Breast Cancer Prediction Case Study - Bayesian Logistic Regression with Comparison of Frequentist and Bayesian Variable Selection Methods

Laura Silvana Alvarez Luque, Florencia Luque, Nicolas Bühringer, Simon Schmetz

2025-03-14

Introduction

This case study is based on the Breast Cancer Wisconsin (Diagnostic) Data Set (https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data). The data set contains 569 observations and 32 variables. The data set is available at the UCI Machine Learning Repository. The data set contains mean (and at times min and max) values of the patient for the following numeric (continious) variables:

a) radius (mean of distances from center to points on the perimeter)
b) texture (standard deviation of gray-scale values)
c) perimeter
d) area
e) smoothness (local variation in radius lengths)
f) compactness (perimeter^2 / area - 1.0)
g) concavity (severity of concave portions of the contour)
h) concave points (number of concave portions of the contour)
i) symmetry
j) fractal dimension ("coastline approximation" - 1)

The data set also contains the following Binary variables:

2) Diagnosis (M = malignant, B = benign)

Where Malignant (M) means the tumor is cancerous, while Benign (B): means that the tumor is non-cancerous.

Read Data

```r
data <- read.csv("data.csv", header = TRUE, sep = ",")
data <- dplyr::select(data, -c(X,id))
names(data) <- gsub("\\.", "_", names(data))
data$diagnosis <- ifelse(data$diagnosis == "M", 1, 0)
```

Exploratory Data Analysis

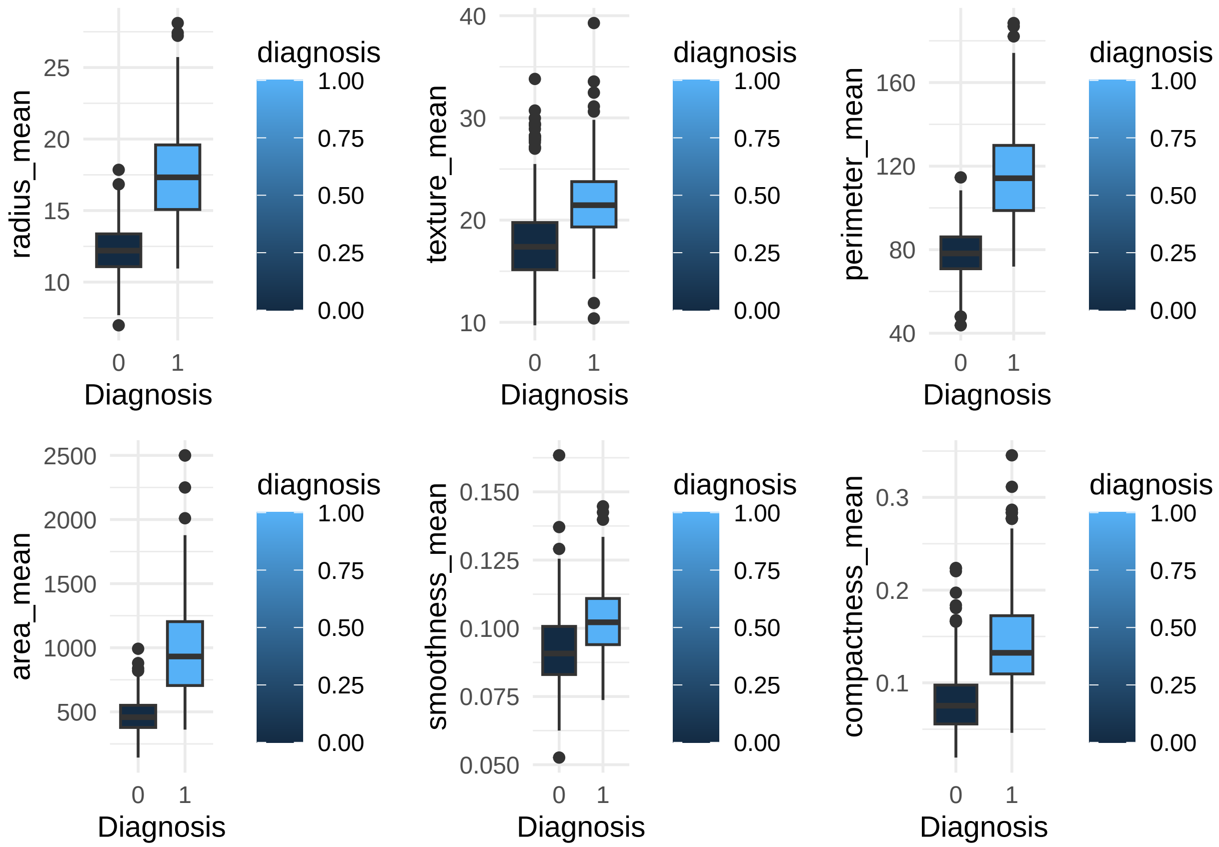
Relation with response var

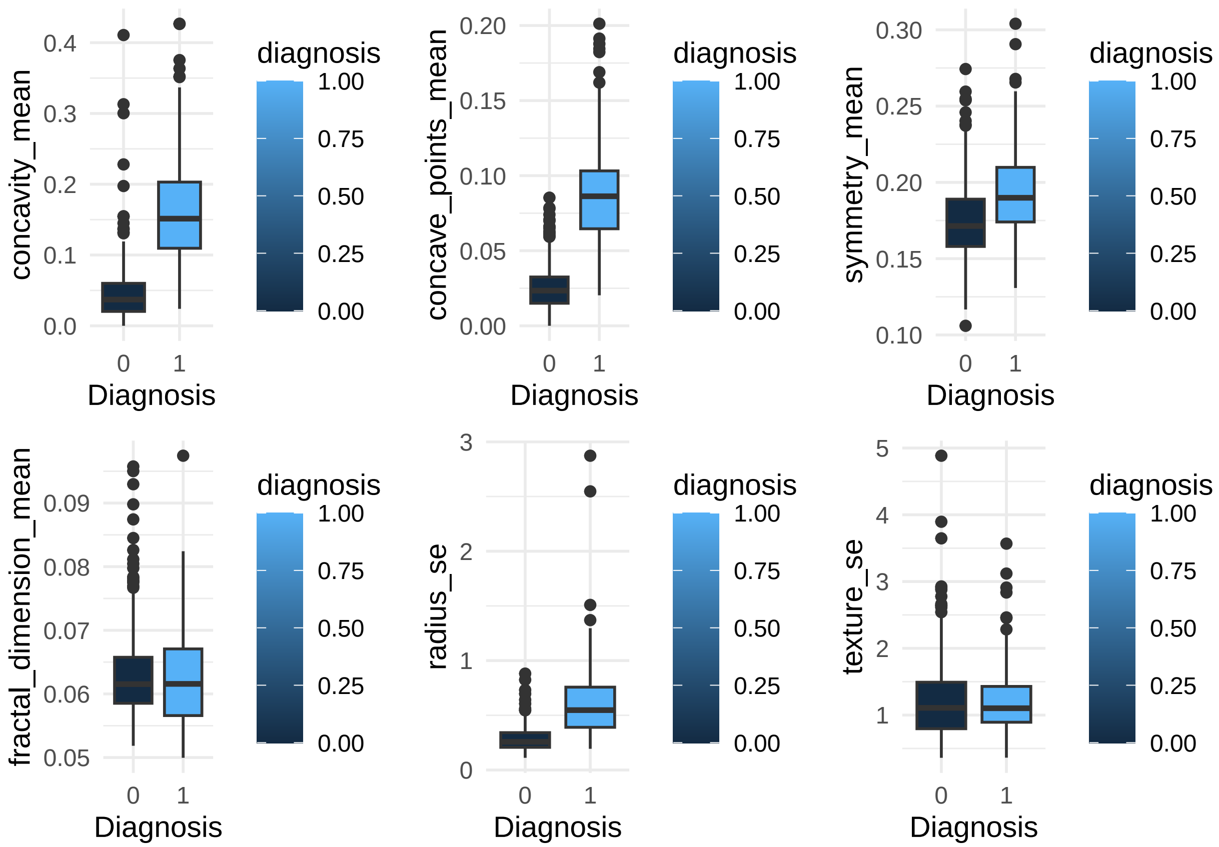
```r
numeric_vars <- data %>%select_if(is.numeric) %>% colnames()
numeric_vars <- setdiff(numeric_vars, "diagnosis")
```

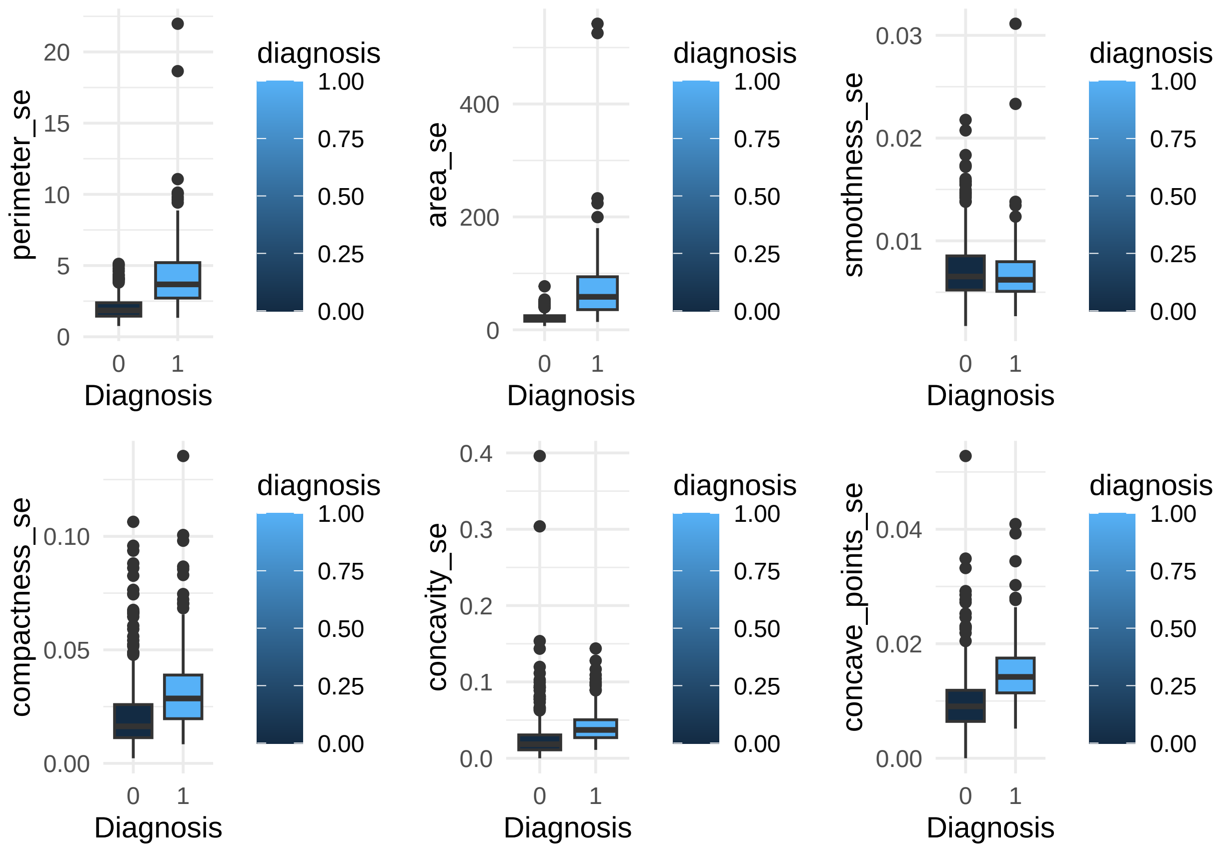
```
plots <- lapply(numeric_vars, function(var) {
  ggplot(data, aes(x = factor(diagnosis), y = .data[[var]], fill = diagnosis)) +
    geom_boxplot() +
    labs(x = "Diagnosis", y = var) +
    theme_minimal()
})

# Print all plots
# install.packages("gridExtra")
library(gridExtra)
```

