------------------------------------------------------------------------
# given values
#------------------------------------------------------------------------
mu <- 0
tau <- 2
```

To use optim function, we create the following function LogPostLogistic, then we can get the optimized $\tilde{\beta}$ and $J_y^{-1}(\tilde{\beta})$

```r
#------------------------------------------------------------------------
# Functions that returns the log posterior for the logistic
#------------------------------------------------------------------------

LogPostLogistic <- function(betas,y,X,mu,sigma){
  linPred <- X%*%betas;
  logLik <- sum( linPred*y - log(1 + exp(linPred)) );
  # Likelihood is not finite, stear the optimizer away from here!
  # Idea from cource code by teacher
  if (abs(logLik) == Inf)
    logLik = -20000;
  # prior follows multi-normal distribution
  logPrior <- dmvnorm(betas, mu, sigma, log=TRUE);
  return(logLik + logPrior)
}

#------------------------------------------------------------------------
# get the optimized beta and inverse jacobian
#------------------------------------------------------------------------

# number of features
n <- dim(X)[2]
# setting up the prior
mu    <- as.vector(rep(mu,n))  # Prior mean vector
sigma <- tau^2 * diag(n)  # Prior variance matrix

# use random initial values
init_val <- as.vector(rnorm(dim(X)[2]))
```

```r
# optimize the log posterior
OptimRes <- optim(init_val,
                  LogPostLogistic,
                  y = y,
                  X = X,
                  mu = mu,
                  sigma = sigma,
                  method=c("BFGS"),
                  control=list(fnscale=-1),
                  hessian=TRUE)
# set values to print out
posterior_mode  <- OptimRes$par

# hessian is the negative of the observed Hessian evaluated at the posterior mode
# Jacobian = (-hessian)
beta_jacobian <- -OptimRes$hessian
beta_inverse_jacobian <- solve(beta_jacobian)
```

```
## [1] "The posterior beta is:"

##    Constant  HusbandInc   EducYears    ExpYears         Age NSmallChild
## -0.04114676 -0.03730710  0.17871035  0.12072802 -0.04617730 -1.47239519
##   NBigChild
## -0.02010522

## [1] "The Inverse Jacobian of beta is:"

##          Constant    HusbandInc     EducYears      ExpYears           Age
## [1,]  1.909867851  4.032714e-03 -6.280855e-02  1.041111e-03 -2.575540e-02
## [2,]  0.004032714  4.833220e-04 -9.147811e-04 -2.665790e-05 -6.428917e-05
## [3,] -0.062808545 -9.147811e-04  7.958283e-03  5.508827e-05 -3.180934e-04
## [4,]  0.001041111 -2.665790e-05  5.508827e-05  1.112824e-03 -2.844885e-04
## [5,] -0.025755398 -6.428917e-05 -3.180934e-04 -2.844885e-04  7.547482e-04
## [6,] -0.137699771  1.585477e-03 -1.438738e-02 -1.336368e-03  5.547782e-03
## [7,] -0.088874755  4.968080e-06  1.134845e-04  7.206901e-04  1.044887e-03
##       NSmallChild     NBigChild
## [1,] -0.137699771 -8.887475e-02
## [2,]  0.001585477  4.968080e-06
## [3,] -0.014387384  1.134845e-04
## [4,] -0.001336368  7.206901e-04
## [5,]  0.005547782  1.044887e-03
## [6,]  0.227969888  1.122568e-02
## [7,]  0.011225684  2.690182e-02
```

We can find that NSmallChild has the biggest influence on the work value, which is -1.47239519. From the beta values, we find that the number corresponding to the variable NSmallChild is also a negative number, which means the more NSmallChild, the less likely to work.

Then we compute an approximate 95% equal tail posterior probability interval for the regression coefficient to the variable NSmallChild.We find that the whole interval is negative, which means NSmallChild has an obvious negative impact on women's work or not.

```r
#-------------------------------------------------------------------
# Compute an approximate 95% equal tail posterior probability interval
#-------------------------------------------------------------------
beta_sim <- mvtnorm::rmvnorm(n = 1000, mean = posterior_mode, sigma = beta_inverse_jacobian)
data.NSmallChild <- beta_sim[,6]
```

```r
# calculate the 95% credible interval
pred_interval <- quantile(data.NSmallChild, probs = c(0.025,0.975))
pred_interval
```

```
##      2.5%     97.5%
## -2.400252 -0.554989
```

We also use glm to genertate logistic regression model.

