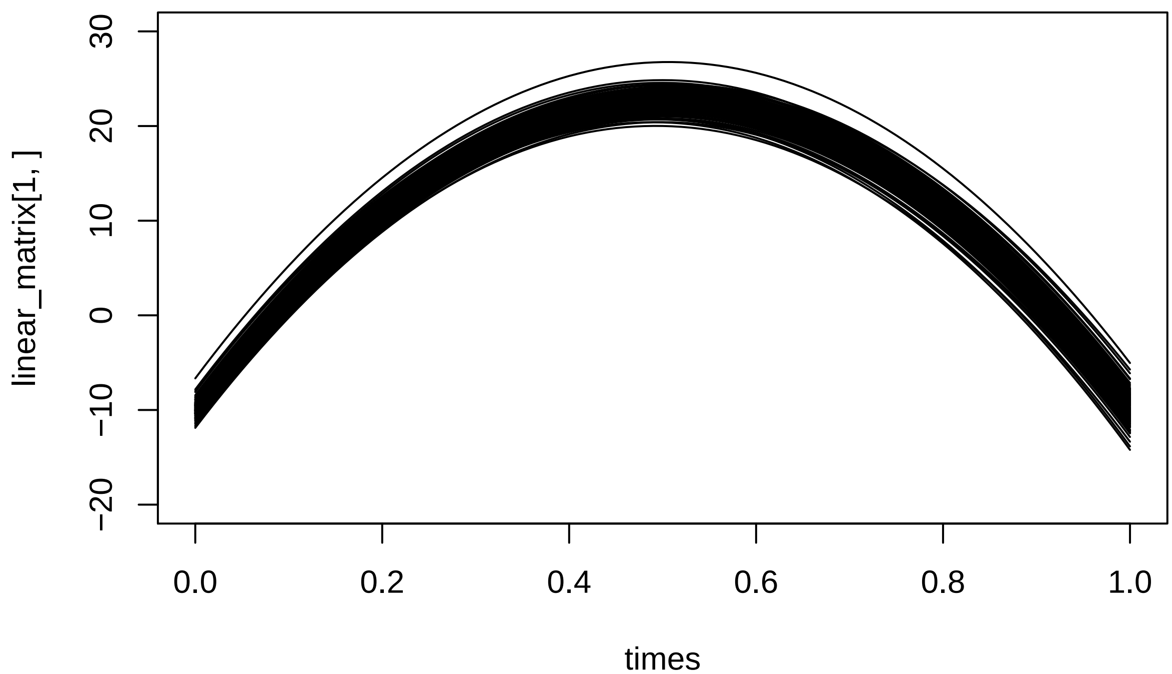
Lab 2 - TDDE07

Axel Holmberg (axeho681), Wilhelm Hansson (wilha431)

