# Package ‘sodavis’

October 14, 2022

**Type** Package

**Title** SODA: Main and Interaction Effects Selection for Logistic Regression, Quadratic Discriminant and General Index Models

**Version** 1.2

**Depends** R (>= 3.0.0), nnet, MASS, mvtnorm

**Date** 2018-05-12

**Author** Yang Li, Jun S. Liu

**Maintainer** Yang Li <yangli.stat@gmail.com>

**Description** Variable and interaction selection are essential to classification in high-dimensional setting. In this package, we provide the implementation of SODA procedure, which is a forward-backward algorithm that selects both main and interaction effects under logistic regression and quadratic discriminant analysis. We also provide an extension, S-SODA, for dealing with the variable selection problem for semi-parametric models with continuous responses.

**License** GPL-2

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2018-05-13 21:24:03 UTC

## R topics documented:

## mich_lung *Gene expression data for Michigan lung cancer study in Beer et al. (2002)*

### Description

Gene expression data of 5217 genes for n = 86 subjects, with 62 subjects in "good outcomes" (class 1) and 24 subjects in "poor outcomes" (class 2), from the microarray study of Beer et al. (2002).

### Usage

```
data(mich_lung)
```

### Format

Response variable vector and design matrix on 86 observations for expression of 5217 genes.

### References

Beer et al. (1999) Gene-expression profiles predict survival of patients with lung adenocarcinoma. *Nature medicine*, 286(8): 816-824.

## pumadyn *Pumadyn dataset*

### Description

This is a dataset synthetically generated from a realistic simulation of the dynamics of a Unimation Puma 560 robot arm.

### Usage

```
data(pumadyn)
```

### Format

Response variable vector and design matrix on 4499 in-sample and 3693 out-sample observations for 32 predictor variables.

### References

Corke, P. I. (1996). A Robotics Toolbox for MATLAB. *IEEE Robotics and Automation Magazine*, 3 (1): 24-32.

## soda — *SODA algorithm for variable and interaction selection*

### Description

SODA is a forward-backward variable and interaction selection algorithm under logistic regression model with second-order terms. In the forward stage, a stepwise procedure is conducted to screen for important predictors with both main and interaction effects, and in the backward stage SODA remove insignificant terms so as to optimize the extended BIC (EBIC) criterion. SODA is applicable for variable selection for logistic regression, linear/quadratic discriminant analysis and other discriminant analysis with generative model being in exponential family.

### Usage

```
soda(xx, yy, norm = F, debug = F, gam = 0, minF = 3)
```

### Arguments

| | |
|---|---|
| `xx` | The design matrix, of dimensions n * p, without an intercept. Each row is an observation vector. |
| `yy` | The response vector of dimension n * 1. |
| `norm` | Logical flag for xx variable quantile normalization to standard normal, prior to performing SODA algorithm. Default is norm=FALSE. Quantile-normalization is suggested if the data contains obvious outliers. |
| `debug` | Logical flag for printing debug information. |
| `gam` | Tuning paramter gamma in extended BIC criterion.<br>EBIC for selected set S:<br>EBIC = -2 * log-likelihood + \|S\| * log(n) + 2 * \|S\| * gamma * log(p) |
| `minF` | Minimum number of steps in forward interaction screening. Default is minF=3. |

### Value

| | |
|---|---|
| `EBIC` | Trace of extended Bayesian information criterion (EBIC) score. |
| `Type` | Trace of step type ("Forward (Main)", "Forward (Int)", "Backward"). |
| `Var` | Trace of selected variables. |
| `Term` | Trace of selected main and interaction terms. |
| `final_EBIC` | Final selected term set EBIC score. |
| `final_Var` | Final selected variables. |
| `final_Term` | Final selected main and interaction terms. |

### Author(s)

Yang Li, Jun S. Liu

### References

Li Y, Liu JS. (2017). Robust Variable and Interaction Selection for Logistic Regression and Multiple Index Models. *Technical Report*.

### Examples

```
# # (uncomment the code to run)
# # simulation study with 1 main effect and 2 interactions
# N = 250;
# p = 1000;
# r = 0.5;
# s = 1;
# H = abs(outer(1:p, 1:p, "-"))
# S = s * r^H;
# S[cbind(1:p, 1:p)] = S[cbind(1:p, 1:p)] * s

# xx = as.matrix(data.frame(mvrnorm(N, rep(0,p), S)));
# zz = 1 + xx[,1] - xx[,10]^2 + xx[,10]*xx[,20];
# yy = as.numeric(runif(N) < exp(zz) / (1+exp(zz)))

# res_SODA = soda(xx, yy, gam=0.5);
# cv_SODA  = soda_trace_CV(xx, yy, res_SODA)
# cv_SODA

# # Michigan lung cancer dataset
# data(mich_lung);
# res_SODA = soda(mich_lung_xx, mich_lung_yy, gam=0.5);
# cv_SODA  = soda_trace_CV(mich_lung_xx, mich_lung_yy, res_SODA)
# cv_SODA
```

---

## soda_trace_CV    *Calculate a trace of cross-validation error rate for SODA forward-backward procedure*

---

### Description

This function takes a SODA result variable as input, and calculates the cross-validation error for each step of the SODA procedure.

### Usage

```
soda_trace_CV(xx, yy, res_SODA)
```

### Arguments

`xx` The design matrix, of dimensions n * p, without an intercept. Each row is an observation vector.