```r
#------------------------------------------------------------------
# logistic regression
#------------------------------------------------------------------
glmModel <- glm(Work ~ 0 + ., data = WomenAtWork, family = binomial)
summary(glmModel)
```

```
##
## Call:
## glm(formula = Work ~ 0 + ., family = binomial, data = WomenAtWork)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## Constant     0.02263    1.93083   0.012 0.990649
## HusbandInc  -0.03796    0.02229  -1.703 0.088573 .
## EducYears    0.18447    0.10007   1.844 0.065253 .
## ExpYears     0.12132    0.03353   3.618 0.000297 ***
## Age         -0.04858    0.03323  -1.462 0.143686
## NSmallChild -1.56485    0.51078  -3.064 0.002187 **
## NBigChild   -0.02526    0.17716  -0.143 0.886618
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 182.99  on 132  degrees of freedom
## Residual deviance: 146.73  on 125  degrees of freedom
## AIC: 160.73
##
## Number of Fisher Scoring iterations: 4
```

Compare with the result above, we can find that the posterior result is similar to the result of logistic regression generated by glm.

2.b. Write a function that simulates draws from the posterior predictive distribution and plot

According to the question, and the formula provided in 2.a

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

So we know that

$$Pr(y=0|x,\beta) = 1 - Pr(y=1|x,\beta)$$

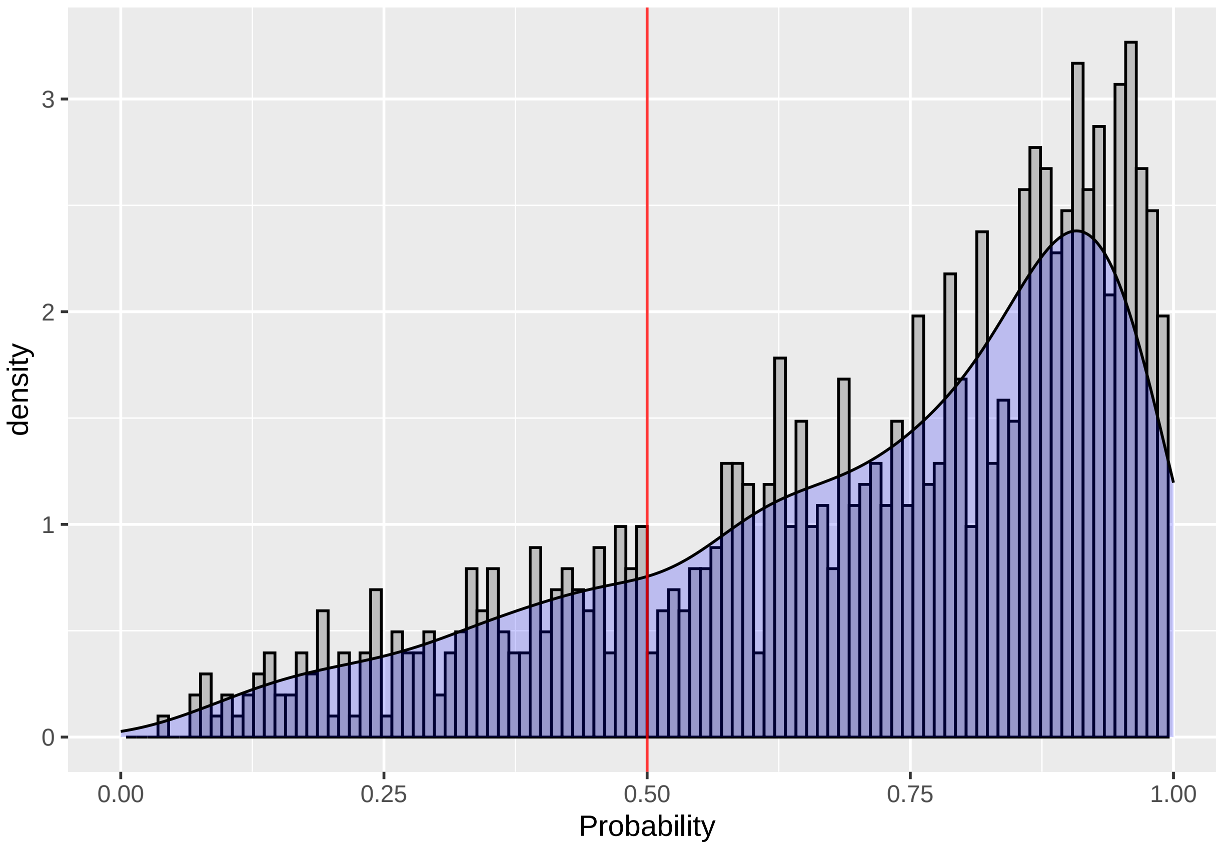
Then we write the following function.