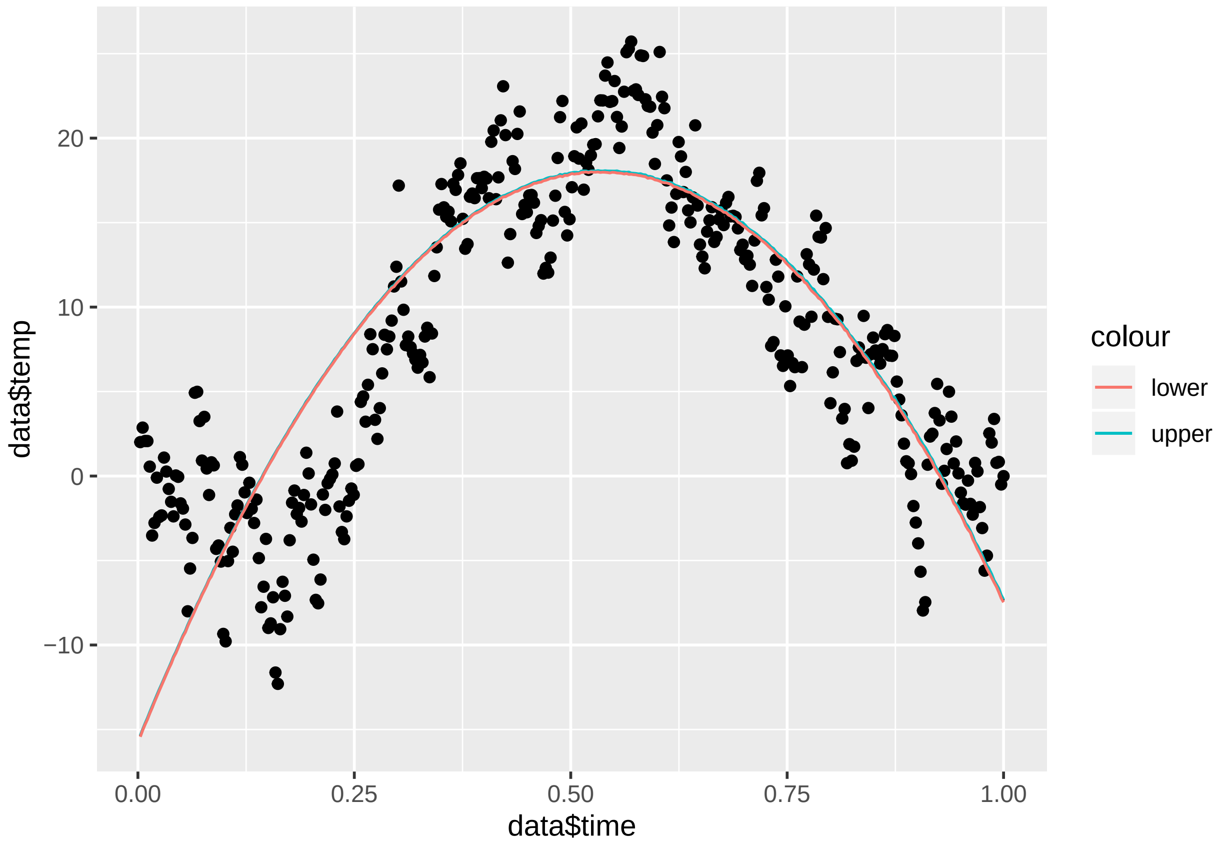
1.

a)



The plot above shows the regression curve for the temperature. The tuning of the joint prior parameters was decided by looking at the values of the actual data. From the data the mean, min and max values were identified. Further the knowledge that 0 = January and 1 = December was used to unnderstand when the peak of the quadratic curve wsa suppose to be (around 0.5 ~ June). So, variance between the individual curves were allowed to avoid to much bias.

b)



The band does not contain most of the data points. This should be the case as it is the confidence interval of our posterior and not of the data points.

c)

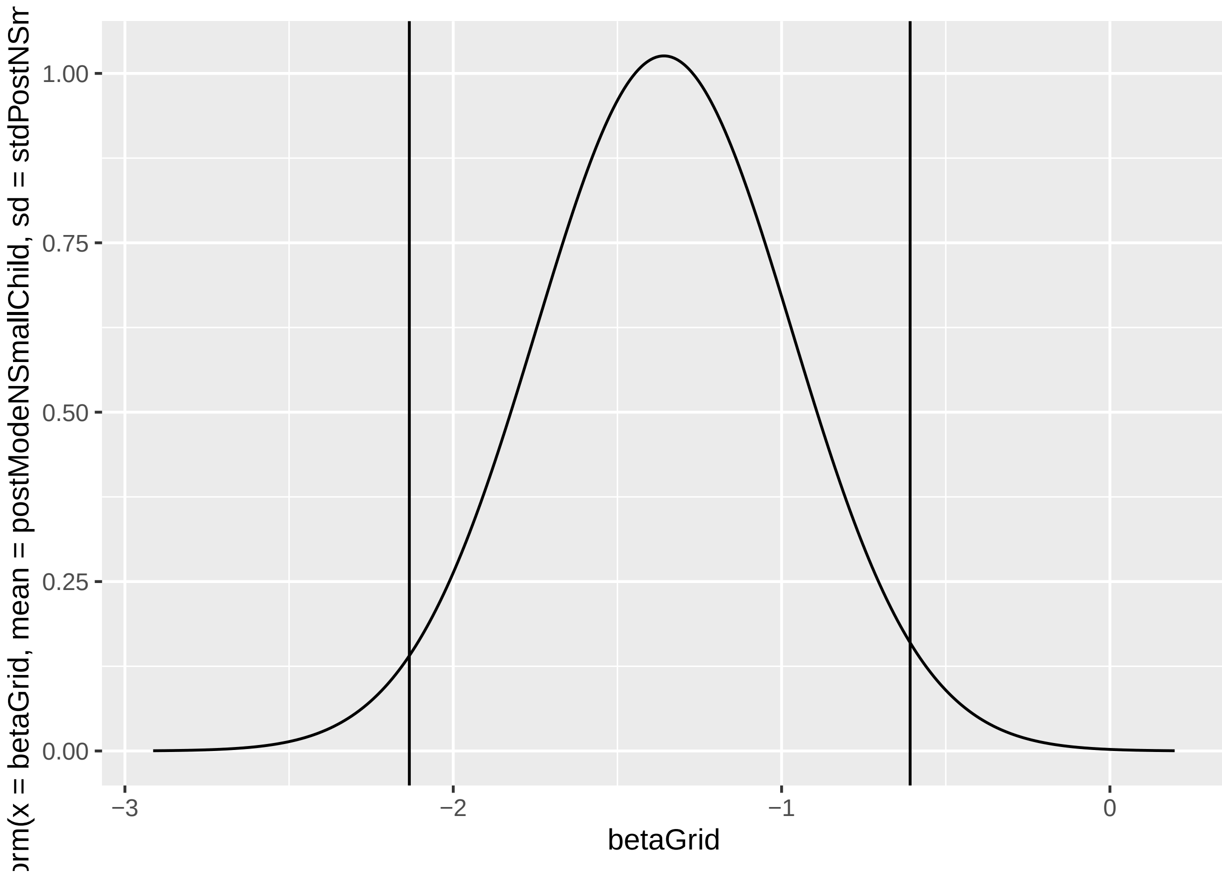
$\tilde{x} = 0.517808$, which means that the hottest day is sometime in the middle of July.

