

# Package ‘bayesreg’

June 3, 2019

**Type** Package

**Title** Bayesian Regression Models with Global-Local Shrinkage Priors

**Version** 1.1

**Date** 2019-06-03

**Author** Daniel F. Schmidt [aut, cph, cre],  
Enes Makalic [aut, cph]

**Maintainer** Daniel F. Schmidt <daniel.schmidt@monash.edu>

**Description** Fits linear or logistic regression model using Bayesian global-local shrinkage prior hierarchies as described in Polson and Scott (2010) <doi:10.1093/acprof:oso/9780199694587.003.0017>. Handles ridge, lasso, horseshoe and horseshoe+ regression with logistic, Gaussian, Laplace or Student-t distributed targets using the algorithms summarized in Makalic and Schmidt (2016) <arXiv:1611.06649>.

**License** GPL (>= 3)

**Imports** stats (>= 3.0), pgdraw (>= 1.0)

**RoxygenNote** 6.1.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2019-06-03 11:30:19 UTC

## R topics documented:

bayesreg-package . . . . .	2
bayesreg . . . . .	5
predict.bayesreg . . . . .	9
spambase . . . . .	12
summary.bayesreg . . . . .	14
<b>Index</b>	<b>17</b>

**Description**

This is a comprehensive, user-friendly package implementing the state-of-the-art in Bayesian linear regression and Bayesian logistic regression. Features of the toolbox include:

- Supports Gaussian, Laplace, Student-t and logistic binary data models.
- Efficient and numerically stable implementations of Bayesian ridge, Bayesian lasso, horseshoe and horseshoe+ regression.
- Provides variable ranking and importance, credible intervals and diagnostics such as the widely applicable information criterion.
- Factor variables are automatically grouped together and additional shrinkage is applied to the set of indicator variables to which they expand.
- Prediction tools for generating credible intervals and Bayesian averaging of predictions.

The lasso, horseshoe and horseshoe+ priors are recommended for data sets where the number of predictors is greater than the sample size. The non-Gaussian models are based on scale-mixture representations. Logistic regression utilises the Polya-gamma sampler implemented in the pgdraw package.

**Details**

To support analysis of data with outliers, we provide two heavy-tailed error models in our implementation of Bayesian linear regression: Laplace and Student-t distribution errors. The widely applicable information criterion (WAIC) is routinely calculated and displayed to assist users in selecting an appropriate prior distribution for their particular problem, i.e., choice of regularisation or data model. Most features are straightforward to use.

Further information on the particular algorithms/methods implemented in this package provided by the literature referenced below.

**Note**

To cite this package please reference:

Makalic, E. & Schmidt, D. F. High-Dimensional Bayesian Regularised Regression with the BayesReg Package arXiv:1611.06649 [stat.CO], 2016 <https://arxiv.org/pdf/1611.06649.pdf>

A MATLAB-compatible implementation of this package can be obtained from:

<https://au.mathworks.com/matlabcentral/fileexchange/60823>

**Author(s)**

Daniel Schmidt <daniel.schmidt@monash.edu>

Faculty of Information Technology, Monash University, Australia

Enes Makalic <emakalic@unimelb.edu.au>

Centre for Epidemiology and Biostatistics, The University of Melbourne, Australia

## References

- Bhadra, A.; Datta, J.; Polson, N. G. & Willard, B. The Horseshoe+ Estimator of Ultra-Sparse Signals Bayesian Analysis, 2016
- Bhattacharya, A.; Chakraborty, A. & Mallick, B. K. Fast sampling with Gaussian scale-mixture priors in high-dimensional regression arXiv:1506.04778, 2016
- Carvalho, C. M.; Polson, N. G. & Scott, J. G. The horseshoe estimator for sparse signals Biometrika, Vol. 97, pp. 465-480, 2010
- Makalic, E. & Schmidt, D. F. A Simple Sampler for the Horseshoe Estimator IEEE Signal Processing Letters, Vol. 23, pp. 179-182, 2016
- Park, T. & Casella, G. The Bayesian Lasso Journal of the American Statistical Association, Vol. 103, pp. 681-686, 2008
- Polson, N. G.; Scott, J. G. & Windle, J. Bayesian inference for logistic models using Polya-Gamma latent variables Journal of the American Statistical Association, Vol. 108, pp. 1339-1349, 2013
- Rue, H. Fast sampling of Gaussian Markov random fields Journal of the Royal Statistical Society (Series B), Vol. 63, pp. 325-338, 2001
- Xu, Z., Schmidt, D.F., Makalic, E., Qian, G. & Hopper, J.L. Bayesian Grouped Horseshoe Regression with Application to Additive Models AI 2016: Advances in Artificial Intelligence, pp. 229-240, 2016

