Paris House Pricing Prediction Model Using Bayesian Method

Elvina Benedicta Santoso

2023-12-29

Load library

```
library(coda)
library(rjags)
```

```
## Linked to JAGS 4.3.2

## Loaded modules: basemod,bugs
```

```
library(knitr)
library(purrr)
library(tidyr)
library(ggplot2)
```

Set Seed

```
set.seed(123)
```

Load Data

```
df <- read.csv('./ParisHousing.csv')
head(df)
```

```
##   squareMeters numberOfRooms hasYard hasPool floors cityCode cityPartRange
## 1        75523             3       0       1     63     9373             3
## 2        80771            39       1       1     98    39381             8
## 3        55712            58       0       1     19    34457             6
## 4        32316            47       0       0      6    27939            10
## 5        70429            19       1       1     90    38045             3
## 6        39223            36       0       1     17    39489             8
##   numPrevOwners made isNewBuilt hasStormProtector basement attic garage
## 1             8 2005          0                 1     4313  9005    956
## 2             6 2015          1                 0     3653  2436    128
```

```
## 3            8 2021         0                  0     2937  8852    135
## 4            4 2012         0                  1      659  7141    359
## 5            7 1990         1                  0     8435  2429    292
## 6            6 2012         0                  1     2009  4552    757
##   hasStorageRoom hasGuestRoom   price
## 1              0            7 7559082
## 2              1            2 8085990
## 3              1            9 5574642
## 4              0            3 3232561
## 5              1            4 7055052
## 6              0            1 3926647
```

Model 1

Set Dataframe as matrix and split dependent and independent variable

```
price <- as.matrix(df$price)
Y <- price

X <- df[, c("squareMeters", "numberOfRooms", "hasYard", "hasPool", "floors", "cityCode", "numPrevOwners

names <- c("price","Square Meters", "Number of Rooms", "Has Yard", "Has Pool", "Number of Floors", "City
```

Delete Missing Value

```
junk <- is.na(rowSums(X))
Y <- Y[!junk]
X <- X[!junk,]
```

Standardize Covariates

```
X <- as.matrix(scale(X))
```

JAGS

Put Data in JAGS Format

```
n <- length(Y)
p <- ncol(X)

data <- list(Y=Y,X=X,n=n,p=p)
params <- c("alpha","beta")
burn <- 500
n.iter <- 2000
n.chains <- 3
thin <- 5
```

Make Jags Model

```
model_string <- textConnection("model{
  # Likelihood
  for(i in 1:n){
    Y[i] ~ dnorm(alpha+mu[i],taue)
    mu[i] <- inprod(X[i,],beta[])
  }

  # Priors
  for(j in 1:p){
    beta[j] ~ dnorm(0,0.001)
  }

  alpha ~ dnorm(0,0.001)
  taue ~ dgamma(0.1, 0.1)

  # WAIC calculations
  for(i in 1:n){
    like[i] <- dnorm(Y[i],mu[i],taue)
  }

}")
```

Set Initial Value

```
inits = list()

inits$alpha = rnorm(1)
for(i in 1:p) {
    inits$beta[i] = rnorm(1)
}
inits$taue = 10
```

Compile Model

```
model1 <- jags.model(model_string, data = data, n.chains=n.chains, quiet=TRUE, inits = inits)
```

Update Model

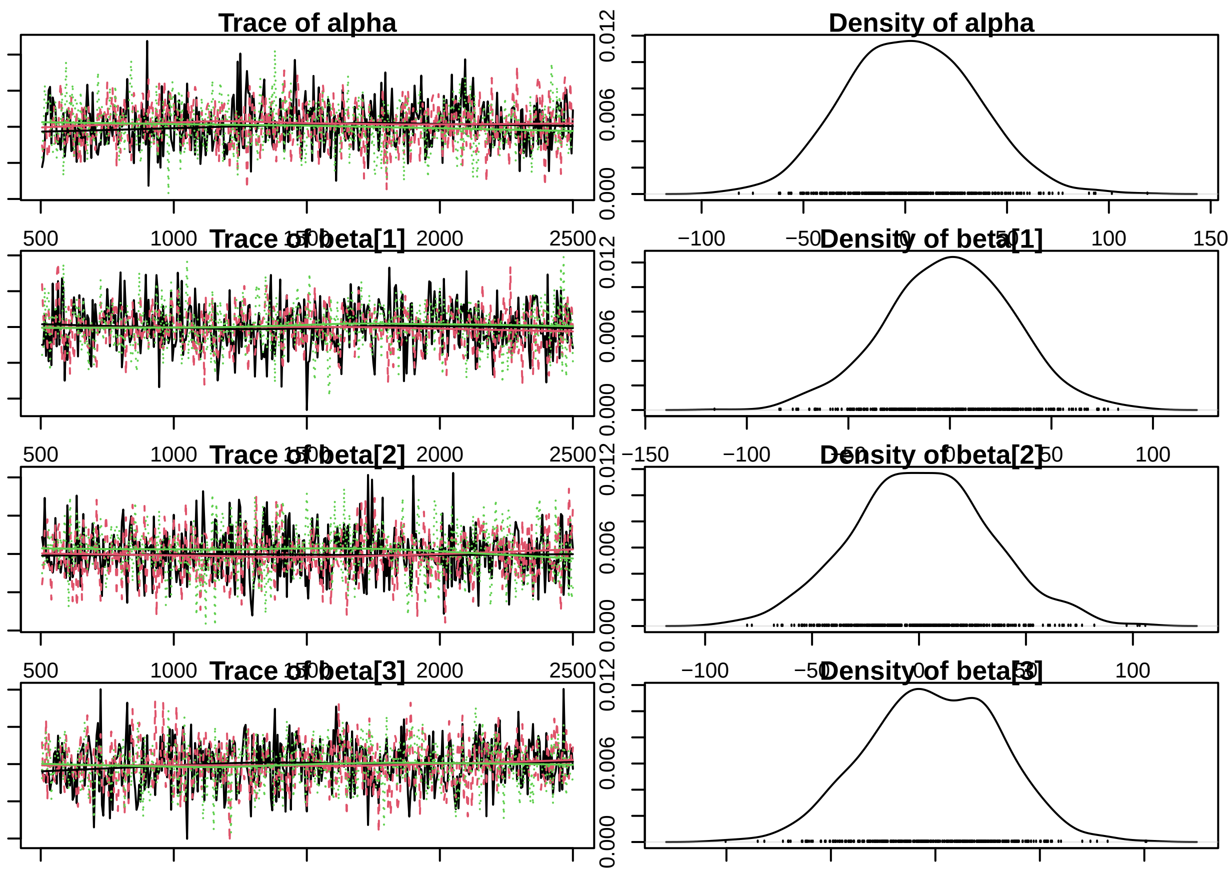
```
update(model1, burn)
```

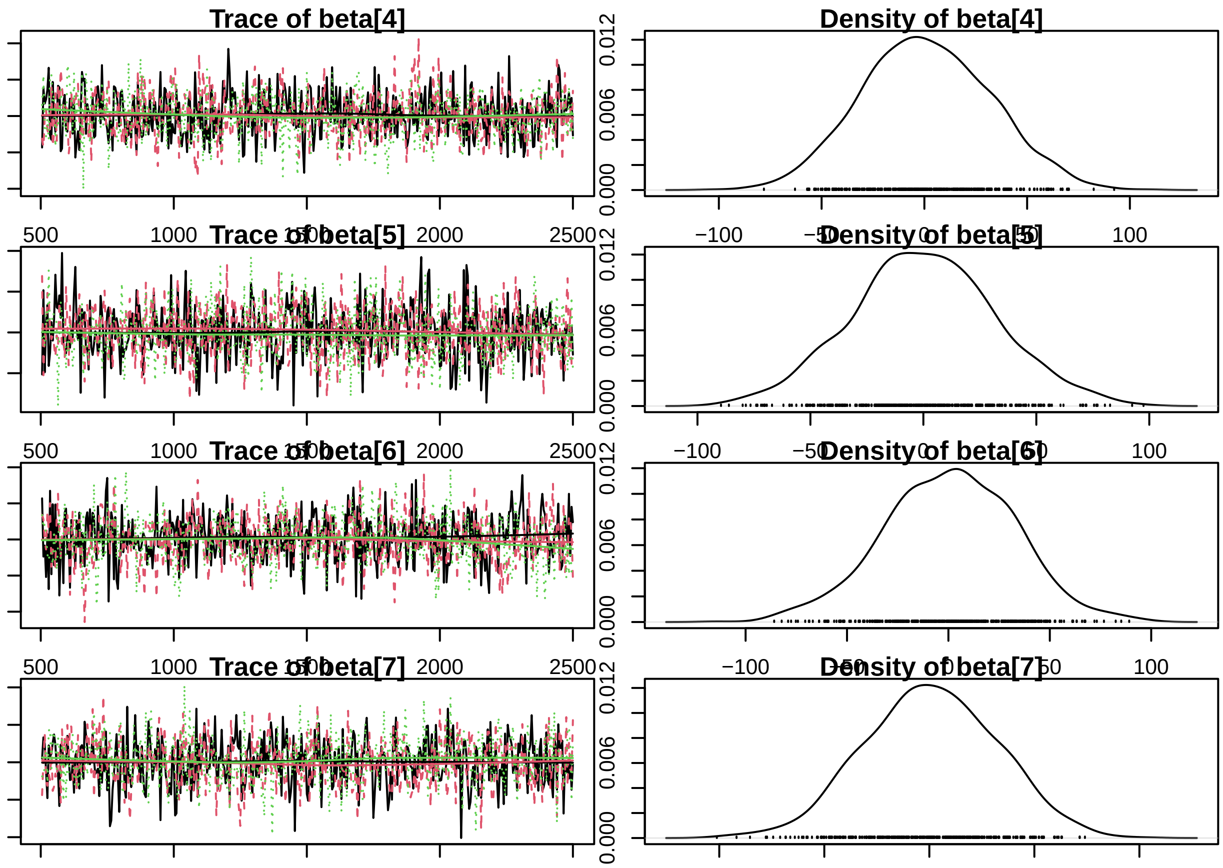
Get Posterior Samples from the model

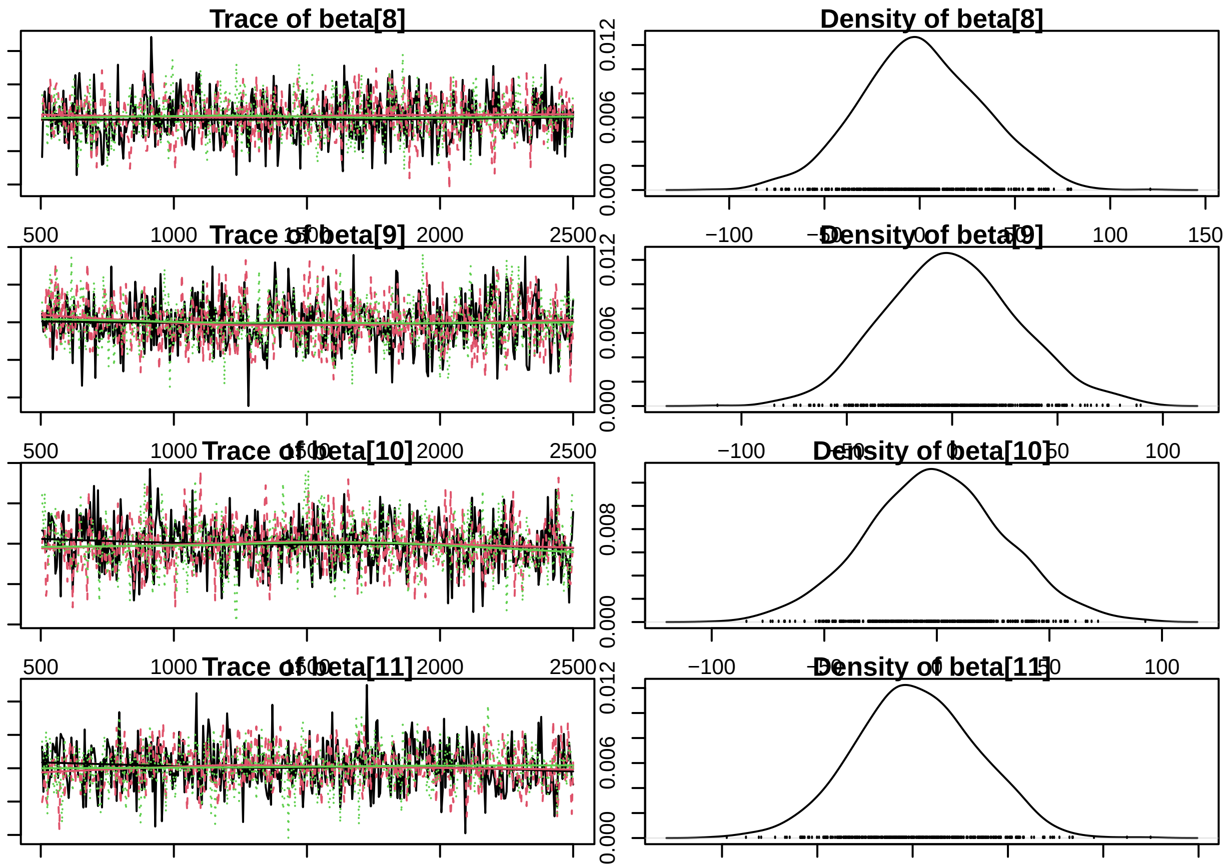
```
samples1 <- coda.samples(model1, variable.names=params, n.iter=n.iter, thin=thin)
```

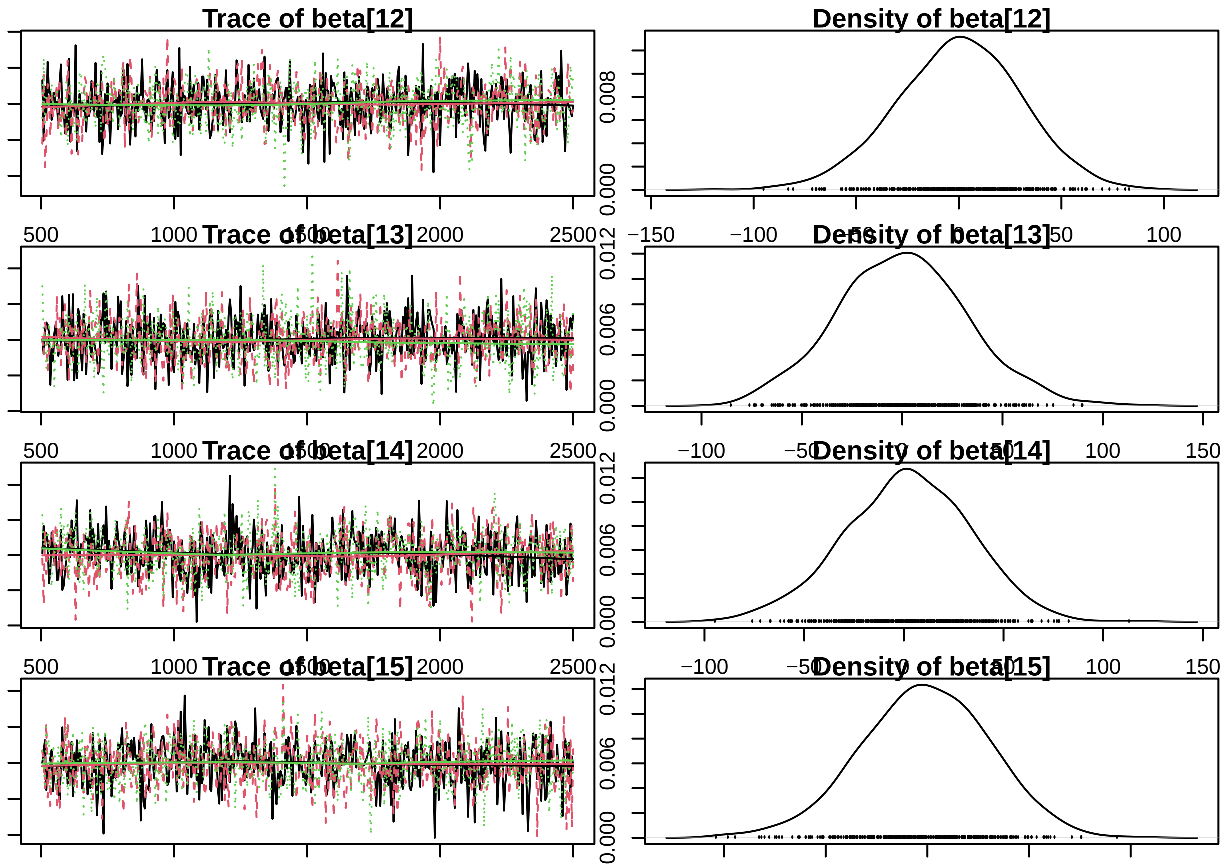
Plot MCMC Chain Trace and Features Posterior Density

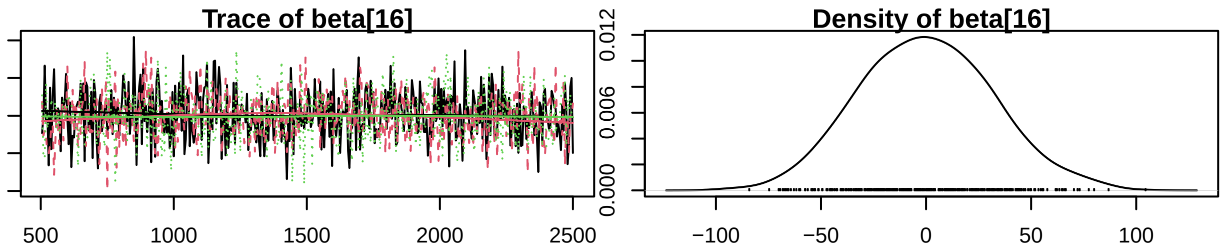
```
par(mar=c(1,1,1,1))
plot(samples1)
```




Trace of beta[4]
Density of beta[4]
Trace of beta[5]
Density of beta[5]
Trace of beta[6]
Density of beta[6]
Trace of beta[7]
Density of beta[7]
500
1000
1500
2000
2500
−100
−50
0
50
100
0.000
0.006
0.012


Trace of beta[8]
Density of beta[8]
Trace of beta[9]
Density of beta[9]
Trace of beta[10]
Density of beta[10]
Trace of beta[11]
Density of beta[11]
500
1000
1500
2000
2500
−100
−50
0
50
100
150
0.000
0.006
0.012
0.008


Trace of beta[12]
Density of beta[12]
Trace of beta[13]
Density of beta[13]
Trace of beta[14]
Density of beta[14]
Trace of beta[15]
Density of beta[15]



Descriptive Statistics of model1

```
summary(samples1)
```

```
##
## Iterations = 505:2500
## Thinning interval = 5
## Number of chains = 3
## Sample size per chain = 400
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##              Mean    SD Naive SE Time-series SE
## alpha     2.60364 31.57   0.9114         0.9116
## beta[1]  -0.21763 31.38   0.9060         0.9130
## beta[2]   1.15939 32.31   0.9326         0.9308
## beta[3]  -0.87051 31.43   0.9073         0.8733
## beta[4]   0.93194 31.05   0.8965         0.8726
## beta[5]   0.20884 31.86   0.9197         0.8642
## beta[6]   0.48302 32.30   0.9324         1.0516
## beta[7]   0.70716 31.39   0.9062         0.9366
## beta[8]   0.88142 32.11   0.9268         0.9262
```

```
## beta[9]  -0.31983 31.65   0.9136         0.9286
## beta[10] -0.70227 30.40   0.8775         0.8781
## beta[11]  1.80388 32.08   0.9262         0.9268
## beta[12] -0.03802 30.38   0.8770         0.8429
## beta[13]  0.20978 32.52   0.9388         0.8954
## beta[14]  1.67043 31.64   0.9133         0.9997
## beta[15] -1.10066 31.62   0.9129         0.9132
## beta[16]  0.06349 31.74   0.9161         0.8650
##
## 2. Quantiles for each variable:
##
##            2.5%    25%      50%   75% 97.5%
## alpha    -55.77 -19.23  2.67669 24.23 64.64
## beta[1]  -65.83 -20.58  0.03270 20.89 60.69
## beta[2]  -61.43 -20.34  0.69101 21.66 68.19
## beta[3]  -61.42 -22.11 -1.18361 22.04 57.21
## beta[4]  -57.08 -20.94 -0.73974 21.99 61.84
## beta[5]  -61.96 -20.11  0.14642 21.04 66.78
## beta[6]  -66.80 -20.70  1.61000 23.38 63.22
## beta[7]  -60.63 -21.00  0.68301 22.07 60.83
## beta[8]  -62.31 -20.68  0.29102 22.32 63.86
## beta[9]  -58.84 -22.13 -1.40274 20.06 65.29
## beta[10] -61.23 -21.55 -1.17282 18.71 59.85
## beta[11] -59.89 -19.66  1.44799 23.07 62.35
## beta[12] -61.62 -19.84  0.85211 20.42 58.22
## beta[13] -62.15 -22.02 -0.08204 21.68 65.45
## beta[14] -61.97 -19.62  2.24538 23.47 61.71
## beta[15] -64.65 -21.94 -0.98620 20.45 58.52
## beta[16] -59.98 -22.17 -0.52528 21.80 64.92
```