d)

A suitable prior to mitigate the potential problem of overfitting would be to set parameters of μ_0 and Ω_0 to values the would result in a $\beta = 0$ for the higher order terms. In this case the $\mu_{4...7} = 0$, and since it is Ω^{-1} that is used the values of Ω should be large.

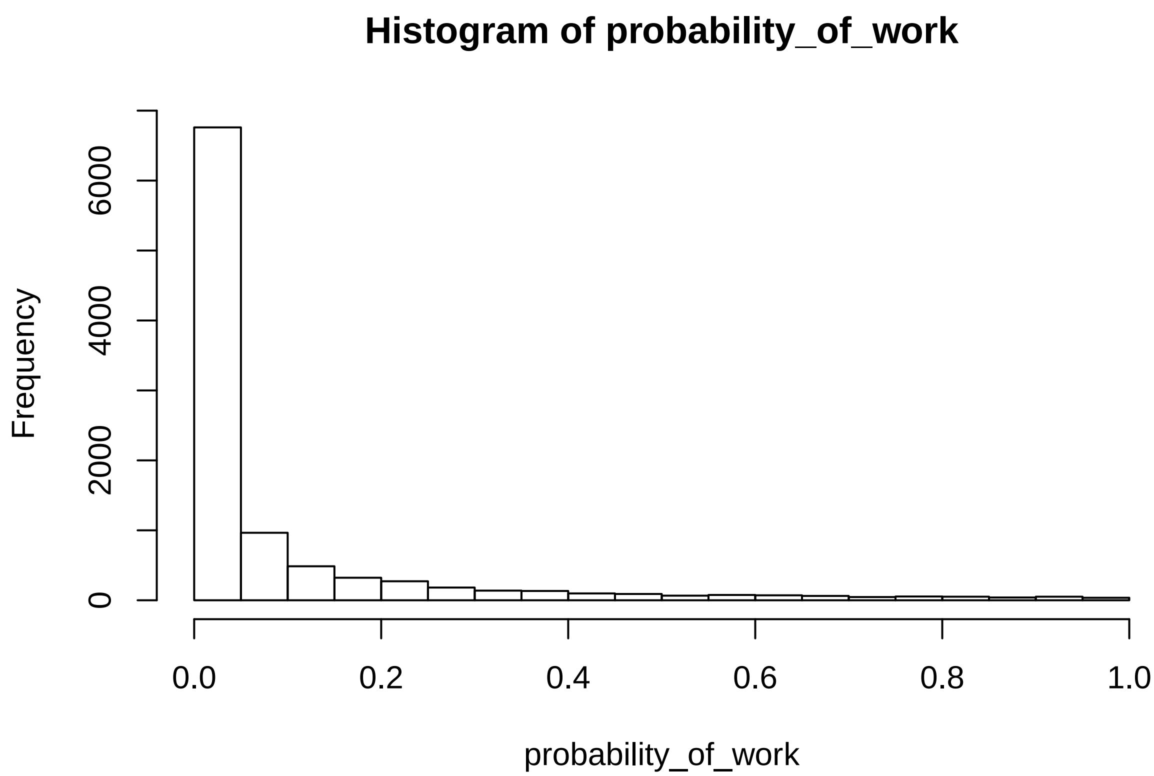
2.

