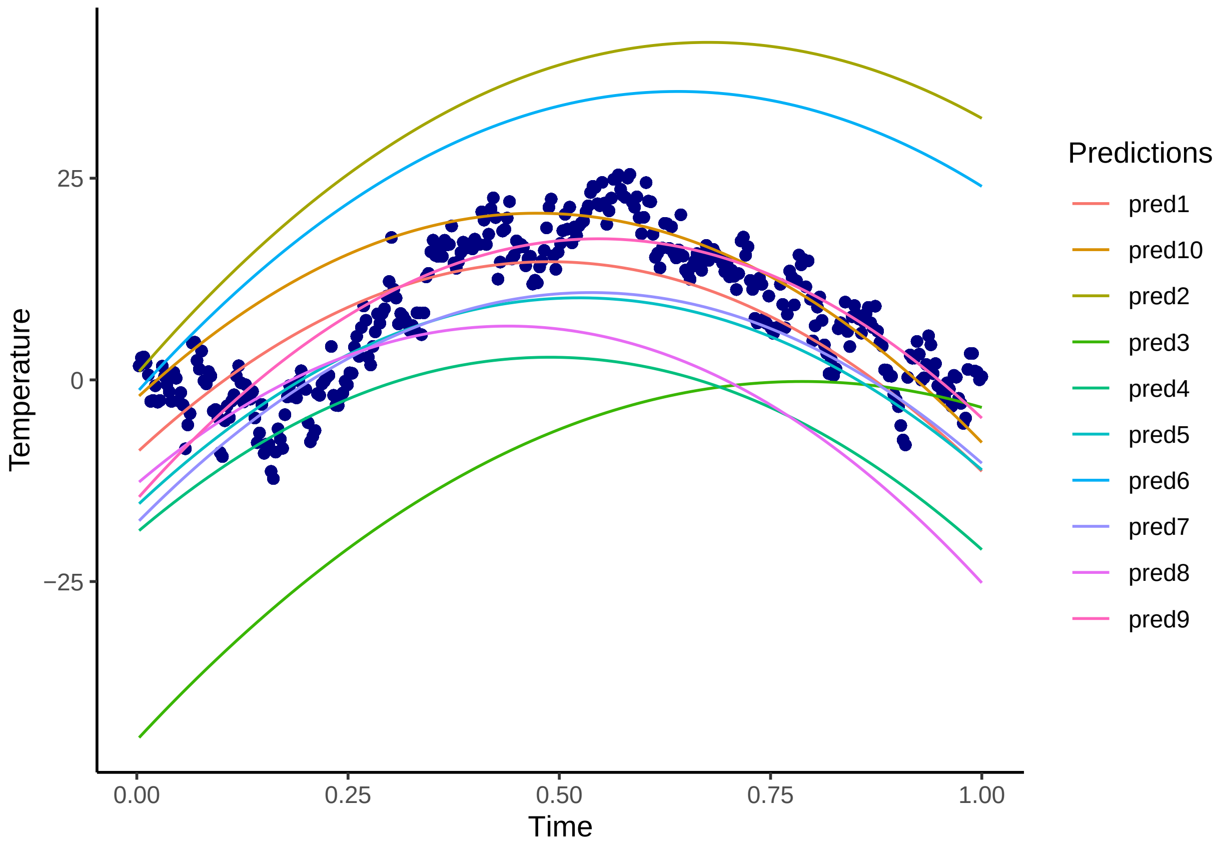
Bayesian Learning (732A91) Lab2 Report

Christoforos Spyretos (chrsp415) & Marketos Damigos (marda352)

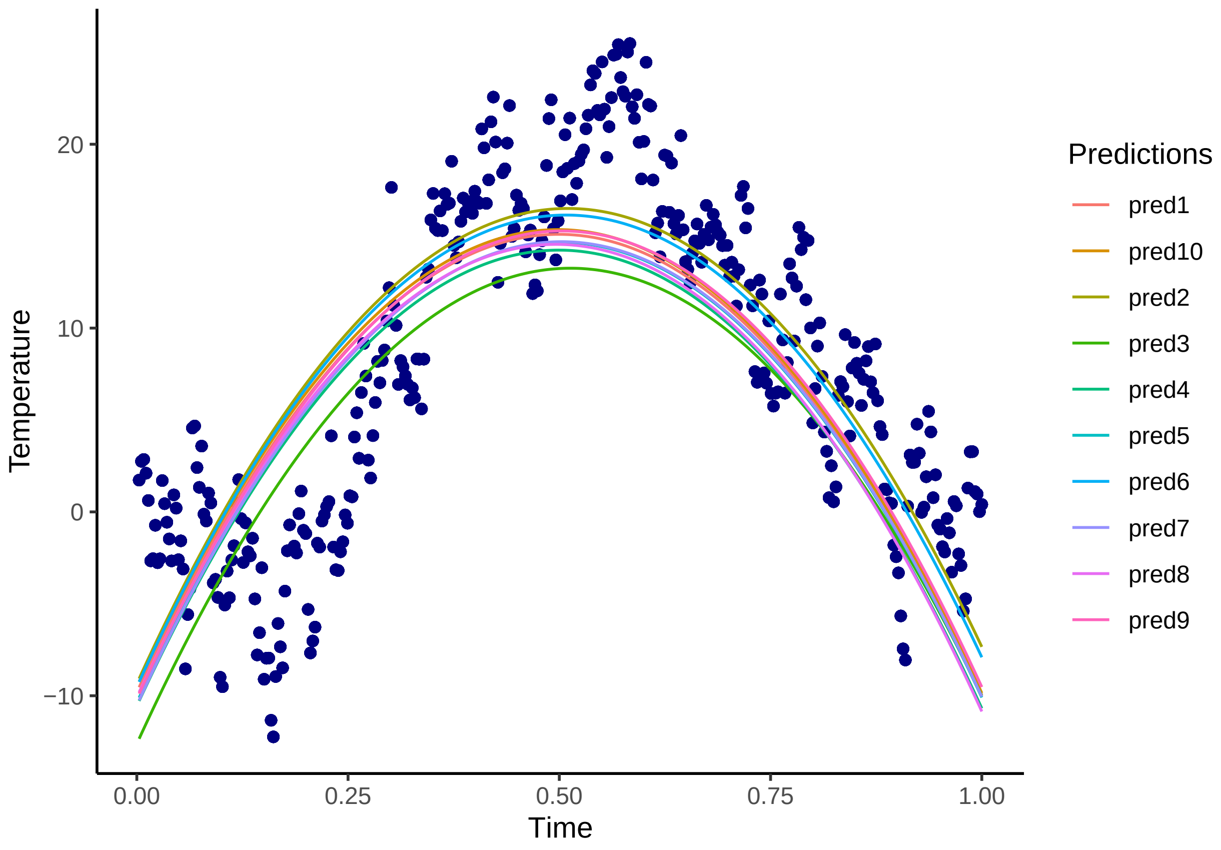
2022-05-24

Assignment 1 *Linear and polynomial regression*

Task 1a



The above graph illustrates the actual temperatures (blue points) and a collection of regression curves, one for each draw from the prior. The graph provides mixed results almost half of the curves are not placed in the region of the actual temperatures, but it is not that bad. Two represent a little higher temperatures but follow the same pattern as the actual ones. Additionally, two curves represent the actual temperatures initially, but as time passes, they illustrate lower temperatures than the actual ones. Finally, only one curve shows poor results; in the very beginning has low temperatures, but in the end, the curve is in the region of the actual temperatures. Thus, the prior hyper-parameters need small configurations.