Show Alpha and Betas to the corresponding Features

```
sum <- summary(samples1)
rownames(sum$statistics) <- names
rownames(sum$quantiles) <- names
sum$statistics <- round(sum$statistics,3)
sum$quantiles <- round(sum$quantiles,3)
sum
```

```
##
## Iterations = 505:2500
## Thinning interval = 5
## Number of chains = 3
## Sample size per chain = 400
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                              Mean    SD Naive SE Time-series SE
## price                       2.604 31.57    0.911          0.912
## Square Meters              -0.218 31.38    0.906          0.913
## Number of Rooms             1.159 32.30    0.933          0.931
```

```
## Has Yard                    -0.871 31.43    0.907          0.873
## Has Pool                     0.932 31.05    0.896          0.873
## Number of Floors             0.209 31.86    0.920          0.864
## City Code                    0.483 32.30    0.932          1.052
## Number of Previous Owners    0.707 31.39    0.906          0.937
## Year Made                    0.881 32.11    0.927          0.926
## is Newly Built              -0.320 31.65    0.914          0.929
## Has Storm Protector         -0.702 30.40    0.877          0.878
## Basement Area                1.804 32.08    0.926          0.927
## Attic Area                  -0.038 30.38    0.877          0.843
## Garage Area                  0.210 32.52    0.939          0.895
## Has Storage Room             1.670 31.64    0.913          1.000
## Has Guest Room              -1.101 31.62    0.913          0.913
## City Part Range              0.063 31.73    0.916          0.865
##
## 2. Quantiles for each variable:
##
##                               2.5%    25%    50%   75% 97.5%
## price                       -55.77 -19.23  2.677 24.23 64.64
## Square Meters               -65.83 -20.58  0.033 20.89 60.69
## Number of Rooms             -61.43 -20.34  0.691 21.66 68.19
## Has Yard                    -61.42 -22.11 -1.184 22.04 57.21
## Has Pool                    -57.08 -20.94 -0.740 21.99 61.84
## Number of Floors            -61.96 -20.11  0.146 21.04 66.78
## City Code                   -66.80 -20.70  1.610 23.38 63.22
## Number of Previous Owners -60.63 -21.00  0.683 22.07 60.83
## Year Made                   -62.30 -20.68  0.291 22.32 63.86
## is Newly Built              -58.84 -22.13 -1.403 20.06 65.29
## Has Storm Protector         -61.23 -21.55 -1.173 18.71 59.85
## Basement Area               -59.89 -19.66  1.448 23.07 62.35
## Attic Area                  -61.62 -19.84  0.852 20.42 58.22
## Garage Area                 -62.15 -22.02 -0.082 21.68 65.45
## Has Storage Room            -61.97 -19.62  2.245 23.47 61.71
## Has Guest Room              -64.65 -21.94 -0.986 20.45 58.52
## City Part Range             -59.98 -22.17 -0.525 21.80 64.92
```

Check Convergence

```r
gelman.diag(samples1)
```

```
## Potential scale reduction factors:
##
##          Point est. Upper C.I.
## alpha         1.006      1.024
## beta[1]       1.005      1.014
## beta[2]       1.005      1.019
## beta[3]       0.999      1.001
## beta[4]       1.001      1.007
## beta[5]       0.999      1.002
## beta[6]       1.001      1.008
## beta[7]       1.005      1.024
## beta[8]       1.002      1.013
```

```
## beta[9]       0.999      1.000
## beta[10]      1.000      1.003
## beta[11]      1.001      1.003
## beta[12]      1.000      1.001
## beta[13]      1.007      1.013
## beta[14]      1.001      1.006
## beta[15]      1.002      1.006
## beta[16]      0.999      0.999
##
## Multivariate psrf
##
## 1.02
```

Compile results

```
ESS1 <- effectiveSize(samples1)
out1 <- summary(samples1)$quantiles
rownames(out1)<-names

ESS1
```

```
##     alpha   beta[1]   beta[2]   beta[3]   beta[4]   beta[5]   beta[6]   beta[7]
## 1200.0000 1181.8287 1200.0000 1317.1512 1282.1286 1412.7931  976.1667 1131.0232
##   beta[8]   beta[9]  beta[10]  beta[11]  beta[12]  beta[13]  beta[14]  beta[15]
## 1200.0000 1165.6151 1200.0000 1200.0000 1324.9619 1337.5107 1047.6516 1200.0000
##  beta[16]
## 1375.3569
```

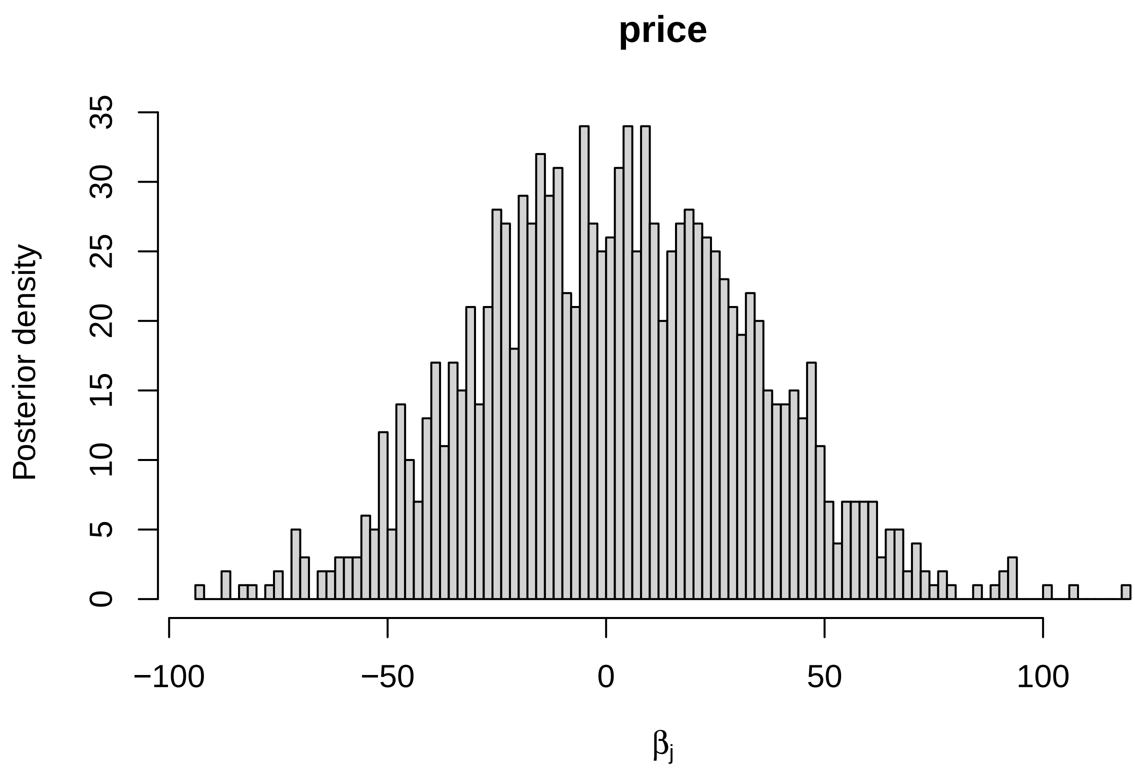
Compute DIC & WAIC

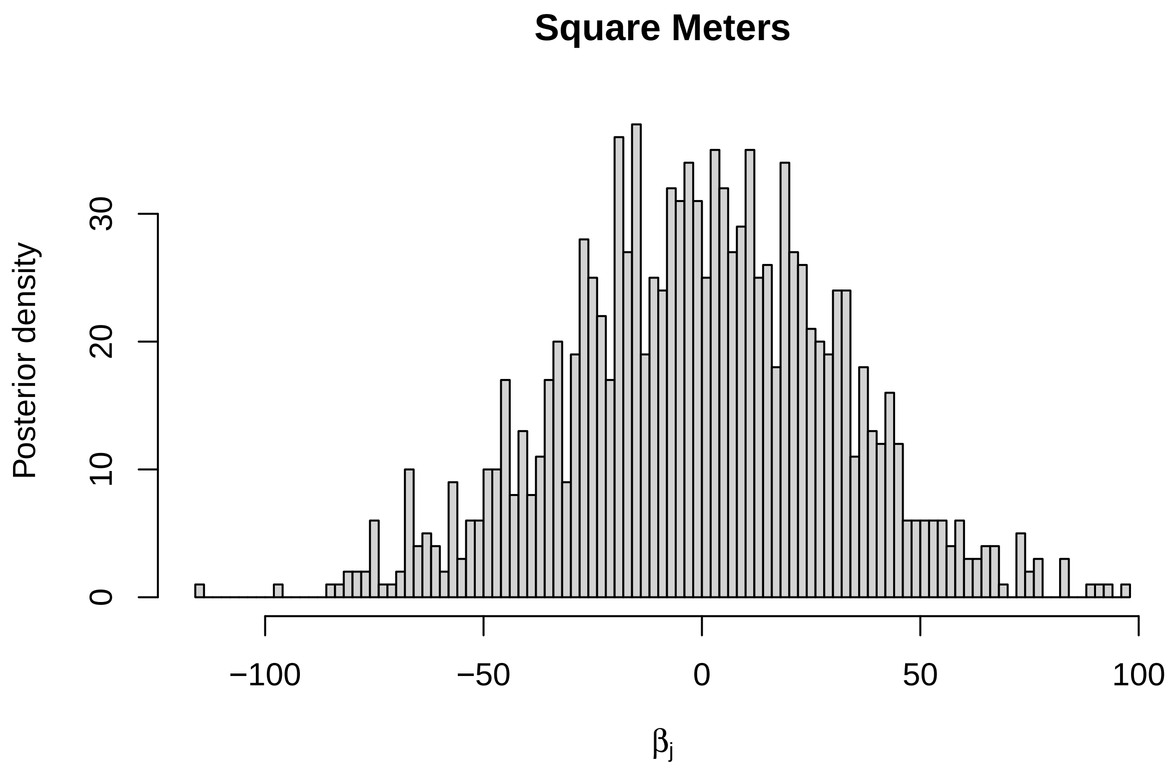
```
# DIC
dic1 <- dic.samples(model1,n.iter=n.iter)

# WAIC
waic1 <- coda.samples(model1, variable.names=c("like"), n.iter=n.iter)
like1 <- waic1[[1]]
fbar1 <- colMeans(like1)
P1 <- sum(apply(log(like1),2,var))
WAIC1 <- -2*sum(log(fbar1))+2*P1
```

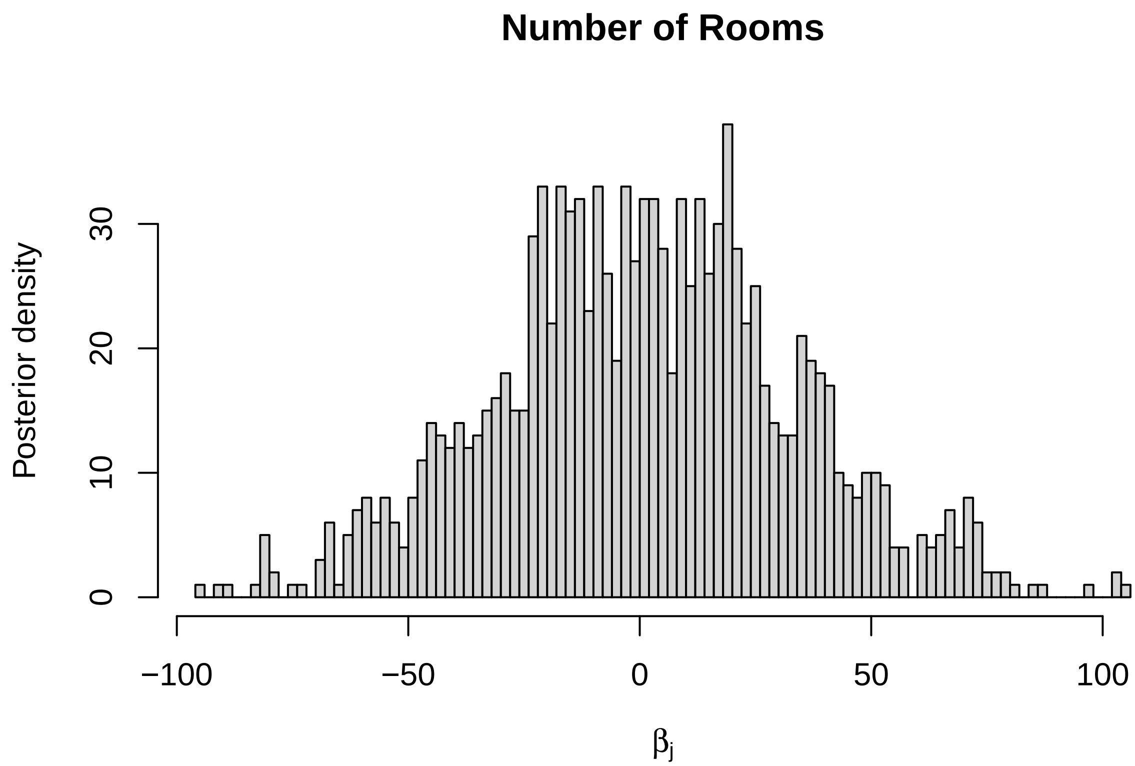
Plot Posterior Density Individually

```
beta <- NULL
for(l in 1:n.chains){
  beta <- rbind(beta,samples1[[l]])
}
colnames(beta) <- names
for(j in 1:17){
```

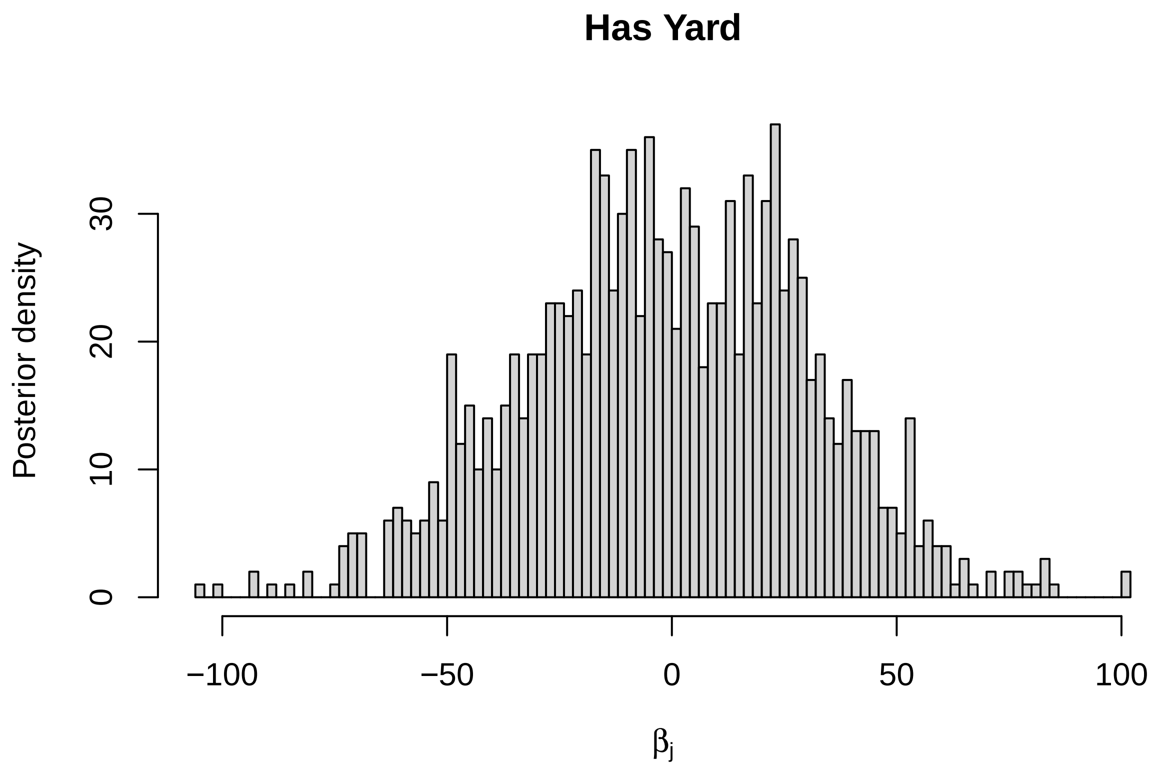
```
  hist(beta[,j],xlab=expression(beta[j]),ylab="Posterior density",
  breaks=100,main=names[j])
}
```



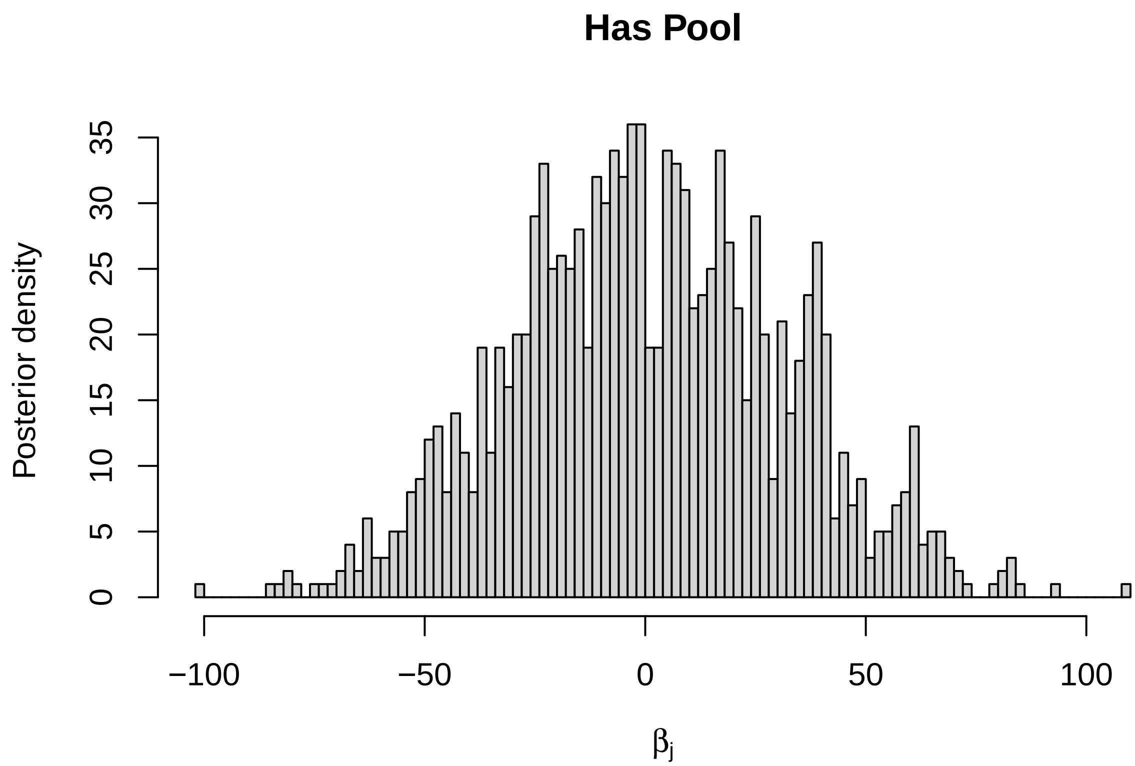

Square Meters
Posterior density
0
10
20
30
−100
−50
0
50
100
β_j



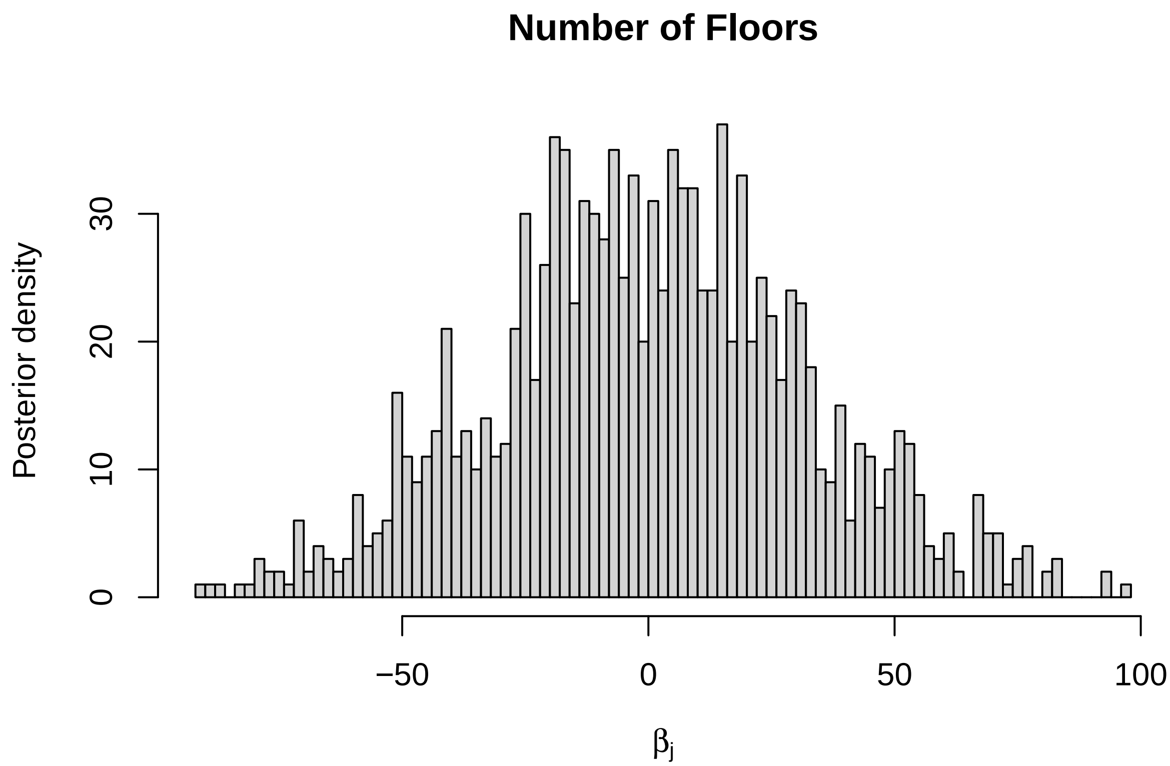
Number of Rooms
Posterior density
0
10
20
30
−100
−50
0
50
100
β_j