a)



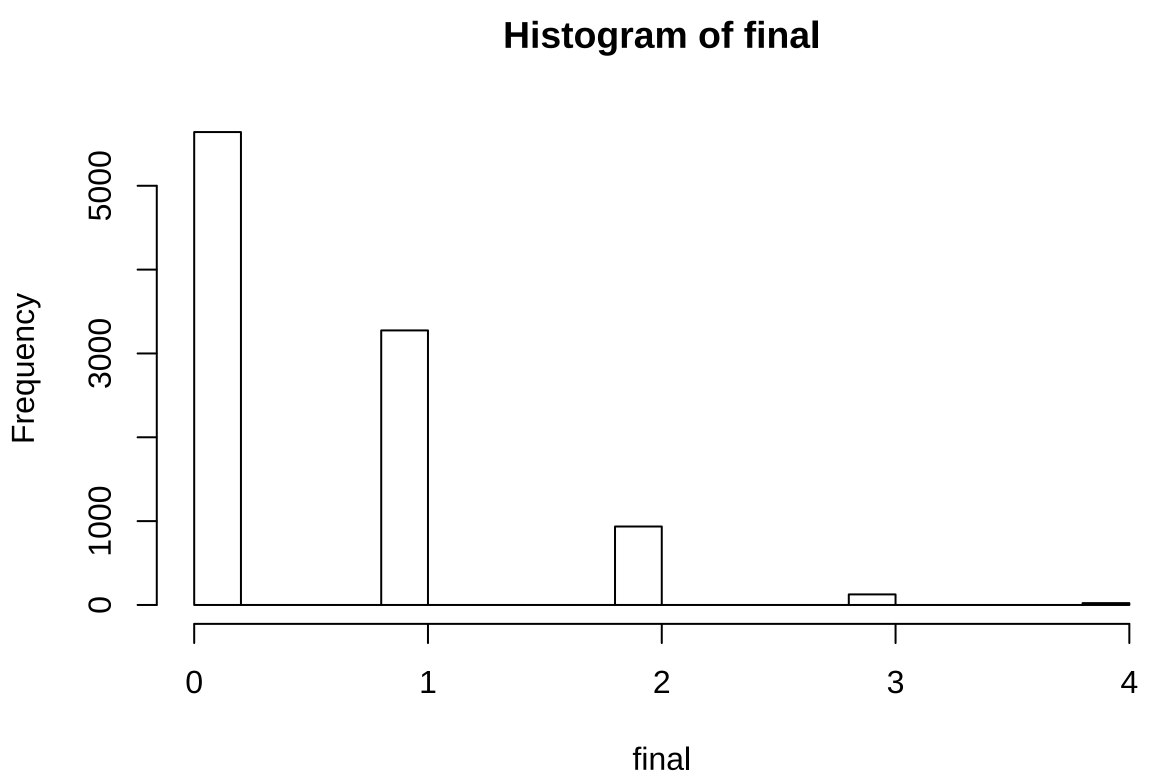
As the variable for NSmallChild has a 95 % confidence interval between -2.1336941 and -0.6084974 we would consider this feature as an important determinant for the probability wether a woman works or not.

b)



As one can see in the plot above there is a low probability for a women to work given the feature vector.

c)



Considering our plot above it can be expected that in most of the cases none of the 10 women will be working, and in this case a maximum of four women will work.

