BSR(Bayesian Subset Regression) Package

July 9, 2013

```
> library(BSR)
```

Detailed example of calling the BSR function

Load the training and testing data frames.

```
> f <- system.file("sampleData", "data.rda", package = "BSR")
> load(f)
```

Both the training and testing data frames must have the same variables in the same order.

```
> train.data[1:5, 1:5]
```

```
  id y intercept x.1 x.2
1  1 1         1   1   2
2  2 1         1   2   0
3  3 1         1   1   2
4  4 1         1   1   1
5  5 1         1   2   1
```

```
> test.data[1:5, 1:5]
```

```
  id y intercept x.1 x.2
1  1 1         1   2   2
2  2 1         1   1   2
3  3 1         1   2   0
4  4 1         1   2   1
5  5 1         1   1   0
```

The response variable is "y", the id variable is "id", and the intercept column is "intercept". We will let the intercept be the only variable that is in every model.

```
> response.var <- "y"
> id.var <- "id"
> vars0 <- "intercept"
```

We need to choose a directory that has write access to serve as the directory where the output files will be created. For this example, let this directory be the working directory.

```
> outdir <- getwd()
> print(outdir)
```

```
[1] "C:/Users/wheelerb/AppData/Local/Temp/RtmpHcczcg/Rbuild6e4f3a4f/BSR/inst/doc"
```

Define the list of groups of variables.

```
> groups <- list(group1 = c("x.1", "x.2"), group2 = c("x.3", "x.4",
+     "x.5"), group3 = paste("x.", 20:25, sep = ""), group4 = paste("x.",
+     47:50, sep = ""), group5 = "x.40", group6 = "x.41", group7 = "x.42")
> groups
```

```
$group1
[1] "x.1" "x.2"

$group2
[1] "x.3" "x.4" "x.5"

$group3
[1] "x.20" "x.21" "x.22" "x.23" "x.24" "x.25"

$group4
[1] "x.47" "x.48" "x.49" "x.50"

$group5
[1] "x.40"

$group6
[1] "x.41"

$group7
[1] "x.42"
```

Let us impose the conditions that groups 6 and 7 must be in the model whenever group 5 is in the model, and that groups 1 and 2 must be in the model whenever group 3 is in the model.

```
> interactions <- list(c("group5", "group7", "group6"), c("group3",
+     "group1", "group2"))
> interactions
```

```
[[1]]
[1] "group5" "group7" "group6"

[[2]]
[1] "group3" "group1" "group2"
```

Define the options for training.

```
> op.train <- list(stepscale = 1000, total_iteration = 2e+05, warm = 20000,
+     savestep = 50, lowE = 570, maxE = 670, scale = 1, rho = 1,
+     temperature = 1, maxvar = 10, maxbeta = 10, seed = 10, gamma = 0.4,
+     model_maxn = 1e+05, delete = 0)
```

Define the options for resampling.

```
> op.resample <- list(total_iteration = 5e+05, warm = 2000, savestep = 10,
+     temperature = 1, seed = 10, model_maxn = 1e+05, delete = 0)
```

Define the options for prediction and the list of options for the BSR function.

```
> op.prediction <- list(n.burn = 100)
> op <- list(train.op = op.train, resample.op = op.resample, prediction.op = op.prediction
+     out.string = "_EXAMPLE")
```

Call the BSR function (not run due to the time it would take). **(BSR(train.data, test.data, outdir, response.var, id.var, vars0, groups=groups, interactions=interactions, op=op)**

Read in the output files created from the BSR function.

```
> f1 <- system.file("sampleData", "PREDICTION_EXAMPLE.TRAIN.txt.xls",
+     package = "BSR")
> f2 <- system.file("sampleData", "PREDICTION_EXAMPLE.TEST.txt.xls",
+     package = "BSR")
> mat.train <- read.table(f1, header = T, sep = "\t")
> mat.test <- read.table(f2, header = T, sep = "\t")
```

Compute the prediction errors.

```
> sum(abs(mat.train[, 1] - mat.train[, 3]))/nrow(mat.train)

[1] 0.3711305

> sum(abs(mat.train[, 2] - mat.train[, 3]))/nrow(mat.train)

[1] 0.371901

> sum(abs(mat.test[, 1] - mat.test[, 3]))/nrow(mat.test)

[1] 0.3870201

> sum(abs(mat.test[, 2] - mat.test[, 3]))/nrow(mat.test)

[1] 0.3866208
```