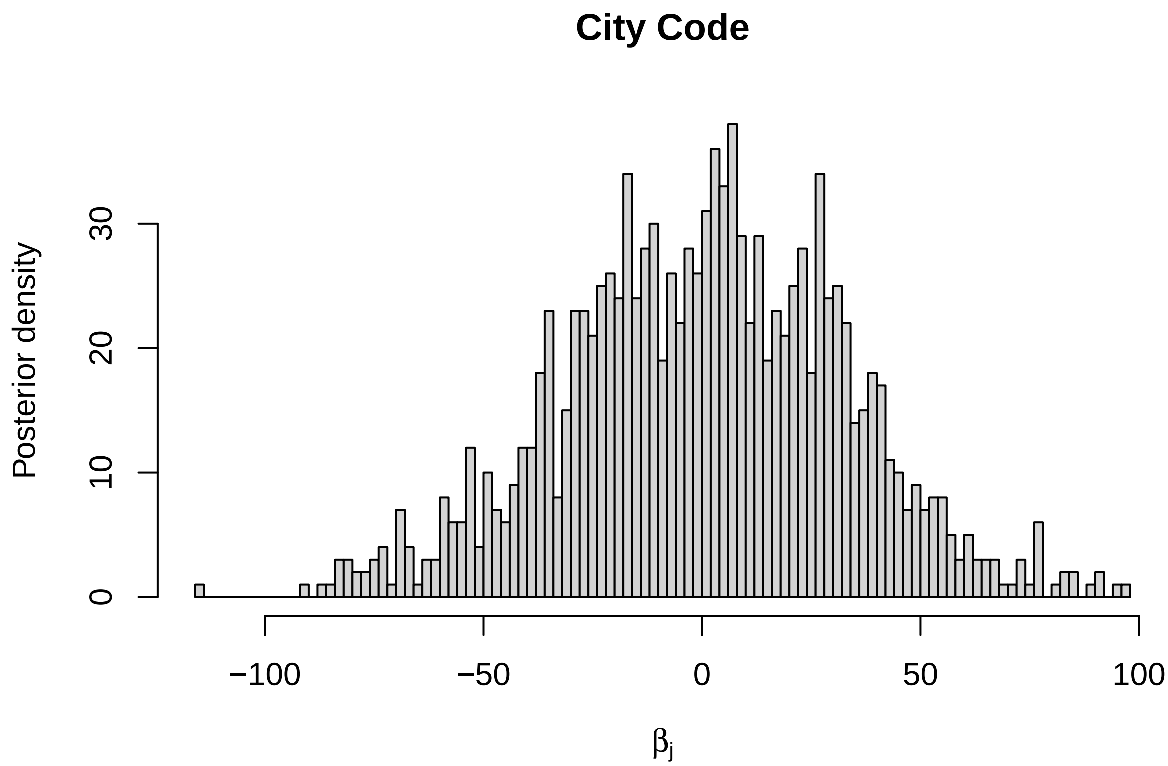


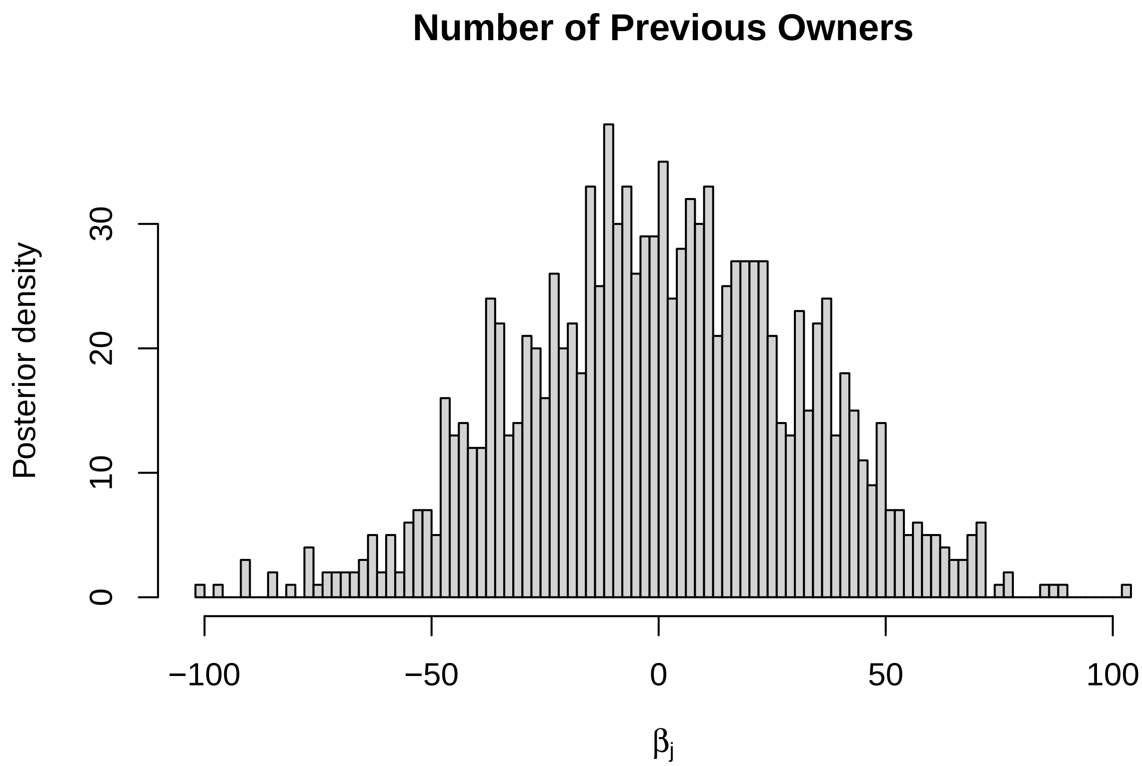
Has Yard
Posterior density
0
10
20
30
−100
−50
0
50
100
β_j

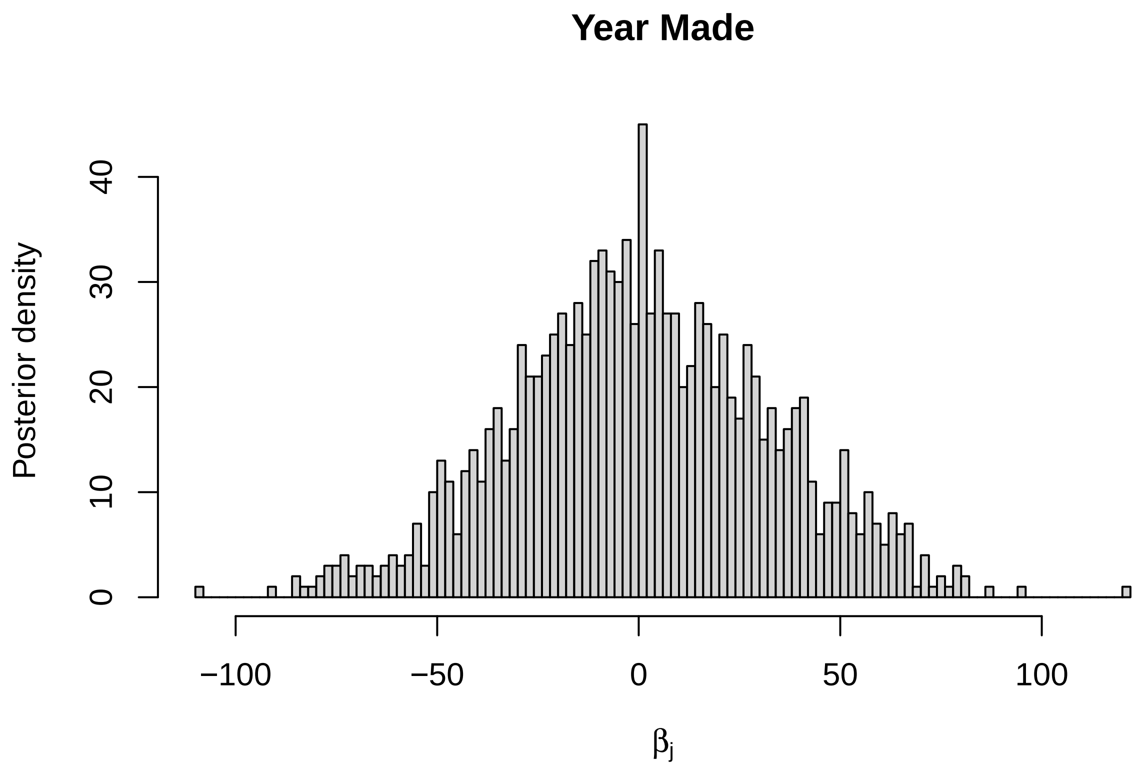


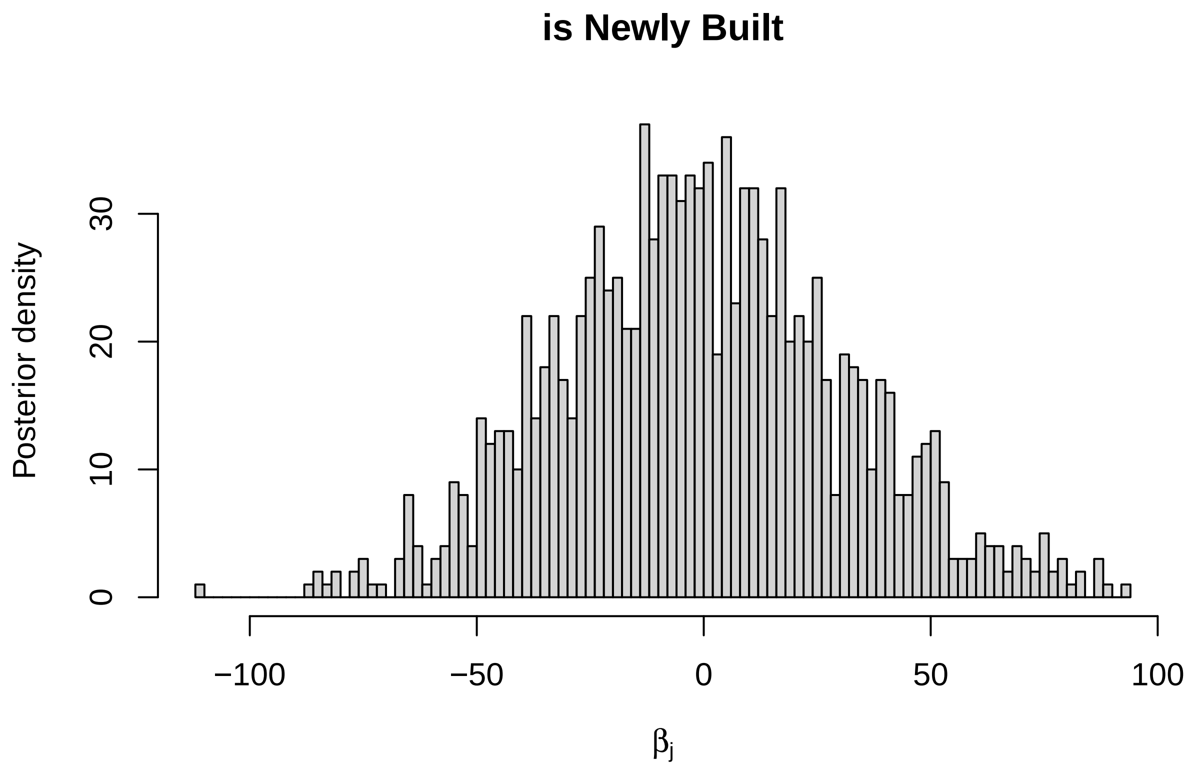
Has Pool
Posterior density
0
5
10
15
20
25
30
35
−100
−50
0
50
100
β_j

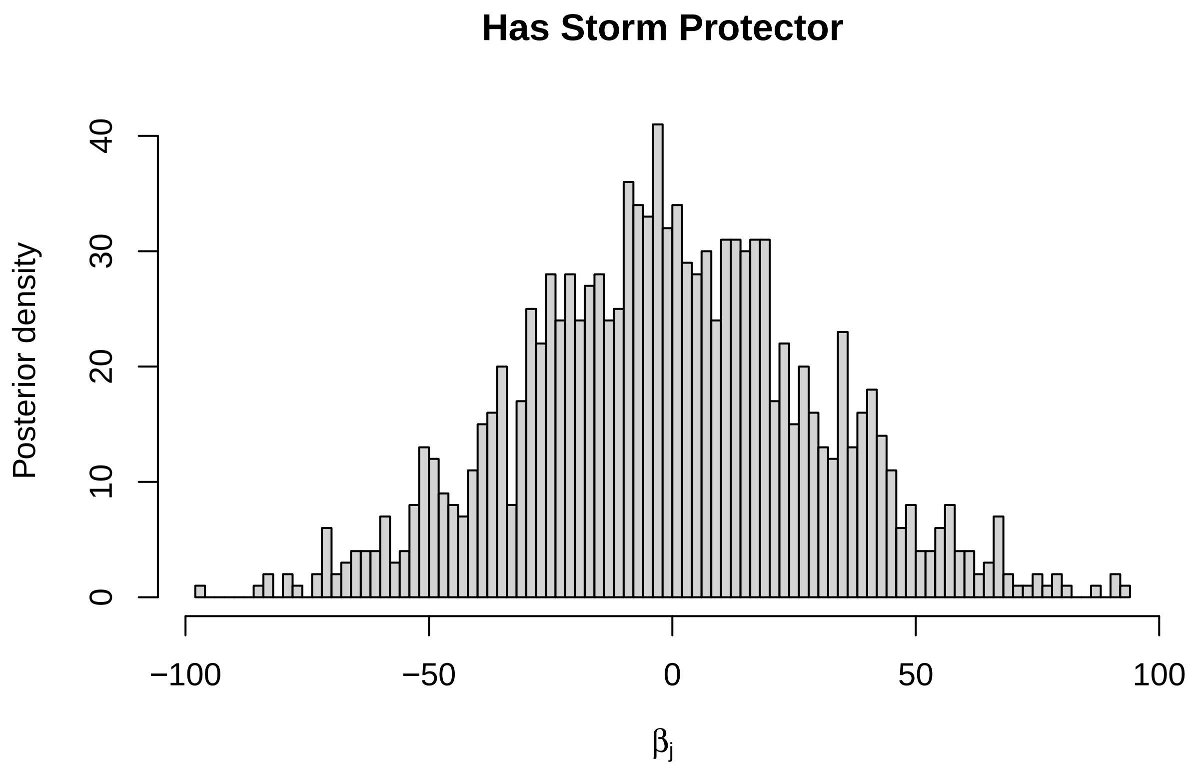

Number of Floors
Posterior density
0
10
20
30
−50
0
50
100
βj


City Code
Posterior density
0
10
20
30
−100
−50
0
50
100
βj

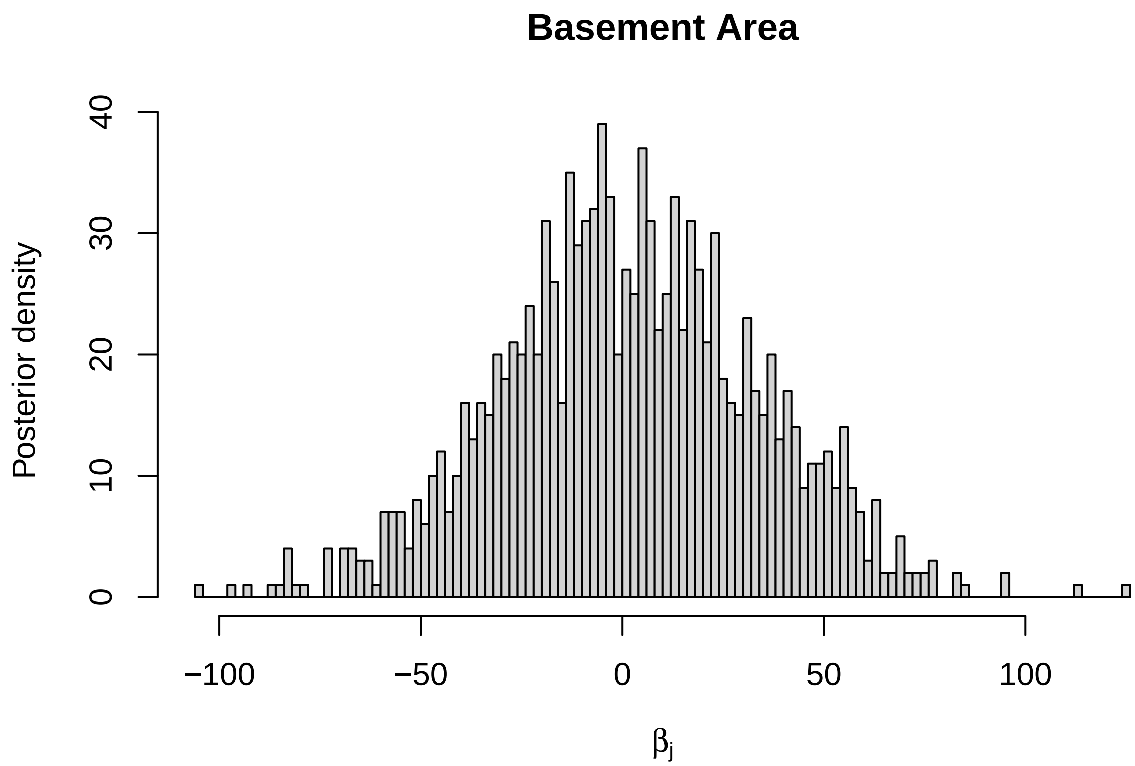

Number of Previous Owners
Posterior density
0
10
20
30
−100
−50
0
50
100
βj

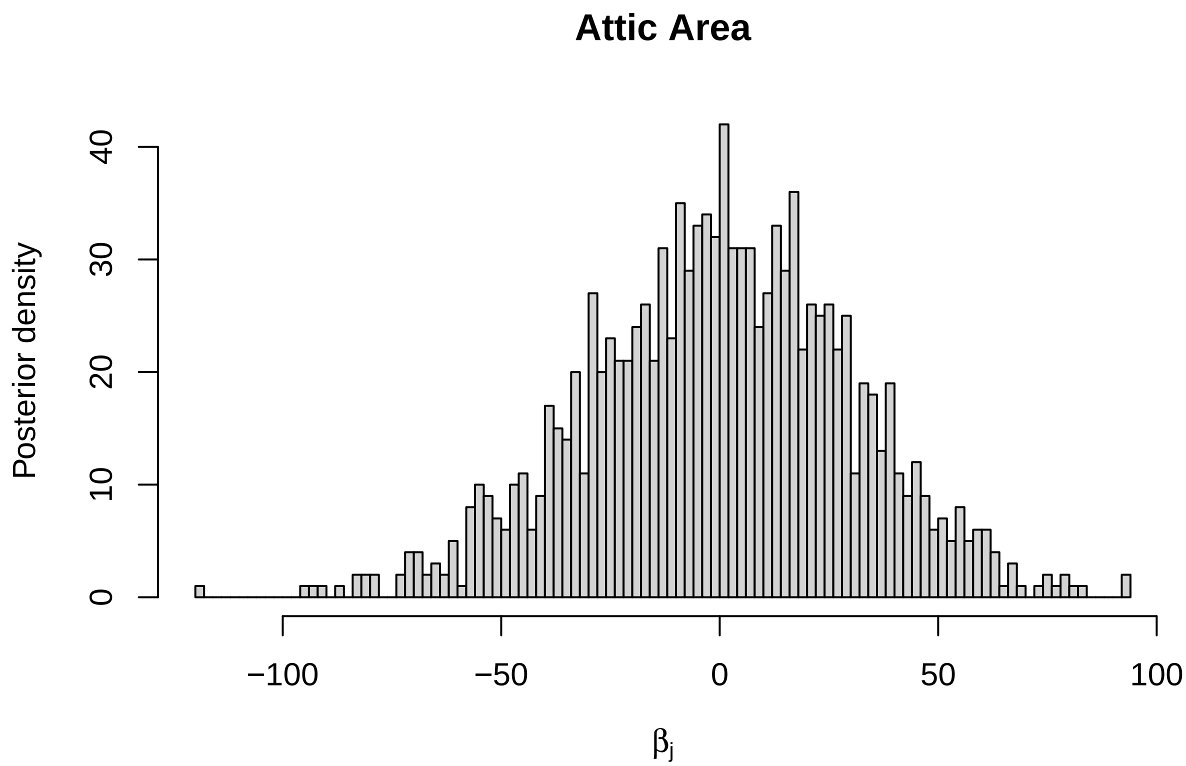

Year Made
Posterior density
0
10
20
30
40
−100
−50
0
50
100
βj


is Newly Built
Posterior density
0
10
20
30
−100
−50
0
50
100
βj

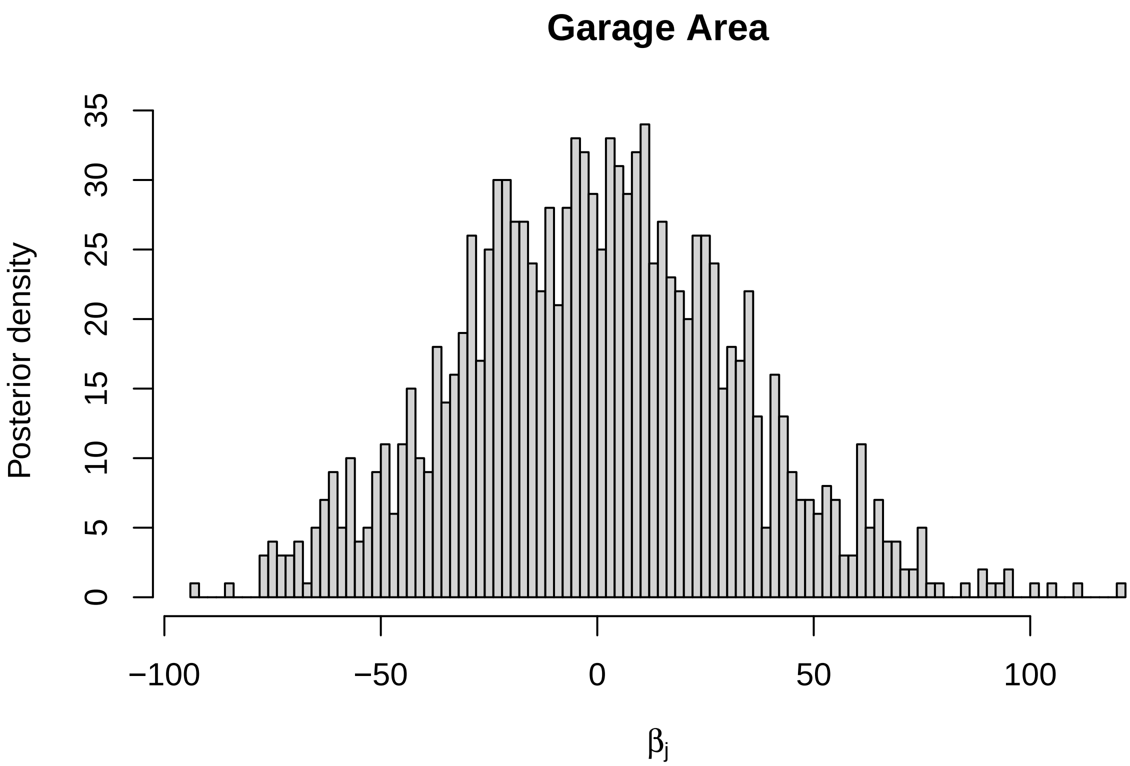


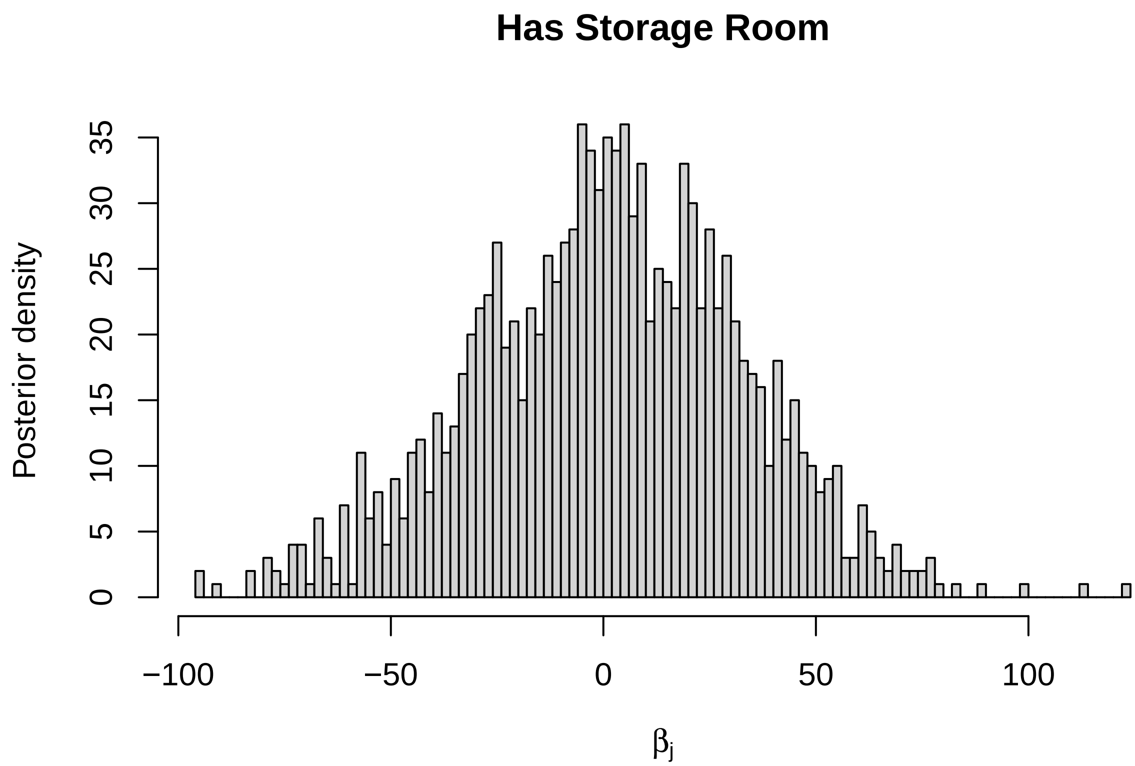
Has Storm Protector
Posterior density
0
10
20
30
40
−100
−50
0
50
100
βj