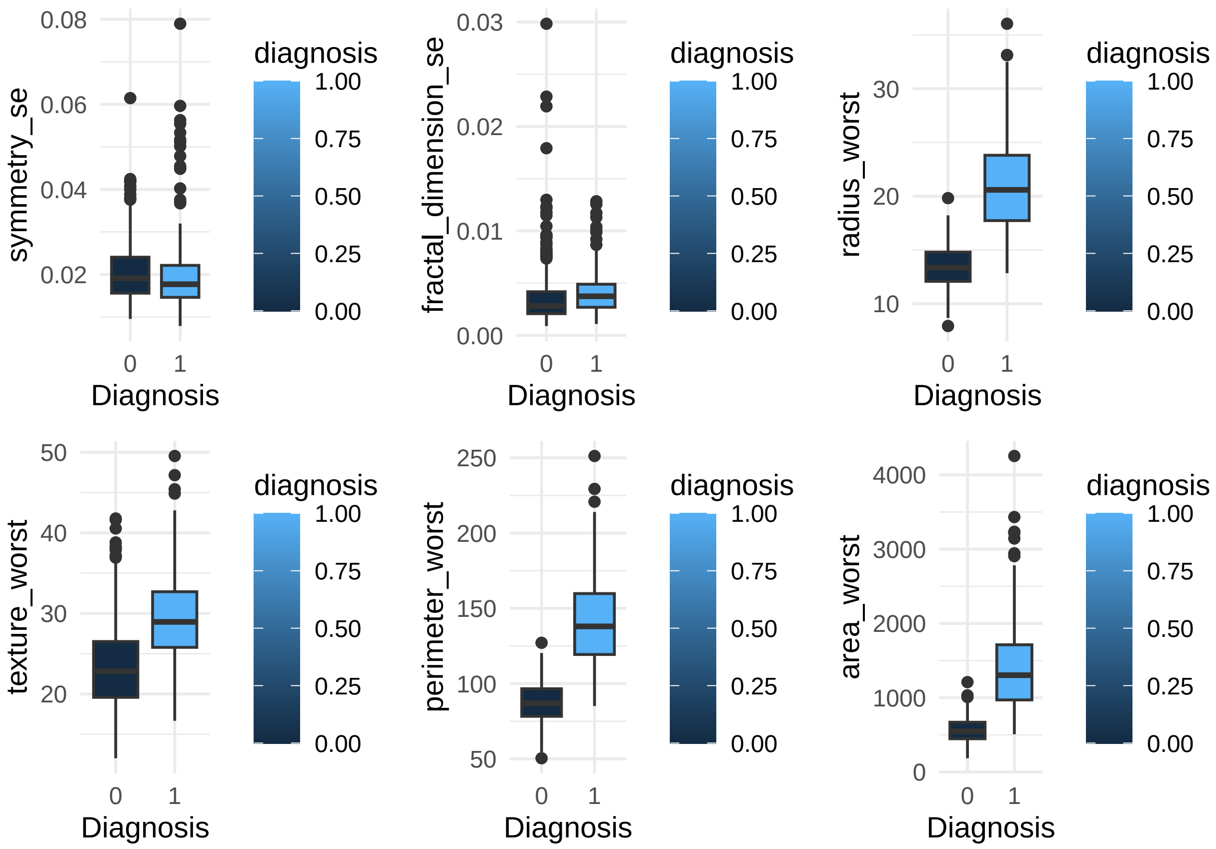
```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

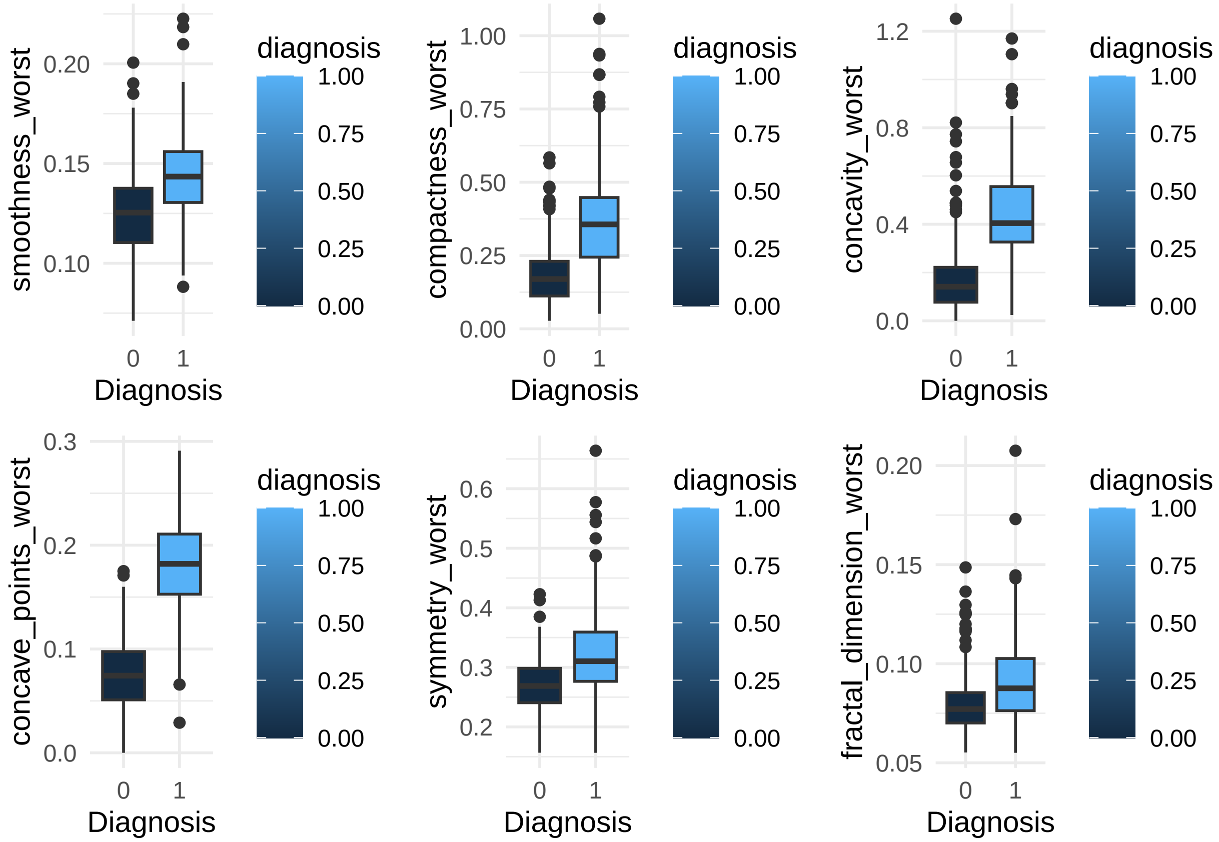
```
# Display plots in batches of 6 (2 rows × 3 columns)
num_plots <- length(plots)
batch_size <- 6

for(i in seq(1, num_plots, batch_size)) {
  end_idx <- min(i + batch_size - 1, num_plots)
  batch_plots <- plots[i:end_idx]
  grid.arrange(grobs = batch_plots, ncol = 3)
}
```




concavity_mean
0.0
0.1
0.2
0.3
0.4
0
1
Diagnosis
diagnosis
1.00
0.75
0.50
0.25
0.00
concave_points_mean
0.00
0.05
0.10
0.15
0.20
0
1
Diagnosis
diagnosis
1.00
0.75
0.50
0.25
0.00
symmetry_mean
0.10
0.15
0.20
0.25
0.30
0
1
Diagnosis
diagnosis
1.00
0.75
0.50
0.25
0.00
fractal_dimension_mean
0.05
0.06
0.07
0.08
0.09
0
1
Diagnosis
diagnosis
1.00
0.75
0.50
0.25
0.00
radius_se
0
1
2
3
0
1
Diagnosis
diagnosis
1.00
0.75
0.50
0.25
0.00
texture_se
1
2
3
4
5
0
1
Diagnosis
diagnosis
1.00
0.75
0.50
0.25
0.00


perimeter_se
0
5
10
15
20
0
1
Diagnosis
diagnosis
1.00
0.75
0.50
0.25
0.00
area_se
0
200
400
0
1
Diagnosis
diagnosis
1.00
0.75
0.50
0.25
0.00
smoothness_se
0.01
0.02
0.03
0
1
Diagnosis
diagnosis
1.00
0.75
0.50
0.25
0.00
compactness_se
0.00
0.05
0.10
0
1
Diagnosis
diagnosis
1.00
0.75
0.50
0.25
0.00
concavity_se
0.0
0.1
0.2
0.3
0.4
0
1
Diagnosis
diagnosis
1.00
0.75
0.50
0.25
0.00
concave_points_se
0.00
0.02
0.04
0
1
Diagnosis
diagnosis
1.00
0.75
0.50
0.25
0.00


symmetry_se
0.08
0.06
0.04
0.02
Diagnosis
0
1
diagnosis
1.00
0.75
0.50
0.25
0.00
fractal_dimension_se
0.03
0.02
0.01
0.00
Diagnosis
0
1
diagnosis
1.00
0.75
0.50
0.25
0.00
radius_worst
30
20
10
Diagnosis
0
1
diagnosis
1.00
0.75
0.50
0.25
0.00
texture_worst
50
40
30
20
Diagnosis
0
1
diagnosis
1.00
0.75
0.50
0.25
0.00
perimeter_worst
250
200
150
100
50
Diagnosis
0
1
diagnosis
1.00
0.75
0.50
0.25
0.00
area_worst
4000
3000
2000
1000
0
Diagnosis
0
1
diagnosis
1.00
0.75
0.50
0.25
0.00

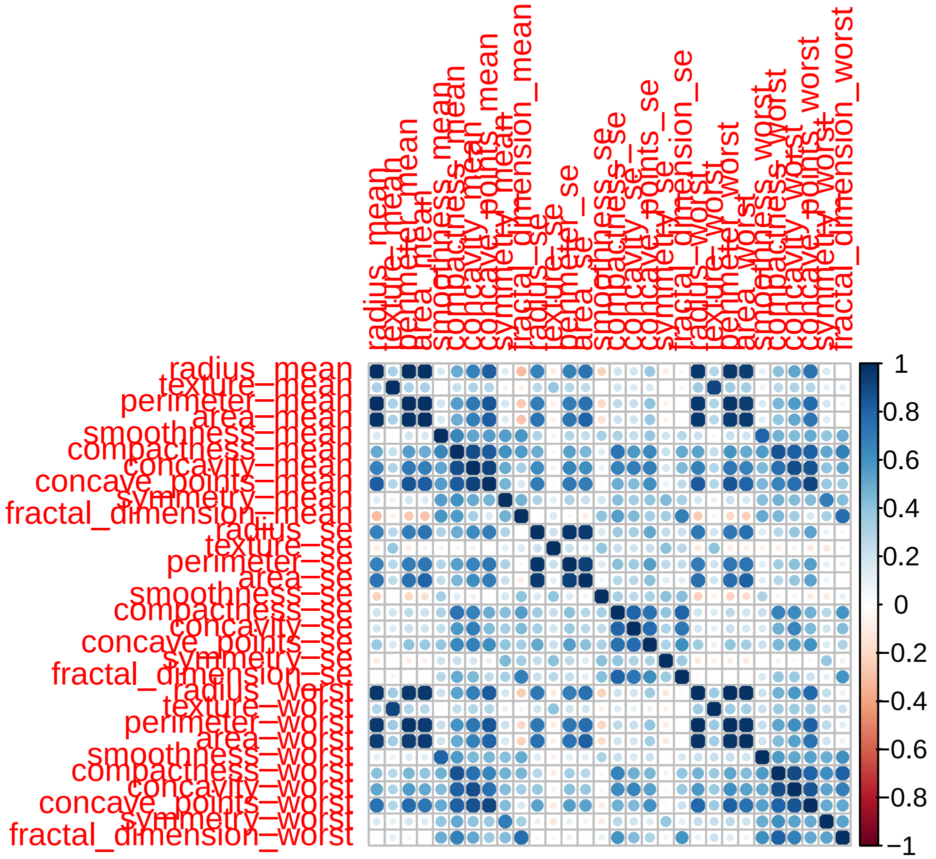


```
summary_stat <- data %>%
  group_by(factor(diagnosis)) %>%
  summarise(across(all_of(numeric_vars),
                   list(
                     mean = ~mean(.x, na.rm = TRUE),
                     sd = ~sd(.x, na.rm = TRUE),
                     median = ~median(.x, na.rm = TRUE),
                     min = ~min(.x, na.rm = TRUE),
                     max = ~max(.x, na.rm = TRUE)
                   )
            ))


# For easier viewing, you can pivot longer
summary_long <- summary_stat %>%
  pivot_longer(cols = -`factor(diagnosis)`,
               names_to = c("variable", "stat"),
               names_pattern = "(.*)_(.*)")

summary_long
```

```
## # A tibble: 300 x 4
##    `factor(diagnosis)` variable    stat  value
##    <fct>               <chr>       <chr> <dbl>
##  1 0                   radius_mean mean  12.1
##  2 0                   radius_mean sd     1.78
##  3 0                     radius_mean  median 12.2
##  4 0                     radius_mean  min     6.98
##  5 0                     radius_mean  max    17.8
##  6 0                     texture_mean mean   17.9
##  7 0                     texture_mean sd      4.00
##  8 0                     texture_mean median 17.4
##  9 0                     texture_mean min     9.71
## 10 0                     texture_mean max    33.8
## # i 290 more rows
```

Correlation

```r
# Check correlation between numeric variables
cor_matrix <- cor(data[, numeric_vars])
corrplot(cor_matrix, method = "circle")
```



```r
# Or find highly correlated variables
high_cor <- findCorrelation(cor_matrix, cutoff = 0.8)
problematic_vars <- numeric_vars[high_cor]
print(problematic_vars)
```

```
##  [1] "concavity_mean"       "concave_points_mean"  "compactness_mean"
##  [4] "concave_points_worst" "concavity_worst"      "perimeter_worst"
##  [7] "radius_worst"         "perimeter_mean"       "compactness_worst"
## [10] "area_worst"           "radius_mean"          "perimeter_se"
## [13] "compactness_se"       "area_se"              "smoothness_mean"
## [16] "texture_mean"
```

Variable Selection

Frequentist Approach

Check VIF and remove variables with extremly high values

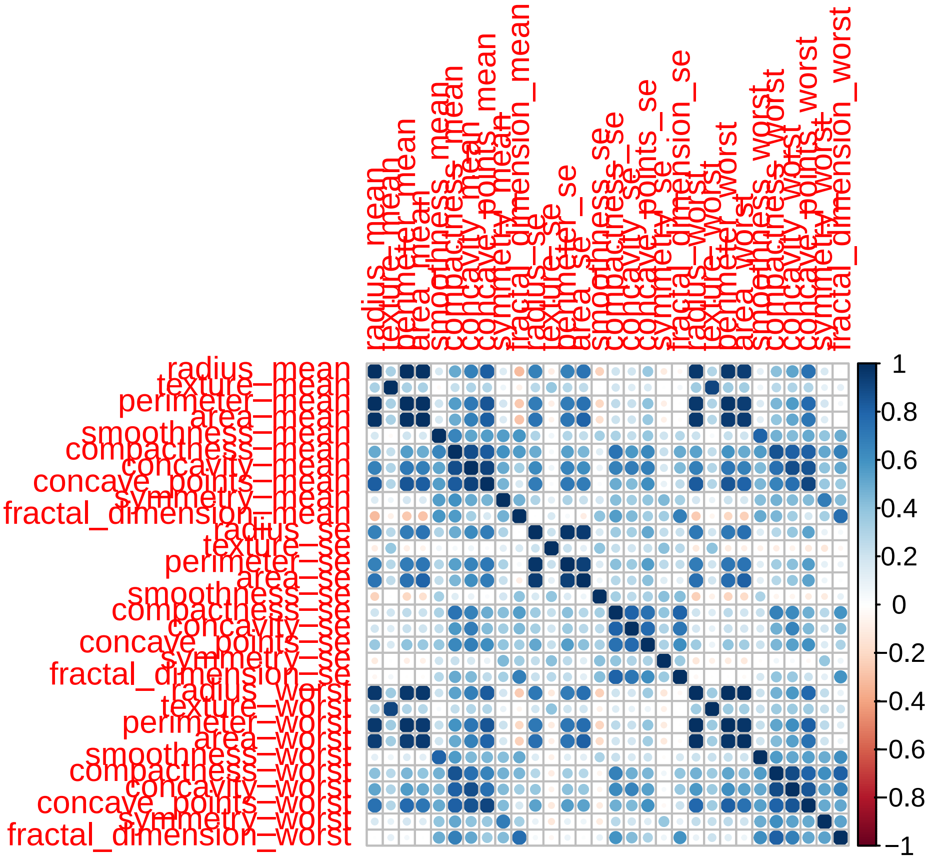
```
predictors <- setdiff(names(data), c("diagnosis"))
formula_str <- paste("diagnosis ~", paste(predictors, collapse = " + "))
formula <- as.formula(formula_str)

l_reg = lm(formula, data)
vif_values <- vif(l_reg)

vif_df <- data.frame(
  Variable = names(vif_values),
  VIF = vif_values
)
vif_df <- vif_df %>% arrange(desc(VIF))
print(vif_df)
```