## See Also

[bayesreg](#)

## Examples

```
# -----
# Example 1: Gaussian regression
X = matrix(rnorm(100*20),100,20)
b = matrix(0,20,1)
b[1:5] = c(5,4,3,2,1)
y = X %*% b + rnorm(100, 0, 1)

df <- data.frame(X,y)
rv.lm <- lm(y~.,df) # Regular least-squares
summary(rv.lm)

rv.hs <- bayesreg(y~.,df,prior="hs") # Horseshoe regression
rv.hs.s <- summary(rv.hs)

# Expected squared prediction error for least-squares
coef_ls = coef(rv.lm)
as.numeric(sum( (as.matrix(coef_ls[-1]) - b)^2 ) + coef_ls[1]^2)

# Expected squared prediction error for horseshoe
as.numeric(sum( (rv.hs$mu.beta - b)^2 ) + rv.hs$mu.beta^2)

# -----
```

```

# Example 2: Gaussian v Student-t robust regression
X = 1:10;
y = c(-0.6867, 1.7258, 1.9117, 6.1832, 5.3636, 7.1139, 9.5668, 10.0593, 11.4044, 6.1677);
df = data.frame(X,y)

# Gaussian ridge
rv.G <- bayesreg(y~., df, model = "gaussian", prior = "ridge", n.samples = 1e3)

# Student-t ridge
rv.t <- bayesreg(y~., df, model = "t", prior = "ridge", t.dof = 5, n.samples = 1e3)

# Plot the different estimates with credible intervals
plot(df$X, df$y, xlab="x", ylab="y")

yhat_G <- predict(rv.G, df, bayes.avg=TRUE)
lines(df$X, yhat_G[,1], col="blue", lwd=2.5)
lines(df$X, yhat_G[,3], col="blue", lwd=1, lty="dashed")
lines(df$X, yhat_G[,4], col="blue", lwd=1, lty="dashed")

yhat_t <- predict(rv.t, df, bayes.avg=TRUE)
lines(df$X, yhat_t[,1], col="darkred", lwd=2.5)
lines(df$X, yhat_t[,3], col="darkred", lwd=1, lty="dashed")
lines(df$X, yhat_t[,4], col="darkred", lwd=1, lty="dashed")

legend(1,11,c("Gaussian","Student-t (dof=5)"),lty=c(1,1),col=c("blue","darkred"),
      lwd=c(2.5,2.5), cex=0.7)

## Not run:
# -----
# Example 3: Logistic regression on spambase
data(spambase)

# bayesreg expects binary targets to be factors
spambase$is.spam <- factor(spambase$is.spam)

# First take a subset of the data (1/10th) for training, reserve the rest for testing
spambase.tr = spambase[seq(1,nrow(spambase),10),]
spambase.tst = spambase[-seq(1,nrow(spambase),10),]

# Fit a model using logistic horseshoe for 2,000 samples
rv <- bayesreg(is.spam ~ ., spambase.tr, model = "logistic", prior = "horseshoe", n.samples = 2e3)

# Summarise, sorting variables by their ranking importance
rv.s <- summary(rv,sort.rank=TRUE)

# Make predictions about testing data -- get class predictions and class probabilities
y_pred <- predict(rv, spambase.tst, type='class')

# Check how well did our predictions did by generating confusion matrix
table(y_pred, spambase.tst$is.spam)

# Calculate logarithmic loss on test data
y_prob <- predict(rv, spambase.tst, type='prob')

```

```

cat('Neg Log-Like for no Bayes average, posterior mean estimates: ', sum(-log(y_prob[,1])), '\n')
y_prob <- predict(rv, spambase.tst, type='prob', sum.stat="median")
cat('Neg Log-Like for no Bayes average, posterior median estimates: ', sum(-log(y_prob[,1])), '\n')
y_prob <- predict(rv, spambase.tst, type='prob', bayes.avg=TRUE)
cat('Neg Log-Like for Bayes average: ', sum(-log(y_prob[,1])), '\n')

## End(Not run)

```

---

bayesreg

*Fitting Bayesian Regression Models with Continuous Shrinkage Priors*


---

## Description

Fit a linear or logistic regression model using Bayesian continuous shrinkage prior distributions. Handles ridge, lasso, horseshoe and horseshoe+ regression with logistic, Gaussian, Laplace or Student-t distributed targets. See [bayesreg-package](#) for more details on the features available in this package.