Appendix for code

```
setwd("~/Programming/TDDE07/Lab 2")
library(ggplot2)
data <-
    read.delim("TempLinkoping.txt",
                     header = TRUE,
                     sep = "\t",
                     dec = ".")

library(LaplacesDemon)
library(mvtnorm)
set.seed(12345)
my_0 <- matrix(c(-10, 130, -130), nrow = 3, ncol = 1)

omega_0 <- 3 * diag(3)

v_0 <- 10
sigma_sq_0 <- 2

number_of_draws <- 200

sigma_sq <- rinvchisq(number_of_draws, v_0, sigma_sq_0)

omega_0.inv <- solve(omega_0)


betas <- c()


for (sigma in sigma_sq) {
    beta <- rmvnorm(1, my_0, sigma * omega_0.inv)
    #tmp <- cbind(sigma, beta)
    betas <- rbind(betas, beta)
}


linear_matrix <- c()

times <- seq(0, 1, 0.01)
for (rownumber in 1:nrow(betas)) {
    tmprow <- c()
    for (time in times) {
        tmp <- betas[rownumber, 1] + betas[rownumber, 2] * time + betas[rownumber, 3] *
            time ^ 2
        tmprow <- cbind(tmprow, tmp)
    }
    linear_matrix <- rbind(linear_matrix, tmprow)
}

plot(times,linear_matrix[1,], type='l', ylim=c(-20,30))
for (i in 2:nrow(linear_matrix)) {
    lines(times, linear_matrix[i,])
```

```r
}

library(Rmisc)
X <- cbind(rep.int(1, 365), data[, 1], I(data[, 1] ^ 2))
Y <- data[, 2]

beta_hat <- solve((t(X) %*% X)) %*% t(X) %*% Y
my_n <-
    solve(t(X) %*% X + omega_0) %*% (t(X) %*% X %*% beta_hat + omega_0 %*% my_0)
omega_n <- t(X) %*% X + omega_0
v_n <- v_0 + nrow(X)
v_n_sigma_sq_n <-
    v_0 * sigma_sq_0 + (t(Y) %*% Y + t(my_0) %*% omega_0 %*% my_0 - t(my_n) %*%
                                                omega_n %*% my_n)
n_draws <- 200

sigmas_sq <- rinvchisq(n_draws, v_n, v_n_sigma_sq_n / v_n)
hist(sqrt(sigmas_sq))


betas_posterior <- c()
for (sigma_sq in sigmas_sq) {
    beta_posterior <- rmvnorm(1, my_n, sigma_sq * solve(omega_n))
    betas_posterior <- rbind(betas_posterior, beta_posterior)
}

hist(betas_posterior[, 1], xlab = "B0 (intercept)")
hist(betas_posterior[, 2], xlab = "B1 (slope)")
hist(betas_posterior[, 3], xlab = "B2 (curve)")

calctemp <- function(x) {
    return(
        median(betas_posterior[, 1]) + median(betas_posterior[, 2]) * x + median(betas_posterior[, 3]) *
            x ^ 2
    )
}



conf_i <- function(betas) {
    tmp <- sort(betas)
    tmp_cols <- cbind(tmp[,0.025 * nrow(betas)], tmp[,0.975 * nrow(betas)])
    return(tmp_cols)
}

out <- c()
for (time in data$time) {
    ys <- c()
    for (i in 1:10) { #INCREASE
        betas_posterior_tmp <- c()
        for (sigma_sq in sigmas_sq) {
            beta_posterior_tmp <- rmvnorm(1, my_n, sigma_sq * solve(omega_n))
            betas_posterior_tmp <-
```

```r
                rbind(betas_posterior_tmp, beta_posterior_tmp)
        }

        b0m_tmp <- median(betas_posterior_tmp[, 1])
        b1m_tmp <- median(betas_posterior_tmp[, 2])
        b2m_tmp <- median(betas_posterior_tmp[, 3])

        y <-  b0m_tmp + b1m_tmp * time + b2m_tmp * time ^ 2
        ys <- cbind(ys, y)
    }

    out <- rbind(out,ys)

}

cis <- c()
for (i in 1:nrow(data)) {
    ci_tmp <- CI(out[i,])
    cis <- rbind(cis,cbind(ci_tmp[1],median(out[i,]),ci_tmp[3]))

}


ggplot() + geom_point(aes(x = data$time, y = data$temp)) + geom_line(aes(x =
    data$time, y = cis[, 2])) + geom_line(aes(x = data$time, y = cis[, 1], color =
    "upper")) + geom_line(aes(x = data$time, y = cis[, 3], color = "lower"))

x_tilde <- data$time[which(max(cis[,2]) == cis[,2])]


################################### Ass 2 #####################################################
