Package ‘s2net’

January 16, 2020

Type Package
Title The Generalized Semi-Supervised Elastic-Net
Version 1.0.1
Date 2020-01-08
Description Implements the generalized semi-supervised elastic-net. This method extends the supervised elastic-net problem, and thus it is a practical solution to the problem of feature selection in semi-supervised contexts. Its mathematical formulation is presented from a general perspective, covering a wide range of models. We focus on linear and logistic responses, but the implementation could be easily extended to other losses in generalized linear models. We develop a flexible and fast implementation, written in 'C++' using 'RcppArmadillo' and integrated into R via 'Rcpp' modules. See Culp, M. 2013 <doi:10.1080/10618600.2012.657139> for references on the Joint Trained Elastic-Net.
License GPL (>= 2)
Imports Rcpp, methods, MASS
Depends stats
LinkingTo Rcpp, RcppArmadillo
Suggests knitr, rmarkdown, glmnet, Metrics, testthat
VignetteBuilder knitr

URL https://github.com/jlaria/s2net

BugReports https://github.com/jlaria/s2net/issues
Encoding UTF-8

NeedsCompilation yes
Author Juan C. Laria [aut, cre] (<https://orcid.org/0000-0001-7734-9647>), Line H. Clemmensen [aut]
Maintainer Juan C. Laria <juank.laria@gmail.com>
Repository CRAN
Date/Publication 2020-01-16 13:20:06 UTC
s2net-package

R topics documented:

s2net-package ................................................................. 2
auto_mpg ................................................................. 4
predict.s2netR ............................................................. 5
predict_Rcpp_s2net ......................................................... 6
print.s2Data ............................................................... 7
Rcpp_s2net-class ........................................................... 8
s2Data ................................................................. 9
s2Fista ................................................................. 11
s2netR ................................................................. 12
s2Params ................................................................. 13
simulate_extra ............................................................ 14
simulate_groups .......................................................... 15

Index 16

s2net-package  The Generalized Semi-Supervised Elastic-Net

Description

Implements the generalized semi-supervised elastic-net. This method extends the supervised elastic-net problem, and thus it is a practical solution to the problem of feature selection in semi-supervised contexts. Its mathematical formulation is presented from a general perspective, covering a wide range of models. We focus on linear and logistic responses, but the implementation could be easily extended to other losses in generalized linear models. We develop a flexible and fast implementation, written in 'C++' using 'RcppArmadillo' and integrated into R via 'Rcpp' modules. See Culp, M. 2013 <doi:10.1080/10618600.2012.657139> for references on the Joint Trained Elastic-Net.

Details

The DESCRIPTION file:

Package: s2net
Type: Package
Title: The Generalized Semi-Supervised Elastic-Net
Version: 1.0.1
Date: 2020-01-08
Authors@R: c(person("Juan C.", "Laria", role = c("aut", "cre"), email = "juank.laria@gmail.com", comment = c(ORCID = "0000-0001-7734-9647"),)
Description: Implements the generalized semi-supervised elastic-net. This method extends the supervised elastic-net problem, and thus it is a practical solution to the problem of feature selection in semi-supervised contexts. Its mathematical formulation is presented from a general perspective, covering a wide range of models. We focus on linear and logistic responses, but the implementation could be easily extended to other losses in generalized linear models. We develop a flexible and fast implementation, written in 'C++' using 'RcppArmadillo' and integrated into R via 'Rcpp' modules. See Culp, M. 2013 <doi:10.1080/10618600.2012.657139> for references on the Joint Trained Elastic-Net.

License: GPL (>= 2)
Imports: Rcpp, methods, MASS
Depends: stats
LinkingTo: Rcpp, RcppArmadillo
Suggests: knitr, rmarkdown, glmnet, Metrics, testthat
VignetteBuilder: knitr
URL: https://github.com/jlaria/s2net
Index of help topics:

- Rcpp_s2net-class: Class 's2net'
- auto_mpg: Auto MPG Data Set
- predict.s2netR: S3 Methods for 's2netR' objects.
- predict_Rcpp_s2net: Predict method for 's2net' C++ class.
- print.s2Data: Print methods for S3 objects
- s2Data: Data wrapper for 's2net'.
- s2Fista: Hyper-parameter wrapper for FISTA.
- s2Params: Hyper-parameter wrapper for 's2net'
- s2net: The Generalized Semi-Supervised Elastic-Net
- s2netR: Trains a generalized extended linear joint trained model using semi-supervised data.
- simulate_extra: Simulate extrapolated data
- simulate_groups: Simulate data (two groups design)

This package includes a very easy-to-use interface for handling data, with the s2Data function. The main function of the package is the s2netR function, which is a wrapper for the Rcpp_s2net (s2net) class.