## Usage

```

bayesreg(formula, data, model = "normal", prior = "ridge",
  n.samples = 1000, burnin = 1000, thin = 5, t.dof = 5)

```

## Arguments

formula	An object of class " <a href="#">formula</a> ": a symbolic description of the model to be fitted using the standard R formula notation.
data	A data frame containing the variables in the model.
model	The distribution of the target (y) variable. Continuous or numeric variables can be distributed as per a Gaussian distribution (model="gaussian" or model="normal"), Laplace distribution (model = "laplace" or model = "l1") or Student-t distribution ("model" = "studentt" or "model" = "t"). For binary targets (factors with two levels) either model="logistic" or "model"="binomial" should be used.
prior	The continuous shrinkage prior distribution over the regression coefficients that is to be used. Options include ridge regression (prior="rr" or prior="ridge"), lasso regression (prior="lasso"), horseshoe regression (prior="hs" or prior="horseshoe") and horseshoe+ regression (prior="hs+" or prior="horseshoe+")
n.samples	Number of posterior samples to generate.
burnin	Number of burn-in samples.
thin	Desired level of thinning.
t.dof	Degrees of freedom for the Student-t distribution.

**Value**

An object with S3 class "bayesreg" containing the results of the sampling process, plus some additional information.

beta	Posterior samples the regression model coefficients.
beta0	Posterior samples of the intercept parameter.
sigma2	Posterior samples of the square of the scale parameter; for Gaussian distributed targets this is equal to the variance. For binary targets this is empty.
mu.beta	The mean of the posterior samples for the regression coefficients.
mu.beta0	The mean of the posterior samples for the intercept parameter.
mu.sigma2	The mean of the posterior samples for squared scale parameter.
tau2	Posterior samples of the global shrinkage parameter.
t.stat	Posterior t-statistics for each regression coefficient.
var.ranks	Ranking of the covariates by their importance, with "1" denoting the most important covariate.
log.l	The log-likelihood at the posterior means of the model parameters
waic	The Widely Applicable Information Criterion (WAIC) score for the model
waic.dof	The effective degrees-of-freedom of the model, as estimated by the WAIC.

The returned object also stores the parameters/options used to run bayesreg:

formula	The object of type "formula" describing the fitted model.
model	The distribution of the target (y) variable.
prior	The shrinkage prior used to fit the model.
n.samples	The number of samples generated from the posterior distribution.
burnin	The number of burnin samples that were generated.
thin	The level of thinning.
n	The sample size of the data used to fit the model.
p	The number of covariates in the fitted model.

**Details**

Draws a series of samples from the posterior distribution of a linear (Gaussian, Laplace or Student-t) or logistic regression model with specified continuous shrinkage prior distribution (ridge regression, lasso, horseshoe and horseshoe+) using Gibbs sampling. The intercept parameter is always included, and is never penalised.

While only `n.samples` are returned, the total number of samples generated is equal to `burnin+n.samples*thin`. To generate the samples of the regression coefficients, the code will use either Rue's algorithm (when the number of samples is twice the number of covariates) or the algorithm of Bhattacharya et al. as appropriate. Factor variables are automatically grouped together and additional shrinkage is applied to the set of indicator variables to which they expand.

**Note**

To cite this toolbox please reference:

Makalic, E. & Schmidt, D. F. High-Dimensional Bayesian Regularised Regression with the BayesReg Package arXiv:1611.06649 [stat.CO], 2016 <https://arxiv.org/pdf/1611.06649.pdf>

A MATLAB implementation of the bayesreg function is also available from:

<https://au.mathworks.com/matlabcentral/fileexchange/60823-bayesian-penalized-regression-with-contin>

**References**

Makalic, E. & Schmidt, D. F. High-Dimensional Bayesian Regularised Regression with the BayesReg Package arXiv:1611.06649 [stat.CO], 2016 <https://arxiv.org/pdf/1611.06649.pdf>

Park, T. & Casella, G. The Bayesian Lasso Journal of the American Statistical Association, Vol. 103, pp. 681-686, 2008