Basement Area
Posterior density
0
10
20
30
40
−100
−50
0
50
100
β_j

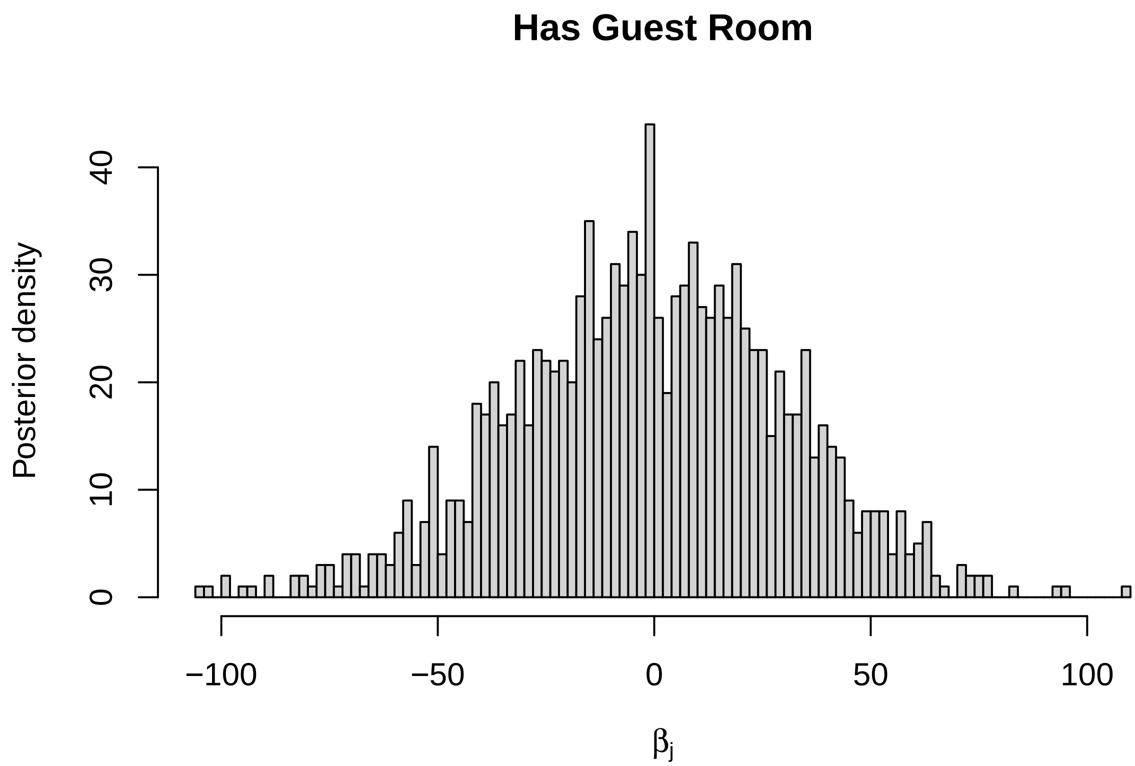


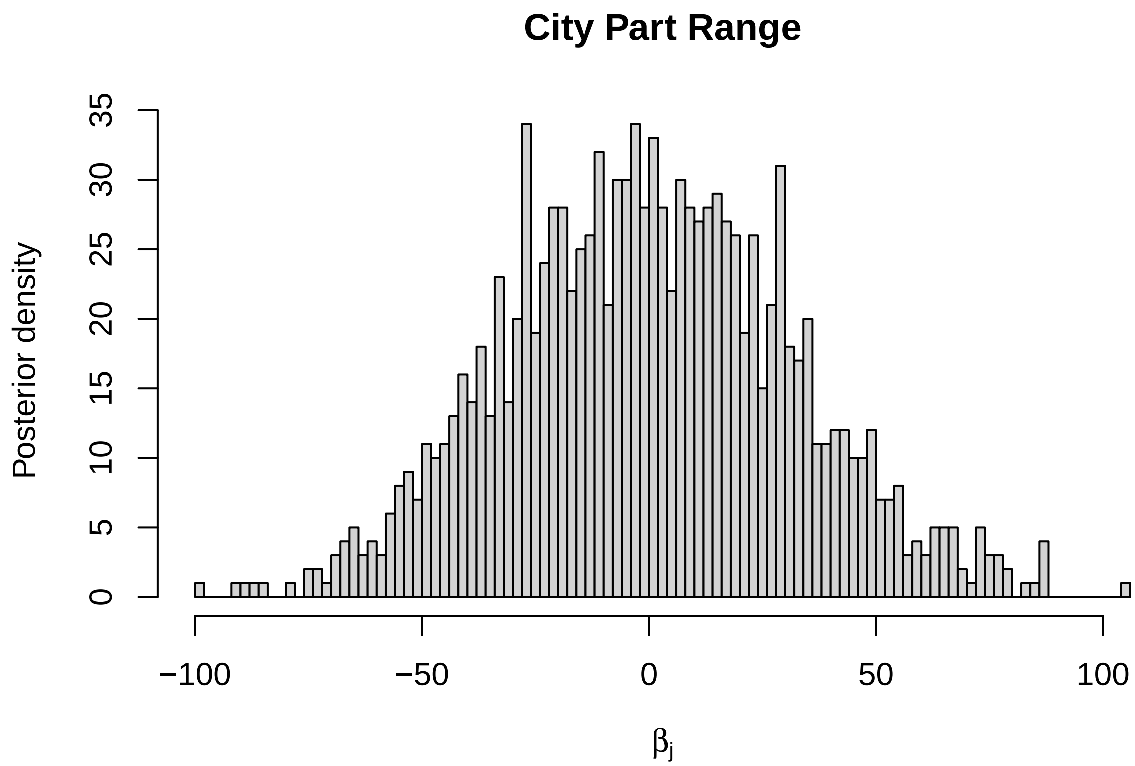
Attic Area
Posterior density
0
10
20
30
40
−100
−50
0
50
100
β_j


Garage Area
Posterior density
0
5
10
15
20
25
30
35
−100
−50
0
50
100
β_j



Has Storage Room
Posterior density
0
5
10
15
20
25
30
35
−100
−50
0
50
100
βj


Has Guest Room
Posterior density
0
10
20
30
40
−100
−50
0
50
100
βj



SSVS

```
library(knitr)
Inc_Prob <- apply(beta!=0,2,mean)
Q <- t(apply(beta,2,quantile,c(0.5,0.05,0.95)))
out <- cbind(Inc_Prob,Q)
kable(round(out,2))
```

	Inc_Prob	50%	5%	95%
price	1	2.68	-47.63	54.44
Square Meters	1	0.03	-52.42	50.00
Number of Rooms	1	0.69	-52.57	53.44
Has Yard	1	-1.18	-52.37	48.27
Has Pool	1	-0.74	-49.06	54.67
Number of Floors	1	0.15	-51.31	53.04
City Code	1	1.61	-53.94	51.74
Number of Previous Owners	1	0.68	-49.18	50.74
Year Made	1	0.29	-50.61	55.12
is Newly Built	1	-1.40	-49.54	51.92
Has Storm Protector	1	-1.17	-50.89	48.88
Basement Area	1	1.45	-50.22	54.90
Attic Area	1	0.85	-52.09	48.73

	Inc_Prob	50%	5%	95%
Garage Area	1	-0.08	-53.91	55.95
Has Storage Room	1	2.25	-52.82	52.03
Has Guest Room	1	-0.99	-52.83	49.99
City Part Range	1	-0.53	-51.09	53.01

Model 2 (Using features that have strong correlation with the target variable based on the frequentist linear regression model)

Build frequentist linear regression model

```
price.lm <- lm(price ~ X, data = df)

summary(price.lm)
```

```
##
## Call:
## lm(formula = price ~ X, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6988.9 -1192.2    -3.2  1198.7  7005.6
##
## Coefficients:
##                      Estimate Std. Error    t value Pr(>|t|)
## (Intercept)         4.993e+06  1.898e+01 263030.498  < 2e-16 ***
## XsquareMeters       2.877e+06  1.900e+01 151480.243  < 2e-16 ***
## XnumberOfRooms      7.257e+00  1.901e+01      0.382 0.702640
## XhasYard            1.506e+03  1.899e+01     79.289  < 2e-16 ***
## XhasPool            1.489e+03  1.900e+01     78.365  < 2e-16 ***
## Xfloors             1.576e+03  1.900e+01     82.943  < 2e-16 ***
## XcityCode          -2.332e+01  1.899e+01     -1.228 0.219513
## XnumPrevOwners     -1.130e+00  1.901e+01     -0.059 0.952580
## Xmade              -2.153e+01  1.899e+01     -1.134 0.256940
## XisNewBuilt         7.903e+01  1.900e+01      4.159 3.22e-05 ***
## XhasStormProtector  7.060e+01  1.899e+01      3.718 0.000202 ***
## Xbasement          -6.057e+00  1.900e+01     -0.319 0.749846
## Xattic             -1.305e+01  1.900e+01     -0.687 0.492064
## Xgarage             2.973e+01  1.901e+01      1.563 0.117967
## XhasStorageRoom     9.743e+00  1.901e+01      0.512 0.608332
## XhasGuestRoom      -1.785e+01  1.901e+01     -0.939 0.347641
## XcityPartRange      1.360e+02  1.900e+01      7.162 8.51e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1898 on 9983 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 1.436e+09 on 16 and 9983 DF,  p-value: < 2.2e-16
```

There are 7 variables that have a strong correlation with the variable price (indicated by 3 stars). These seven variables include squareMeters, hasYard, hasPool, floors, cityPartRange, isNewBuilt, and hasStormProtector.

Split dependent and independent variable

Using 7 variables that have a strong correlation with the variable price

```
X <- df[, c("squareMeters", "hasYard", "hasPool", "floors", "cityPartRange", "isNewBuilt", "hasStormPro
Y <- df$price
names <- c("price", "Square Meters", "Has Yard", "Has Pool", "Floors", "City Part Range", "Is New Built
```

Delete Missing Value

```
junk <- is.na(rowSums(X))
Y <- Y[!junk]
X <- X[!junk,]
```

Standardize Covariates

```
X <- as.matrix(scale(X))
```

JAGS

Put Data in JAGS Format

```
n <- length(Y)
p <- ncol(X)

data <- list(Y=Y,X=X,n=n,p=p)
params <- c("alpha","beta")
burn <- 500
n.iter <- 2000
n.chains <- 3
thin <- 5
```

Make Jags Model

```
model_string <- textConnection("model{
  # Likelihood
  for(i in 1:n){
    Y[i] ~ dnorm(alpha+mu[i],taue)
    mu[i] <- inprod(X[i,],beta[])
  }
```

```
  # Priors
  for(j in 1:p){
    beta[j] ~ dnorm(0,0.001)
  }

  alpha ~ dnorm(0,0.001)
  taue ~ dgamma(0.1, 0.1)

  # WAIC calculations
  for(i in 1:n){
    like[i] <- dnorm(Y[i],mu[i],taue)
  }

}")
```

Set Initial Value

```
inits = list()

inits$alpha = rnorm(1)
for(i in 1:p) {
    inits$beta[i] = rnorm(1)
}
inits$taue = 10
```

Compile Model

```
model2 <- jags.model(model_string, data = data, n.chains=n.chains, quiet=TRUE, inits = inits)
```

Update Model

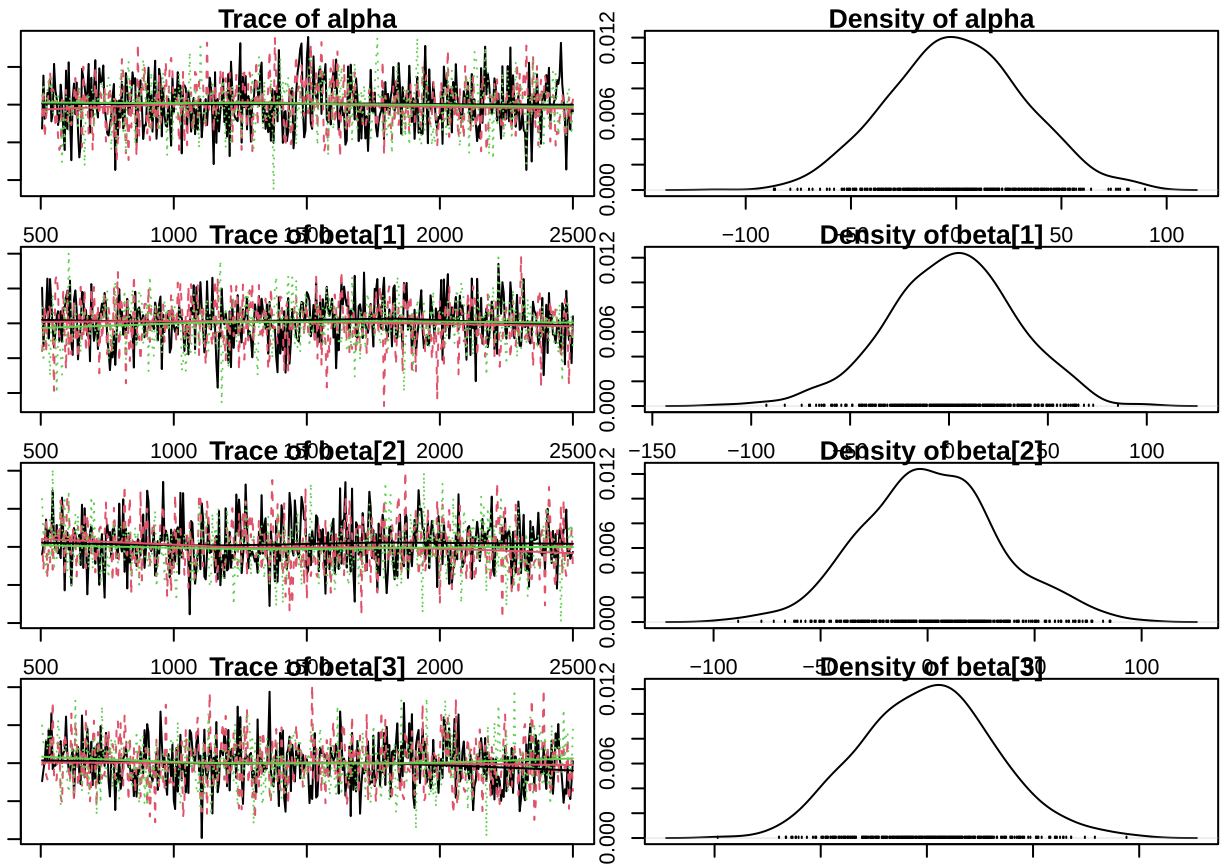
```
update(model2, burn)
```

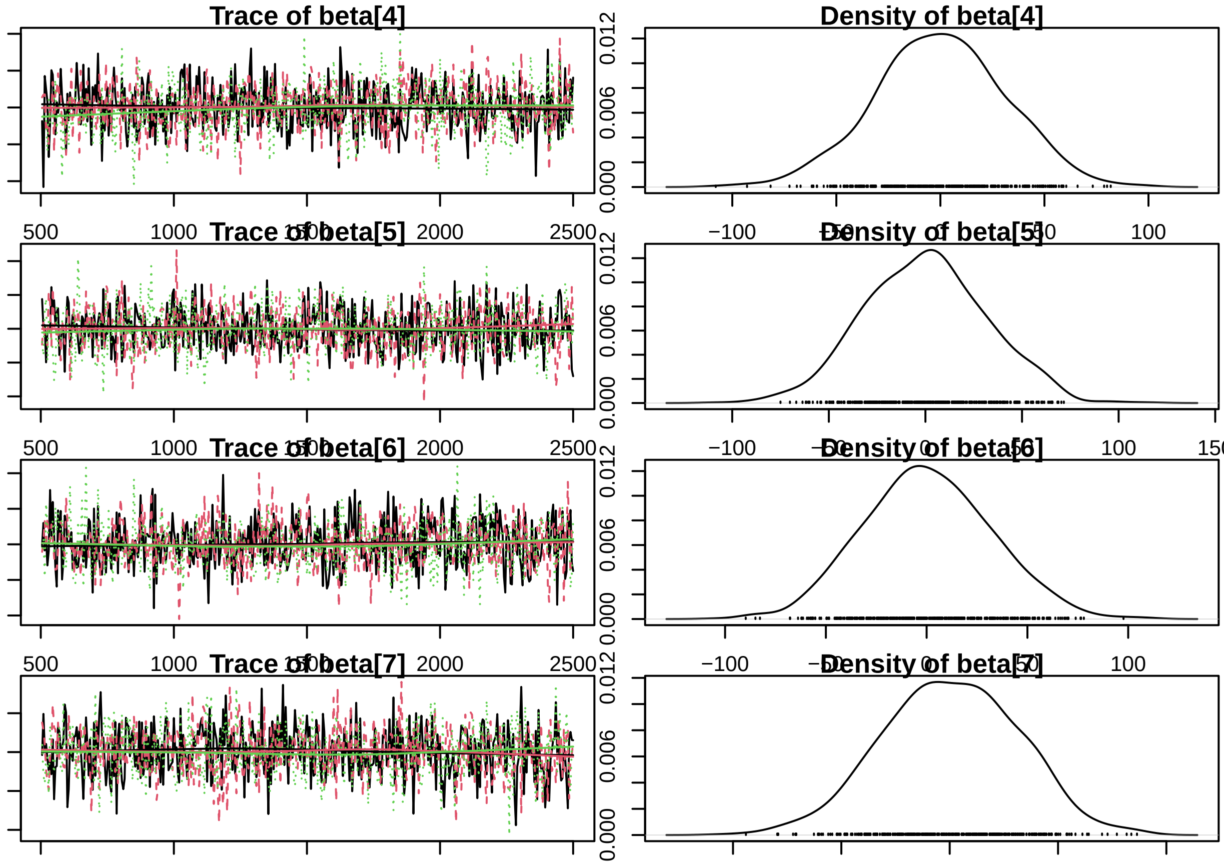
Get Posterior Samples from the model

```
samples2 <- coda.samples(model2, variable.names=params, n.iter=n.iter, thin = thin)
```

Plot MCMC Chain Trace and Features Posterior Density

```
par(mar=c(1,1,1,1))
plot(samples2)
```


Trace of alpha
Density of alpha
Trace of beta[1]
Density of beta[1]
Trace of beta[2]
Density of beta[2]
Trace of beta[3]
Density of beta[3]
500
1000
1500
2000
2500
0.000
0.006
0.012
−150
−100
−50
0
50
100



Descriptive Statistics of model2

```r
summary(samples2)
```

```
##
## Iterations = 505:2500
## Thinning interval = 5
## Number of chains = 3
## Sample size per chain = 400
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##              Mean    SD Naive SE Time-series SE
## alpha     0.23135 32.20   0.9297         0.9303
## beta[1]   0.92347 32.49   0.9378         0.9375
## beta[2]   1.61012 32.01   0.9241         0.9228
## beta[3]   0.60626 31.57   0.9113         0.8673
## beta[4]  -0.18475 31.51   0.9097         0.9262
## beta[5]  -0.07379 31.53   0.9103         0.9110
## beta[6]   0.44872 31.80   0.9181         0.9185
## beta[7]   0.32309 31.36   0.9054         0.9060
##
```

```
## 2. Quantiles for each variable:
##
##           2.5%    25%      50%   75% 97.5%
## alpha   -61.42 -21.19  0.02346 21.67 62.81
## beta[1] -68.17 -20.15  1.89683 22.17 62.71
## beta[2] -59.18 -19.33  0.56575 21.57 67.96
## beta[3] -57.55 -21.50  1.07917 21.09 64.90
## beta[4] -61.80 -21.01 -0.35077 20.29 60.50
## beta[5] -60.01 -21.75  0.08325 20.82 61.43
## beta[6] -57.84 -20.81 -0.72388 21.23 62.17
## beta[7] -61.53 -20.25  0.68807 21.69 59.40
```