```
##                                        Variable         VIF
## radius_mean                         radius_mean 3806.115296
## perimeter_mean                   perimeter_mean 3786.400419
## radius_worst                       radius_worst  799.105946
## perimeter_worst                 perimeter_worst  405.023336
## area_mean                             area_mean  347.878657
## area_worst                           area_worst  337.221924
## radius_se                             radius_se   75.462027
## concavity_mean                   concavity_mean   70.767720
## perimeter_se                       perimeter_se   70.359695
## concave_points_mean         concave_points_mean   60.041733
## compactness_mean               compactness_mean   50.505168
## area_se                                 area_se   41.163091
## compactness_worst             compactness_worst   36.982755
## concave_points_worst       concave_points_worst   36.763714
## concavity_worst                 concavity_worst   31.970723
## fractal_dimension_worst fractal_dimension_worst   18.861533
## texture_worst                     texture_worst   18.569966
## fractal_dimension_mean   fractal_dimension_mean   15.756977
## concavity_se                       concavity_se   15.694833
## compactness_se                   compactness_se   15.366324
## texture_mean                       texture_mean   11.884048
## concave_points_se             concave_points_se   11.520796
## smoothness_worst               smoothness_worst   10.923061
## fractal_dimension_se       fractal_dimension_se    9.717987
## symmetry_worst                   symmetry_worst    9.520570
## smoothness_mean                 smoothness_mean    8.194282
## symmetry_se                         symmetry_se    5.175426
## symmetry_mean                     symmetry_mean    4.220656
## texture_se                           texture_se    4.205423
## smoothness_se                     smoothness_se    4.027923
```

```
vars_to_exclude <- c(head(vif_df,15)$Variable)
# setdiff(problematic_vars , vars_to_exclude)
# setdiff(vars_to_exclude, problematic_vars)
```

Check correlations after excluding x VIF, variables to pay attention if something does not work.

```r
# Check correlation between numeric variables
cor_matrix_f <- cor(data[, setdiff(numeric_vars, vars_to_exclude)])
corrplot(cor_matrix, method = "circle")
```



```r
# Or find highly correlated variables
high_cor_f <- findCorrelation(cor_matrix_f, cutoff = 0.8)
problematic_vars_f <- setdiff(numeric_vars, vars_to_exclude)[high_cor_f]
print(problematic_vars_f)
```

```
## [1] "compactness_se"  "smoothness_mean" "texture_mean"
```

```r
selected_freq <- setdiff(numeric_vars, vars_to_exclude)
```

Bayesian Approach

```r
library(BAS)

# Fit a Bayesian logistic regression with variable selection
model_bas <- bas.glm(diagnosis ~ .,
                     data = data,
                     family = binomial(),
                     method = "MCMC", # or "BAS" for deterministic sampling
                     MCMC.iterations = 10000,
                     modelprior = uniform()) # Prior over model space
```

```
# Summary of results
summary(model_bas)
```

```
##                         P(B != 0 | Y)      model 1   model 2      model 3
## Intercept                      1.0000   1.00000000   1.00000   1.00000000
## radius_mean                    0.3486   0.00000000   0.00000   1.00000000
## texture_mean                   0.2364   0.00000000   0.00000   0.00000000
## perimeter_mean                 0.2586   0.00000000   0.00000   0.00000000
## area_mean                      0.2374   0.00000000   1.00000   0.00000000
## smoothness_mean                0.2168   0.00000000   0.00000   0.00000000
## compactness_mean               0.4635   0.00000000   1.00000   0.00000000
## concavity_mean                 0.3506   0.00000000   0.00000   1.00000000
## concave_points_mean            0.5527   0.00000000   1.00000   0.00000000
## symmetry_mean                  0.1601   0.00000000   0.00000   0.00000000
## fractal_dimension_mean         0.1654   0.00000000   0.00000   0.00000000
## radius_se                      0.4634   1.00000000   0.00000   0.00000000
## texture_se                     0.3690   0.00000000   0.00000   0.00000000
## perimeter_se                   0.2660   0.00000000   0.00000   0.00000000
## area_se                        0.5434   0.00000000   1.00000   1.00000000
## smoothness_se                  0.3558   1.00000000   0.00000   0.00000000
## compactness_se                 0.4479   0.00000000   0.00000   0.00000000
## concavity_se                   0.2973   0.00000000   0.00000   0.00000000
## concave_points_se              0.4684   0.00000000   1.00000   1.00000000
## symmetry_se                    0.2091   0.00000000   0.00000   0.00000000
## fractal_dimension_se           0.5357   1.00000000   1.00000   1.00000000
## radius_worst                   0.2816   0.00000000   0.00000   1.00000000
## texture_worst                  0.8778   1.00000000   1.00000   1.00000000
## perimeter_worst                0.2354   0.00000000   0.00000   0.00000000
## area_worst                     0.6316   1.00000000   1.00000   0.00000000
## smoothness_worst               0.4006   0.00000000   0.00000   1.00000000
## compactness_worst              0.2426   1.00000000   0.00000   0.00000000
## concavity_worst                0.4138   1.00000000   0.00000   0.00000000
## concave_points_worst           0.4240   1.00000000   0.00000   0.00000000
## symmetry_worst                 0.5546   0.00000000   1.00000   0.00000000
## fractal_dimension_worst        0.4300   1.00000000   1.00000   1.00000000
## BF                                 NA   0.03514088   1.00000   0.06543644
## PostProbs                          NA   0.00500000   0.00460   0.00270000
## R2                                 NA   0.92380000   0.93790   0.92540000
## dim                                NA  10.00000000  11.00000  10.00000000
## logmarg                            NA -52.96576776 -49.61738 -52.34405369
##                              model 4     model 5
## Intercept                 1.00000000   1.0000000
## radius_mean               1.00000000   0.0000000
## texture_mean              0.00000000   1.0000000
## perimeter_mean            0.00000000   0.0000000
## area_mean                 0.00000000   0.0000000
## smoothness_mean           0.00000000   0.0000000
## compactness_mean          1.00000000   1.0000000
## concavity_mean            0.00000000   0.0000000
## concave_points_mean       1.00000000   1.0000000
## symmetry_mean             0.00000000   0.0000000
## fractal_dimension_mean    0.00000000   0.0000000
## radius_se                 0.00000000   0.0000000
## texture_se                0.00000000   0.0000000
## perimeter_se              1.00000000   0.0000000
## area_se                   1.00000000   1.0000000
## smoothness_se             0.00000000   0.0000000
## compactness_se            0.00000000   0.0000000
## concavity_se              0.00000000   0.0000000
## concave_points_se         1.00000000   0.0000000
## symmetry_se               0.00000000   1.0000000
## fractal_dimension_se      1.00000000   1.0000000
## radius_worst              0.00000000   1.0000000
## texture_worst             1.00000000   0.0000000
## perimeter_worst           1.00000000   0.0000000
## area_worst                0.00000000   0.0000000
## smoothness_worst          0.00000000   1.0000000
## compactness_worst         1.00000000   0.0000000
## concavity_worst           0.00000000   1.0000000
## concave_points_worst      0.00000000   0.0000000
## symmetry_worst            1.00000000   1.0000000
## fractal_dimension_worst   1.00000000   0.0000000
## BF                        0.06601369   0.0113394
## PostProbs                 0.00240000   0.0022000
## R2                        0.94130000   0.9263000
## dim                      13.00000000  11.0000000
## logmarg                 -52.33527083 -54.0968492
```

```
# Posterior inclusion probabilities
pip <- model_bas$probne0
variable_names <- names(pip)
#pip_df <- data.frame(Variable = numeric_vars,
#                     InclusionProb = pip)
#pip_df <- pip_df[order(pip_df$InclusionProb, decreasing = TRUE),]
#print(pip_df)

selected_bayes <- c( "perimeter_mean", "concave_points_mean", "compactness_mean",
                     "concavity_mean", "area_se", "smoothness_se", "concave_points_se",
                     "fractal_dimension_se","radius_worst", "texture_worst",
                     "fractal_dimension_worst")
```

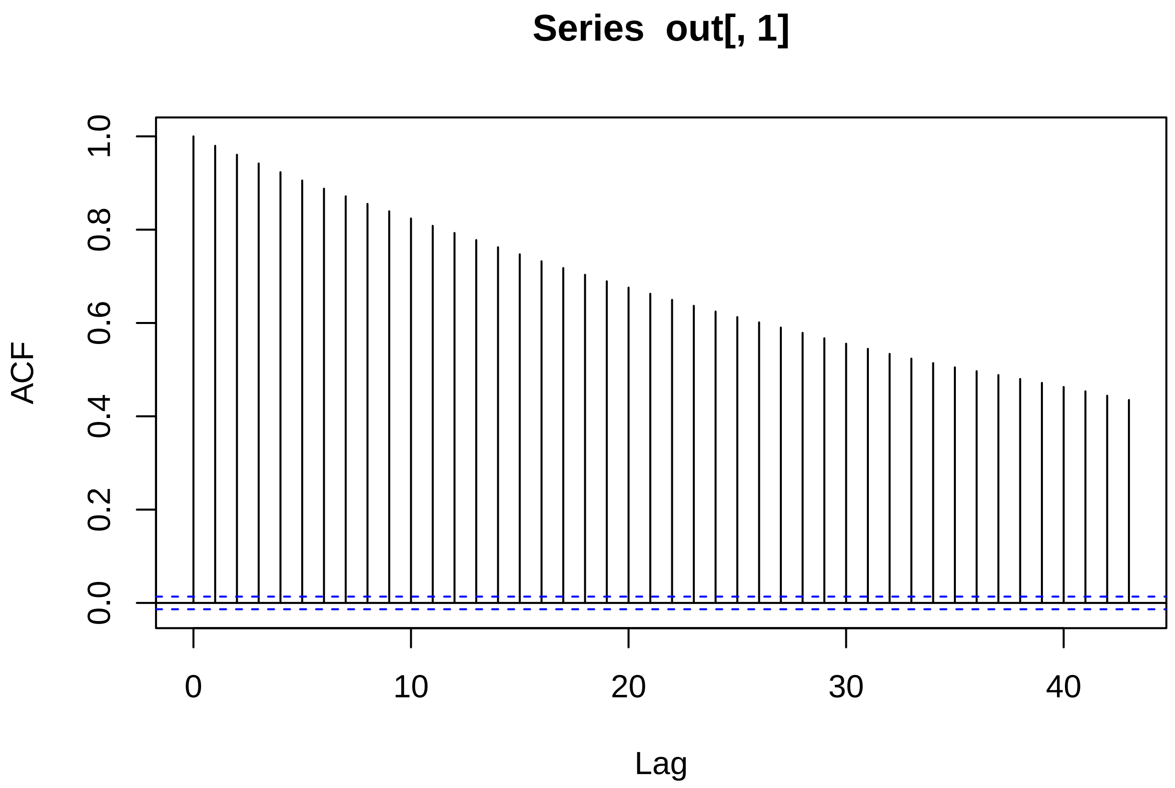
Logistic Models

Freq var selection

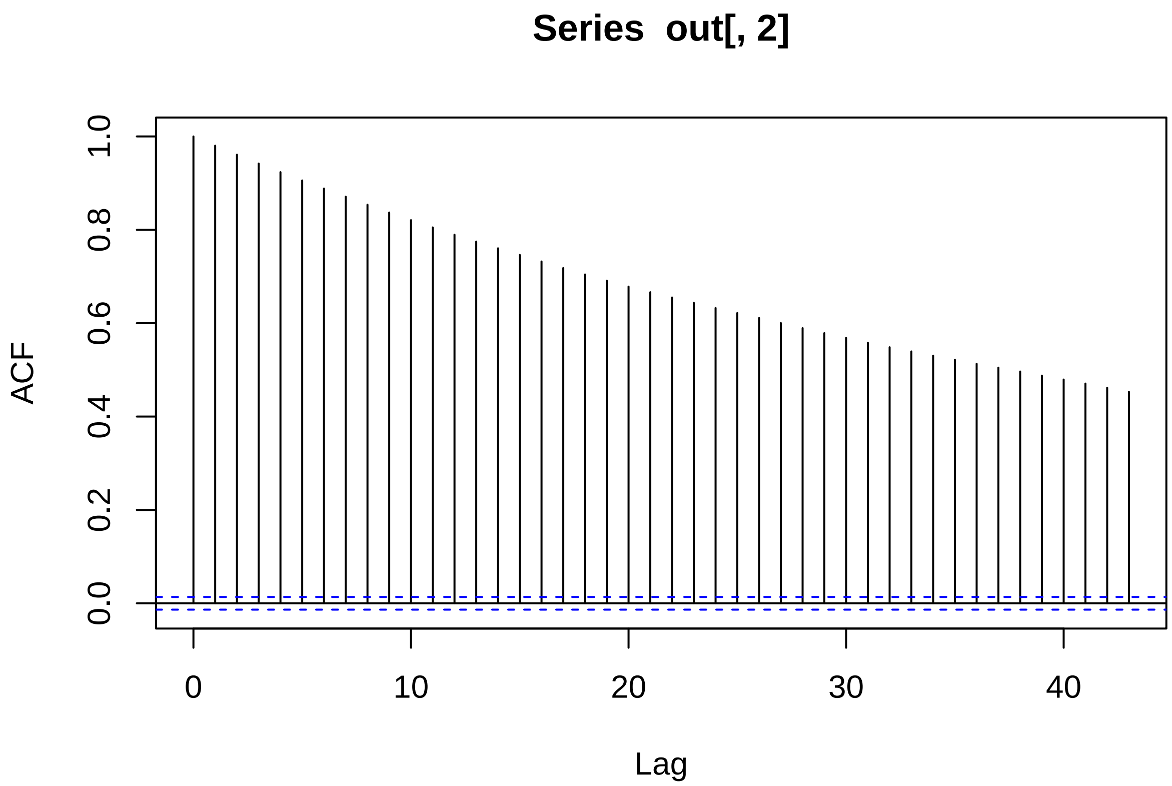
```
formula_str <- paste("diagnosis ~", paste(selected_freq, collapse = " + "))
formula <- as.formula(formula_str)

freq_model1<-lm(formula, data = data)
beta.start1 <- coef(freq_model1)

out = MCMClogit(formula, data, burnin=1000, mcmc=21000, beta.start = beta.start1)
summary(out)
```