Carvalho, C. M.; Polson, N. G. & Scott, J. G. The horseshoe estimator for sparse signals Biometrika, Vol. 97, 465-480, 2010

Makalic, E. & Schmidt, D. F. A Simple Sampler for the Horseshoe Estimator IEEE Signal Processing Letters, Vol. 23, pp. 179-182, 2016 <https://arxiv.org/pdf/1508.03884v4.pdf>

Bhadra, A.; Datta, J.; Polson, N. G. & Willard, B. The Horseshoe+ Estimator of Ultra-Sparse Signals Bayesian Analysis, 2016

Polson, N. G.; Scott, J. G. & Windle, J. Bayesian inference for logistic models using Polya-Gamma latent variables Journal of the American Statistical Association, Vol. 108, 1339-1349, 2013

Rue, H. Fast sampling of Gaussian Markov random fields Journal of the Royal Statistical Society (Series B), Vol. 63, 325-338, 2001

Bhattacharya, A.; Chakraborty, A. & Mallick, B. K. Fast sampling with Gaussian scale-mixture priors in high-dimensional regression arXiv:1506.04778, 2016

**See Also**

The prediction function [predict.bayesreg](#) and summary function [summary.bayesreg](#)

**Examples**

```
# -----
# Example 1: Gaussian regression
X = matrix(rnorm(100*20),100,20)
b = matrix(0,20,1)
b[1:5] = c(5,4,3,2,1)
y = X %*% b + rnorm(100, 0, 1)

df <- data.frame(X,y)
rv.lm <- lm(y~.,df) # Regular least-squares
summary(rv.lm)

rv.hs <- bayesreg(y~.,df,prior="hs") # Horseshoe regression
rv.hs.s <- summary(rv.hs)

# Expected squared prediction error for least-squares
```

```

coef_ls = coef(rv.lm)
as.numeric(sum( (as.matrix(coef_ls[-1]) - b)^2 ) + coef_ls[1]^2)

# Expected squared prediction error for horseshoe
as.numeric(sum( (rv.hs$mu.beta - b)^2 ) + rv.hs$mu.beta^2)

# -----
# Example 2: Gaussian v Student-t robust regression
X = 1:10;
y = c(-0.6867, 1.7258, 1.9117, 6.1832, 5.3636, 7.1139, 9.5668, 10.0593, 11.4044, 6.1677);
df = data.frame(X,y)

# Gaussian ridge
rv.G <- bayesreg(y~., df, model = "gaussian", prior = "ridge", n.samples = 1e3)

# Student-t ridge
rv.t <- bayesreg(y~., df, model = "t", prior = "ridge", t.dof = 5, n.samples = 1e3)

# Plot the different estimates with credible intervals
plot(df$X, df$y, xlab="x", ylab="y")

yhat_G <- predict(rv.G, df, bayes.avg=TRUE)
lines(df$X, yhat_G[,1], col="blue", lwd=2.5)
lines(df$X, yhat_G[,3], col="blue", lwd=1, lty="dashed")
lines(df$X, yhat_G[,4], col="blue", lwd=1, lty="dashed")

yhat_t <- predict(rv.t, df, bayes.avg=TRUE)
lines(df$X, yhat_t[,1], col="darkred", lwd=2.5)
lines(df$X, yhat_t[,3], col="darkred", lwd=1, lty="dashed")
lines(df$X, yhat_t[,4], col="darkred", lwd=1, lty="dashed")

legend(1,11,c("Gaussian", "Student-t (dof=5)"),lty=c(1,1),col=c("blue", "darkred"),
      lwd=c(2.5,2.5), cex=0.7)

## Not run:
# -----
# Example 3: Logistic regression on spambase
data(spambase)

# bayesreg expects binary targets to be factors
spambase$is.spam <- factor(spambase$is.spam)

# First take a subset of the data (1/10th) for training, reserve the rest for testing
spambase.tr = spambase[seq(1,nrow(spambase),10),]
spambase.tst = spambase[-seq(1,nrow(spambase),10),]

# Fit a model using logistic horseshoe for 2,000 samples
rv <- bayesreg(is.spam ~ ., spambase.tr, model = "logistic", prior = "horseshoe", n.samples = 2e3)

# Summarise, sorting variables by their ranking importance
rv.s <- summary(rv,sort.rank=TRUE)