Show Alpha and Betas to the corresponding Features

```r
sum <- summary(samples2)
rownames(sum$statistics) <- names
rownames(sum$quantiles) <- names
sum$statistics <- round(sum$statistics,3)
sum$quantiles <- round(sum$quantiles,3)
sum
```

```
##
## Iterations = 505:2500
## Thinning interval = 5
## Number of chains = 3
## Sample size per chain = 400
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                        Mean    SD Naive SE Time-series SE
## price                 0.231 32.20    0.930          0.930
## Square Meters         0.923 32.49    0.938          0.938
## Has Yard              1.610 32.01    0.924          0.923
## Has Pool              0.606 31.57    0.911          0.867
## Floors               -0.185 31.52    0.910          0.926
## City Part Range      -0.074 31.53    0.910          0.911
## Is New Built          0.449 31.80    0.918          0.919
## Has Storm Protector   0.323 31.36    0.905          0.906
##
## 2. Quantiles for each variable:
##
##                        2.5%    25%    50%   75% 97.5%
## price                -61.42 -21.19  0.023 21.66 62.81
## Square Meters        -68.17 -20.15  1.897 22.17 62.71
## Has Yard             -59.18 -19.33  0.566 21.57 67.96
## Has Pool             -57.55 -21.50  1.079 21.09 64.90
## Floors               -61.80 -21.01 -0.351 20.29 60.50
## City Part Range      -60.01 -21.75  0.083 20.82 61.43
## Is New Built         -57.84 -20.81 -0.724 21.23 62.17
## Has Storm Protector -61.53 -20.25  0.688 21.69 59.40
```

Check Convergence

```r
gelman.diag(samples2)
```

```
## Potential scale reduction factors:
##
##          Point est. Upper C.I.
## alpha         1.000       1.00
## beta[1]       1.007       1.03
## beta[2]       1.008       1.03
## beta[3]       1.002       1.01
## beta[4]       1.002       1.01
## beta[5]       0.999       1.00
## beta[6]       1.002       1.01
## beta[7]       0.999       1.00
##
## Multivariate psrf
##
## 1.02
```

Compile results

```r
ESS2 <- effectiveSize(samples2)
out2 <- summary(samples2)$quantiles
rownames(out2)<-names

ESS2
```

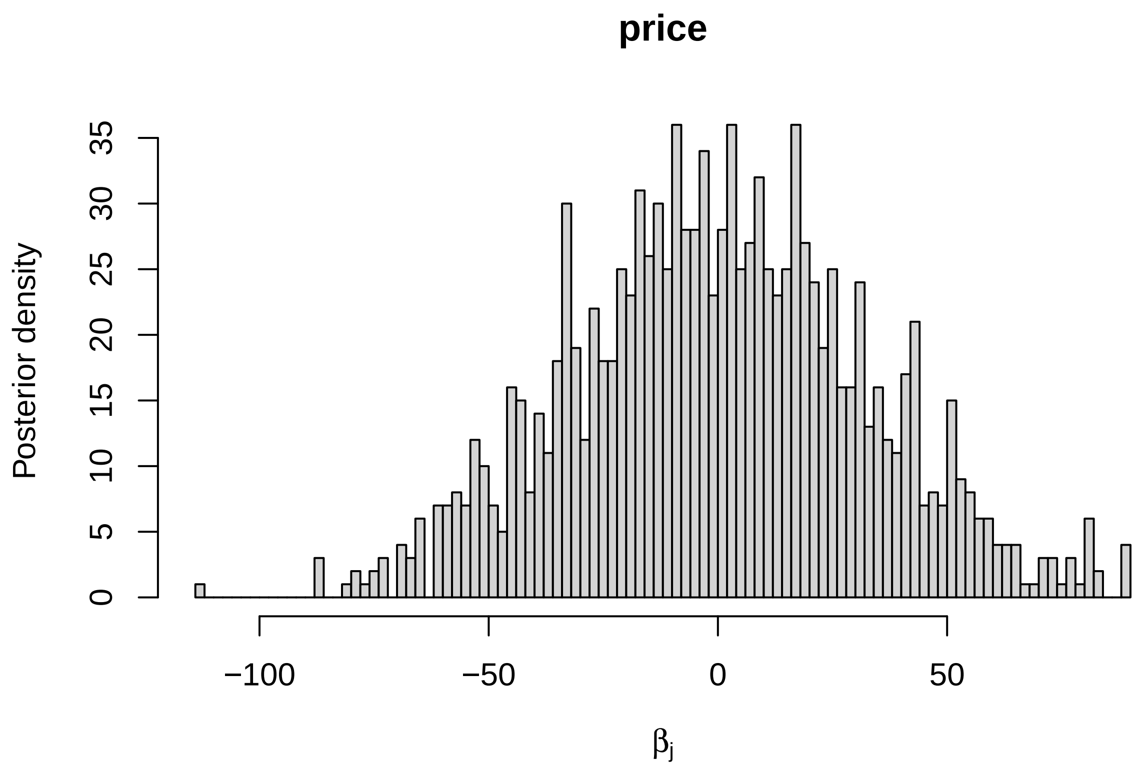
```
##    alpha  beta[1]  beta[2]  beta[3]  beta[4]  beta[5]  beta[6]  beta[7]
## 1200.000 1200.000 1200.000 1338.351 1161.042 1200.000 1200.000 1200.000
```

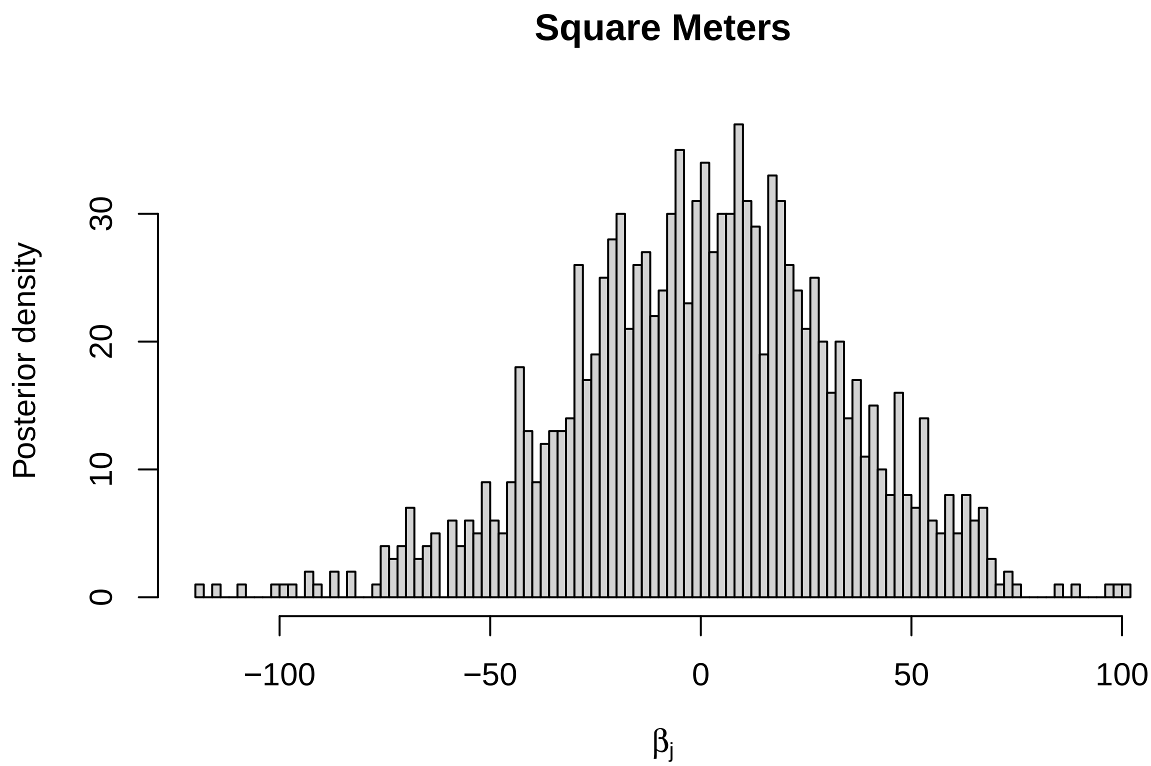
Compute DIC & WAIC

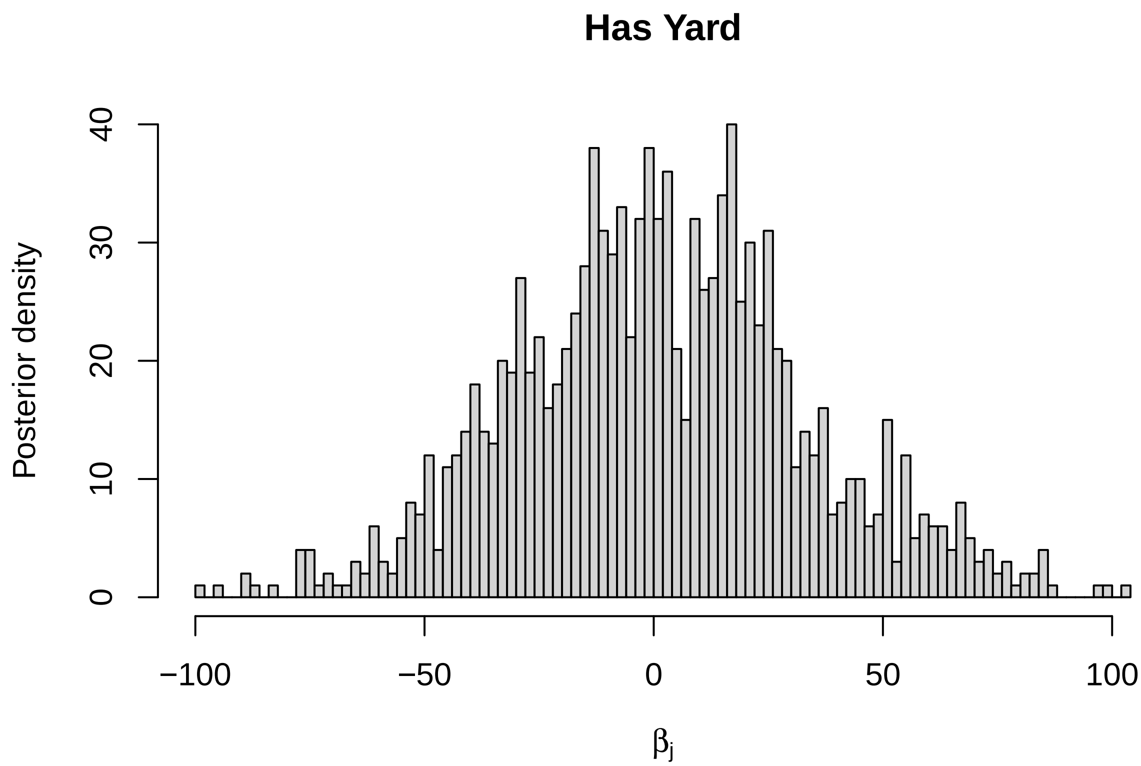
```r
# DIC
dic2 <- dic.samples(model2,n.iter=n.iter)

# WAIC
waic2 <- coda.samples(model2, variable.names=c("like"), n.iter=n.iter)
like2 <- waic2[[1]]
fbar2 <- colMeans(like2)
P2 <- sum(apply(log(like2),2,var))
WAIC2 <- -2*sum(log(fbar2))+2*P2
```

Plot Posterior Density Individually

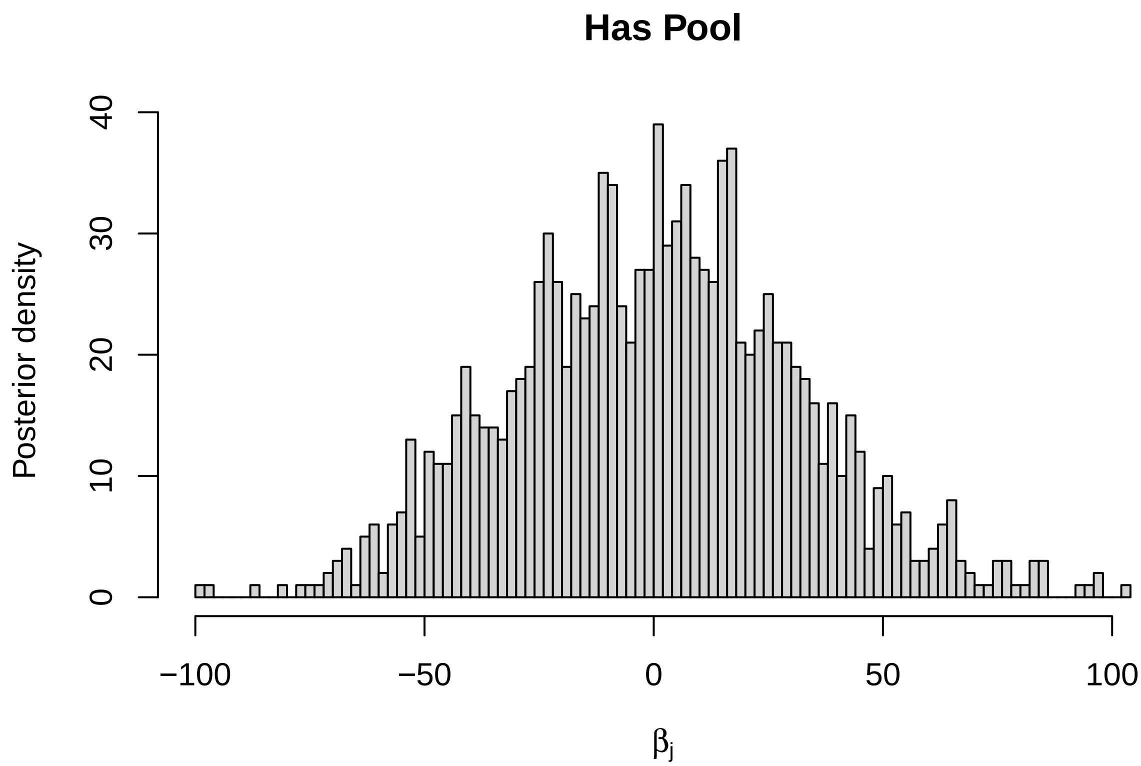
```r
beta <- NULL
for(l in 1:n.chains){
  beta <- rbind(beta,samples2[[l]])
}
colnames(beta) <- names
for(j in 1:8){
  hist(beta[,j],xlab=expression(beta[j]),ylab="Posterior density",
  breaks=100,main=names[j])
}
```

