Freie Universität Berlin
Chair of Statistics
Location: Berlin
Summer Term 2017
Lecture: Einführung in die Bayes-Statistik
Examiner: Dr. Florian Meinfelder

Program leave-one-out posterior predictive checking in R

Johannes Brinkmann (), jojo-brinkmann@gmx.de
Carlo Michaelis (5043128), carlo.michaelis@gmail.com
Max Reinhardt (), max_reinhardt@me.com
Adrian Rolf (), adrian.rolf@gmx.de

Master Statistics
August 16, 2017

Contents

List of Figures

List of Tables

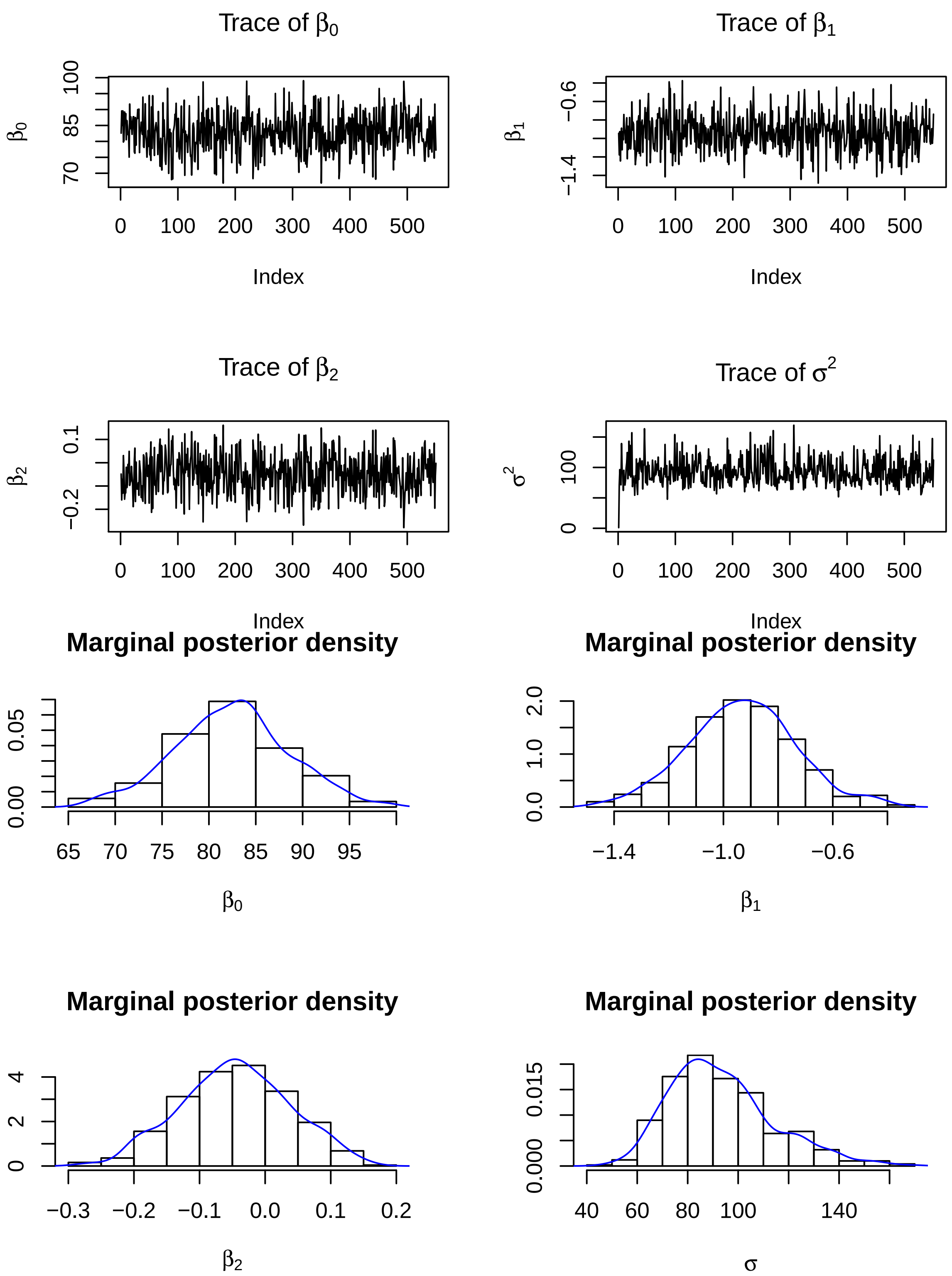
1 Introduction

2 Code

```r
# Swiss data
dat <- swiss

# Response variable
Y <- dat$Fertility

# Design matrix
n <- nrow(dat)
X <- matrix(c(rep(1,n), dat$Education, dat$Agriculture), nrow=n)
p <- ncol(X)

# Number of samples
b <- 50   # Burn in
R <- 500 # Random draws to evaluate
B <- R + b

plotSampling <- function(betas, sigma, traces = TRUE, density = TRUE) {
  # Get number of parameters and adjust plot frame height
  q <- ncol(betas) + 1
  frameRows <- round(q/2+0.5)

  # Traces
  if(traces == TRUE) {
    par(mfrow = c(2,frameRows))
    for(i in 1:ncol(betas)) {
      plot(betas[,i], type='l', ylab=bquote(beta[.(i-1)]), main=bquote("Trace of" ~ beta[.(i-1)]))
    }
    plot(sigma, type='l', ylab=bquote(sigma^2), main=bquote("Trace of" ~ sigma^2))
  }

  # Marginal posterior densities (remove burn in)
  if(density == TRUE) {
    # Function to draw plot
    drawHistDensity <- function(para, para_name) {
      # para      : Parameter (e.b. Beta, Sigma)
      # para_name: Title of plot

      # Estimate density for parameter values
      density <- density(para)

      # Draw histogram and add estimated density line
      hist(para, freq = FALSE, ylim = c(0,max(density$y)), xlab = para_name,
           ylab=NULL, main = "Marginal posterior density")
      lines(density, col="blue")
    }

    # Adjust frame and plot all parameters