The above graph illustrates the actual temperatures (blue points) and a collection of regression curves, one for each draw from the prior, but with the modified hyper-parameters. Only two parameters were changed, $\Omega_0 = 0.05$ and $\sigma^2 = 0.2$. As a result, all the curves are in the actual temperature region.

Task 1b

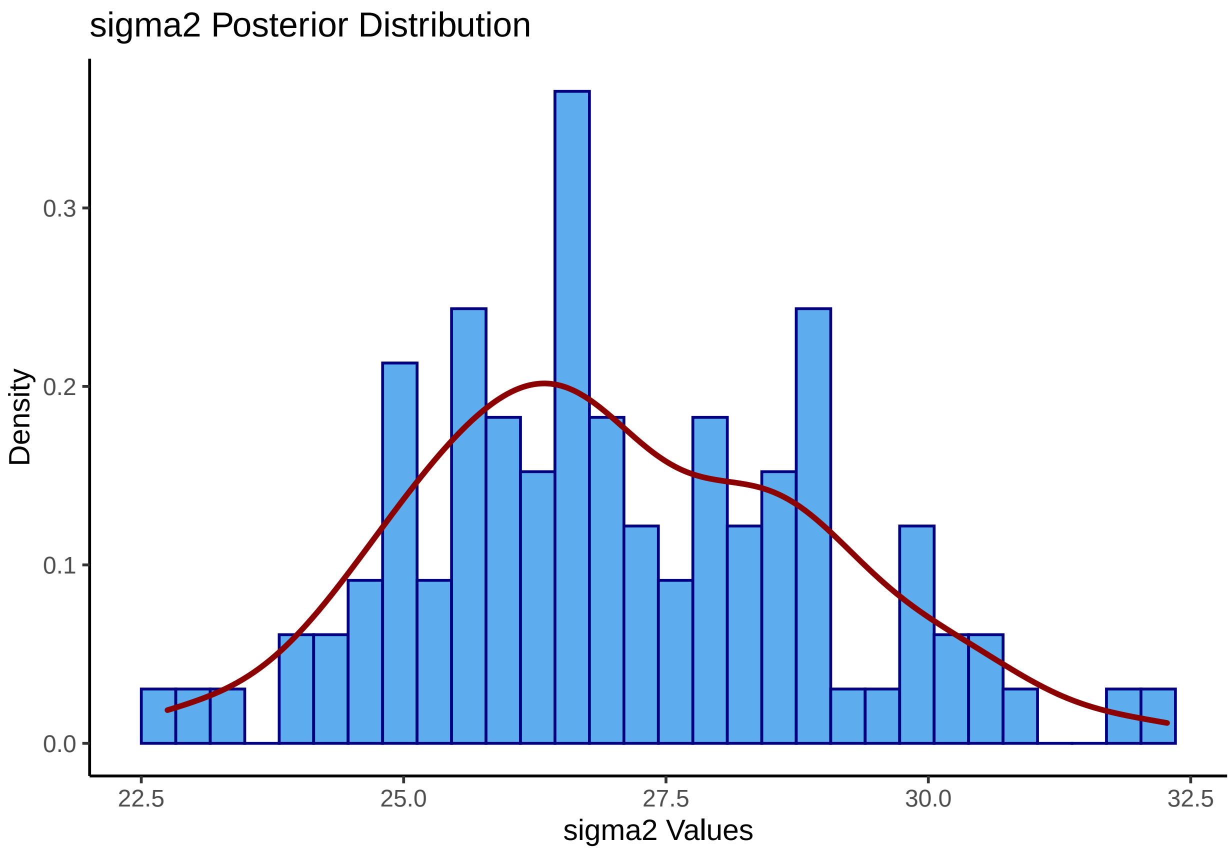
The joint posterior for β and σ^2 is given by:

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

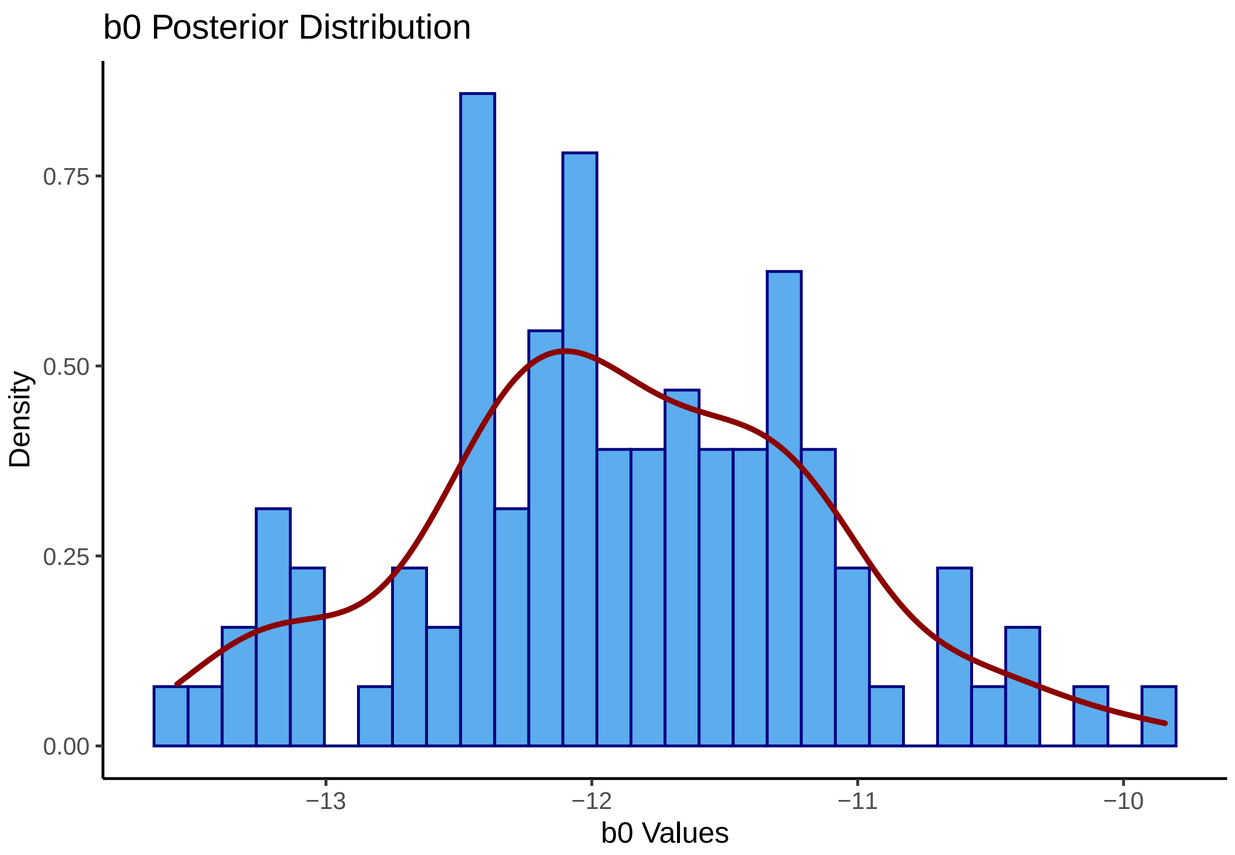
Where:

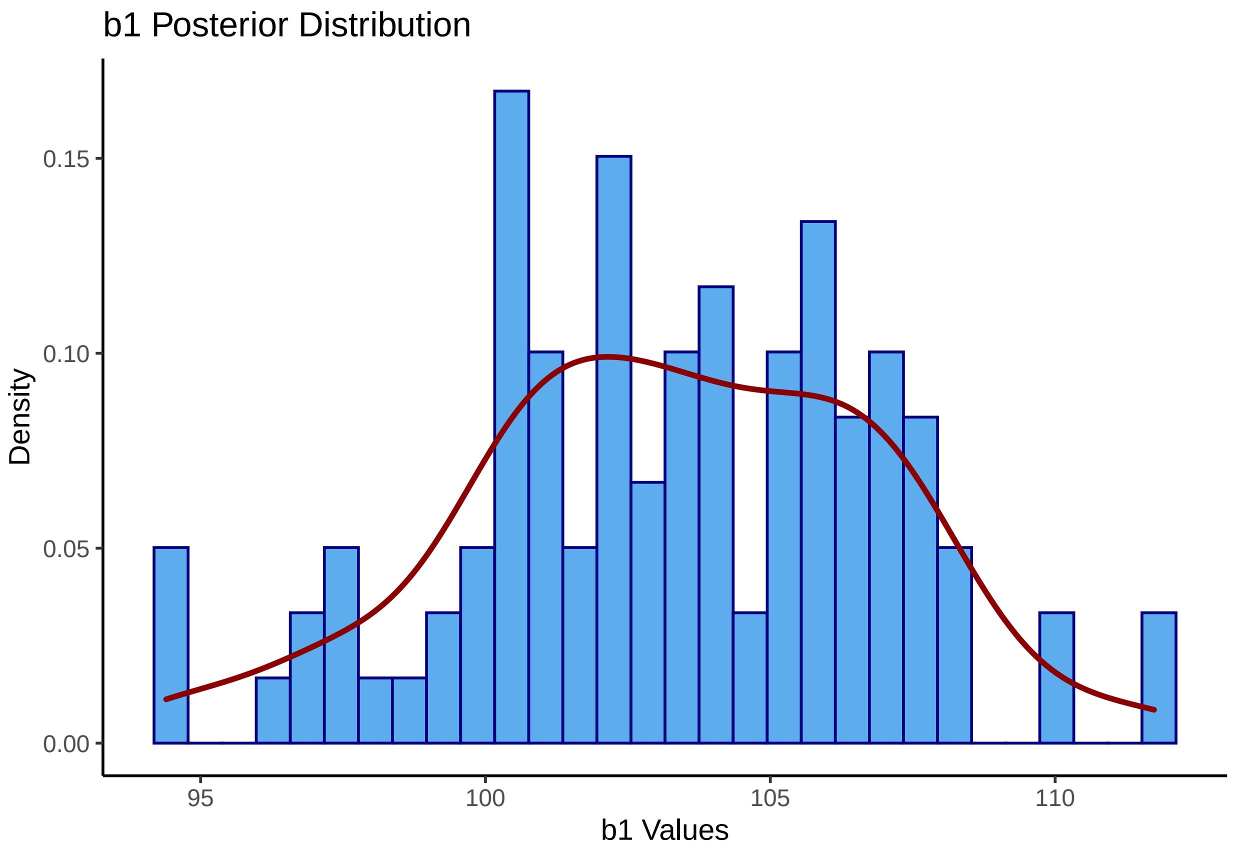
$$\widehat{\beta} = (X'X)^{-1}X'y$$
$$\mu_n = (X'X + \Omega_0)^{-1}(X'X\widehat{\beta} + \Omega_0\mu_0)$$
$$\Omega_n = X'X + \Omega_0$$
$$\nu_n = \nu_0 + n$$
$$\sigma_n^2 = \frac{\nu_0\sigma_0^2 + (y'y + \mu_0'\Omega_0\mu_0 - \mu_n'\Omega_n\mu_n)}{\nu_n}$$