```
##
## Iterations = 1001:22000
## Thinning interval = 1
## Number of chains = 1
## Sample size per chain = 21000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                              Mean       SD  Naive SE Time-series SE
## (Intercept)              -7.0448   2.7936 0.0192780       0.194235
## texture_mean              0.1402   0.1332 0.0009190       0.009217
## smoothness_mean         212.5324  39.8023 0.2746618       2.930954
## symmetry_mean             6.1214  13.6580 0.0942492       1.029750
## fractal_dimension_mean -692.6680  86.5229 0.5970649       6.070409
## texture_se               -1.4180   0.7611 0.0052523       0.051882
## smoothness_se          -244.3942 113.0283 0.7799695       7.684728
## compactness_se           22.7986  27.5534 0.1901367       2.002600
## concavity_se             -1.5990  12.5088 0.0863193       0.973608
## concave_points_se       419.3653  67.1342 0.4632698       4.945145
## symmetry_se               0.1725  46.5293 0.3210826       3.015775
## fractal_dimension_se   -254.7769 237.3704 1.6380116      16.019617
## texture_worst             0.2644   0.1237 0.0008536       0.008767
## smoothness_worst         -0.7513  26.7957 0.1849080       1.771922
## symmetry_worst           12.2862   8.1680 0.0563646       0.596335
## fractal_dimension_worst  155.0970  37.1387 0.2562813       2.341975
##
## 2. Quantiles for each variable:
##
##                               2.5%        25%       50%       75%      97.5%
## (Intercept)              -12.45556   -8.79360   -7.0801   -5.1257   -1.54107
## texture_mean              -0.12287    0.04874    0.1382    0.2356    0.40810
## smoothness_mean          133.04711  188.72514  208.3203  240.4691  292.78315
## symmetry_mean            -20.78431   -2.58098    6.6107   15.3513   33.61856
## fractal_dimension_mean  -865.96422 -744.19734 -688.0979 -640.5828 -522.87375
## texture_se                -2.98513   -1.91367   -1.4060   -0.9046   -0.02046
## smoothness_se           -451.60599 -315.71718 -248.1793 -175.6460  -15.88348
## compactness_se           -30.94268    3.70100   23.7214   42.2483   72.24119
## concavity_se             -25.05991  -10.98320   -2.0636    7.6320   23.35489
## concave_points_se        285.73948  381.17421  414.5063  459.1587  565.12707
## symmetry_se              -90.66652  -27.98892    0.3550   29.4079   98.93149
## fractal_dimension_se    -735.58558 -423.48834 -261.8735  -89.4792  204.34667
## texture_worst              0.02234    0.18775    0.2620    0.3426    0.52044
## smoothness_worst         -55.15123  -18.34344    1.8441   17.3100   53.51026
## symmetry_worst            -2.53864    7.09859   11.6060   17.1774   30.02571
## fractal_dimension_worst   81.74641  131.53188  155.7603  178.0896  233.41108
```

```
acf(out[,1])
```



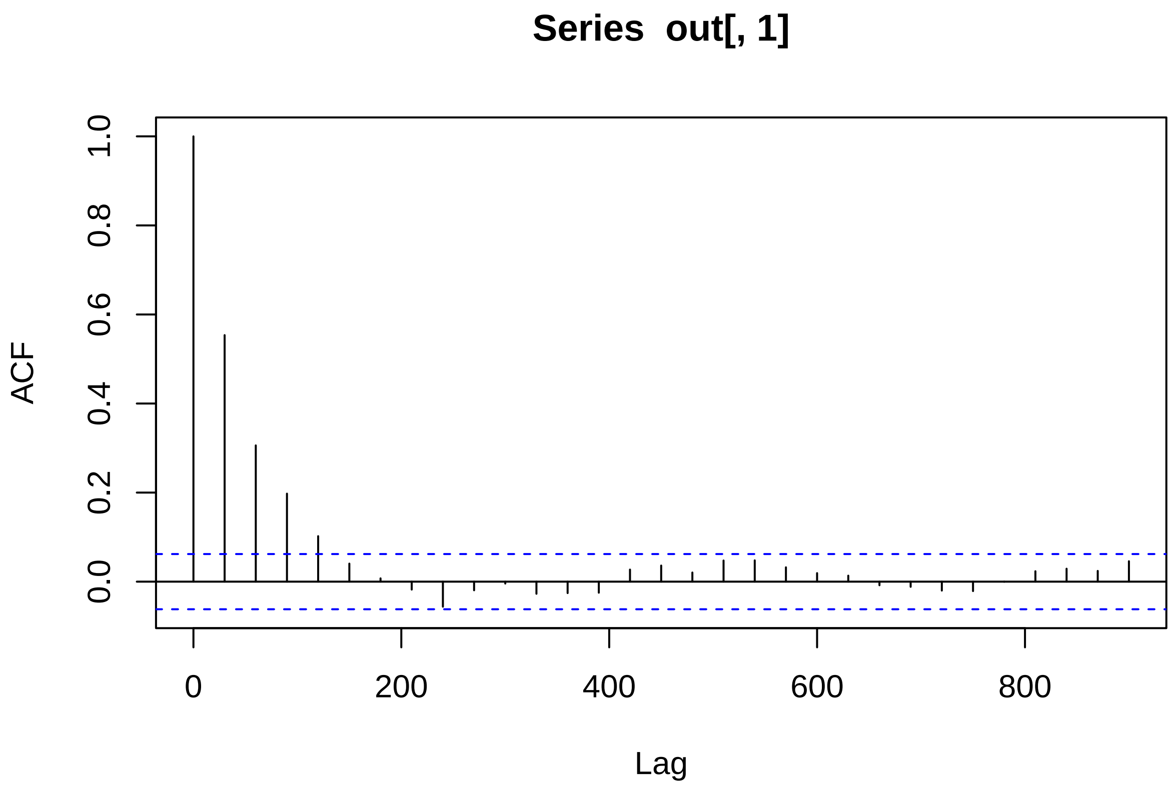
```
acf(out[,2])
```



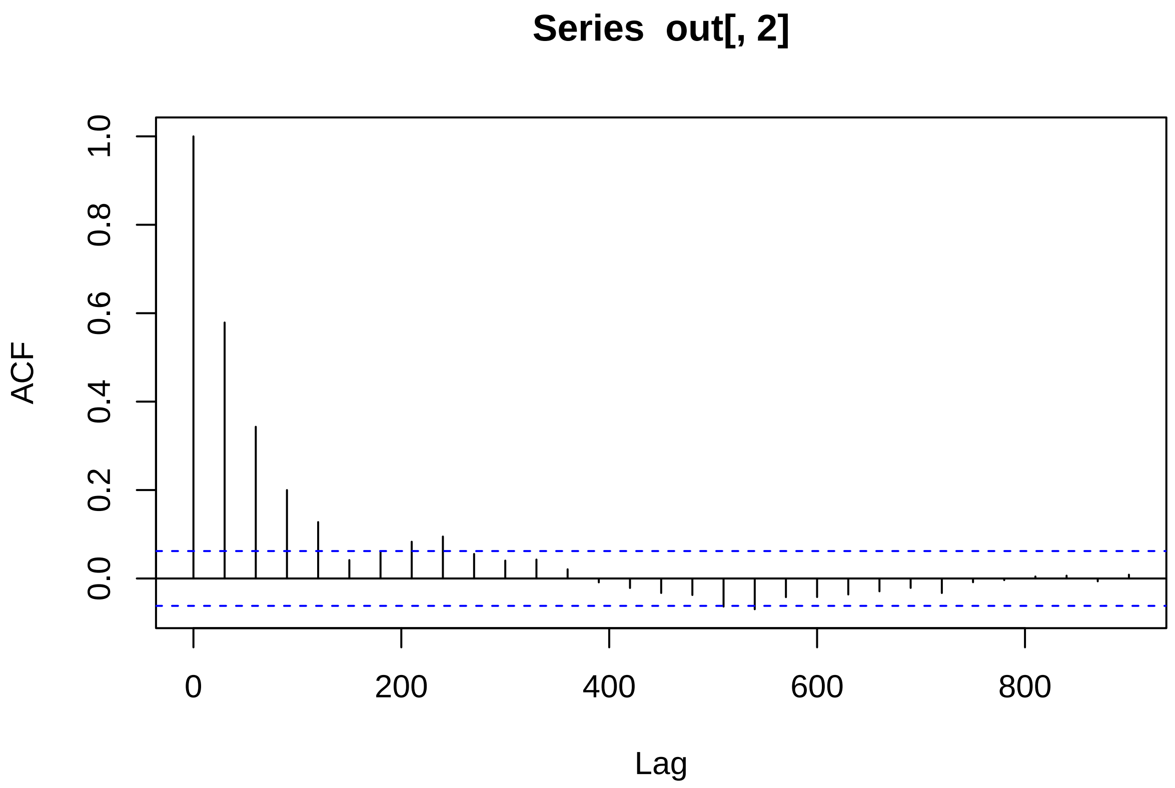
```
# Correct autocorrelation

out = MCMClogit(formula, data, burnin=5000, mcmc=30000, thin = 30,
                beta.start = beta.start1)
summary(out)
```