    par(mfrow = c(2,frameRows))
    for(i in 1:ncol(betas)) {
      drawHistDensity(betas[-(1:b),i], bquote(beta[.(i-1)]))
    }
    drawHistDensity(sigma[-(1:b)], bquote(sigma))
```

```
  }
}

# The Gibbs Sampler
gibbsSampler <- function(X, Y, B, show = FALSE) {
  # Size of design matrix
  n <- nrow(X)
  p <- ncol(X)

  # Variables to store the samples in
  betas <- matrix(NA, nrow = B, ncol = p)
  sigma <- c(1, rep(NA, B))

  # Sampling
  for(i in 1:B){
    # OLS of beta
    V <- solve(t(X)%*%X)     # (X^T X)^-1
    beta_hat <- V%*%t(X)%*%Y # (X^T X)^-1 X^T Y

    # OLS of sigma
    sigma_hat <- t(Y-X%*%beta_hat)%*%(Y-X%*%beta_hat)/(n-p)

    # Sample beta from the full conditional
    betas[i,] <- rmvnorm(1,beta_hat,sigma[i]*V)

    # Sample sigma from the full conditional
    sigma[i+1] <- 1/rgamma(1,(n-p)/2,(n-p)*sigma_hat/2)
  }

  if(show == TRUE) {
    plotSampling(betas, sigma)
  }

  return(list(betas = betas, sigma = sigma))
}

crossValidation <- function(X, Y, B) {
  Yhat <- rep(NA, n)
  betas <- matrix(NA, nrow = n, ncol = p)

  for(i in 1:n) {
    # Remove i-th row from data
    Xi <- X[-i,]
    Yi <- Y[-i]

    # Run gibbs sampler to get sampled parameters and plot results from first run
    res <- gibbsSampler(Xi, Yi, B, ifelse(i == 1, TRUE, FALSE))

    # Calculate posterior mean from sampled betas
    betas[i,] <- apply(res$betas, 2, mean)

    # Predict value with posterior mean
    Yhat[i] <- X[i,]%*%betas[i,]
  }

  # Calculate beta estimate
  beta_cv <- mean(betas)
```

```
  # Calculate MSE
  mse <- sum((Y-Yhat)^2)

  return(list(betas = betas, mse = mse))
}

res <- crossValidation(X, Y, B)
```



```r
print(res$mse)
```

```
## [1] 4396.357
```

```r
# Compare with frequentist linear regression
#lm(dat$Fertility ~ dat$Education + dat$Agriculture)
```

3 References