```



```

# Make predictions about testing data -- get class predictions and class probabilities
y_pred <- predict(rv, spambase.tst, type='class')

# Check how well did our predictions did by generating confusion matrix
table(y_pred, spambase.tst$is.spam)

# Calculate logarithmic loss on test data
y_prob <- predict(rv, spambase.tst, type='prob')
cat('Neg Log-Like for no Bayes average, posterior mean estimates: ', sum(-log(y_prob[,1])), '\n')
y_prob <- predict(rv, spambase.tst, type='prob', sum.stat="median")
cat('Neg Log-Like for no Bayes average, posterior median estimates: ', sum(-log(y_prob[,1])), '\n')
y_prob <- predict(rv, spambase.tst, type='prob', bayes.avg=TRUE)
cat('Neg Log-Like for Bayes average: ', sum(-log(y_prob[,1])), '\n')

## End(Not run)

```

---

predict.bayesreg	<i>Prediction method for Bayesian penalised regression (bayesreg) models</i>
------------------	--

---

## Description

Predict values based on Bayesian penalised regression ([bayesreg](#)) models.

## Usage

```

## S3 method for class 'bayesreg'
predict(object, newdata, type = "linpred",
        bayes.avg = FALSE, sum.stat = "mean", ...)

```

## Arguments

object	an object of class "bayesreg" created as a result of a call to <a href="#">bayesreg</a> .
newdata	A data frame providing the variables from which to produce predictions.
type	The type of predictions to produce; if type="linpred" it will return the linear predictor for both binary and continuous data. For binary data, if type="prob" it will return predictive probability estimates, and if type="class" and the data is binary, it will return the best guess at the class of the target variable.
bayes.avg	logical; whether to produce predictions using Bayesian averaging.
sum.stat	The type of summary statistic to use; either sum.stat="mean" or sum.stat="median".
...	Further arguments passed to or from other methods.

## Value

predict.bayesreg produces a vector or matrix of predictions of the specified type. If bayes.avg is FALSE a matrix with a single column pred is returned, containing the predictions.

If bayes.avg is TRUE, three additional columns are returned: se(pred), which contains standard errors for the predictions, and CI 2.5% and CI 97.5% which contain 95% credible intervals for the predictions.

## Details

predict.bayesreg produces predicted values using variables from the specified data frame. The type of predictions produced depend on the value of the parameter type.

If type="linpred", the predictions that are returned will be the value of the linear predictor formed from the model coefficients and the provided data.

If type="prob", the predictions will be probabilities. If the specified data frame includes a column with the same name as the target variable on which the model was created, the predictions will then be the probability density values for these target values. For binary data, if the specified data frame does not include a column with the same name as the target variable, the predictions will be the probability of the target being equal to the second level of the factor variable.

If type="class" and the target variable is binary, the predictions will be the most likely class.

If bayes.avg is FALSE the predictions will be produced by using a summary of the posterior samples of the coefficients and scale parameters as estimates for the model. If bayes.avg is TRUE, the predictions will be produced by posterior averaging over the posterior samples of the coefficients and scale parameters, allowing the uncertainty in the estimation process to be explicitly taken into account in the prediction process.

If sum.stat="mean" and bayes.avg is FALSE, the mean of the posterior samples will be used as point estimates for making predictions. Likewise, if sum.stat="median" and bayes.avg is FALSE, the co-ordinate wise posterior medians will be used as estimates for making predictions. If bayes.avg is TRUE and type!="prob", the posterior mean (median) of the predictions from each of the posterior samples will be used as predictions. The value of sum.stat has no effect if type="prob".

## See Also

The model fitting function [bayesreg](#) and summary function [summary.bayesreg](#)