```
##
## Iterations = 5001:34971
## Thinning interval = 30
## Number of chains = 1
## 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
## (Intercept)               -6.9619   2.9564 0.093488       0.174505
## texture_mean               0.1449   0.1338 0.004230       0.008603
## smoothness_mean          210.3856  38.8092 1.227254       2.364636
## symmetry_mean              6.7972  13.7695 0.435430       0.874829
## fractal_dimension_mean  -692.9451  85.9090 2.716683       5.109411
## texture_se                -1.4312   0.7698 0.024345       0.043598
## smoothness_se           -245.3894 120.4758 3.809779       6.431210
## compactness_se            20.8806  28.9179 0.914463       1.881365
## concavity_se              -2.1651  11.8429 0.374504       0.841066
## concave_points_se        419.8216  68.7222 2.173187       4.788362
## symmetry_se                2.8399  51.6192 1.632341       3.064199
## fractal_dimension_se    -235.3213 230.1363 7.277549      12.459032
## texture_worst              0.2585   0.1235 0.003904       0.008201
## smoothness_worst           0.4922  26.7191 0.844933       1.548509
## symmetry_worst            12.2236   8.5569 0.270594       0.595713
## fractal_dimension_worst  153.6350  38.6341 1.221717       2.154905
##
## 2. Quantiles for each variable:
##
##                               2.5%        25%       50%       75%      97.5%
## (Intercept)              -1.257e+01   -8.82359   -7.1256   -4.9263   -1.06303
## texture_mean             -1.264e-01    0.05609    0.1493    0.2347    0.42288
## smoothness_mean           1.346e+02  185.56985  207.4429  237.4691  285.25906
## symmetry_mean            -2.133e+01   -2.07081    7.2447   16.8415   33.46638
## fractal_dimension_mean   -8.685e+02 -748.06764 -687.5111 -638.0705 -535.08845
## texture_se               -3.011e+00   -1.90562   -1.3896   -0.9116   -0.01158
## smoothness_se            -4.695e+02 -329.76049 -253.4453 -166.7919    2.89369
## compactness_se           -3.356e+01    1.54953   21.3018   40.3877   76.28100
## concavity_se             -2.669e+01  -10.76965   -2.0245    6.0901   20.70983
## concave_points_se         2.861e+02  374.51992  416.7823  463.6432  566.28489
## symmetry_se              -1.014e+02  -29.30766    2.0131   37.1225   99.63610
## fractal_dimension_se     -7.050e+02 -385.97951 -235.5333  -66.6496  194.69516
## texture_worst             2.987e-03    0.17517    0.2570    0.3330    0.52035
## smoothness_worst         -5.199e+01  -17.05354   -0.2086   17.8640   54.16787
## symmetry_worst           -2.732e+00    6.80055   11.5672   17.4516   30.08029
## fractal_dimension_worst   7.957e+01  126.52236  155.9399  178.0914  231.92666
```

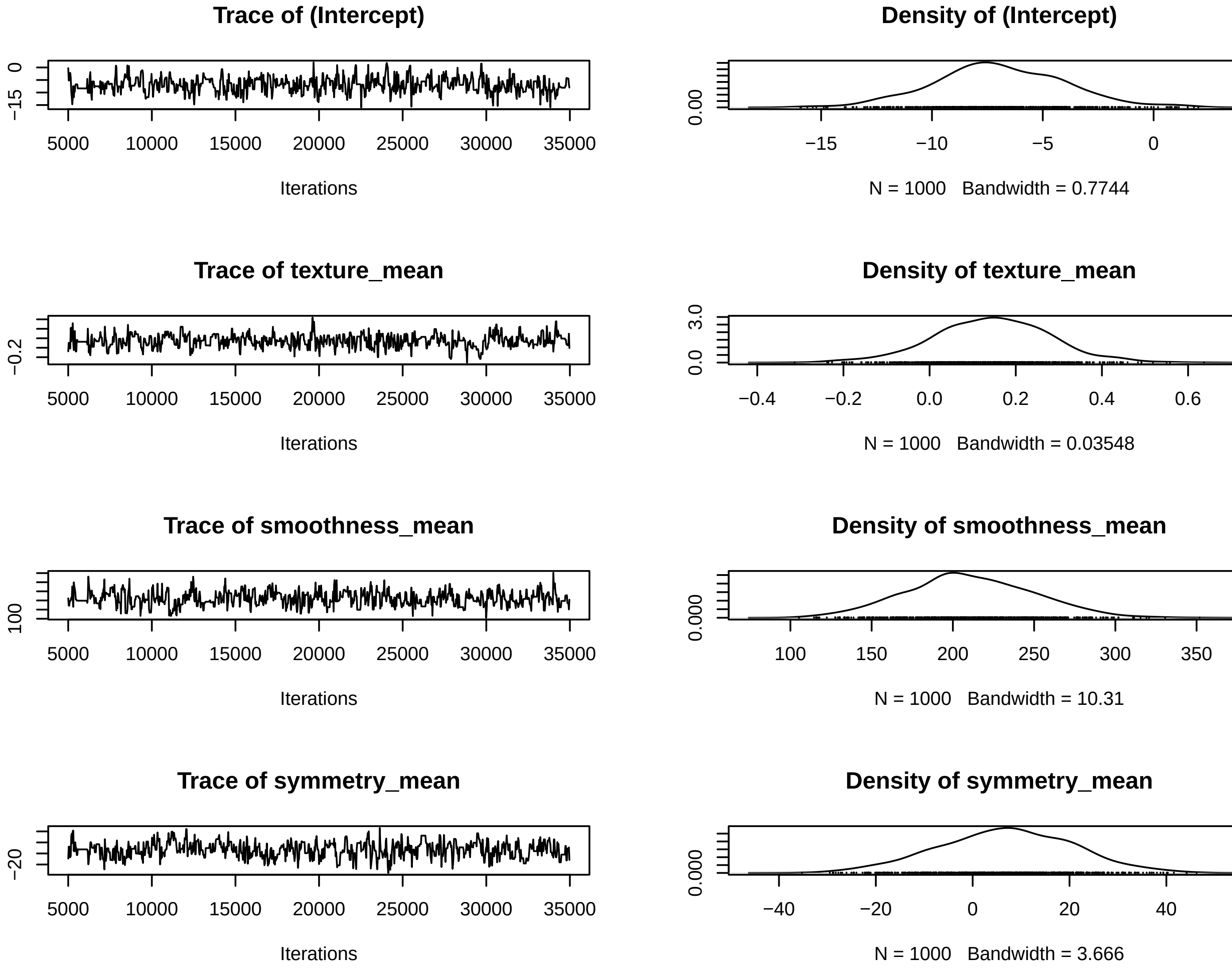
```
acf(out[,1])
```

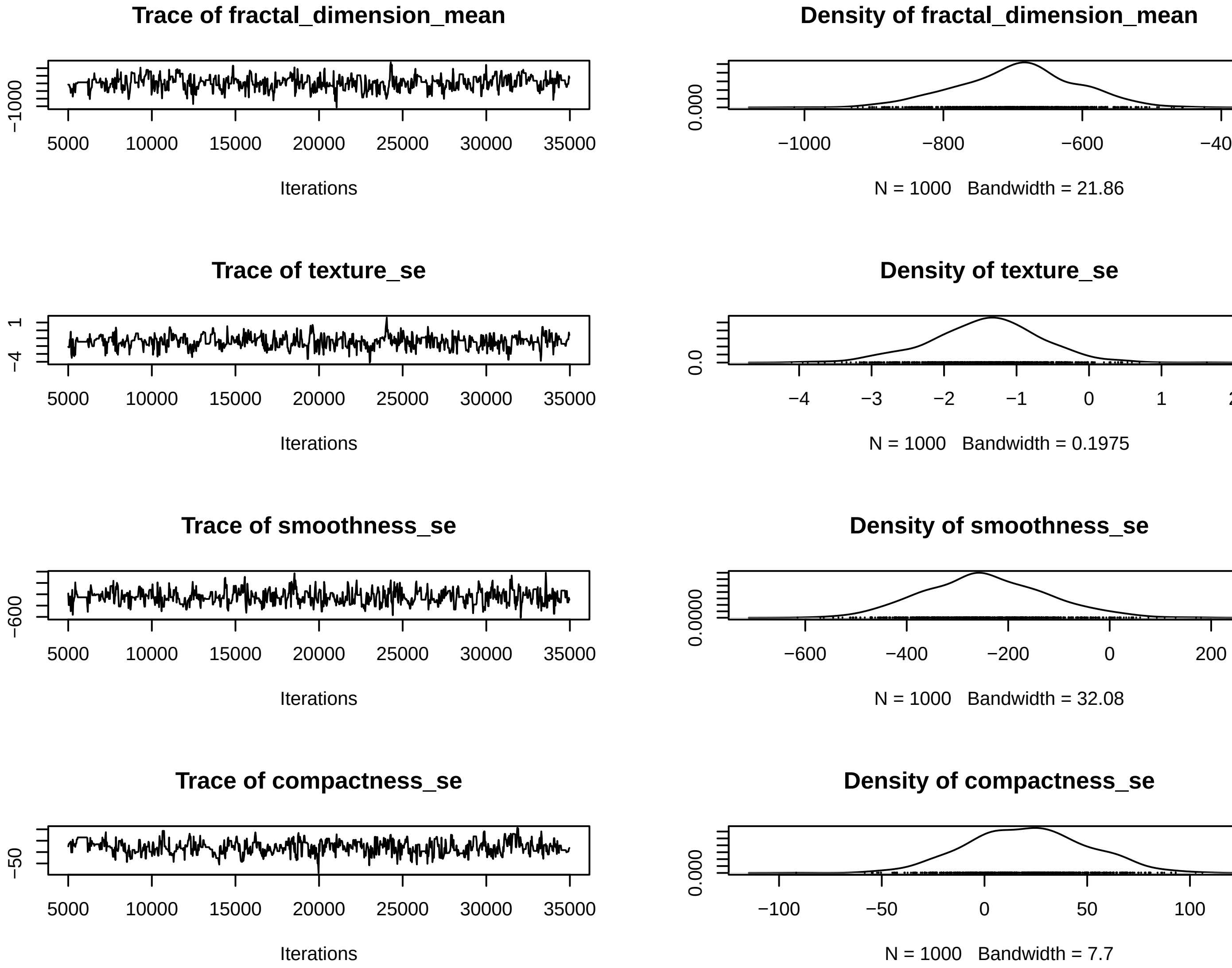


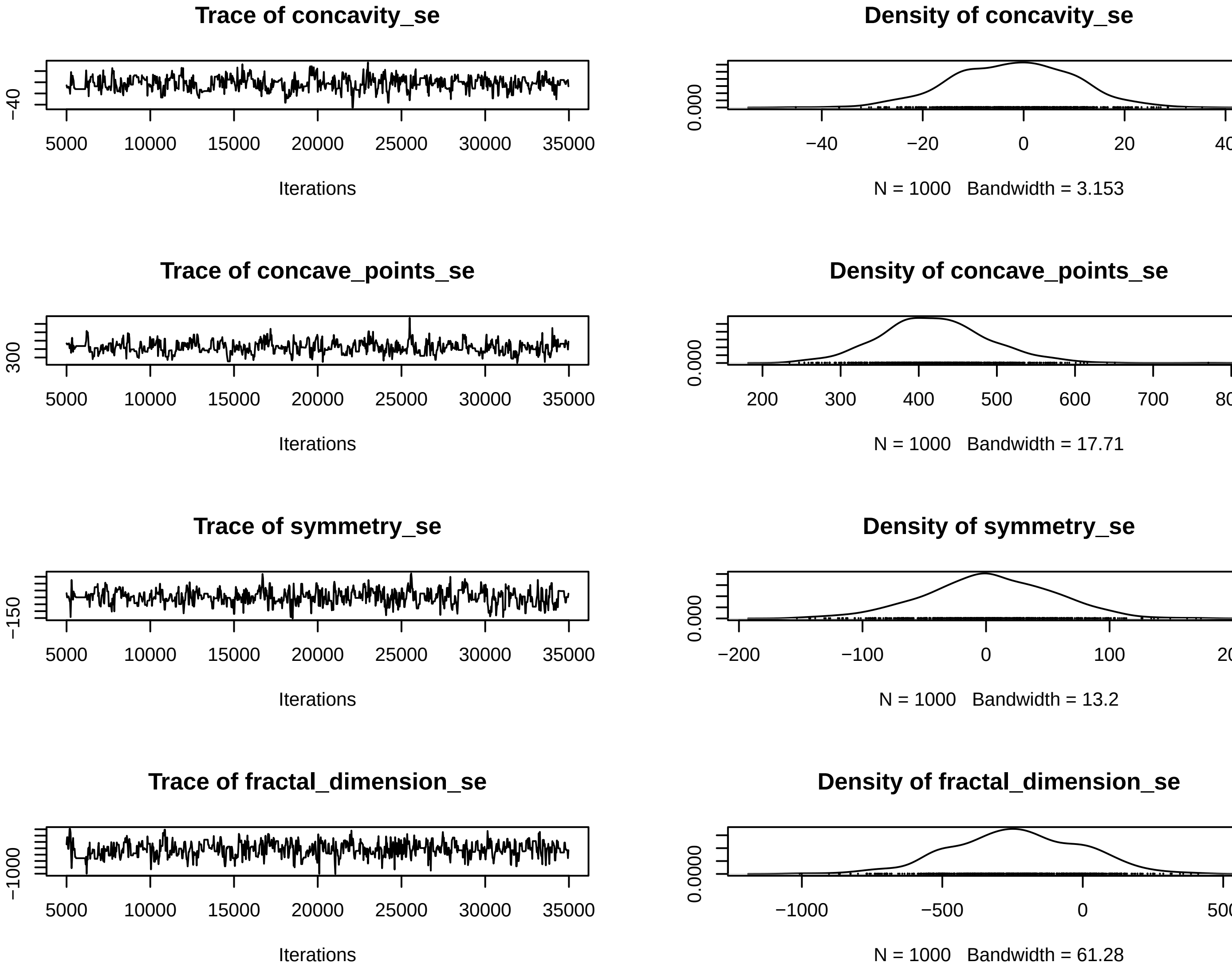
```
acf(out[,2])
```

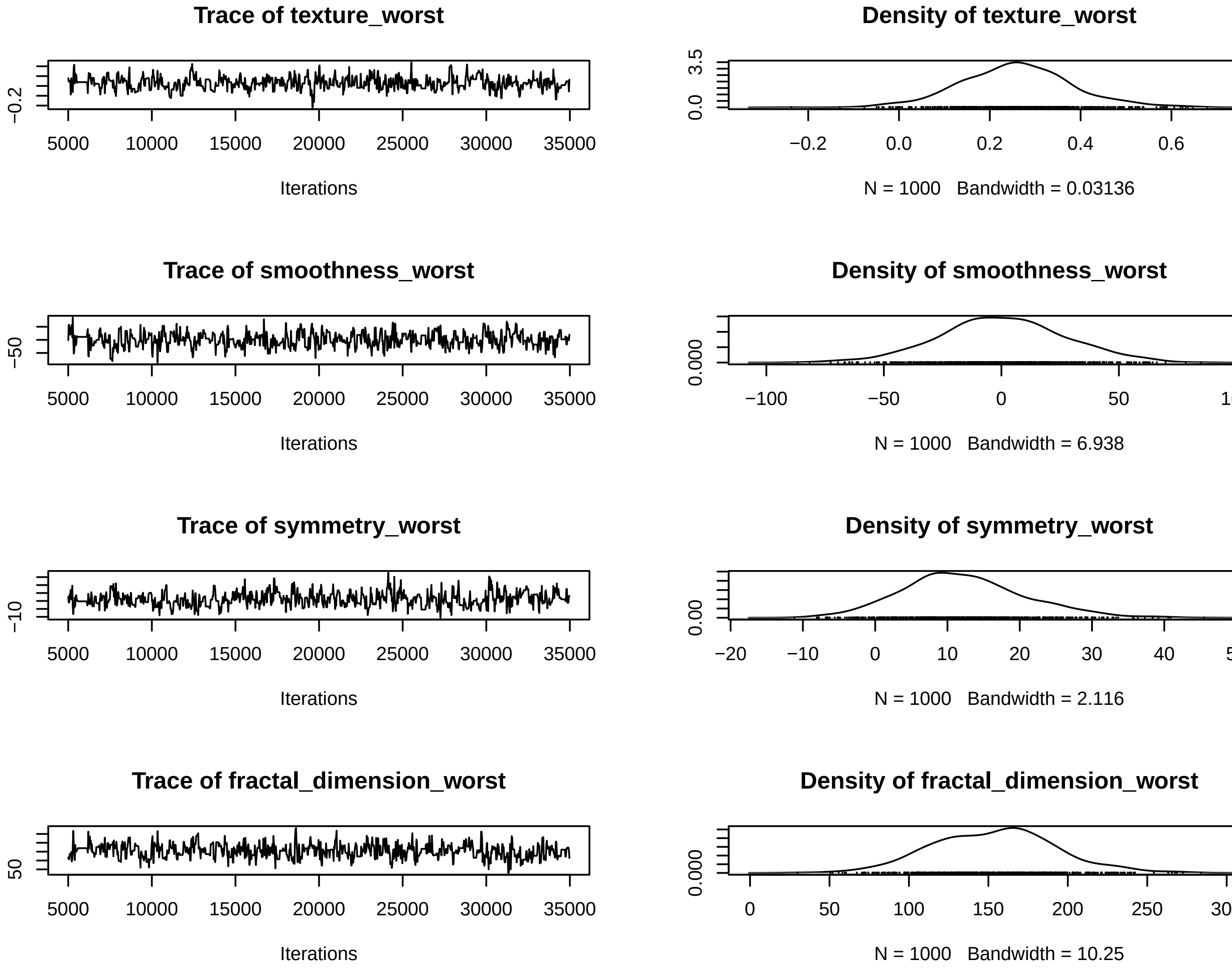