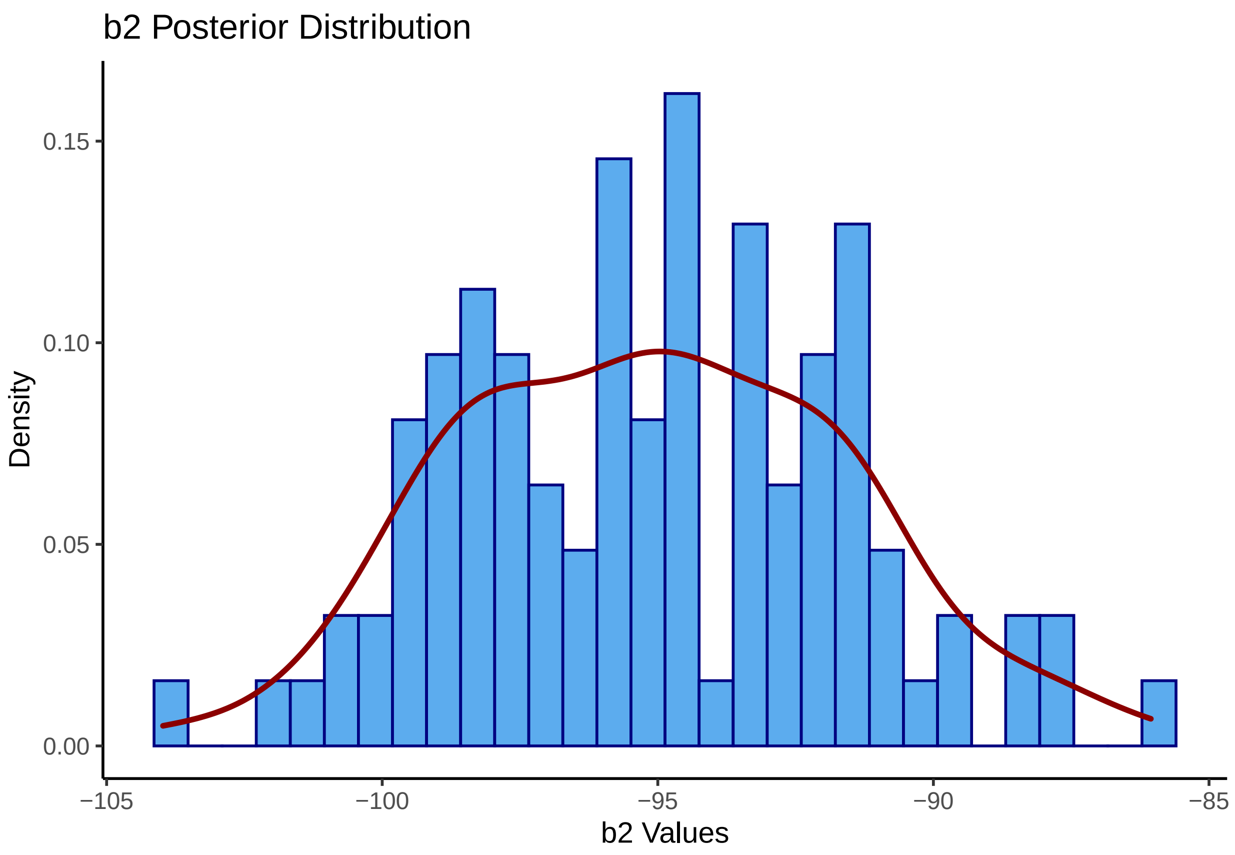
i)



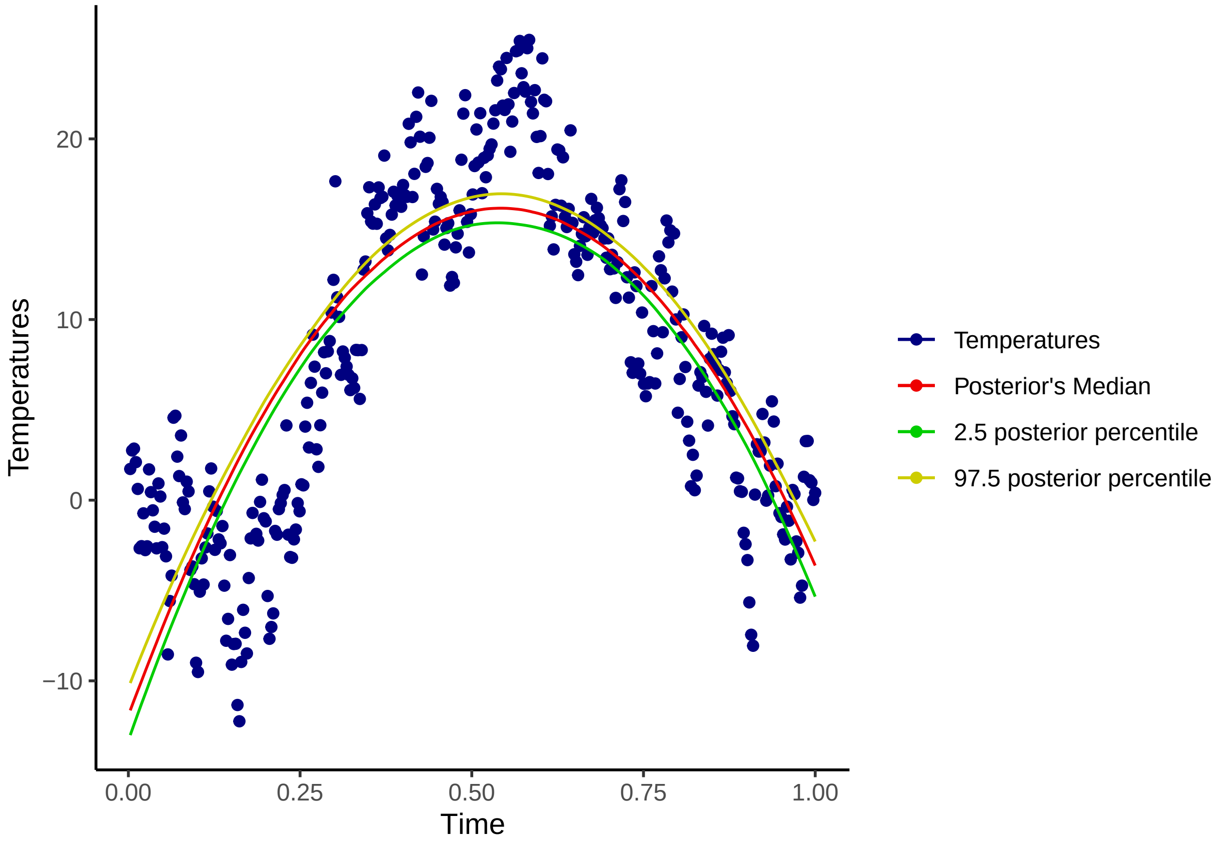
sigma2 Posterior Distribution
Density
0.0
0.1
0.2
0.3
22.5
25.0
27.5
30.0
32.5
sigma2 Values


b0 Posterior Distribution
Density
0.75
0.50
0.25
0.00
−13
−12
−11
−10
b0 Values


b1 Posterior Distribution
Density
0.15
0.10
0.05
0.00
95
100
105
110
b1 Values


b2 Posterior Distribution
Density
0.15
0.10
0.05
0.00
−105
−100
−95
−90
−85
b2 Values

ii)



It is evident that 95% equal tail posterior probability intervals do not contain most data points. It should not contain most data points because the interval illustrates the uncertainty around the median value.