## Examples

```
# -----
# Example 1: Fitting linear models to data and generating credible intervals
X = 1:10;
y = c(-0.6867, 1.7258, 1.9117, 6.1832, 5.3636, 7.1139, 9.5668, 10.0593, 11.4044, 6.1677);
df = data.frame(X,y)

# Gaussian ridge
rv.L <- bayesreg(y~., df, model = "laplace", prior = "ridge", n.samples = 1e3)

# Plot the different estimates with credible intervals
plot(df$X, df$y, xlab="x", ylab="y")

yhat <- predict(rv.L, df, bayes.avg=TRUE)
lines(df$X, yhat[,1], col="blue", lwd=2.5)
lines(df$X, yhat[,3], col="blue", lwd=1, lty="dashed")
lines(df$X, yhat[,4], col="blue", lwd=1, lty="dashed")
yhat <- predict(rv.L, df, bayes.avg=TRUE, sum.stat = "median")
lines(df$X, yhat[,1], col="red", lwd=2.5)
```

```

legend(1,11,c("Posterior Mean (Bayes Average)","Posterior Median (Bayes Average)"),
      lty=c(1,1),col=c("blue","red"),lwd=c(2.5,2.5), cex=0.7)

# -----
# Example 2: Predictive density for continuous data
X = 1:10;
y = c(-0.6867, 1.7258, 1.9117, 6.1832, 5.3636, 7.1139, 9.5668, 10.0593, 11.4044, 6.1677);
df = data.frame(X,y)

# Gaussian ridge
rv.G <- bayesreg(y~., df, model = "gaussian", prior = "ridge", n.samples = 1.5e3)

# Produce predictive density for X=2
df.tst = data.frame(y=seq(-7,12,0.01),X=2)
prob_noavg_mean <- predict(rv.G, df.tst, bayes.avg=FALSE, type="prob", sum.stat = "mean")
prob_noavg_med <- predict(rv.G, df.tst, bayes.avg=FALSE, type="prob", sum.stat = "median")
prob_avg <- predict(rv.G, df.tst, bayes.avg=TRUE, type="prob")

# Plot the density
plot(NULL, xlim=c(-7,12), ylim=c(0,0.14), xlab="y", ylab="p(y)")
lines(df.tst$y, prob_noavg_mean[,1],lwd=1.5)
lines(df.tst$y, prob_noavg_med[,1], col="red",lwd=1.5)
lines(df.tst$y, prob_avg[,1], col="green",lwd=1.5)

legend(-7,0.14,c("Mean (no averaging)","Median (no averaging)","Bayes Average"),
      lty=c(1,1,1),col=c("black","red","green"),lwd=c(1.5,1.5,1.5), cex=0.7)

title('Predictive densities for X=2')

## Not run:
# -----
# Example 3: Logistic regression on spambase
data(spambase)

# bayesreg expects binary targets to be factors
spambase$is.spam <- factor(spambase$is.spam)

# First take a subset of the data (1/10th) for training, reserve the rest for testing
spambase.tr = spambase[seq(1,nrow(spambase),10),]
spambase.tst = spambase[-seq(1,nrow(spambase),10),]

# Fit a model using logistic horseshoe for 2,000 samples
rv <- bayesreg(is.spam ~ ., spambase.tr, model = "logistic", prior = "horseshoe", n.samples = 2e3)

# Summarise, sorting variables by their ranking importance
rv.s <- summary(rv,sort.rank=TRUE)

# Make predictions about testing data -- get class predictions and class probabilities
y_pred <- predict(rv, spambase.tst, type='class')
y_prob <- predict(rv, spambase.tst, type='prob')

```

```
# Check how well our predictions did by generating confusion matrix
table(y_pred, spambase.tst$is.spam)

# Calculate logarithmic loss on test data
y_prob <- predict(rv, spambase.tst, type='prob')
cat('Neg Log-Like for no Bayes average, posterior mean estimates: ', sum(-log(y_prob[,1])), '\n')
y_prob <- predict(rv, spambase.tst, type='prob', sum.stat="median")
cat('Neg Log-Like for no Bayes average, posterior median estimates: ', sum(-log(y_prob[,1])), '\n')
y_prob <- predict(rv, spambase.tst, type='prob', bayes.avg=TRUE)
cat('Neg Log-Like for Bayes average: ', sum(-log(y_prob[,1])), '\n')

## End(Not run)
```

---

spambase

*Spambase*


---

## Description

This is a well known dataset with a binary target obtainable from the UCI machine learning dataset archive. Each row is an e-mail, which is considered to be either spam or not spam. The dataset contains 48 attributes that measure the percentage of times a particular word appears in the email, 6 attributes that measure the percentage of times a particular character appeared in the email, plus three attributes measuring run-lengths of capital letters.

## Usage

```
data(spambase)
```

## Format

A data frame with 4,601 rows and 58 variables (1 categorical, 57 continuous).