```r
plot(out)
```


Trace of (Intercept)
0
−15
5000
10000
15000
20000
25000
30000
35000
Iterations
Density of (Intercept)
0.00
−15
−10
−5
0
N = 1000 Bandwidth = 0.7744
Trace of texture_mean
−0.2
5000
10000
15000
20000
25000
30000
35000
Iterations
Density of texture_mean
3.0
0.0
−0.4
−0.2
0.0
0.2
0.4
0.6
N = 1000 Bandwidth = 0.03548
Trace of smoothness_mean
100
5000
10000
15000
20000
25000
30000
35000
Iterations
Density of smoothness_mean
0.000
100
150
200
250
300
350
N = 1000 Bandwidth = 10.31
Trace of symmetry_mean
−20
5000
10000
15000
20000
25000
30000
35000
Iterations
Density of symmetry_mean
0.000
−40
−20
0
20
40
N = 1000 Bandwidth = 3.666


Trace of fractal_dimension_mean
−1000
5000
10000
15000
20000
25000
30000
35000
Iterations
Density of fractal_dimension_mean
0.000
−1000
−800
−600
−40
N = 1000 Bandwidth = 21.86
Trace of texture_se
1
−4
5000
10000
15000
20000
25000
30000
35000
Iterations
Density of texture_se
0.0
−4
−3
−2
−1
0
1
2
N = 1000 Bandwidth = 0.1975
Trace of smoothness_se
−600
5000
10000
15000
20000
25000
30000
35000
Iterations
Density of smoothness_se
0.0000
−600
−400
−200
0
200
N = 1000 Bandwidth = 32.08
Trace of compactness_se
−50
5000
10000
15000
20000
25000
30000
35000
Iterations
Density of compactness_se
0.000
−100
−50
0
50
100
N = 1000 Bandwidth = 7.7


Trace of concavity_se
−40
5000
10000
15000
20000
25000
30000
35000
Iterations
Density of concavity_se
0.000
−40
−20
0
20
40
N = 1000 Bandwidth = 3.153
Trace of concave_points_se
300
5000
10000
15000
20000
25000
30000
35000
Iterations
Density of concave_points_se
0.000
200
300
400
500
600
700
80
N = 1000 Bandwidth = 17.71
Trace of symmetry_se
−150
5000
10000
15000
20000
25000
30000
35000
Iterations
Density of symmetry_se
0.000
−200
−100
0
100
20
N = 1000 Bandwidth = 13.2
Trace of fractal_dimension_se
−1000
5000
10000
15000
20000
25000
30000
35000
Iterations
Density of fractal_dimension_se
0.0000
−1000
−500
0
500
N = 1000 Bandwidth = 61.28



Bayes var selection

```
formula_str_b <- paste("diagnosis ~", paste(selected_bayes, collapse = " + "))
formula_b <- as.formula(formula_str_b)

# starting point
freq_model<-lm(formula_b, data = data)
beta.start <- coef(freq_model)

out_b = MCMClogit(formula_b, data, burnin=1000, mcmc=21000)
```

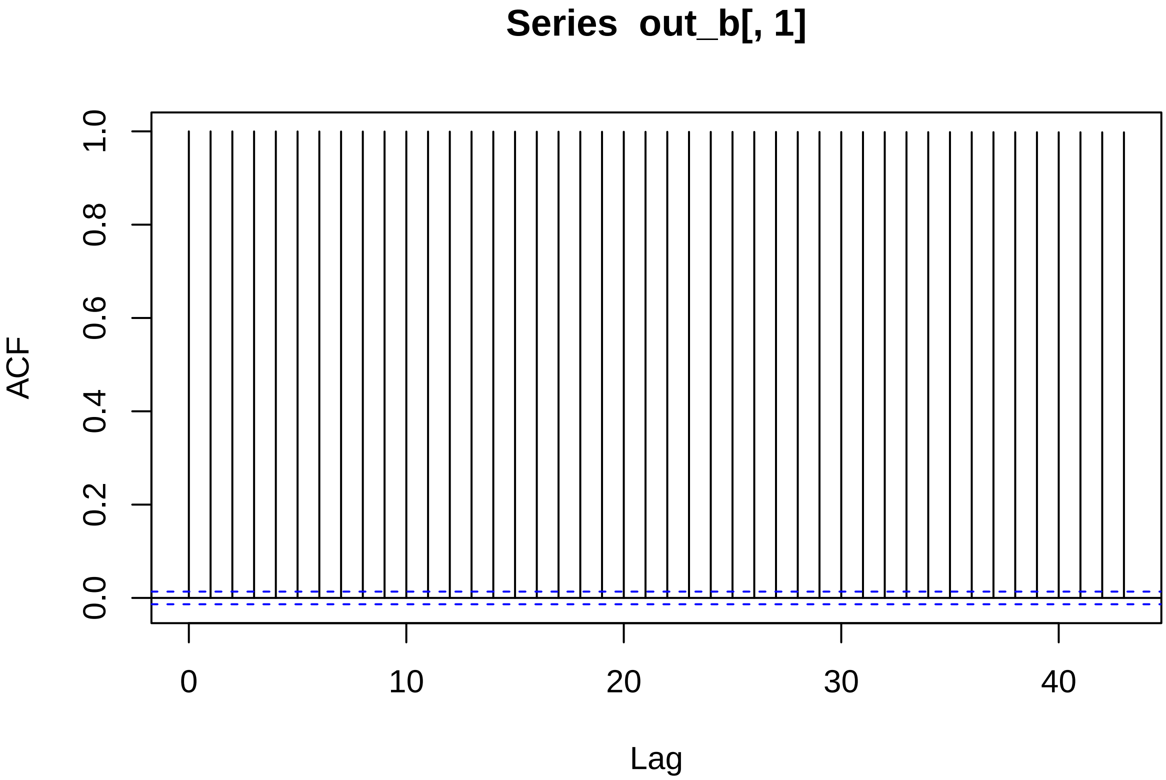
```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(out_b)
```

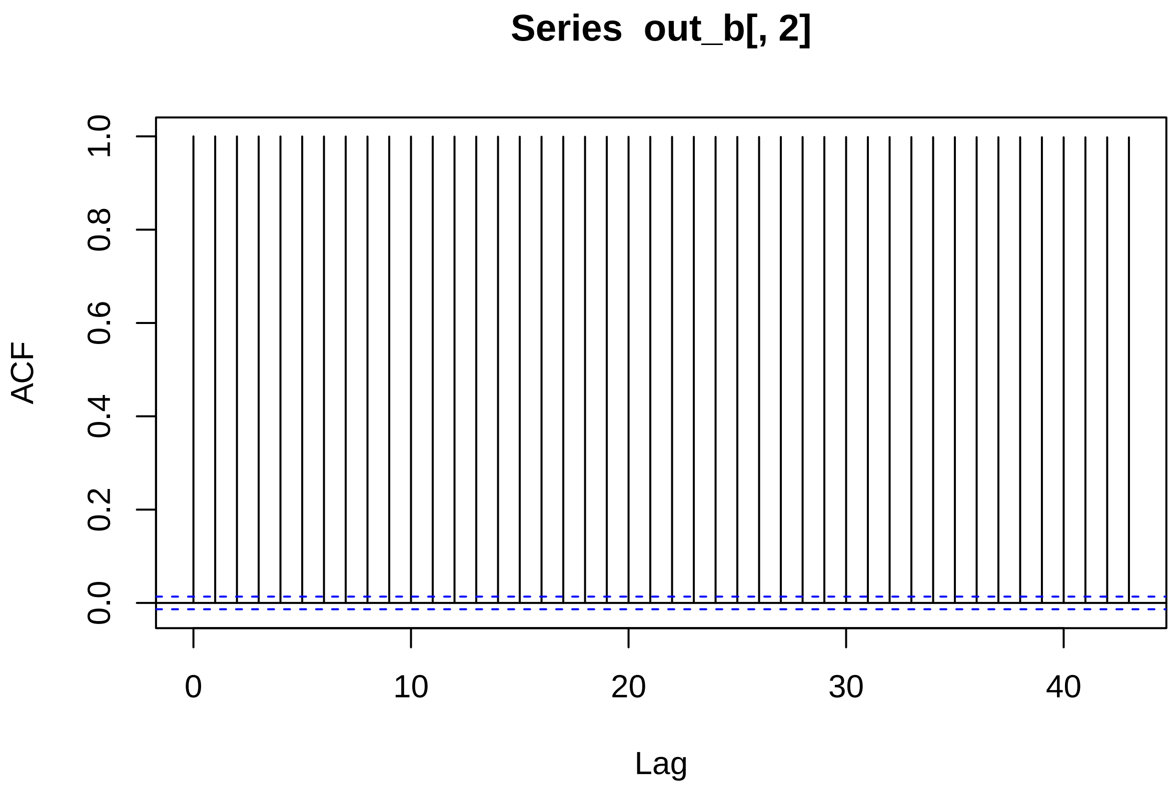
```
##
```

```
## Iterations = 1001:22000
## Thinning interval = 1
## Number of chains = 1
## Sample size per chain = 21000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                                Mean SD Naive SE Time-series SE
## (Intercept)                -66.8917  0        0              0
## perimeter_mean              -0.3481  0        0              0
## concave_points_mean        113.2543  0        0              0
## compactness_mean           -67.0638  0        0              0
## concavity_mean              36.5866  0        0              0
## area_se                      0.1428  0        0              0
## smoothness_se              433.2403  0        0              0
## concave_points_se          428.0355  0        0              0
## fractal_dimension_se     -2564.0379  0        0              0
## radius_worst                 3.4530  0        0              0
## texture_worst                0.4372  0        0              0
## fractal_dimension_worst    318.5212  0        0              0
##
## 2. Quantiles for each variable:
##
##                                2.5%        25%        50%        75%      97.5%
## (Intercept)                -66.8917   -66.8917   -66.8917   -66.8917   -66.8917
## perimeter_mean              -0.3481    -0.3481    -0.3481    -0.3481    -0.3481
## concave_points_mean        113.2543   113.2543   113.2543   113.2543   113.2543
## compactness_mean           -67.0638   -67.0638   -67.0638   -67.0638   -67.0638
## concavity_mean              36.5866    36.5866    36.5866    36.5866    36.5866
## area_se                      0.1428     0.1428     0.1428     0.1428     0.1428
## smoothness_se              433.2403   433.2403   433.2403   433.2403   433.2403
## concave_points_se          428.0355   428.0355   428.0355   428.0355   428.0355
## fractal_dimension_se     -2564.0379 -2564.0379 -2564.0379 -2564.0379 -2564.0379
## radius_worst                 3.4530     3.4530     3.4530     3.4530     3.4530
## texture_worst                0.4372     0.4372     0.4372     0.4372     0.4372
## fractal_dimension_worst    318.5212   318.5212   318.5212   318.5212   318.5212
```

```
acf(out_b[,1])
```



```
acf(out_b[,2])
```



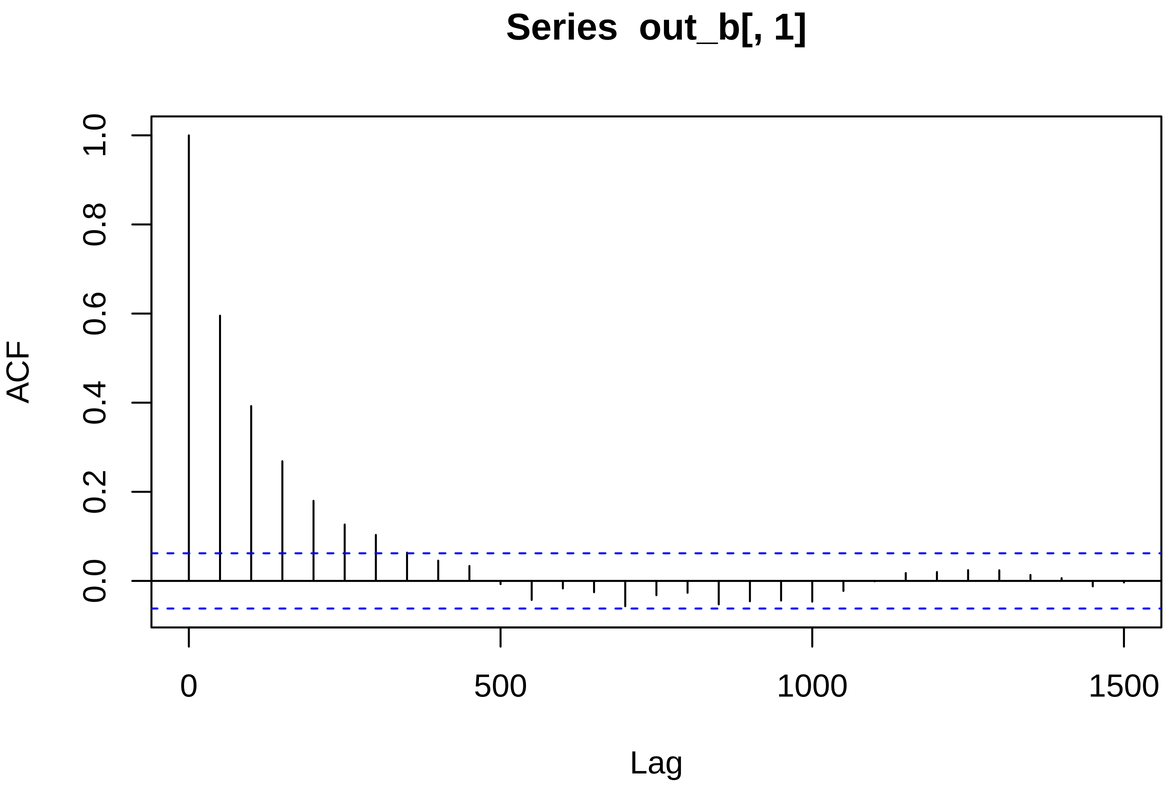
```
out_b = MCMClogit(formula_b, data, burnin=5000, mcmc=50000,
                  beta.start = beta.start, thin = 50, tune=0.5)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