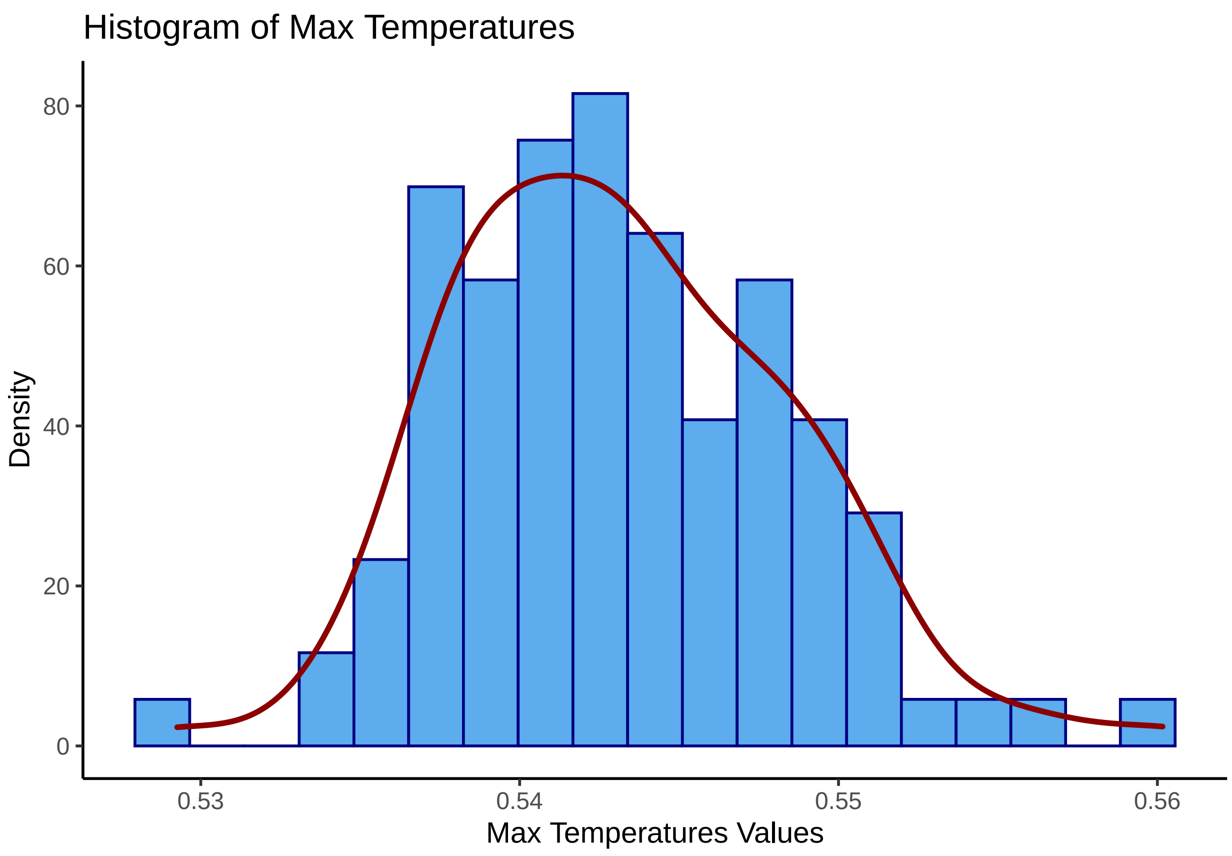
Task 1c

It is given that the regression function is $f(time) = \beta_0 + \beta_1 \cdot time + \beta_2 \cdot time^2$. In order to locate the time with the highest expected temperature; the time, where f(time) is maximal, is needed to be found. Thus, the derivative of $f(time)$ is needed to be found and set it equal to zero to find the maximal.

Set $time = x$; thus, $f(x) = \beta_0 + \beta_1 \cdot x + \beta_2 \cdot x^2$

Calculate the derivative, $f'(x) = \beta_1 + 2 \cdot \beta_2 \cdot x$

Find the maximal: $f'(x) = 0 \Leftrightarrow \beta_1 + 2 \cdot \beta_2 \cdot x = 0 \Leftrightarrow x = \frac{-\beta_1}{2 \cdot \beta_2}$



The time with the highest expected temperature is approximately 0.56.

Task 1d

Estimating a polynomial regression of order 8 with the suspicion that higher-order terms may not be needed because it might lead to overfitting the data. The main problem that could lead to overfitting is the number of knots. A solution to this problem is to introduce a regularised prior, $\beta_i|\sigma^2 \sim N(\mu_0, \frac{\sigma^2}{\lambda})$, where $Omega_0 = \lambda \cdot I_3$. Where the parameter λ will control the variance of β_i. If λ is larger, β_i would be much closer to μ_0.

Assignment 2 *Posterior approximation for classification with logistic regression*

Task 2a

The below table illustrates the $J_y^{-1}(\widetilde{\beta})$.

Constant	HusbandInc	EducYears	ExpYears	Age	NSmallChild	NBigChild
2.5292633	0.0039371	-0.0775073	0.0008893	-0.0341000	-0.2281033	-0.1137008
0.0039371	0.0003915	-0.0008069	-0.0000011	-0.0000516	0.0011500	-0.0000640
-0.0775073	-0.0008069	0.0073415	0.0000610	0.0000496	-0.0080245	0.0019256
0.0008893	-0.0000011	0.0000610	0.0009006	-0.0002492	-0.0009930	0.0006122
-0.0341000	-0.0000516	0.0000496	-0.0002492	0.0008072	0.0060957	0.0012787
-0.2281033	0.0011500	-0.0080245	-0.0009930	0.0060957	0.1918381	0.0094209
-0.1137008	-0.0000640	0.0019256	0.0006122	0.0012787	0.0094209	0.0222163

The below table illustrates the posterior mode values of every feature of the dataset.

	Value
Constant	0.9929529
HusbandInc	-0.0343502
EducYears	0.1794176
ExpYears	0.1230628
Age	-0.0727923
NSmallChild	-1.6227735
NBigChild	-0.0838883

From the above table, it could be seen that the posterior mode of the feature *NSmallChild* has the lowest value; thus, it could be assumed that this variable plays a significant role if a woman is employed.