`is.spam` Is the email considered to be spam? (0=no,1=yes)

`word.freq.make` Percentage of times the word 'make' appeared in the e-mail

`word.freq.address` Percentage of times the word 'address' appeared in the e-mail

`word.freq.all` Percentage of times the word 'all' appeared in the e-mail

`word.freq.3d` Percentage of times the word '3d' appeared in the e-mail

`word.freq.our` Percentage of times the word 'our' appeared in the e-mail

`word.freq.over` Percentage of times the word 'over' appeared in the e-mail

`word.freq.remove` Percentage of times the word 'remove' appeared in the e-mail

`word.freq.internet` Percentage of times the word 'internet' appeared in the e-mail

`word.freq.order` Percentage of times the word 'order' appeared in the e-mail

`word.freq.mail` Percentage of times the word 'mail' appeared in the e-mail

`word.freq.receive` Percentage of times the word 'receive' appeared in the e-mail

`word.freq.will` Percentage of times the word 'will' appeared in the e-mail

word.freq.people Percentage of times the word 'people' appeared in the e-mail  
word.freq.report Percentage of times the word 'report' appeared in the e-mail  
word.freq.addresses Percentage of times the word 'addresses' appeared in the e-mail  
word.freq.free Percentage of times the word 'free' appeared in the e-mail  
word.freq.business Percentage of times the word 'business' appeared in the e-mail  
word.freq.email Percentage of times the word 'email' appeared in the e-mail  
word.freq.you Percentage of times the word 'you' appeared in the e-mail  
word.freq.credit Percentage of times the word 'credit' appeared in the e-mail  
word.freq.your Percentage of times the word 'your' appeared in the e-mail  
word.freq.font Percentage of times the word 'font' appeared in the e-mail  
word.freq.000 Percentage of times the word '000' appeared in the e-mail  
word.freq.money Percentage of times the word 'money' appeared in the e-mail  
word.freq.hp Percentage of times the word 'hp' appeared in the e-mail  
word.freq.hpl Percentage of times the word 'hpl' appeared in the e-mail  
word.freq.george Percentage of times the word 'george' appeared in the e-mail  
word.freq.650 Percentage of times the word '650' appeared in the e-mail  
word.freq.lab Percentage of times the word 'lab' appeared in the e-mail  
word.freq.labs Percentage of times the word 'labs' appeared in the e-mail  
word.freq.telnet Percentage of times the word 'telnet' appeared in the e-mail  
word.freq.857 Percentage of times the word '857' appeared in the e-mail  
word.freq.data Percentage of times the word 'data' appeared in the e-mail  
word.freq.415 Percentage of times the word '415' appeared in the e-mail  
word.freq.85 Percentage of times the word '85' appeared in the e-mail  
word.freq.technology Percentage of times the word 'technology' appeared in the e-mail  
word.freq.1999 Percentage of times the word '1999' appeared in the e-mail  
word.freq.parts Percentage of times the word 'parts' appeared in the e-mail  
word.freq.pm Percentage of times the word 'pm' appeared in the e-mail  
word.freq.direct Percentage of times the word 'direct' appeared in the e-mail  
word.freq.cs Percentage of times the word 'cs' appeared in the e-mail  
word.freq.meeting Percentage of times the word 'meeting' appeared in the e-mail  
word.freq.original Percentage of times the word 'original' appeared in the e-mail  
word.freq.project Percentage of times the word 'project' appeared in the e-mail  
word.freq.re Percentage of times the word 're' appeared in the e-mail  
word.freq.edu Percentage of times the word 'edu' appeared in the e-mail  
word.freq.table Percentage of times the word 'table' appeared in the e-mail  
word.freq.conference Percentage of times the word 'conference' appeared in the e-mail  
char.freq.; Percentage of times the character ';' appeared in the e-mail

char.freq.( Percentage of times the character '(' appeared in the e-mail  
 char.freq.[ Percentage of times the character '[' appeared in the e-mail  
 char.freq.! Percentage of times the character '!' appeared in the e-mail  
 char.freq.\$ Percentage of times the character '\$' appeared in the e-mail  
 char.freq.# Percentage of times the character '#' appeared in the e-mail  
 capital.run.length.average Average length of contiguous runs of capital letters in the e-mail  
 capital.run.length.longest Maximum length of contiguous runs of capital letters in the e-mail  
 capital.run.length.total Total number of capital letters in the e-mail

### Source

<https://archive.ics.uci.edu/ml/datasets/spambase/>

---

summary.bayesreg	<i>Summarization method for Bayesian penalised regression (bayesreg) models</i>
------------------	---

---

### Description

summary method for Bayesian regression models fitted using [bayesreg](#).