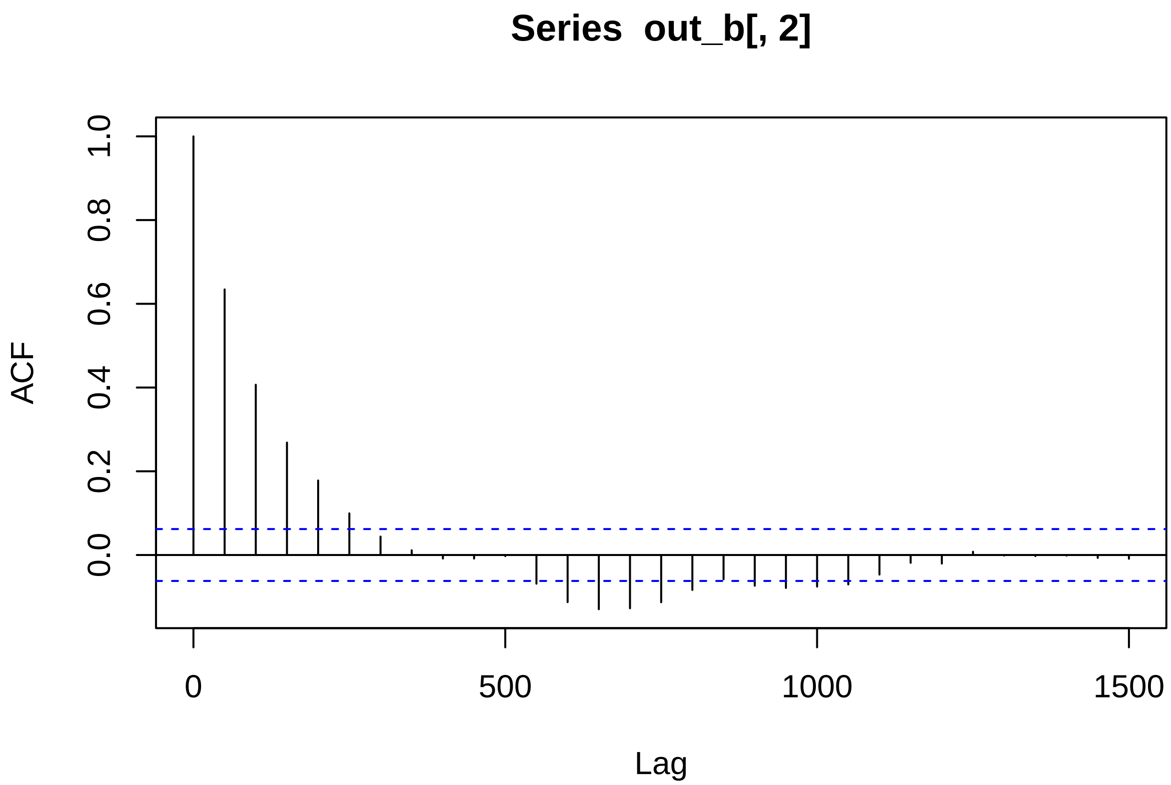
```
summary(out_b)
```

```
##
## Iterations = 5001:54951
## Thinning interval = 50
## Number of chains = 1
## 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
## (Intercept)             -4.089e+01   6.10325 1.930e-01      4.069e-01
## perimeter_mean          -8.767e-02   0.08165 2.582e-03      5.462e-03
## concave_points_mean      7.789e+01  33.64842 1.064e+00      2.578e+00
## compactness_mean        -4.411e+01  20.79019 6.574e-01      1.713e+00
## concavity_mean           1.849e+01  10.76886 3.405e-01      8.339e-01
## area_se                 -7.147e-03   0.01074 3.396e-04      5.185e-04
## smoothness_se            2.135e+02 150.05698 4.745e+00      1.353e+01
## concave_points_se        2.921e+02 101.48434 3.209e+00      5.821e+00
## fractal_dimension_se    -1.196e+03 340.24976 1.076e+01      1.952e+01
## radius_worst             1.662e+00   0.44882 1.419e-02      2.645e-02
## texture_worst            2.749e-01   0.05205 1.646e-03      3.775e-03
## fractal_dimension_worst  1.559e+02  46.60596 1.474e+00      3.093e+00
##
## 2. Quantiles for each variable:
##
##                               2.5%        25%        50%        75%      97.5%
## (Intercept)             -5.348e+01 -4.504e+01 -4.084e+01 -3.637e+01  -29.41492
## perimeter_mean          -2.363e-01 -1.449e-01 -9.337e-02 -3.226e-02    0.08183
## concave_points_mean      1.352e+01  5.524e+01  7.861e+01  1.017e+02  141.59098
## compactness_mean        -8.455e+01 -5.933e+01 -4.349e+01 -2.957e+01   -4.49315
## concavity_mean          -2.063e+00  1.172e+01  1.872e+01  2.589e+01   39.71891
## area_se                 -2.774e-02 -1.452e-02 -6.903e-03 -1.231e-04    0.01324
## smoothness_se           -8.381e+01  1.137e+02  2.063e+02  3.125e+02  512.29411
## concave_points_se        8.751e+01  2.252e+02  2.941e+02  3.573e+02  499.56987
## fractal_dimension_se    -1.905e+03 -1.411e+03 -1.177e+03 -9.488e+02 -589.19519
## radius_worst             7.393e-01  1.375e+00  1.711e+00  1.994e+00    2.41334
## texture_worst            1.752e-01  2.396e-01  2.744e-01  3.086e-01    0.37594
## fractal_dimension_worst  7.238e+01  1.237e+02  1.556e+02  1.866e+02  255.52220
```