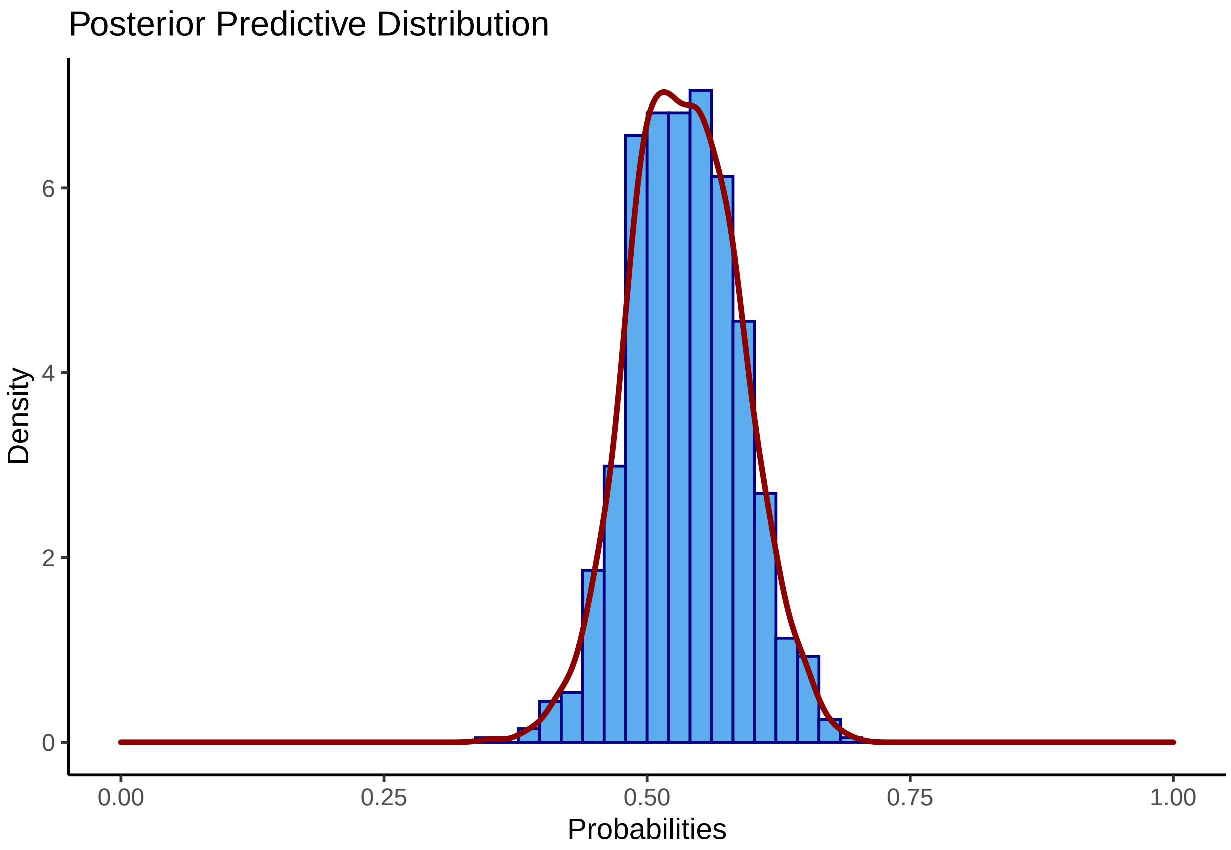
The below table illustrates the approximate posterior standard deviation values of every feature of the dataset.

	Value
Constant	1.5903658
HusbandInc	0.0197857
EducYears	0.0856826
ExpYears	0.0300097
Age	0.0284107
NSmallChild	0.4379933
NBigChild	0.1490514

The approximate 95% equal tail posterior probability interval for the regression coefficient to the variable *NSmallChild* has a lower and an upper bound less than 0; as a result, it strengthens the assumption mentioned above that the *NSmallChild* feature plays a significant role if a woman is employed.

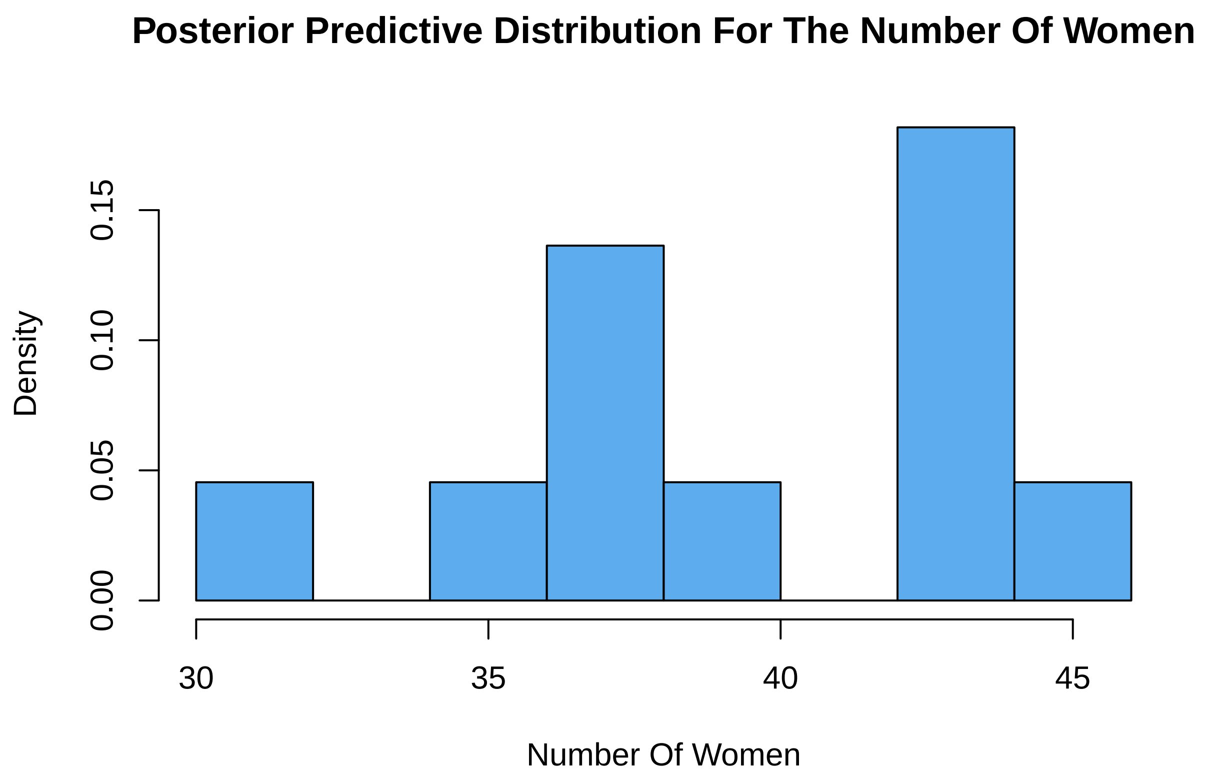
	lower bound	upper bound
95% Equal Tail Credible Interval	-2.445008	-0.8075428

Task 2b



From the above plot, it could be assumed that a 43-year-old woman with two children (7 and 10 years old), 12 years of education, 8 years of experience, and a husband with an income of 20.000 SEK; is more likely to be employed, but there is still a considerable number of occasions that she might without work.

Task 2c



From the above histogram it could be seen that all the observations have a density around 0.5.