### Usage

```
## S3 method for class 'bayesreg'
summary(object, sort.rank = FALSE,
        display.OR = FALSE, ...)
```

### Arguments

object	An object of class "bayesreg" created as a result of a call to <a href="#">bayesreg</a> .
sort.rank	logical; if TRUE, the variables in the summary will be sorted by their importance as determined by their rank estimated by the Bayesian feature ranking algorithm.
display.OR	logical; if TRUE, the variables will be summarised in terms of their cross-sectional odds-ratios rather than their regression coefficients (logistic regression only).
...	Further arguments passed to or from other methods.

### Value

Returns an object with the following fields:

log.l	The log-likelihood of the model at the posterior mean estimates of the regression coefficients.
waic	The Widely Applicable Information Criterion (WAIC) score of the model.
waic.dof	The effective degrees-of-freedom of the model, as estimated by the WAIC.

r2	For non-binary data, the $R^2$ statistic.
sd.error	For non-binary data, the estimated standard deviation of the errors.
p.r2	For binary data, the pseudo- $R^2$ statistic.
mu.coef	The posterior means of the regression coefficients.
se.coef	The posterior standard deviations of the regression coefficients.
CI.coef	The posterior 95% credible interval for the regression coefficients.
med.OR	For binary data, the posterior median of the cross-sectional odds-ratios.
se.OR	For binary data, the posterior standard deviation of the cross-sectional odds-ratios.
CI.OR	For binary data, the posterior 95% credible interval for the cross-sectional odds-ratios.
t.stat	The posterior t-statistic for the coefficients.
n.stars	The significance level for the variable (see above).
rank	The variable importance rank as estimated by the Bayesian feature ranking algorithm (see above).
ESS	The effective sample size for the variable.
log.l0	For binary data, the log-likelihood of the null model (i.e., with only an intercept).

### Details

The `summary` method computes a number of summary statistics and displays these for each variable in a table, along with suitable header information.

For continuous target variables, the header information includes a posterior estimate of the standard deviation of the random disturbances (errors), the  $R^2$  statistic and the Widely applicable information criterion (WAIC) statistic. For logistic regression models, the header information includes the negative log-likelihood at the posterior mean of the regression coefficients, the pseudo  $R^2$  score and the WAIC statistic.

The main table summarises properties of the coefficients for each of the variables. The first column is the variable name. The second and third columns are either the mean and standard error of the coefficients, or the median and standard error of the cross-sectional odds-ratios if `display.OR=TRUE`.

The fourth and fifth columns are the 95% credible intervals of the coefficients (odds-ratios). The sixth column displays the posterior  $t$ -statistic, calculated as the ratio of the posterior mean on the posterior standard deviation for the coefficient. The seventh column is the importance rank assigned to the variable by the Bayesian feature ranking algorithm.

In between the seventh and eighth columns are up to two asterisks indicating significance; a variable scores a first asterisk if the 75% credible interval does not include zero, and scores a second asterisk if the 95% credible interval does not include zero. The final column gives an estimate of the effective sample size for the variable, ranging from 0 to 100, which indicates how much autocorrelation is present in the sample chain for this variable (smaller values indicating high levels of autocorrelation).

### See Also

The model fitting function [bayesreg](#) and prediction function [predict.bayesreg](#).

**Examples**

```
X = matrix(rnorm(100*20),100,20)
b = matrix(0,20,1)
b[1:9] = c(0,0,0,0,5,4,3,2,1)
y = X %*% b + rnorm(100, 0, 1)
df <- data.frame(X,y)

rv.hs <- bayesreg(y~.,df,prior="hs")      # Horseshoe regression

# Summarise without sorting by variable rank
rv.hs.s <- summary(rv.hs)

# Summarise sorting by variable rank
rv.hs.s <- summary(rv.hs, sort.rank = TRUE)
```



# Index

## \*Topic **datasets**

spambase, [12](#)

bayesreg, [3](#), [5](#), [9](#), [10](#), [14](#), [15](#)

bayesreg-package, [2](#)

formula, [5](#), [6](#)

predict.bayesreg, [7](#), [9](#), [15](#)

spambase, [12](#)

summary.bayesreg, [7](#), [10](#), [14](#)