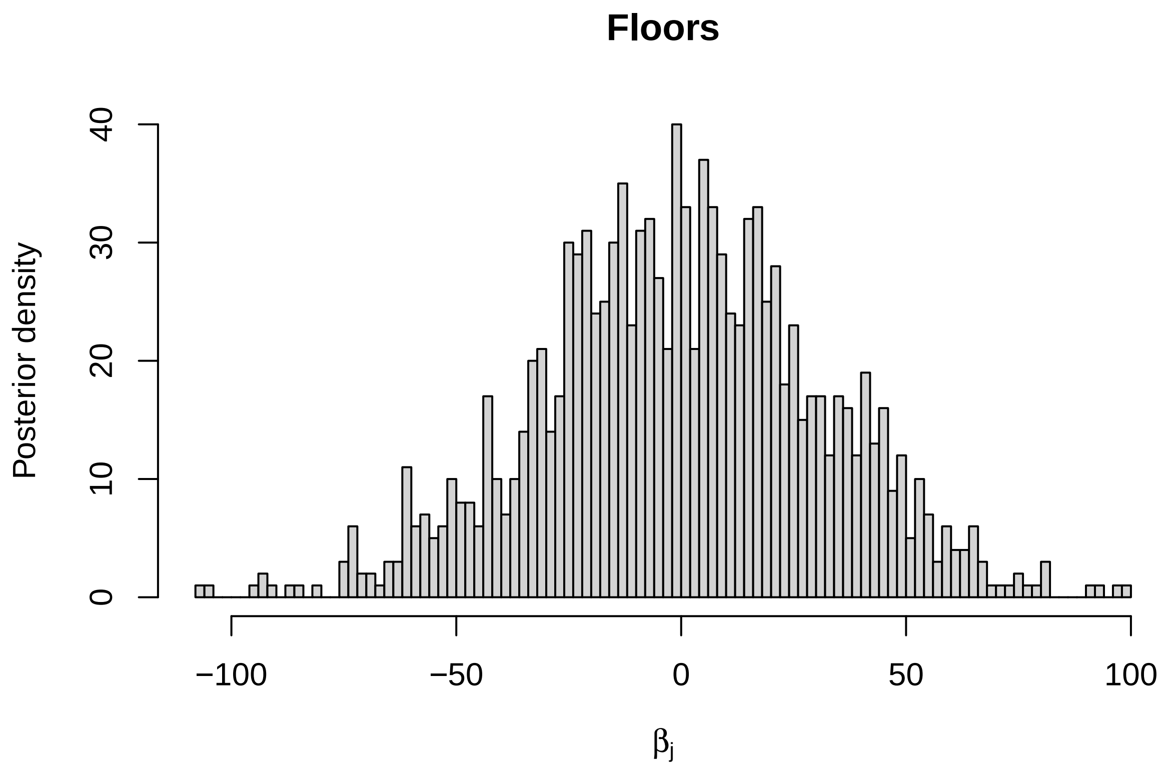


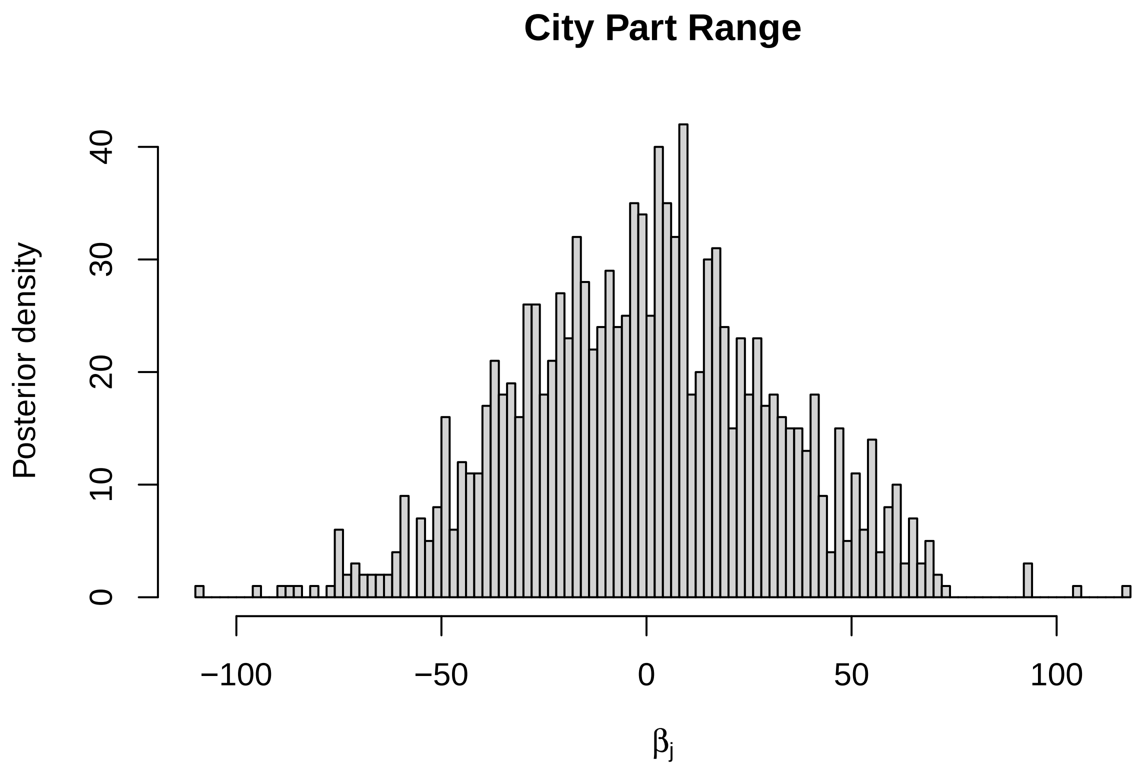

Square Meters
Posterior density
0
10
20
30
−100
−50
0
50
100
β_j

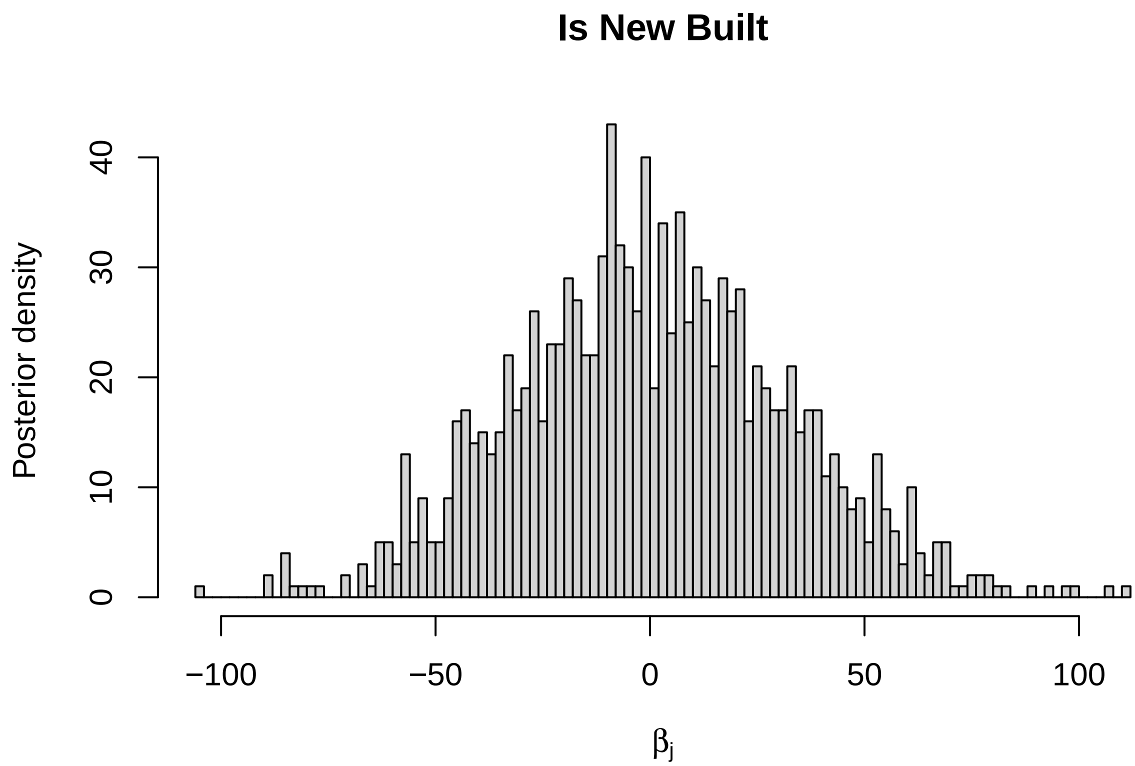


Has Yard
Posterior density
0
10
20
30
40
−100
−50
0
50
100
βj

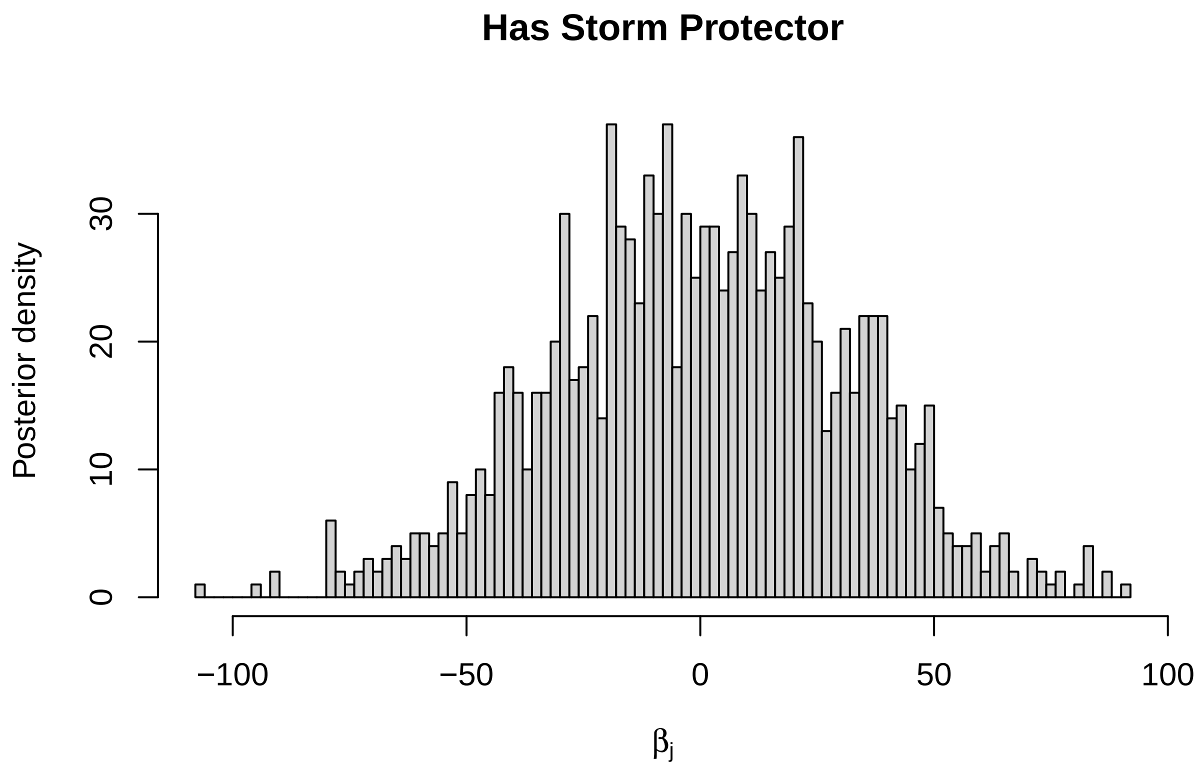

Has Pool
Posterior density
0
10
20
30
40
−100
−50
0
50
100
βj


Floors
Posterior density
0
10
20
30
40
−100
−50
0
50
100
β_j


City Part Range
Posterior density
0
10
20
30
40
−100
−50
0
50
100
β_j



Is New Built
Posterior density
0
10
20
30
40
−100
−50
0
50
100
β_j



Model 3 (Same like model 2, but using initial values (alpha and beta coefficients) from the estimation of frequentist linear model)

Split dependent and independent variable

```
X <- df[, c("squareMeters", "hasYard", "hasPool", "floors", "cityPartRange", "isNewBuilt", "hasStormPro
Y <- df$price
names <- c("price", "Square Meters", "Has Yard", "Has Pool", "Floors", "City Part Range", "Is New Built
```

Delete Missing Value

```
junk <- is.na(rowSums(X))
Y <- Y[!junk]
X <- X[!junk,]
```

Standardize Covariates

```r
X <- as.matrix(scale(X))
```

JAGS

Put Data in JAGS Format

```r
n <- length(Y)
p <- ncol(X)

data <- list(Y=Y,X=X,n=n,p=p)
params <- c("alpha","beta")
burn <- 500
n.iter <- 2000
n.chains <- 3
thin <- 5
```

Make Jags Model

```r
model_string <- textConnection("model{
  # Likelihood
  for(i in 1:n){
    Y[i] ~ dnorm(alpha+mu[i],taue)
    mu[i] <- inprod(X[i,],beta[])
  }

  # Priors
  for(j in 1:p){
    beta[j] ~ dnorm(0,0.001)
  }

  alpha ~ dnorm(0,0.001)
  taue ~ dgamma(0.1, 0.1)

  # WAIC calculations
  for(i in 1:n){
    like[i] <- dnorm(Y[i],mu[i],taue)
  }

}")
```

Frequentist Linear Regression

```r
price.lm2 <- lm(price ~ X, data = df)

summary(price.lm2)
```

```
##
## Call:
## lm(formula = price ~ X, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6976.4 -1193.3    -5.9  1201.2  6968.3
##
## Coefficients:
##                     Estimate Std. Error   t value Pr(>|t|)
## (Intercept)        4.993e+06  1.898e+01 2.631e+05  < 2e-16 ***
## XsquareMeters      2.877e+06  1.899e+01 1.516e+05  < 2e-16 ***
## XhasYard           1.506e+03  1.899e+01 7.930e+01  < 2e-16 ***
## XhasPool           1.489e+03  1.899e+01 7.841e+01  < 2e-16 ***
## Xfloors            1.576e+03  1.898e+01 8.303e+01  < 2e-16 ***
## XcityPartRange     1.355e+02  1.899e+01 7.135e+00 1.04e-12 ***
## XisNewBuilt        7.868e+01  1.899e+01 4.144e+00 3.43e-05 ***
## XhasStormProtector 7.105e+01  1.899e+01 3.742e+00 0.000183 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1898 on 9992 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 3.282e+09 on 7 and 9992 DF,  p-value: < 2.2e-16
```

Linear regression model : Yi = 4993000 + 2877000 * squareMeters + 1506 * hasYard + 1489 * hasPool + 1576 * floors + 135.5 * cityPartRange + 78.68 * isNewBuilt + 71.05 * hasStormProtector

Set Initial Value

We use the estimation alpha and betas from Frequentist Linear Regression Model

```r
inits = list()

inits$alpha = 4993000
inits$beta[1] = 2877000
inits$beta[2] = 1506
inits$beta[3] = 1489
inits$beta[4] = 1576
inits$beta[5] = 135.5
inits$beta[6] = 78.68
inits$beta[7] = 71.05
inits$taue = 10
```

Compile Model

```r
model3 <- jags.model(model_string, data = data, n.chains=n.chains, quiet=TRUE, inits = inits)
```

Update Model

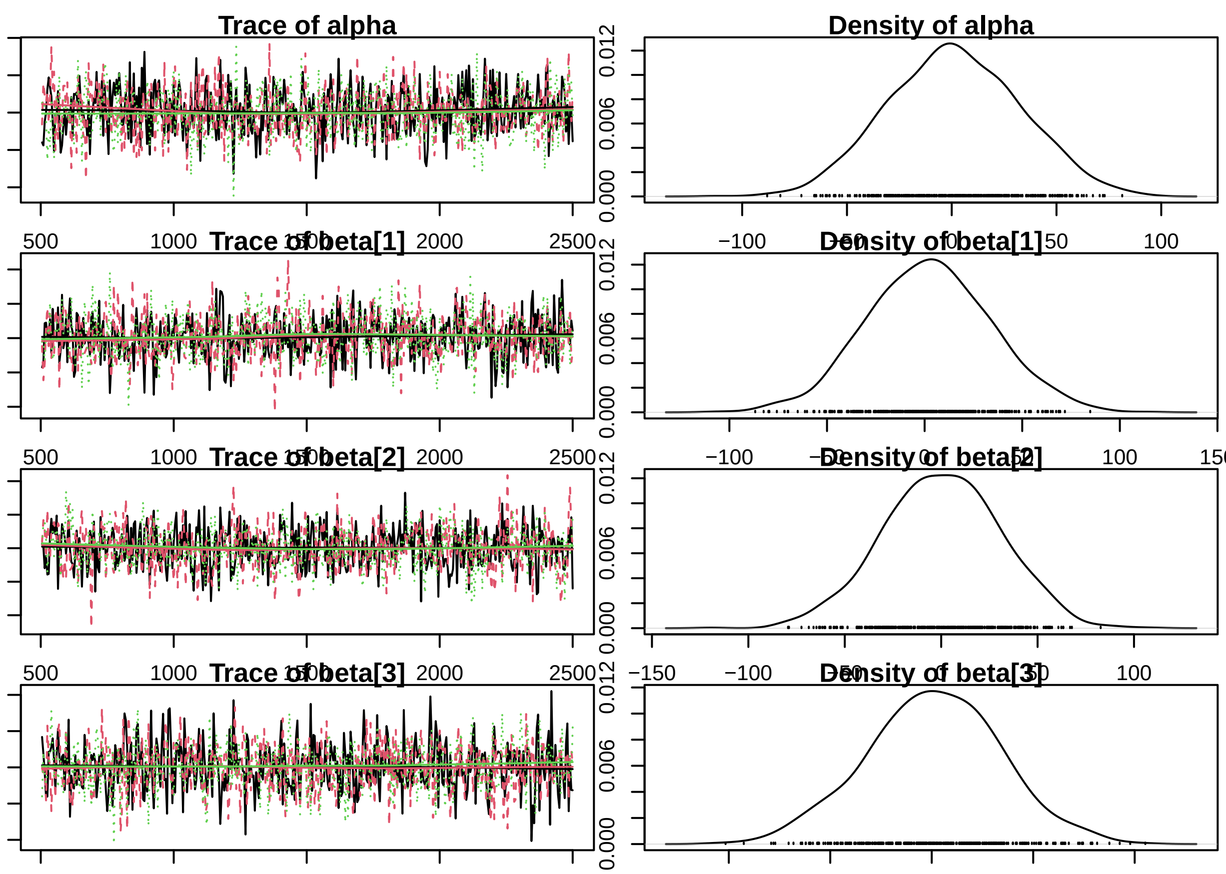
```r
update(model3, burn)
```

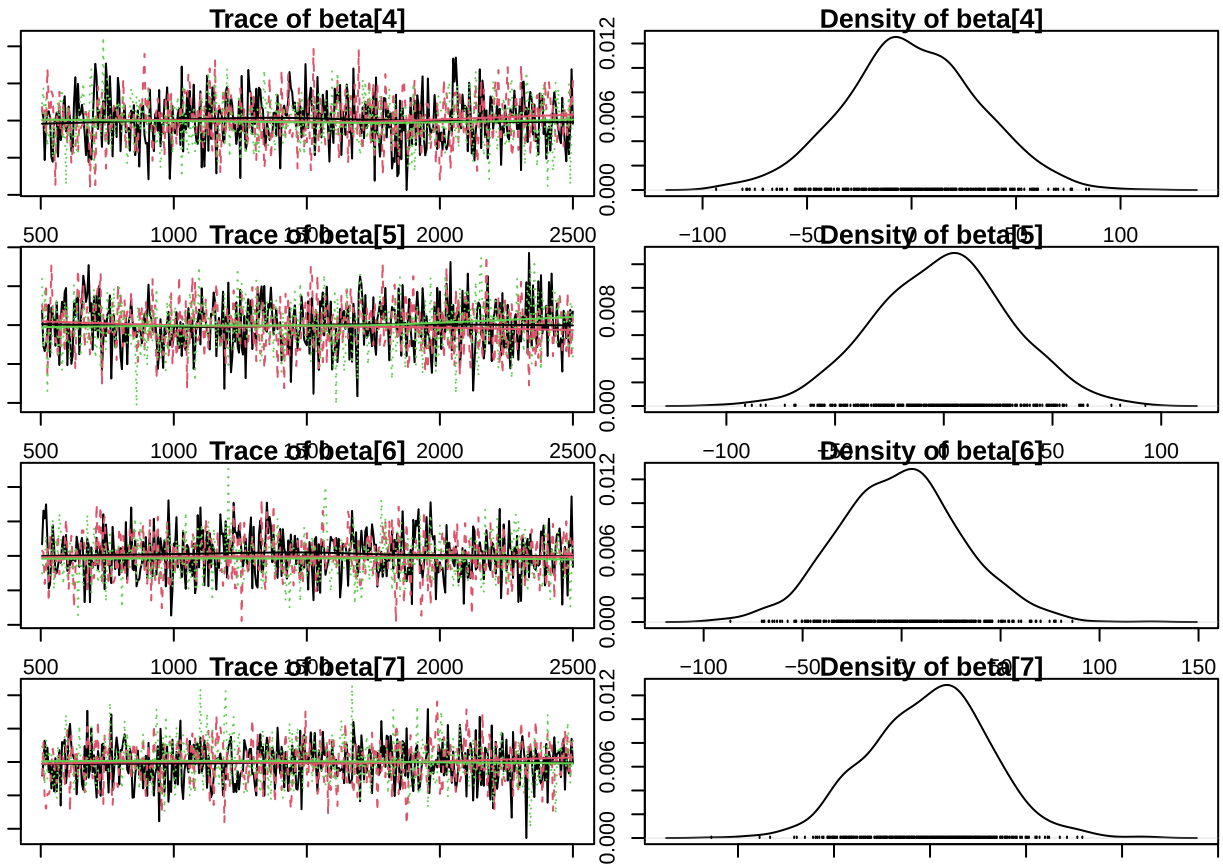
Get Posterior Samples from the model

```r
samples3 <- coda.samples(model3, variable.names=params, n.iter=n.iter, thin=thin)
```

Plot MCMC Chain Trace and Features Posterior Density

```r
par(mar=c(1,1,1,1))
plot(samples3)
```





Descriptive Statistics of model 3

```r
summary(samples3)
```

```
##
## Iterations = 505:2500
## Thinning interval = 5
## Number of chains = 3
## Sample size per chain = 400
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##              Mean    SD Naive SE Time-series SE
## alpha     1.22899 31.44   0.9077         0.9086
## beta[1]   2.29544 31.77   0.9172         0.8927
## beta[2]   0.19762 30.60   0.8833         0.8840
## beta[3]   0.77166 32.96   0.9515         0.9280
## beta[4]   0.05361 32.10   0.9266         0.8893
## beta[5]  -0.59112 31.12   0.8983         0.8730
## beta[6]  -0.54344 30.89   0.8916         0.8592
## beta[7]  -0.26234 31.18   0.9000         0.8367
##
```

```
## 2. Quantiles for each variable:
##
##           2.5%    25%     50%   75% 97.5%
## alpha   -58.37 -21.22  0.7081 22.85 62.84
## beta[1] -60.26 -19.72  2.2413 23.55 65.54
## beta[2] -61.12 -20.05  0.6380 20.82 57.49
## beta[3] -65.04 -21.10  1.0293 22.73 66.52
## beta[4] -64.37 -21.01 -0.9178 20.64 63.50
## beta[5] -59.81 -21.62  1.0063 19.64 61.29
## beta[6] -62.92 -21.34 -0.1015 18.76 60.65
## beta[7] -60.82 -20.83  1.5131 20.19 60.37
```

Show Alpha and Betas to the corresponding Features

```r
sum <- summary(samples3)
rownames(sum$statistics) <- names
rownames(sum$quantiles) <- names
sum$statistics <- round(sum$statistics,3)
sum$quantiles <- round(sum$quantiles,3)
sum
```

```
##
## Iterations = 505:2500
## Thinning interval = 5
## Number of chains = 3
## Sample size per chain = 400
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                         Mean    SD Naive SE Time-series SE
## price                  1.229 31.44    0.908          0.909
## Square Meters          2.295 31.77    0.917          0.893
## Has Yard               0.198 30.60    0.883          0.884
## Has Pool               0.772 32.96    0.951          0.928
## Floors                 0.054 32.10    0.927          0.889
## City Part Range       -0.591 31.12    0.898          0.873
## Is New Built          -0.543 30.89    0.892          0.859
## Has Storm Protector -0.262 31.18    0.900          0.837
##
## 2. Quantiles for each variable:
##
##                         2.5%    25%    50%   75% 97.5%
## price                 -58.37 -21.22  0.708 22.85 62.84
## Square Meters         -60.26 -19.72  2.241 23.55 65.54
## Has Yard              -61.12 -20.05  0.638 20.82 57.49
## Has Pool              -65.04 -21.10  1.029 22.73 66.52
## Floors                -64.37 -21.01 -0.918 20.64 63.50
## City Part Range       -59.81 -21.62  1.006 19.64 61.29
## Is New Built          -62.92 -21.34 -0.102 18.76 60.65
## Has Storm Protector -60.82 -20.83  1.513 20.19 60.37
```

Check Convergence

```
gelman.diag(samples3)
```

```
## Potential scale reduction factors:
##
##          Point est. Upper C.I.
## alpha        0.998      0.999
## beta[1]      1.002      1.009
## beta[2]      1.001      1.002
## beta[3]      1.001      1.007
## beta[4]      1.001      1.007
## beta[5]      1.002      1.011
## beta[6]      1.003      1.016
## beta[7]      0.999      0.999
##
## Multivariate psrf
##
## 1.01
```

Compile results

```
ESS3 <- effectiveSize(samples3)
out3 <- summary(samples3)$quantiles
rownames(out3)<-names

ESS3
```

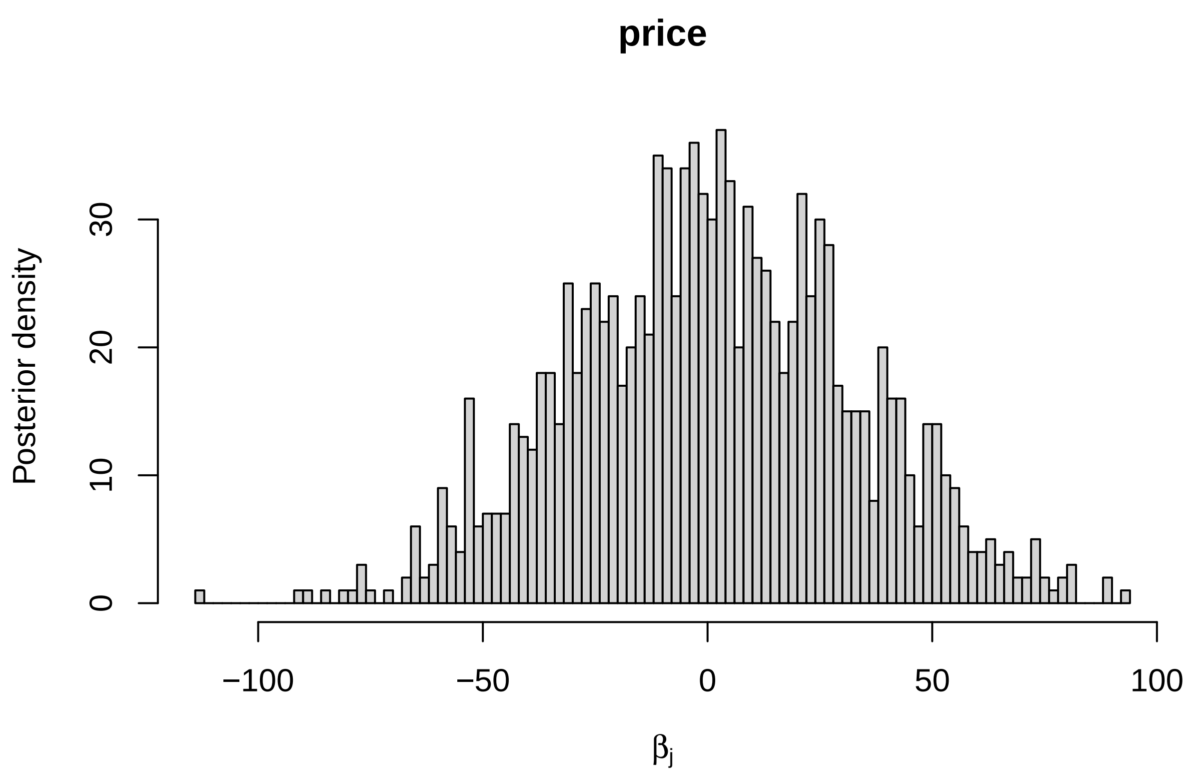
```
##    alpha  beta[1]  beta[2]  beta[3]  beta[4]  beta[5]  beta[6]  beta[7]
## 1213.985 1267.530 1200.000 1264.617 1325.921 1279.350 1302.553 1425.877
```