```r
#----------------------------------------------------------------------
# Simulates draws from the posterior predictive distribution
#----------------------------------------------------------------------
post_pred <- function(X, beta) {
  pred <- beta %*% X #linear predictions
  pred_logit <- 1- exp(pred)/(1 + exp(pred)) #sigmoid function to model probabilities
  return(data.frame(Probability = pred_logit))
}
```

we have the following init value, and we can get the posterior predictive distribution and plot.

```r
#Create matrix of sample features
X_data <- as.matrix(c(0, 18, 11, 7, 40, 1, 1))
#function call on sample data
post_pred_sim <- post_pred(X_data, beta_sim)
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_bar()`).
```



From the density of the posterior prediction, we can find that given the life conditions above , women have more probability to choose not to work.

2.c. Rewrite your function and plot

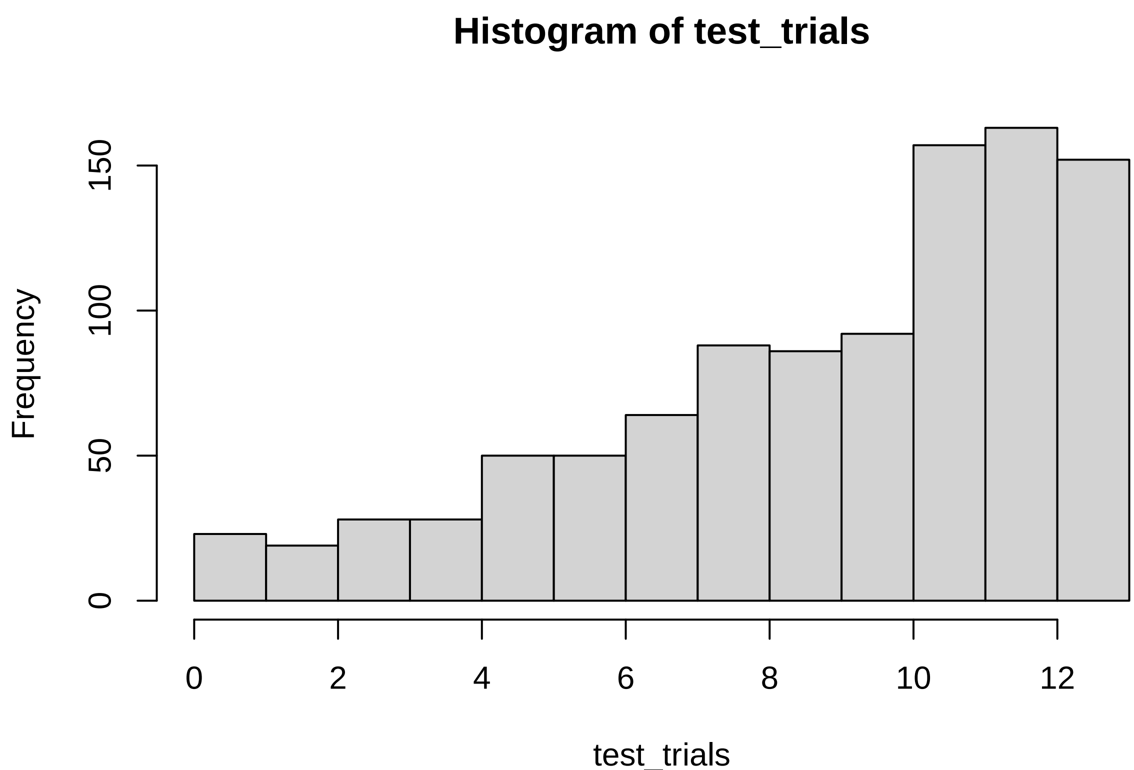
We have 13 women have the same features as in 2b, we rewrite the function and plot the result as below.

```r
#----------------------------------------------------------------------
# Get the trial number and plot
#----------------------------------------------------------------------
```

```r
n <- 13
post_pred_binom <- function(X, beta) {

  pred <- beta %*% X #linear predictions
  pred_logit <- 1- exp(pred)/(1 + exp(pred)) #sigmoid function to model probabilities
  trial <- c()
  for (i in 1:length(pred_logit)) {
    trial[i] <- rbinom(n = 1, size = n, prob = pred_logit[i])
  }
  return(trial)
}

test_trials <- post_pred_binom(X_data, beta_sim)
hist(test_trials)
```



According to the plot above, histogram of test_trials match the result of 2b, and around 10-12 out of 13 are likelily not working given the given features.