Appendix

```r
knitr::opts_chunk$set(echo = TRUE)
knitr::opts_chunk$set(tidy.opts = list(width.cutoff = 60), tidy = TRUE)
library(mvtnorm)
library(ggplot2)
library(tidyr)

# __________________Assignment 1__________________#

# reading data
temperatures <- read.table("TempLambohov.txt", header = TRUE)

# given parameters
m_0 <- c(-10, 100, -100)
omega_0 <- 0.02 * diag(3)
n_0 <- 3
sigma2_0 <- 2

# Task 1a

set.seed(123456)

# number of draws
N <- 10

# matrix to save b0, b1, b2
b <- matrix(NA, nrow = N, ncol = length(m_0))

# matrix to save the predicted temperatures
pred_temp <- matrix(NA, nrow = N, ncol = nrow(temperatures))

for (i in 1:N) {

    # Inv-x simulator from lab 1
    sigma2 <- n_0 * sigma2_0/rchisq(1, n_0)

    # generate b0,b1,b2
    b[i, ] <- rmvnorm(1, mean = m_0, sigma = sigma2 * solve(omega_0))

    # quadratic regression model
    pred_temp[i, ] <- b[i, 1] + b[i, 2] * temperatures$time +
        b[i, 3] * (temperatures$time^2) + rnorm(1, 0, sigma2)
}

# preparing data for plot
rownames(pred_temp) <- c("pred1", "pred2", "pred3", "pred4",
    "pred5", "pred6", "pred7", "pred8", "pred9", "pred10")
pred_temp <- t(pred_temp)

plot_data <- cbind(temperatures, pred_temp)

plot_data <- pivot_longer(plot_data, c(3:12))
```

```r
ggplot(plot_data) + geom_point(aes(x = time, y = temp), color = "navy") +
    geom_line(aes(x = time, y = value, color = name)) + theme(legend.position = "right") +
    guides(color = guide_legend("Predictions")) + xlab("Time") +
    ylab("Temperature") + theme_classic()

# modefied parameters
m_0_mod <- c(-10, 100, -100)
omega_0_mod <- 0.5 * diag(3)  #before 0.2
n_0_mod <- 3
sigma2_0_mod <- 0.2  #before 2

set.seed(123456)

# number of draws
N <- 10

# matrix to save b0, b1, b2
b_mod <- matrix(NA, nrow = N, ncol = length(m_0_mod))

# matrix to save the predicted temperatures
pred_temp_mod <- matrix(NA, nrow = N, ncol = nrow(temperatures))

for (i in 1:N) {

    # Inv-x simulator from lab 1
    sigma2_mod <- n_0_mod * sigma2_0_mod/rchisq(1, n_0)

    # generate b0,b1,b2
    b_mod[i, ] <- rmvnorm(1, mean = m_0_mod, sigma = sigma2_mod *
        solve(omega_0_mod))

    # quadratic regression model
    pred_temp_mod[i, ] <- b_mod[i, 1] + b_mod[i, 2] * temperatures$time +
        b_mod[i, 3] * (temperatures$time^2) + rnorm(1, 0, sigma2_mod)
}

# preparing data for plot
rownames(pred_temp_mod) <- c("pred1", "pred2", "pred3", "pred4",
    "pred5", "pred6", "pred7", "pred8", "pred9", "pred10")

pred_temp_mod <- t(pred_temp_mod)

plot_data_mod <- cbind(temperatures, pred_temp_mod)

plot_data_mod <- pivot_longer(plot_data_mod, c(3:12))

ggplot(plot_data_mod) + geom_point(aes(x = time, y = temp), color = "navy") +
    geom_line(aes(x = time, y = value, color = name)) + theme(legend.position = "right") +
    guides(color = guide_legend("Predictions")) + xlab("Time") +
    ylab("Temperature") + theme_classic()

# Task 1b i
```

```r
set.seed(123456)

# matrix with the times (1st collum with 1 because of the
# intercept)
X <- cbind(rep(1, nrow(temperatures)), temperatures$time, temperatures$time^2)
# vector with target values
y <- temperatures$temp
# calculate b hat
b_hat <- solve(t(X) %*% X) %*% t(X) %*% y

# number of samples
N <- 100

# matrix to store the beta values
b <- matrix(0, N, length(m_0))
# vector to store sigma2 values
sigma2 <- c()

for (i in 1:N) {

    # calculate mu_n
    m_n <- solve(t(X) %*% X + omega_0) %*% (t(X) %*% X %*% b_hat +
        omega_0 %*% m_0)

    # calculate omega_n
    omega_n <- t(X) %*% X + omega_0

    # calculate n_n
    n_n <- n_0 + nrow(temperatures)

    # calculate sigma_n
    sigma2_n <- (n_0 * sigma2_0 + (t(y) %*% y + t(m_0) %*% omega_0 %*%
        m_0 - t(m_n) %*% omega_n %*% m_n))/n_n

    # Inv-x simulator from lab 1
    sigma2[i] <- n_n * sigma2_n/rchisq(1, n_n)

    # generate b0,b1,b2
    b[i, ] <- rmvnorm(1, mean = m_n, sigma = sigma2[i] * solve(omega_n))

}

# df of sigma2 for plot
plot_df_sigma2 <- data.frame(sigma2 = sigma2)

# histogram of sigma2
ggplot(plot_df_sigma2, aes(x = sigma2)) + geom_histogram(bins = 30,
    color = "navy", fill = "steelblue2", aes(y = ..density..)) +
    geom_density(colour = "red4", size = 1) + ggtitle("sigma2 Posterior Distribution") +
    xlab("sigma2 Values") + ylab("Density") + theme_classic()

# df of betas for plot
plot_df_b <- data.frame(b0 = b[, 1], b1 = b[, 2], b2 = b[, 3])
```

```r
# histogram of b0
ggplot(plot_df_b, aes(x = b0)) + geom_histogram(bins = 30, color = "navy",
    fill = "steelblue2", aes(y = ..density..)) + geom_density(colour = "red4",
    size = 1) + ggtitle("b0 Posterior Distribution") + xlab("b0 Values") +
    ylab("Density") + theme_classic()

# histogram of b1
ggplot(plot_df_b, aes(x = b1)) + geom_histogram(bins = 30, color = "navy",
    fill = "steelblue2", aes(y = ..density..)) + geom_density(colour = "red4",
    size = 1) + ggtitle("b1 Posterior Distribution") + xlab("b1 Values") +
    ylab("Density") + theme_classic()

# histogram of b2
ggplot(plot_df_b, aes(x = b2)) + geom_histogram(bins = 30, color = "navy",
    fill = "steelblue2", aes(y = ..density..)) + geom_density(colour = "red4",
    size = 1) + ggtitle("b2 Posterior Distribution") + xlab("b2 Values") +
    ylab("Density") + theme_classic()

# Task 1b ii

# matrix to store the predicted values
pred_temp2 <- matrix(NA, nrow(b), nrow(temperatures))

# calculate the predicted values
for (i in 1:nrow(b)) {
    pred_temp2[i, ] <- b[i, 1] + b[i, 2] * temperatures$time +
        b[i, 3] * (temperatures$time^2)
}

# calculate posterior's median for every value of time
post_median <- c()
for (i in 1:ncol(pred_temp2)) {
    post_median[i] <- median(pred_temp2[, i])
}

# calculate the 2.5 and 97.5 posterior percentiles of f
# (time)
intervals <- matrix(NA, nrow = 2, ncol = ncol(pred_temp2))