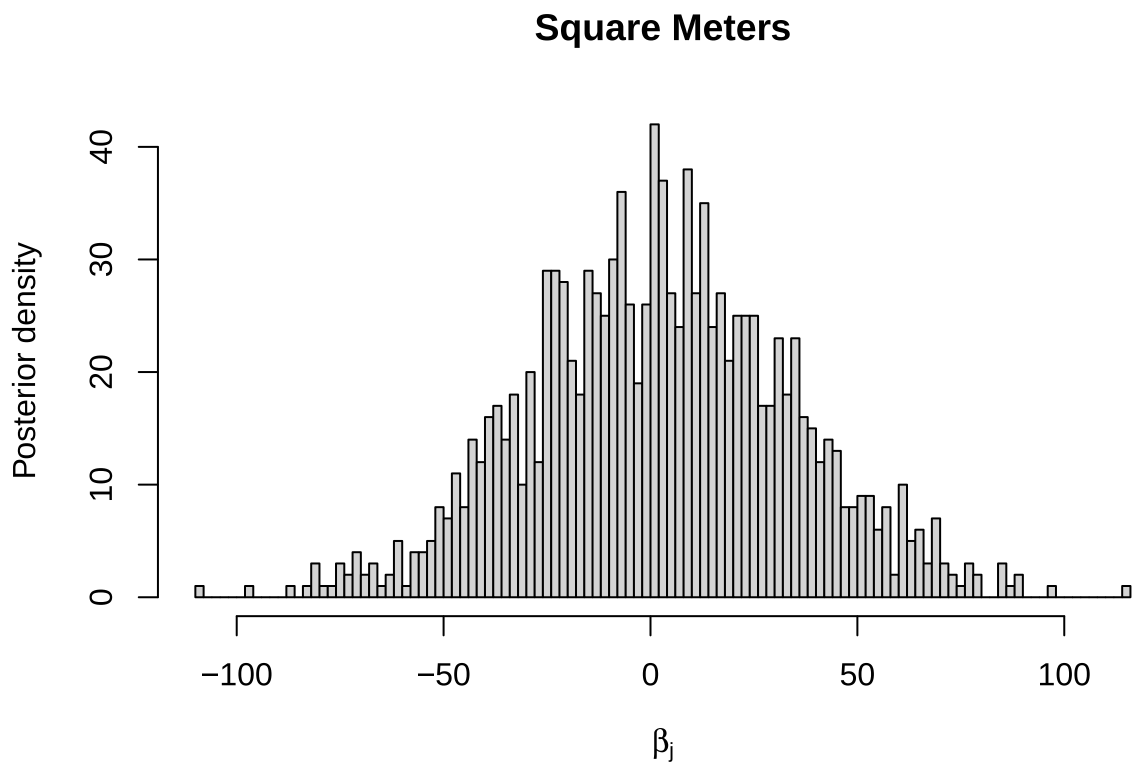
Compute DIC & WAIC

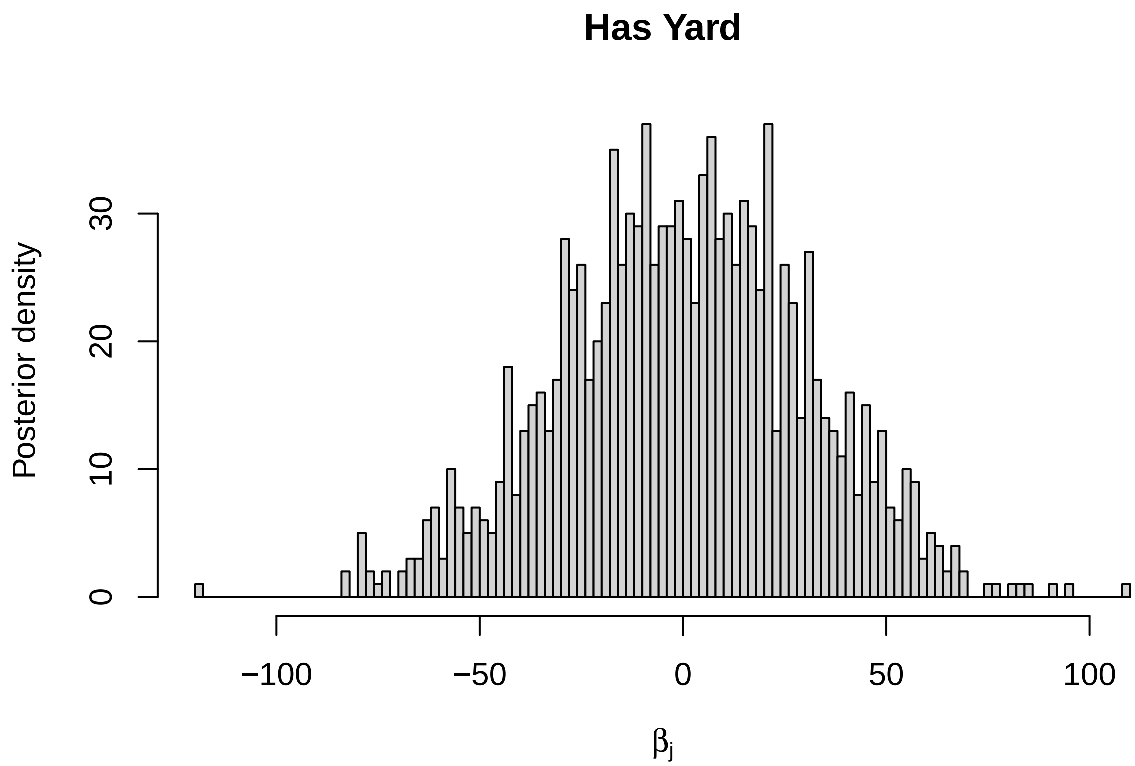
```
# DIC
dic3 <- dic.samples(model3,n.iter=n.iter)

# WAIC
waic3 <- coda.samples(model3, variable.names=c("like"), n.iter=n.iter)
like3 <- waic3[[1]]
fbar3 <- colMeans(like3)
P3 <- sum(apply(log(like3),2,var))
WAIC3 <- -2*sum(log(fbar3))+2*P3
```

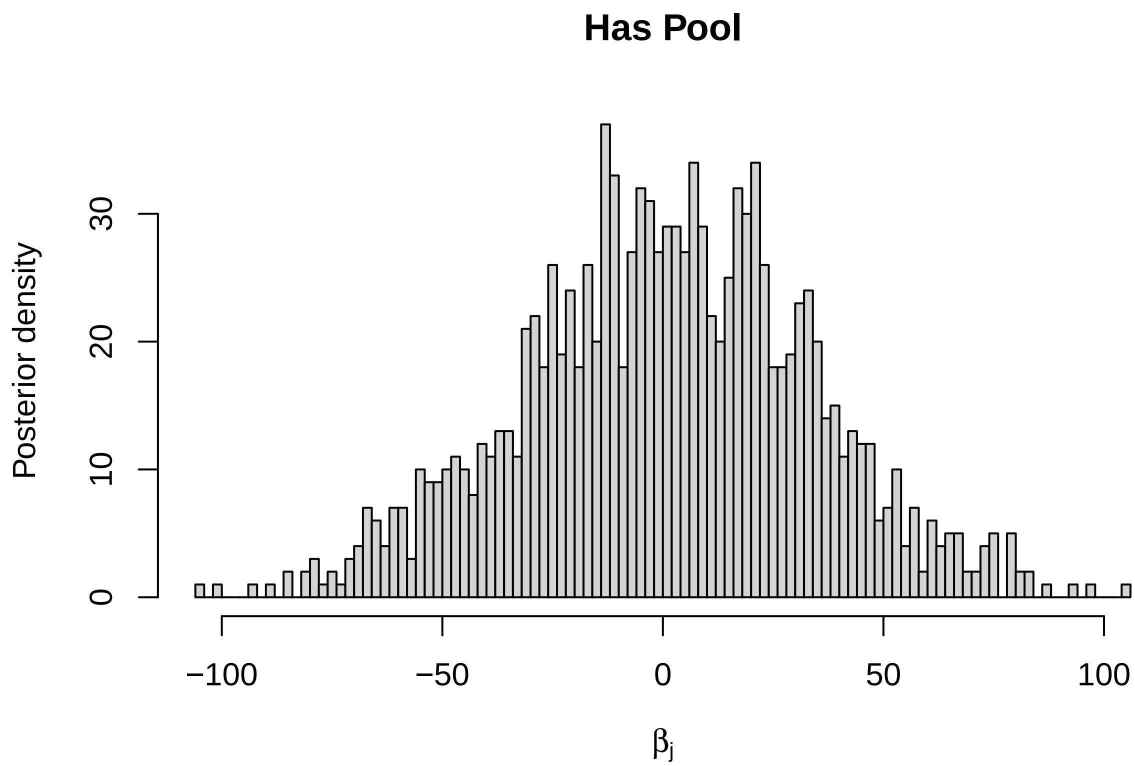
Plot Posterior Density Individually

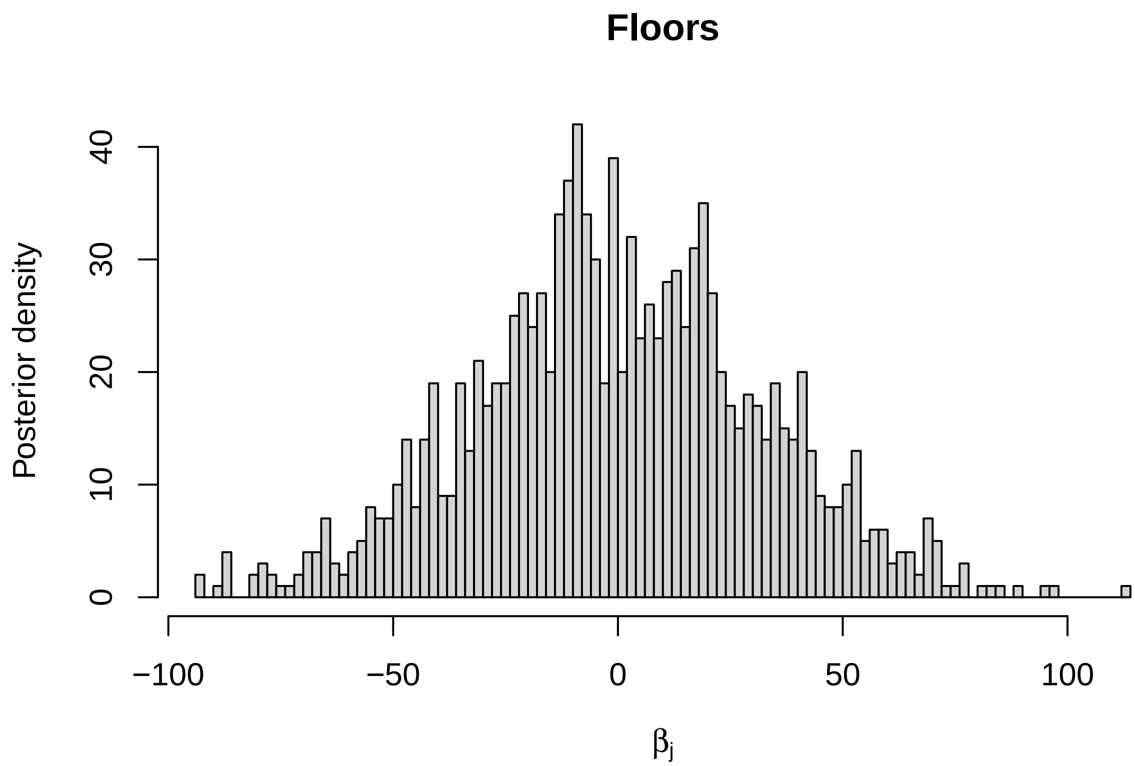
```r
beta <- NULL
for(l in 1:n.chains){
  beta <- rbind(beta,samples3[[l]])
}
colnames(beta) <- names
for(j in 1:8){
  hist(beta[,j],xlab=expression(beta[j]),ylab="Posterior density",
  breaks=100,main=names[j])
}
```

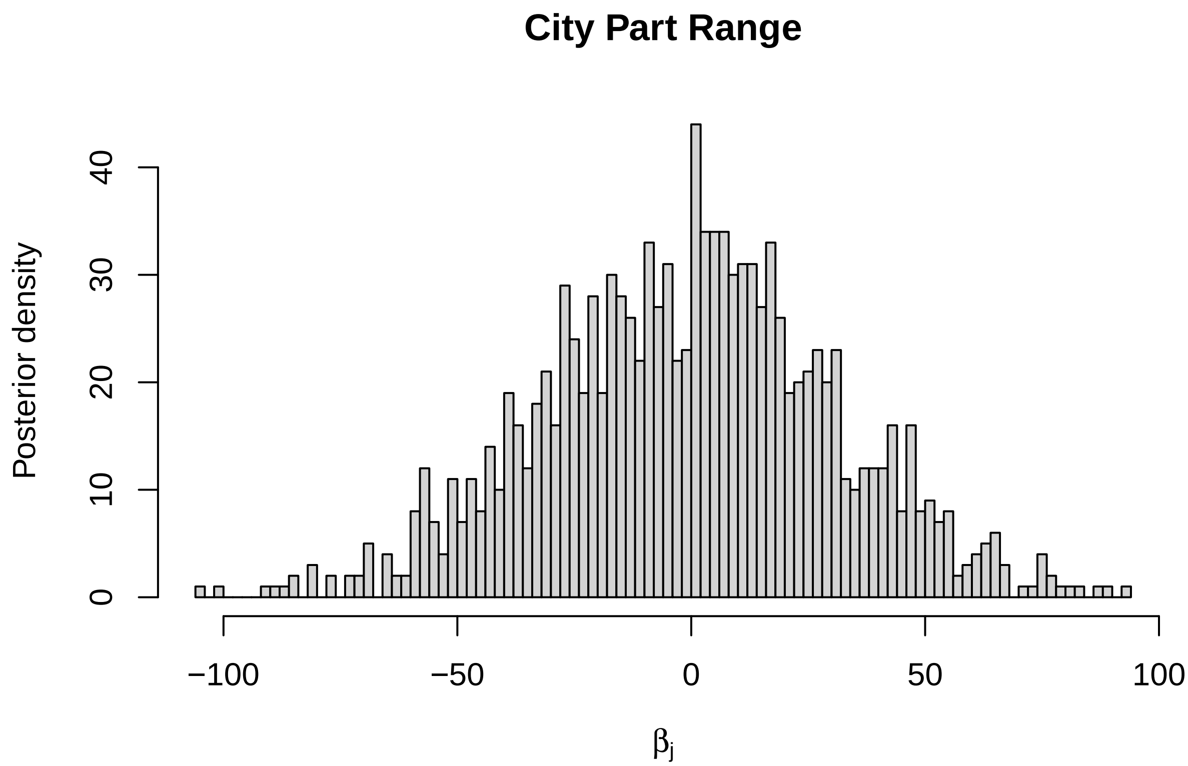


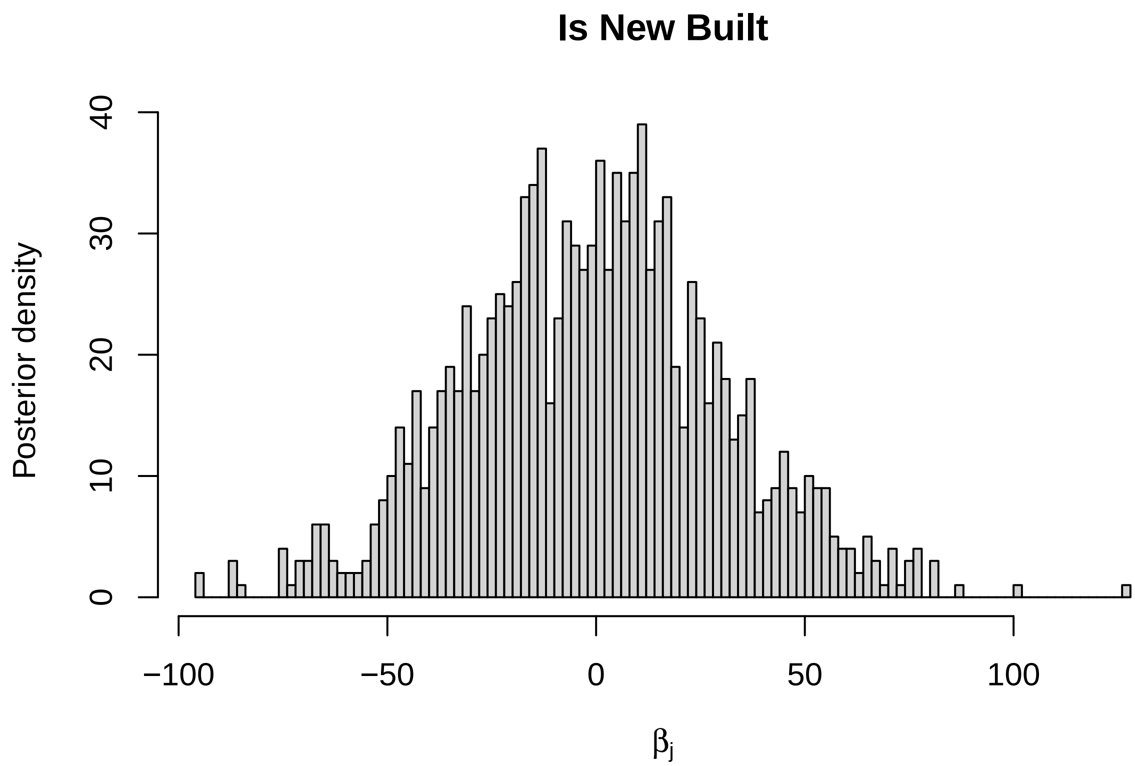

Square Meters
Posterior density
0
10
20
30
40
−100
−50
0
50
100
β_j

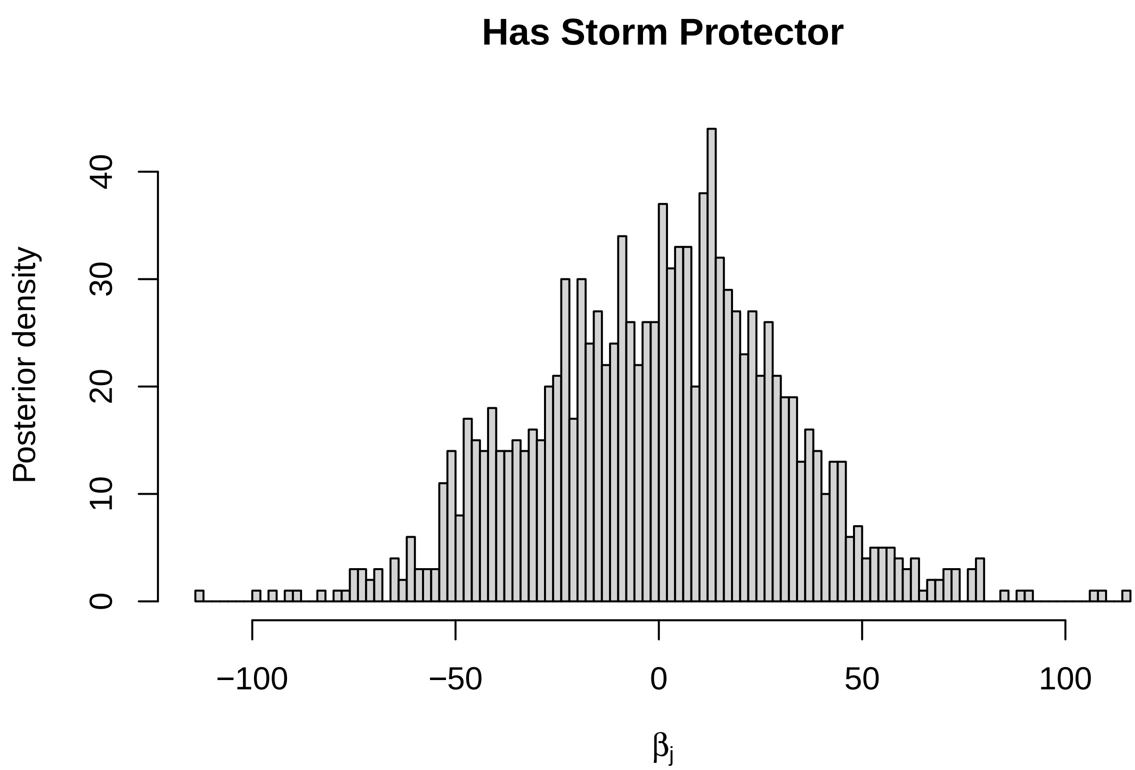

Has Yard
Posterior density
0
10
20
30
−100
−50
0
50
100
βj


Has Pool
Posterior density
0
10
20
30
−100
−50
0
50
100
β_j


Floors
Posterior density
0
10
20
30
40
−100
−50
0
50
100
βj


City Part Range
Posterior density
0
10
20
30
40
−100
−50
0
50
100
βj


Is New Built
Posterior density
0
10
20
30
40
−100
−50
0
50
100
βj



Model 4

Set Dataframe as matrix and split dependent and independent variable

```
price <- as.matrix(df$price)
Y <- price
X <- cbind(df$squareMeters, df$numberOfRooms, df$hasYard, df$hasPool, df$floors, df$cityCode, df$numPrev
names <- c("price","Square Meters", "Number of Rooms", "Has Yard", "Has Pool", "Number of Floors", "City
```

Delete Missing Value

```
junk <- is.na(rowSums(X))
Y <- Y[!junk]
X <- X[!junk,]
```

Standardize Covariates

```
X <- as.matrix(scale(X))
```

JAGS

Put Data in JAGS Format

```
n <- length(Y)
p <- ncol(X)

data <- list(Y=Y,X=X,n=n,p=p)
params <- c("alpha","beta")
burn <- 500
n.iter <- 5000
n.chains <- 3
thin <- 5
```

Make Jags Model

```
model_string <- textConnection("model{
  # Likelihood
  for(i in 1:n){
    Y[i] ~ dnorm(alpha+mu[i],taue)
    mu[i] <- theta[i] + inprod(X[i,],beta[])
  }

  # Random Effects
  for(j in 1:n){
    theta[j] ~ ddexp(0,taue)
  }

  # Priors
  for(j in 1:p){
    beta[j] ~ dnorm(0,0.001)
  }

  alpha ~ dnorm(0,0.001)
  taue ~ dgamma(0.1, 0.1)

  # WAIC calculations
  for(i in 1:n){
    like[i] <- dnorm(Y[i],mu[i],taue)
  }

}")
```