`yy` The response vector of dimension n * 1.

`res_SODA` SODA result varaible. See example below.

### Author(s)

Yang Li, Jun S. Liu

### Examples

```
# Michigan lung cancer dataset (uncomment the code to run)
#data(mich_lung);
#res_SODA = soda(mich_lung_xx, mich_lung_yy, gam=0.5);
#cv_SODA  = soda_trace_CV(mich_lung_xx, mich_lung_yy, res_SODA)
#cv_SODA
```

## `s_soda` *S-SODA algorithm for general index model variable selection*

### Description

S-SODA is an extension of SODA to conduct variable selection for general index models with continuous response. S-SODA first evenly discretizes the continuous response into H slices, and then apply SODA on the discretized response. Compared with existing variable selection methods based on the Sliced Inverse Regression (SIR), SODA requires neither the linearity nor the constant variance condition and is much more robust.

### Usage

```
s_soda(x, y, H = 5, gam = 0, minF = 3, norm = F, debug = F)
```

### Arguments

`x` The design matrix, of dimensions n * p, without an intercept. Each row is an observation vector.

`y` The response vector of dimension n * 1.

`H` The number of slices.

`gam` EBIC penalization coefficient parameter for SODA.

`minF` Minimum number of steps in forward interaction screening. Default is minF=3.

`norm` If set as True, S-SODA first marginally quantile-normalize each predictor to the standard normal distribution.

`debug` If print debug information.

### Value

`BIC` Trace of extended Bayesian information criterion (EBIC) score.

`Var` Trace of selected variables.

`Term` Trace of selected main and interaction terms.

`best_BIC` Final selected term set EBIC score.

`best_Var` Final selected variables.

`best_Term` Final selected main and interaction terms.

## Examples

```
# # (uncomment the code to run)
# # Simulation:  x1 / (1 + x2^2) example
# N = 500
# x1 = runif(N, -3, +3)
# x2 = runif(N, -3, +3)
# x3 = x1 / exp(x2^2) + rnorm(N, 0, 0.2)
# ss = s_soda_model(cbind(x1,x2), x3, H=25)
#
# # true surface in grid
# MM = 50
# xx1 = seq(-3, +3, length.out = MM)
# xx2 = seq(-3, +3, length.out = MM)
# yyy = matrix(0, MM, MM)
# for(i in 1:MM)
#   for(j in 1:MM)
#     yyy[i,j] = xx1[i] / exp(xx2[j]^2)
#
# # predicted surface
# ppp = s_soda_pred_grid(xx1, xx2, ss, po=1)
#
# par(mfrow=c(1, 2), mar=c(1.75, 3, 1.25, 1.5))
# persp(xx1, xx2, yyy, theta=-45, xlab="X1", ylab="X2", zlab="Y")
# persp(xx1, xx2, ppp, theta=-45, xlab="X1", ylab="X2", zlab="Pred")
#
# # Pumadyn dataset
# #data(pumadyn);
# #s_soda(pumadyn_isample_x, pumadyn_isample_y, H=25, gam=0)
```

# s_soda_model *S-SODA model estimation.*

## Description

S-SODA assumes within each slice the X vector follow multivariate normal distribution. This function estimates the mean vector and covariance matrix of X for each slice.

## Usage

```
s_soda_model(x, y, H = 10)
```

## Arguments

| | |
|---|---|
| x | The design matrix, of dimensions n * p, without an intercept. Each row is an observation vector. |
| y | The response vector of dimension n * 1. |
| H | The number of slices. |

### Value

| | |
|---|---|
| int_h | Slice index. |
| int_p | Proportion of samples in each slice. |
| int_l | Length of each slice (max - min response). |
| int_m | Mean vector of covariates in each slice. |
| int_v | Covariance matrix of covariates in each slice. |

## s_soda_pred *Predict the response y using S-SODA model.*

### Description

S-SODA assumes within each slice the X vector follow multivariate normal distribution. This function predicts the response y by reverting the P(X | slice(y)) to P(slice(y) | X), and estimates the E(y|X) as sum_h E(y | slice(y)=h, X) P (slice(y)=h | X)

### Usage

```
s_soda_pred(x, model, po = 1)
```

### Arguments

x The design matrix, of dimensions n * p, without an intercept. Each row is an observation vector.

model S-SODA model estimated from s_soda_model function.

po Order of terms in X to approximate E(y | slice(y)=h, X). If po=0, E(y | slice(y)=h, X) is the mean of y in slice h. If po=1, E(y | slice(y)=h, X) is the linear regression of X to predict y in slice h. If po=2, the linear regression also include 2nd order terms of X.

### Value

Predicted response.

s_soda_pred_grid *Predict the response y using S-SODA model in a 2-dimensional grid.*

## Description

Calls function s_soda_pred in a 2-dimensional grid defined by x1 and x2.

## Usage

```
s_soda_pred_grid(xx1, xx2, model, po = 1)
```

## Arguments

| | |
|---|---|
| `xx1` | Grid breakpoints for predictor 1. |
| `xx2` | Grid breakpoints for predictor 2. |
| `model` | S-SODA model estimated from s_soda_model. |
| `po` | Order of terms in X to approximate E(y \| slice(y)=h, X). |

## Value

Predicted response.

# Index