library("mvtnorm") # This command reads the mvtnorm package into R's memory. NOW we can use dmvnorm fun

data <-
    read.table("WomenWork.dat", header = TRUE)  # Spam data from Hastie et al.

y <- as.vector(data$Work)
X <- as.matrix(data[, 2:9])


tau <- 10
mu <- rep(0, 8)
sigma <- tau ^ 2 * diag(8)

initVal <- rmvnorm(1, mu, sigma)



LogPostLogistic <- function(betaVect, y, X, mu, Sigma) {
    nPara <- length(betaVect)

    linPred <- X %*% betaVect
```

```r
    print(betaVect)
    # evaluating the log-likelihood
    logLik <- sum(linPred * y - log(1 + exp(linPred)))

    if (abs(logLik) == Inf)
        logLik = -20000
    # Likelihood is not finite, stear the optimizer away from here!

    # evaluating the prior
    logPrior <- dmvnorm(betaVect, matrix(0, nPara, 1), Sigma, log = TRUE)

    print(logLik + logPrior)
    print(betaVect)
    # add the log prior and log-likelihood together to get log posterior
    return(logLik + logPrior)
}


OptimResults <-
    optim(
        initVal,
        LogPostLogistic,
        gr = NULL,
        y,
        X,
        mu,
        sigma,
        method = c("BFGS"),
        control = list(fnscale = -1),
        hessian = TRUE
    )

inverse_hessian <- solve(OptimResults$hessian)

postModeNSmallChild <- OptimResults$par[7]
stdPostNSmallChild <- sqrt(diag(-inverse_hessian)[7])



betaGrid <-
    seq(
        postModeNSmallChild - 4 * stdPostNSmallChild,
        postModeNSmallChild + 4 * stdPostNSmallChild,
        length = 1000
    )
ciBetaNsmallChild <-
    quantile(rnorm(10000, mean = postModeNSmallChild, sd = stdPostNSmallChild),
                     c(0.025, 0.975))
ggplot() + geom_line(aes(
    x = betaGrid,
    y = dnorm(x = betaGrid, mean = postModeNSmallChild, sd = stdPostNSmallChild)
)) + geom_vline(xintercept = ciBetaNsmallChild[1]) + geom_vline(xintercept = ciBetaNsmallChild[2])
```

```r
glmModel <- glm(Work ~ 0 + ., data = data, family = binomial)

glmSmallChildVar <- glmModel[["coefficients"]][["NSmallChild"]]

#b)

beta_posterior <- OptimResults$par

features <- c(1, 10.0, 8, (8 / 10) ^ 2 , 10, 40, 1, 1)

prob_working_women <-
    (exp(t(features) %*% t(beta_posterior))) / (1 + exp(t(features) %*% t(beta_posterior)))

func_working_women_prob <- function(feature, betaposterior) {
    tmp <-
        (exp(t(feature) %*% betaposterior)) / (1 + exp(t(feature) %*% betaposterior))
    return(tmp)
}

stdbeta <- diag(-inverse_hessian)

sim_beta <-
    rmvnorm(10000,
                    mean = beta_posterior,
                    sigma = -solve(OptimResults$hessian))

probability_of_work <- c()
for (i in 1:nrow(sim_beta)) {
    probability_of_work <-
        rbind(probability_of_work, (func_working_women_prob(features, sim_beta[i, ])))
}
hist(probability_of_work)


### c)
classified_women <- c()
p_women <- rep(0,10)

for (i in 1:10) {
    probability_of_work <- c()
    sim_beta <-
        rmvnorm(10000,
                        mean = beta_posterior,
                        sigma = -solve(OptimResults$hessian))


    for (j in 1:nrow(sim_beta)) {
        probability_of_work <-
            cbind(probability_of_work, (func_working_women_prob(features, sim_beta[j, ])))
    }

    probability_of_work <- ifelse(probability_of_work>=0.5,1,0)
    #p_women[i] <- sum(probability_of_work)/10000
```

```r
    classified_women <- rbind(classified_women,probability_of_work)

}


final <- c()
for(i in 1:10000){
    final <- cbind(final,sum(classified_women[,i]))
}
hist(final)
```