Set Initial Value

```
inits = list()

inits$alpha = rnorm(1)
for(i in 1:p) {
```

```r
    inits$beta[i] = rnorm(1)
}
for(i in 1:n) {
    inits$theta[i] = rnorm(1)
}
inits$taue = 10
```

Compile Model

```r
model4 <- jags.model(model_string, data = data, n.chains=n.chains, quiet=TRUE, inits = inits)
```

Update Model

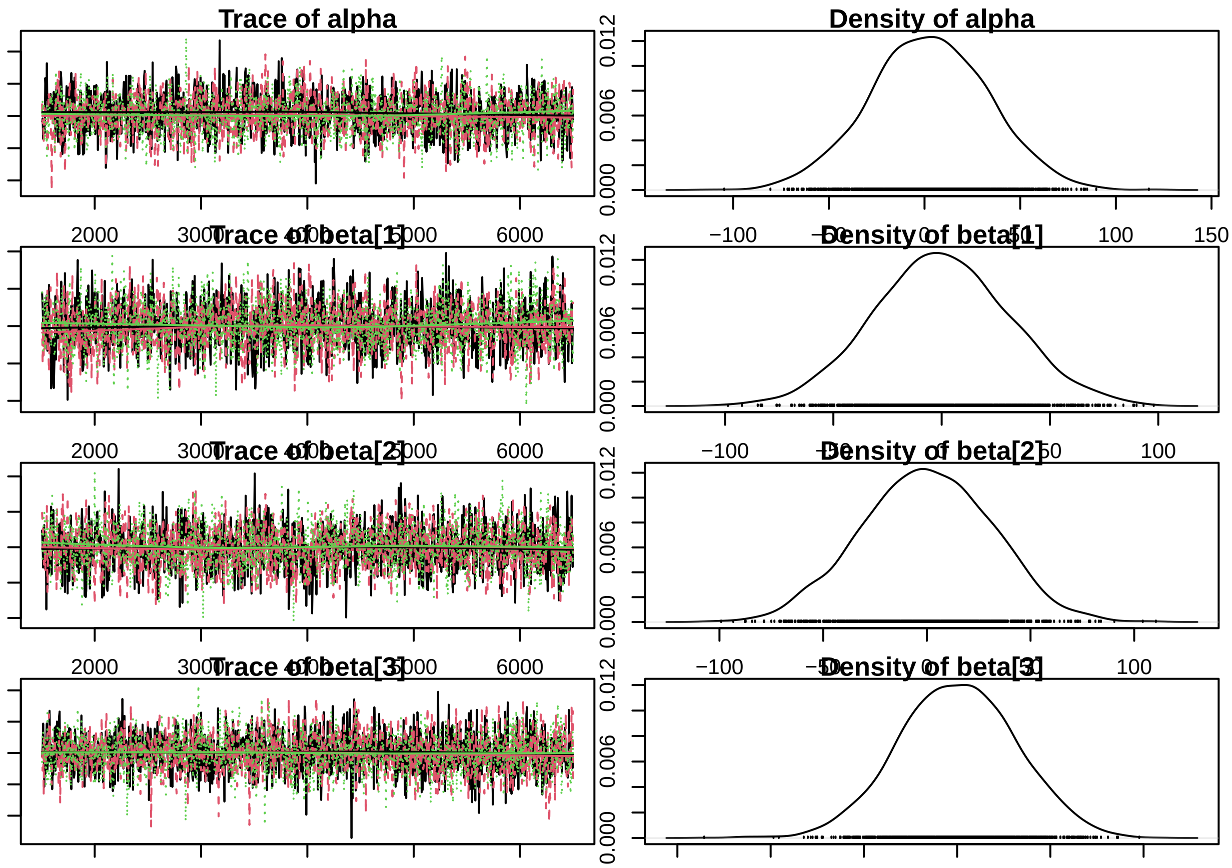
```r
update(model4, burn)
```

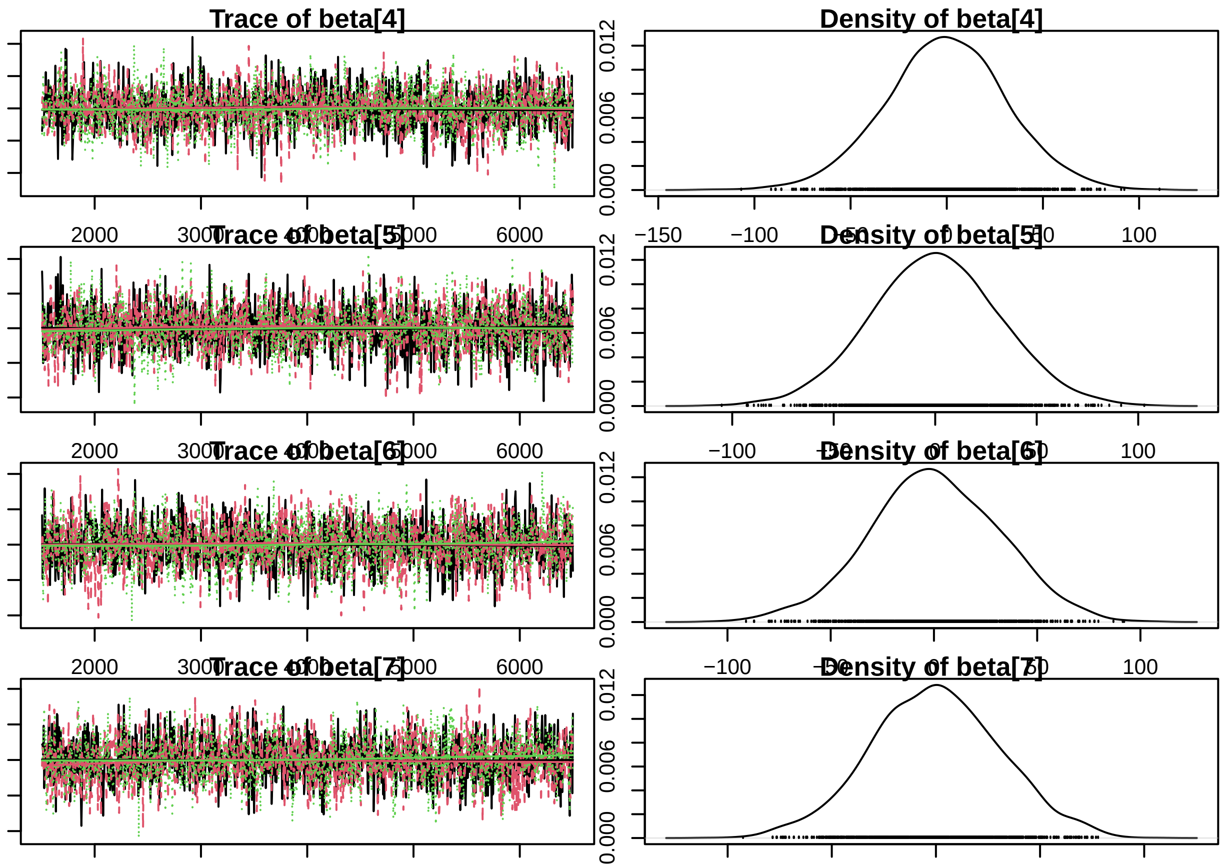
Get Posterior Samples from the model

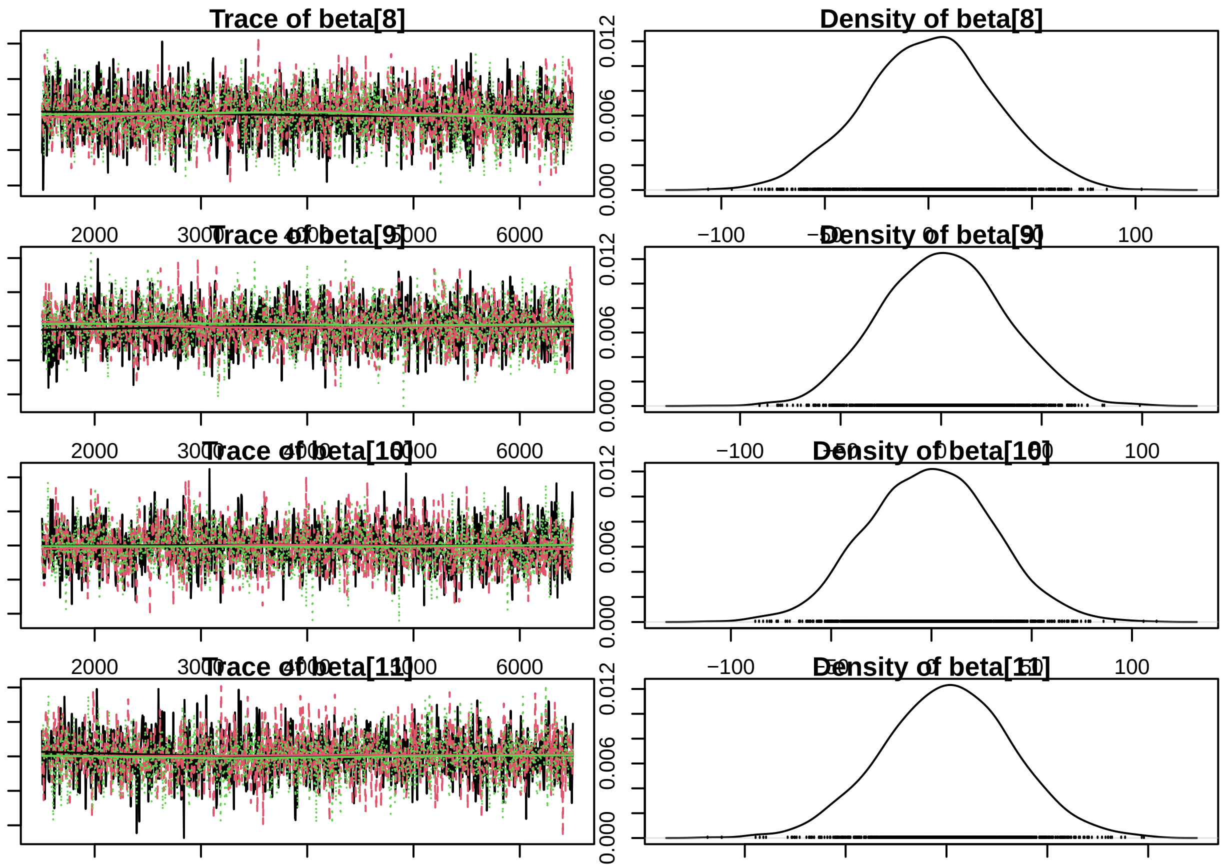
```r
samples4 <- coda.samples(model4, variable.names=params, n.iter=n.iter, thin=thin)
```

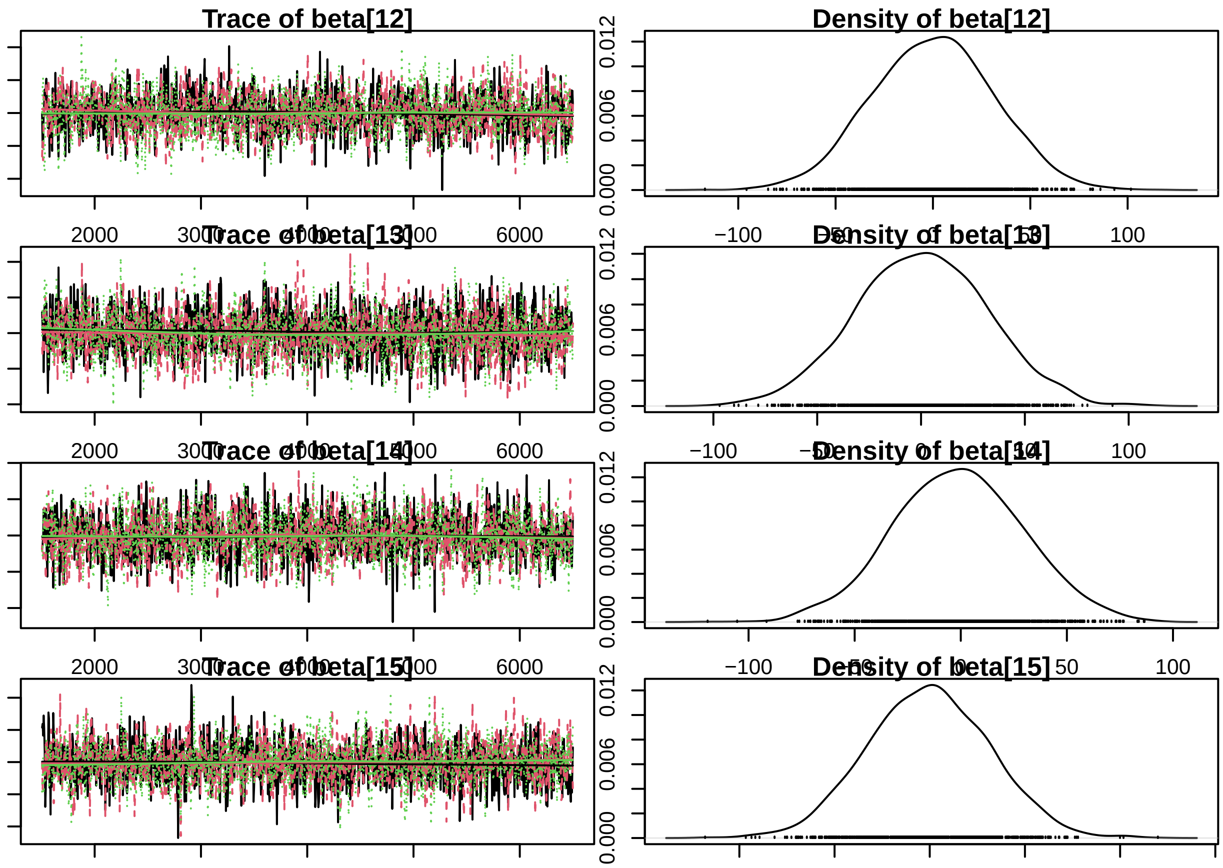
Plot MCMC Chain Trace and Features Posterior Density

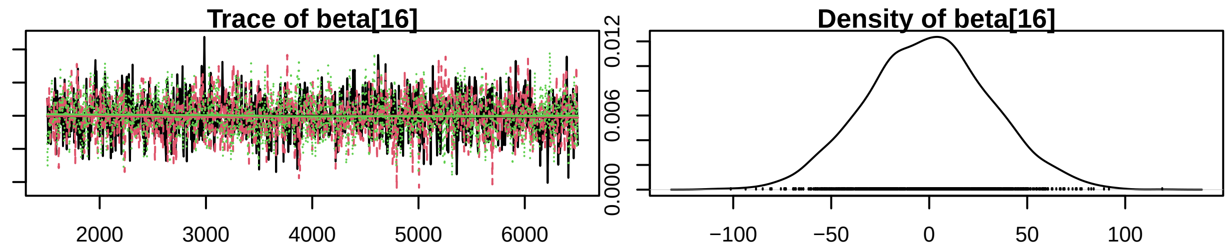
```r
par(mar=c(1,1,1,1))
plot(samples4)
```


Trace of alpha
Density of alpha
Trace of beta[1]
Density of beta[1]
Trace of beta[2]
Density of beta[2]
Trace of beta[3]
Density of beta[3]
2000
3000
4000
5000
6000
−100
−50
0
50
100
150
0.000
0.006
0.012


Trace of beta[4]
Density of beta[4]
Trace of beta[5]
Density of beta[5]
Trace of beta[6]
Density of beta[6]
Trace of beta[7]
Density of beta[7]
2000
3000
4000
5000
6000
0.000
0.006
0.012
−150
−100
−50
0
50
100


Trace of beta[8]
Density of beta[8]
Trace of beta[9]
Density of beta[9]
Trace of beta[10]
Density of beta[10]
Trace of beta[11]
Density of beta[11]
2000
3000
4000
5000
6000
−100
−50
0
50
100
0.000
0.006
0.012


Trace of beta[12]
Density of beta[12]
Trace of beta[13]
Density of beta[13]
Trace of beta[14]
Density of beta[14]
Trace of beta[15]
Density of beta[15]
2000
3000
4000
5000
6000
−100
−50
0
50
100
0.000
0.006
0.012



Show Alpha and Betas to the corresponding Features

```
sum <- summary(samples4)
rownames(sum$statistics) <- names
rownames(sum$quantiles) <- names
sum$statistics <- round(sum$statistics,3)
sum$quantiles <- round(sum$quantiles,3)
sum
```

```
##
## Iterations = 1505:6500
## Thinning interval = 5
## Number of chains = 3
## Sample size per chain = 1000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                              Mean    SD Naive SE Time-series SE
## price                       2.301 31.18    0.569          0.647
## Square Meters              -0.081 31.62    0.577          0.679
## Number of Rooms            -0.296 31.21    0.570          0.633
## Has Yard                    0.100 31.62    0.577          0.607
```

```
## Has Pool                  -0.443 31.62    0.577          0.678
## Number of Floors           0.081 31.89    0.582          0.649
## City Code                  0.294 31.76    0.580          0.680
## Number of Previous Owners -0.153 30.92    0.565          0.698
## Year Made                  0.473 31.86    0.582          0.651
## is Newly Built             1.001 30.72    0.561          0.609
## Has Storm Protector       -0.711 31.36    0.572          0.643
## Basement Area              0.610 32.28    0.589          0.686
## Attic Area                 0.095 31.15    0.569          0.683
## Garage Area                0.382 31.78    0.580          0.652
## Has Storage Room          -1.187 30.53    0.557          0.635
## Has Guest Room            -0.634 31.98    0.584          0.657
## City Part Range           -0.432 31.78    0.580          0.668
##
## 2. Quantiles for each variable:
##
##                             2.5%    25%    50%   75% 97.5%
## price                     -59.02 -18.32  2.424 23.65 62.95
## Square Meters             -61.00 -21.54 -0.234 20.57 63.26
## Number of Rooms           -60.50 -21.33 -0.456 20.94 58.76
## Has Yard                  -61.69 -21.39  0.141 21.91 61.03
## Has Pool                  -63.95 -20.23 -0.380 20.42 62.02
## Number of Floors          -62.81 -21.43  0.112 20.90 63.18
## City Code                 -64.86 -20.66 -0.164 22.14 62.36
## Number of Previous Owners -61.23 -20.79 -0.373 20.73 63.09
## Year Made                 -61.39 -20.81  1.024 21.64 63.30
## is Newly Built            -57.21 -20.38  0.906 21.54 60.80
## Has Storm Protector       -60.77 -21.79 -0.556 20.56 60.84
## Basement Area             -62.55 -20.70  1.014 22.27 64.14
## Attic Area                -60.13 -21.19  0.569 21.08 59.22
## Garage Area               -62.14 -21.12  0.696 22.25 64.13
## Has Storage Room          -62.52 -21.95 -1.237 19.36 59.07
## Has Guest Room            -61.36 -22.13 -0.376 20.90 60.53
## City Part Range           -61.24 -21.51 -0.159 20.80 62.81
```

Check Convergence

```
gelman.diag(samples4)
```

```
## Potential scale reduction factors:
##
##          Point est. Upper C.I.
## alpha             1       1.01
## beta[1]           1       1.00
## beta[2]           1       1.00
## beta[3]           1       1.00
## beta[4]           1       1.00
## beta[5]           1       1.00
## beta[6]           1       1.00
## beta[7]           1       1.01
## beta[8]           1       1.01
## beta[9]           1       1.01
```

```
## beta[10]          1        1.00
## beta[11]          1        1.00
## beta[12]          1        1.00
## beta[13]          1        1.00
## beta[14]          1        1.00
## beta[15]          1        1.02
## beta[16]          1        1.00
##
## Multivariate psrf
##
## 1.01
```

Compile results

```
ESS2 <- effectiveSize(samples4)
out2 <- summary(samples4)$quantiles
rownames(out2)<-names

ESS2
```

```
##    alpha  beta[1]  beta[2]  beta[3]  beta[4]  beta[5]  beta[6]  beta[7]
## 2317.755 2196.744 2434.245 2827.537 2194.905 2428.948 2192.781 1966.371
##  beta[8]  beta[9] beta[10] beta[11] beta[12] beta[13] beta[14] beta[15]
## 2453.230 2549.585 2379.390 2219.239 2098.781 2377.606 2334.667 2376.514
## beta[16]
## 2262.405
```

Compute DIC & WAIC

```
# DIC
dic4 <- dic.samples(model4,n.iter=n.iter)

# WAIC
waic4 <- coda.samples(model4, variable.names=c("like"), n.iter=n.iter)
like4 <- waic4[[1]]
fbar4 <- colMeans(like4)
P4 <- sum(apply(log(like4),2,var))
WAIC4 <- -2*sum(log(fbar4))+2*P4
```

DIC and WAIC comparison

```
print("DIC Model 1:")
```

```
## [1] "DIC Model 1:"
```

```r
print(dic1)
```

```
## Mean deviance:  339719
## penalty 1.036
## Penalized deviance: 339721
```

```r
print("DIC Model 2:")
```

```
## [1] "DIC Model 2:"
```

```r
print(dic2)
```

```
## Mean deviance:  339719
## penalty 0.9815
## Penalized deviance: 339720
```

```r
print("DIC Model 3:")
```

```
## [1] "DIC Model 3:"
```

```r
print(dic3)
```

```
## Mean deviance:  339719
## penalty 0.9966
## Penalized deviance: 339720
```

```r
print("DIC Model 4:")
```

```
## [1] "DIC Model 4:"
```

```r
print(dic4)
```

```
## Mean deviance:  182610
## penalty 10000
## Penalized deviance: 192610
```

```r
print("WAIC Model 1:")
```

```
## [1] "WAIC Model 1:"
```

```r
print(WAIC1)
```

```
## [1] 339719.8
```

```r
print("WAIC Model 2:")
```

```
## [1] "WAIC Model 2:"
```

```r
print(WAIC2)
```

```
## [1] 339719.9
```

```r
print("WAIC Model 3:")
```

```
## [1] "WAIC Model 3:"
```

```r
print(WAIC3)
```

```
## [1] 339719.9
```

```r
print("WAIC Model 4:")
```

```
## [1] "WAIC Model 4:"
```

```r
print(WAIC4)
```

```
## [1] 189554.6
```