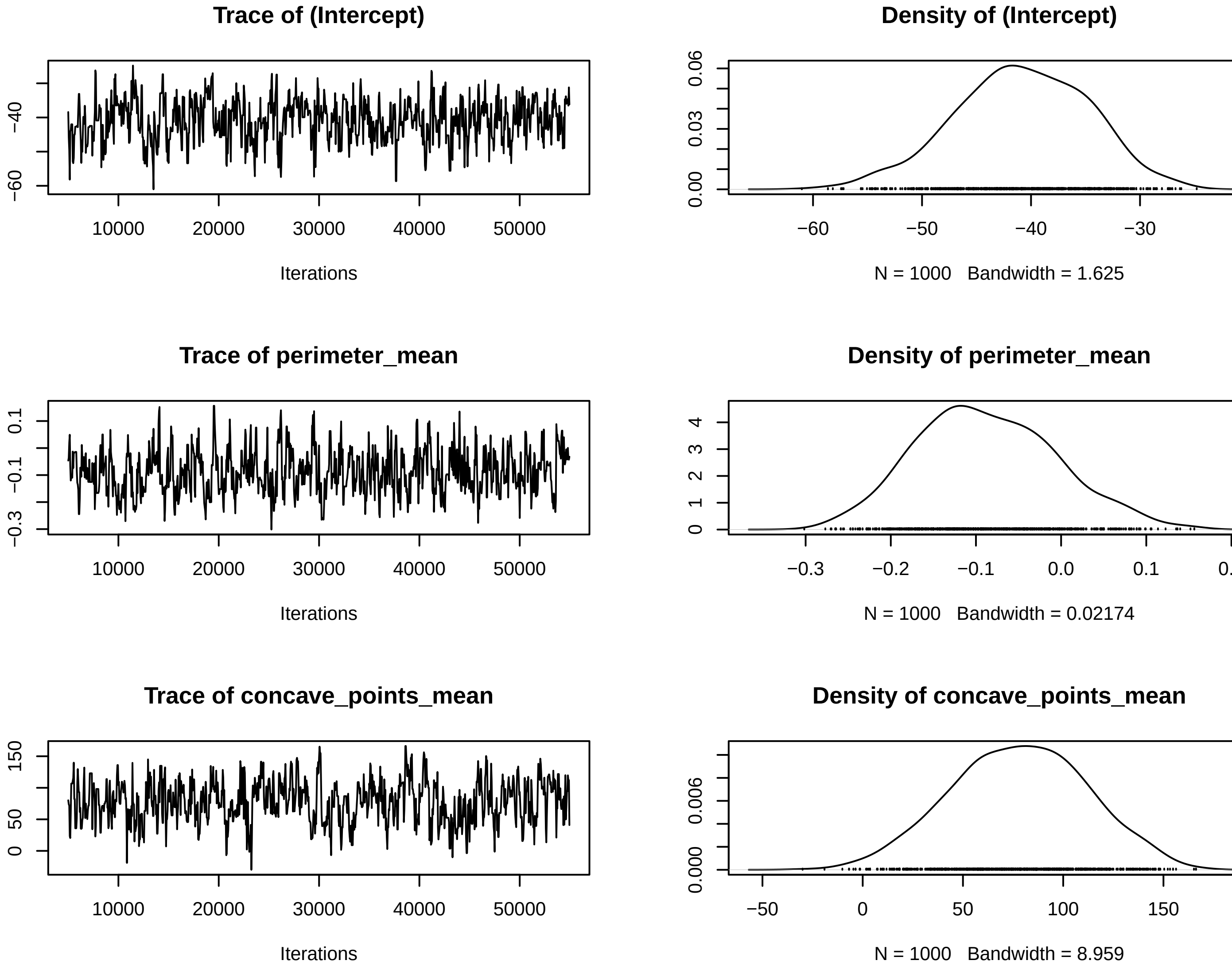
```
acf(out_b[,1])
```

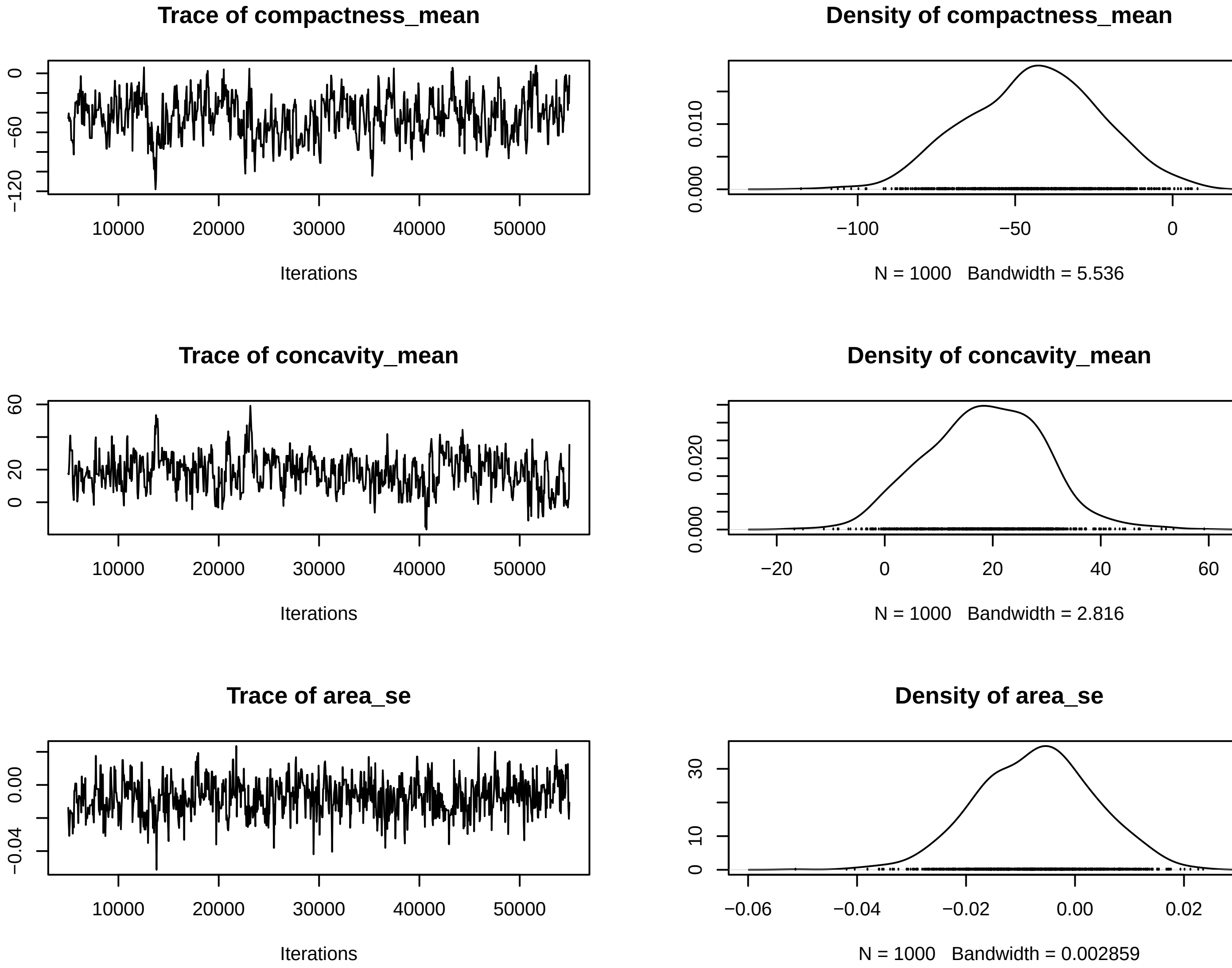


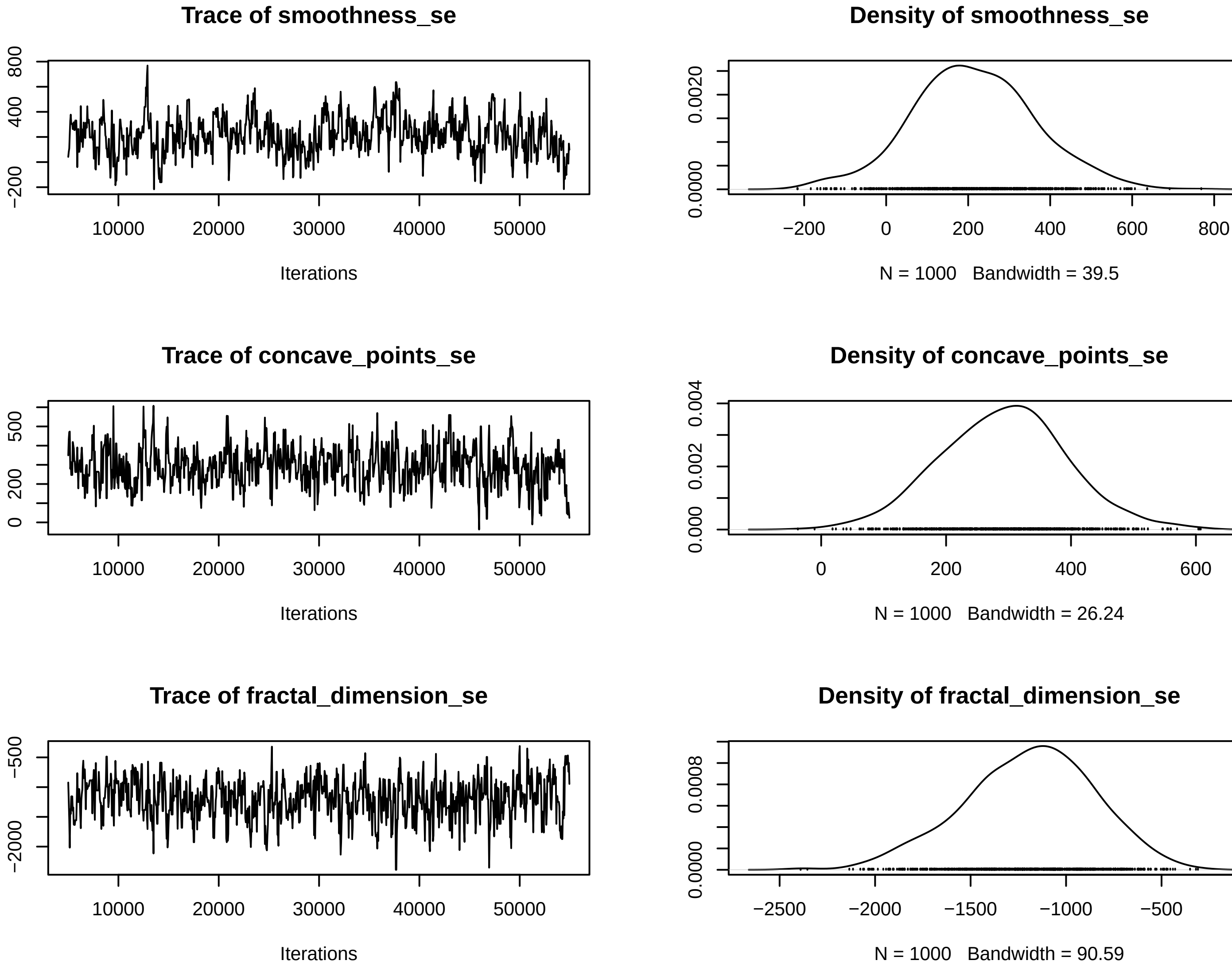
```
acf(out_b[,2])
```

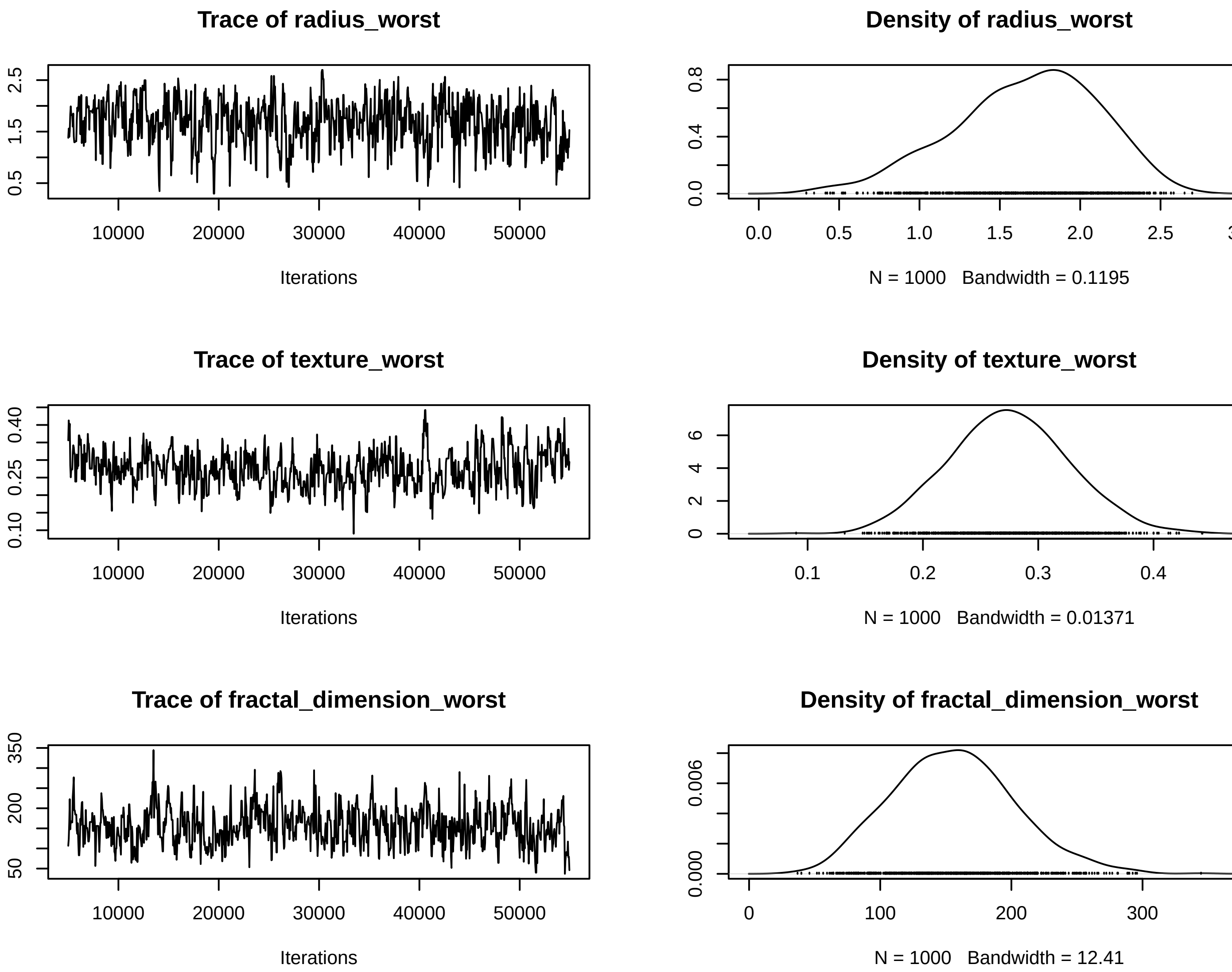


```
plot(out_b)
```


Trace of (Intercept)
−60
−40
10000
20000
30000
40000
50000
Iterations
Density of (Intercept)
0.00
0.03
0.06
−60
−50
−40
−30
N = 1000 Bandwidth = 1.625
Trace of perimeter_mean
−0.3
−0.1
0.1
10000
20000
30000
40000
50000
Iterations
Density of perimeter_mean
0
1
2
3
4
−0.3
−0.2
−0.1
0.0
0.1
N = 1000 Bandwidth = 0.02174
Trace of concave_points_mean
0
50
150
10000
20000
30000
40000
50000
Iterations
Density of concave_points_mean
0.000
0.006
−50
0
50
100
150
N = 1000 Bandwidth = 8.959


Trace of compactness_mean
0
−60
−120
10000
20000
30000
40000
50000
Iterations
Density of compactness_mean
0.000
0.010
−100
−50
0
N = 1000 Bandwidth = 5.536
Trace of concavity_mean
60
20
0
10000
20000
30000
40000
50000
Iterations
Density of concavity_mean
0.000
0.020
−20
0
20
40
60
N = 1000 Bandwidth = 2.816
Trace of area_se
0.00
−0.04
10000
20000
30000
40000
50000
Iterations
Density of area_se
30
10
0
−0.06
−0.04
−0.02
0.00
0.02
N = 1000 Bandwidth = 0.002859


Trace of smoothness_se
800
400
−200
10000
20000
30000
40000
50000
Iterations
Density of smoothness_se
0.0020
0.0000
−200
0
200
400
600
800
N = 1000   Bandwidth = 39.5
Trace of concave_points_se
500
200
0
10000
20000
30000
40000
50000
Iterations
Density of concave_points_se
0.004
0.002
0.000
0
200
400
600
N = 1000   Bandwidth = 26.24
Trace of fractal_dimension_se
−500
−2000
10000
20000
30000
40000
50000
Iterations
Density of fractal_dimension_se
0.0008
0.0000
−2500
−2000
−1500
−1000
−500
N = 1000   Bandwidth = 90.59



```
# Run Bayesian logistic regression with Spike-and-Slab priors (to variable selection)
set.seed(123)
model <- MCMClogit(formula, data = data, burnin = 5000, mcmc = 20000
                   , marginal.likelihood = "Laplace") # , thin = 10, b0 = 0, B0 = 0.1
```

```
## Warning in MCMClogit(formula, data = data, burnin = 5000, mcmc = 20000, : Cannot calculate marginal
```

```
summary(model)
```

```
##
## Iterations = 5001:25000
## Thinning interval = 1
## Number of chains = 1
## Sample size per chain = 20000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
```

```
##                              Mean       SD  Naive SE Time-series SE
## (Intercept)               -6.8170   2.9513 0.0208690       0.219795
## texture_mean               0.1294   0.1348 0.0009531       0.009661
## smoothness_mean          207.3107  41.2005 0.2913315       3.233150
## symmetry_mean              6.9549  13.6898 0.0968017       0.997838
## fractal_dimension_mean  -684.5427  91.4640 0.6467481       7.597434
## texture_se                -1.4813   0.7652 0.0054108       0.054139
## smoothness_se           -234.4591 126.4486 0.8941266       9.664144
## compactness_se            21.2337  26.1173 0.1846775       1.799246
## concavity_se              -1.4548  11.8178 0.0835642       0.846533
## concave_points_se        411.5040  64.7565 0.4578975       4.564841
## symmetry_se               -1.2348  53.9843 0.3817264       3.947217
## fractal_dimension_se    -193.6926 218.7169 1.5465618      12.937977
## texture_worst              0.2698   0.1255 0.0008877       0.009101
## smoothness_worst           0.9483  28.4640 0.2012712       2.053050
## symmetry_worst            12.2825   8.7408 0.0618070       0.658451
## fractal_dimension_worst  147.8957  37.6905 0.2665122       2.554404
##
## 2. Quantiles for each variable:
##
##                               2.5%        25%       50%       75%      97.5%
## (Intercept)              -12.51578   -8.78930   -6.8489   -4.7347   -0.98681
## texture_mean              -0.14451    0.03281    0.1240    0.2273    0.38854
## smoothness_mean          126.36954  181.04922  207.4663  233.5748  289.55512
## symmetry_mean            -20.77997   -1.01465    7.3981   15.9900   33.16468
## fractal_dimension_mean  -876.85892 -744.76209 -676.1047 -624.8130 -519.74380
## texture_se                -2.98677   -1.92703   -1.5112   -0.9390   -0.04385
## smoothness_se           -492.69637 -319.84949 -241.2256 -153.4581   38.46158
## compactness_se           -30.59198    5.34563   19.9356   38.9766   73.12934
## concavity_se             -25.61537   -8.41350   -1.4880    6.0837   21.32527
## concave_points_se        296.11128  368.03285  407.6404  450.6254  546.85401
## symmetry_se             -105.48846  -39.57962    1.0433   37.1174  108.12212
## fractal_dimension_se    -656.33024 -324.37467 -186.6719  -45.9750  200.42144
## texture_worst              0.04377    0.17585    0.2714    0.3511    0.53608
## smoothness_worst         -54.51081  -18.36763    0.8286   21.8650   56.01089
## symmetry_worst            -4.58397    6.13773   12.0131   18.5500   29.77524
## fractal_dimension_worst   82.79661  124.27778  144.7862  172.0066  233.27842
```

```r
apply(model, 2, function(x) mean(x != 0))  # Approximate inclusion probability
```

```
##             (Intercept)            texture_mean         smoothness_mean
##                       1                       1                       1
##           symmetry_mean  fractal_dimension_mean              texture_se
##                       1                       1                       1
##           smoothness_se          compactness_se            concavity_se
##                       1                       1                       1
##       concave_points_se             symmetry_se    fractal_dimension_se
##                       1                       1                       1
##           texture_worst        smoothness_worst          symmetry_worst
##                       1                       1                       1
## fractal_dimension_worst
##                       1
```

Evaluate Models with Deviance Information Criterion (DIC)

In the following code, we will calculate the Deviance Information Criterion (DIC) for both the Frequentist and Bayesian models. The DIC is a measure of model fit that penalizes the complexity of the model. Lower values of DIC indicate better model fit. The DIC is calculated as follows:

$$DIC = \bar{D} + p_D$$

where:
- $\bar{D}$ is the posterior mean deviance:

$$\bar{D} = \mathbb{E}[D(\theta) \mid \mathcal{D}]$$

with $D(\theta) = -2 \log p(\mathcal{D} \mid \theta)$, the deviance evaluated at parameter θ. - p_D a penalization term (effective number of parameters to penalize model complexity):

$$p_D = \bar{D} - D(\hat{\theta})$$

where $\hat{\theta}$ is the posterior mean of θ.

The R implementation of the DIC function is as follows and was developed with help of Prof Michael Wiper:

```r
# DIC Code
DIC = function(model, X, data, target) {
  dev = 0
  # Calculate Average Deviance of MCMC
  for (i in 1:nrow(model)) {
    params <- model[i,]
    p = inv.logit(X %*% params)
    p[data[target] == 0] = 1-p[data[target] == 0]
    dev = dev - 2 * sum(log(p))  # Negative log-likelihood
  }
  D_bar = dev / nrow(model)

  # D_theta: Deviance at the posterior mean (using the average parameter values)
  posterior_means <- colMeans(model)
  linear_predictor <- X %*% posterior_means
  p_post <- inv.logit(linear_predictor)
  p_post[data[target] == 0] = 1-p_post[data[target] == 0]

  D_theta = -2 * sum(log(p_post)) # Deviance at the posterior mean

  # p_D: Posterior deviance penalty
  p_D = D_bar - D_theta

  # DIC
  DIC = D_bar + p_D

  return(list(DIC=DIC, D_bar=D_bar, p_D=p_D))
}
```

We now continue with applying the DIC Score to the model derived from frequentist variable selection and the model derived from Bayesian variable selection. The straight forward conclusion is that the DIC is significantly better (lower) for the model that was set up with the Bayesian Variable Selection approach. Based on this result, we conclude this to be the best model and will use it for further analysis.

```r
# Frequentist
model = out
X <- model.matrix(~ texture_mean + smoothness_mean + symmetry_mean +
    fractal_dimension_mean + texture_se + smoothness_se + compactness_se +
    concavity_se + concave_points_se + symmetry_se + fractal_dimension_se +
    texture_worst + smoothness_worst + symmetry_worst + fractal_dimension_worst, data = data) # model m
target = "diagnosis"

print("Frequentist Variable Selection DIC Score")
```

```
## [1] "Frequentist Variable Selection DIC Score"
```

```r
DIC(model, X, data, target)
```

```
## $DIC
## [1] 245.5246
##
## $D_bar
## [1] 229.9737
##
## $p_D
## [1] 15.5509
```

```r
# Bayesian
model = out_b
X <- model.matrix(~ perimeter_mean + concave_points_mean + compactness_mean +
    concavity_mean + area_se + smoothness_se + concave_points_se +
    fractal_dimension_se + radius_worst + texture_worst + fractal_dimension_worst, data = data) # model
target = "diagnosis"

print("Bayesian Variable Selection DIC Score")
```

```
## [1] "Bayesian Variable Selection DIC Score"
```

```r
DIC(model, X, data, target)
```

```
## $DIC
## [1] 89.86894
##
## $D_bar
## [1] 80.33019
##
## $p_D
## [1] 9.538756
```

Prediction

```r
# TODO: Make predictions
```

Conclusion

TODO: Write Conclusion