for (i in 1:ncol(pred_temp2)) {
    intervals[, i] <- quantile(pred_temp2[, i], probs = c(0.025,
        0.975))
}

# df for plot
plot_df_1b2 <- data.frame(temperatures = temperatures$temp, time = temperatures$time,
    interval1 = intervals[1, ], interval2 = intervals[2, ], medians = post_median)

# scatterplot of the temperatures & curve of the posterior
# median & curves for the 95% equal tail posterior
# probability intervals
ggplot(data = plot_df_1b2) + geom_point(aes(x = time, y = temperatures,
    color = "navy")) + geom_line(aes(x = time, y = medians, color = "red2")) +
    geom_line(aes(x = time, y = interval1, color = "green3")) +
    geom_line(aes(x = time, y = interval2, color = "yellow3")) +
    theme(legend.position = "right") + scale_color_identity(guide = "legend",
    name = "", breaks = c("navy", "red2", "green3", "yellow3"),
    labels = c("Temperatures", "Posterior's Median", "2.5 posterior percentile",
        "97.5 posterior percentile")) + xlab("Time") + ylab("Temperatures") +
    theme_classic()

# Task 1c

# getting the time with the highest expected temperature by
# using the above expression
max_temp <- c()

for (i in 1:N) {
    max_temp[i] <- max(-b[i, 2]/(2 * b[i, 3]))
}

df_max <- data.frame(max_temp = max_temp)

# histogram of max temps
ggplot(df_max, aes(x = max_temp)) + geom_histogram(bins = 19,
    color = "navy", fill = "steelblue2", aes(y = ..density..)) +
    geom_density(colour = "red4", size = 1) + ggtitle("Histogram of Max Temperatures") +
    xlab("Max Temperatures Values") + ylab("Density") + theme_classic()
# __________________Assignment 2__________________#

# reading data
women <- read.table("WomenAtWork.dat", header = TRUE)

# Task 2a

# given parameteres
tau <- 5

# the below code snippets from a demo of logistic
# regression in Lecture 6

# target value
y <- women$Work

# prior inputs Select which covariates/features to include
X <- as.matrix(women[2:8])
Xnames <- colnames(X)

Npar <- dim(X)[2]

# Setting up the prior
mu <- as.matrix(rep(0, Npar))  # Prior mean vector
Sigma <- tau^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 here!
    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 minimize
# 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),
    hessian = TRUE)

names(OptimRes$par) <- Xnames  # Naming the coefficient by covariates
approxPostStd <- sqrt(diag(-solve(OptimRes$hessian)))  # Computing approximate standard deviations.
names(approxPostStd) <- Xnames  # Naming the coefficient by covariates
# print('The posterior mode is:') print(OptimRes$par)
# print('The approximate posterior standard deviation is:')
approxPostStd <- sqrt(diag(-solve(OptimRes$hessian)))
# print(approxPostStd)

invHessian <- data.frame(invhes = -solve(OptimRes$hessian))
colnames(invHessian) <- c("Constant", "HusbandInc", "EducYears",
    "ExpYears", "Age", "NSmallChild", "NBigChild")
knitr::kable(invHessian)

df_post_mode <- data.frame(post_mode = OptimRes$par)
colnames(df_post_mode) <- c("Value")
rownames(df_post_mode) <- c("Constant", "HusbandInc", "EducYears",
    "ExpYears", "Age", "NSmallChild", "NBigChild")
knitr::kable(df_post_mode)

df_approxPostStd <- data.frame(approxPostStd = sqrt(diag(-solve(OptimRes$hessian))))
colnames(df_approxPostStd) <- c("Value")
rownames(df_approxPostStd) <- c("Constant", "HusbandInc", "EducYears",
    "ExpYears", "Age", "NSmallChild", "NBigChild")
knitr::kable(df_approxPostStd)

# draw b
b_draws <- rmvnorm(n = 1000, mean = OptimRes$par, sigma = -solve(OptimRes$hessian))

# calculate quantiles
interval <- quantile(b_draws[, 6], c(0.025, 0.975))
```

```r
df_interval <- data.frame(lower_bound = interval[1], upper_bound = interval[2])
colnames(df_interval) <- c("lower bound", "upper bound")
rownames(df_interval) <- c("95% Equal Tail Credible Interval")
knitr::kable(df_interval)

set.seed(1234567)
women2 <- c(1, 20, 12, 8, 43, 0, 2)

# function that classifies if the women work or not
prediction <- function(N, data, posterior_mode, posterior_covariates) {

    # generate betas
    b <- rmvnorm(N, mean = posterior_mode, sigma = posterior_covariates)

    # calculating the the logistic regression model
    res <- exp(data %*% t(b))/(1 + exp(data %*% t(b)))

    return(res)
}

# predictions
pred <- prediction(1000, women2, OptimRes$par, -solve(OptimRes$hessian))

# df for plot
pred_women <- data.frame(pred = t(pred))

ggplot(pred_women, aes(x = pred)) + geom_histogram(bins = 50,
    color = "navy", fill = "steelblue2", aes(y = ..density..)) +
    geom_density(colour = "red4", size = 1) + scale_x_continuous(limits = c(0,
    1)) + ggtitle("Posterior Predictive Distribution") + xlab("Probabilities") +
    ylab("Density") + theme_classic()

set.seed(1234567)

prediction2 <- function(N, data, posterior_mode, posterior_covariates,
    nwomen) {

    # generate betas
    b <- rmvnorm(N, mean = posterior_mode, sigma = posterior_covariates)

    # calculating the the logistic regression model
    pred <- exp(data %*% t(b))/(1 + exp(data %*% t(b)))

    # vector to store results
    res <- c()

    for (i in 1:nwomen) {
        # binomial distribution
        res[i] <- sum(rbinom(n = 7, size = 11, prob = pred))
    }

    return(res)
}
```

```r
# predictions
pred2 <- prediction2(1000, women2, OptimRes$par, -solve(OptimRes$hessian),
    11)

# pred_women2 <- data.frame('pred' = pred2)
# ggplot(pred_women2 , aes(x=pred)) + geom_histogram(color
# = 'navy', fill = 'steelblue2') + ggtitle('Posterior
# Predictive Distribution For The Number Of Women') +
# xlab('Number Of Women') + ylab('Density') +
# theme_classic()

hist(pred2, col = "steelblue2", main = "Posterior Predictive Distribution For The Number Of Women",
    xlab = "Number Of Women", ylab = "Density", probability = TRUE)
```