Author(s)

NA

References


See Also

s2Data, s2netR, Rcpp_s2net

Examples

data("auto_mpg")
train = s2Data(xL = auto_mpg$P1$xL, yL = auto_mpg$P1$yL, xU = auto_mpg$P1$xU)
model = s2netR(train,
       s2Params(lambda1 = 0.1,
$\lambda_2 = 0,$
$\gamma_1 = 0.1,$
$\gamma_2 = 100,$
$\gamma_3 = 0.1))$

# here we tell it to transform the valid data as we did with train.
valid = s2Data(auto_mpg$P1$xU, auto_mpg$P1$yU, preprocess = train)
ypred = predict(model, valid$xL)

## Not run:
if(require(ggplot2)){
  ggplot() +
  aes(x = ypred, y = valid$yL) + geom_point() +
  geom_abline(intercept = 0, slope = 1, linetype = 2)
}
## End(Not run)

---

auto_mpg

Auto MPG Data Set

Description

This dataset was taken from the UCI Machine Learning Repository [https://archive.ics.uci.edu/ml/datasets/Auto+MPG](https://archive.ics.uci.edu/ml/datasets/Auto+MPG), and processed for the semi-supervised setting (Ryan and Culp, 2015).

Usage

data("auto_mpg")

Format

There are two lists that contain partitions from a data frame with 398 observations on the following 9 variables.

- mpg  a numeric vector
- cylinders an ordered factor with levels 3 < 4 < 5 < 6 < 8
- displacement a numeric vector
- horsepower  a numeric vector
- weight  a numeric vector
- acceleration a numeric vector
- year  a numeric vector
- origin  a factor
Details

This dataset is a slightly modified version of the dataset provided in the StatLib library. In line with the use by Ross Quinlan (1993) in predicting the attribute "mpg", 8 of the original instances were removed because they had unknown values for the "mpg" attribute. "The data concerns city-cycle fuel consumption in miles per gallon, to be predicted in terms of 3 multivalued discrete and 5 continuous attributes." (Quinlan, 1993)

Source


References


Examples

data(auto_mpg)
head(auto_mpg$P1$xL)

predict.s2netR  

S3 Methods for s2netR objects.

Description

Generic predict method. Wrapper for the C++ class method s2net$predict.

Usage

```r
## S3 method for class 's2netR'
predict(object, newX, type = "default", ...)
```

Arguments

- `object` A s2netR object
- `newX` A matrix with the data to make predictions. It should be in the same scale as the original data. See `s2Data` to see how to format the data.
- `type` Type of predictions. One of "default" (figure it out from the train data), "response", "probs", "class".

...
predict_Rcpp_s2net

### Value
A column matrix with predictions.

### See Also
`s2netR`, `s2net`

### Examples
```r
data("auto_mpg")
train = s2Data(xL = auto_mpg$P1$xL, yL = auto_mpg$P1$yL, xU = auto_mpg$P1$xU)

model = s2netR(train,
                 s2Params(lambda1 = 0.1,
                          lambda2 = 0,
                          gamma1 = 0.1,
                          gamma2 = 100,
                          gamma3 = 0.1),
                 loss = "linear",
                 frame = "ExtJT",
                 proj = "auto",
                 fista = s2Fista(5000, 1e-7, 1, 0.8))

valid = s2Data(auto_mpg$P1$xU, auto_mpg$P1$yU, preprocess = train)
ypred = predict(model, valid$xL)

## Not run:
if(require(ggplot2)){
  ggplot() +
  aes(x = ypred, y = valid$yL) + geom_point() +
  geom_abline(intercept = 0, slope = 1, linetype = 2)
}

## End(Not run)
```

---

**predict_Rcpp_s2net**

*Predict method for s2net C++ class.*

### Description
This function provides an interface in R for the method `predict` in C++ class `s2net`.

### Usage
```r
predict_Rcpp_s2net(object, newX, type = "default")
```
Arguments

- **object**: An object of class \texttt{Rcpp_s2net}.
- **newX**: Data to make predictions. Could be a \texttt{s2Data} object (field \texttt{xL} is used) or a matrix (in the same space as the original data where the model was fitted).
- **type**: Type of predictions. One of "default": let the method figure it out; "response": the linear predictor; "probs": fitted probabilities; class: fitted class.

Details

This method is included as a high-level wrapper of \texttt{object$predict}().

Value

Returns a column matrix with the same number of rows/observations as \texttt{newX}.

Author(s)

Juan C. Laria

See Also

\texttt{Rcpp_s2net}

\begin{verbatim}
print.s2Data          Print methods for S3 objects
\end{verbatim}

Description

Very simple print methods to show basic information about these simple S3 objects.

Usage

```r
## S3 method for class 's2Data'
print(x, ...)
## S3 method for class 's2Fista'
print(x, ...)
```

Arguments

- **x**: S3 object of class \texttt{s2Data} or \texttt{s2Fista}

See Also

\texttt{s2Data}
Rcpp_s2net-class

Description

This is the main class of this library, implemented in C++ and exposed to R using Rcpp modules. It can be used in R directly, although some generic S4 methods have been implemented to make it easier to interact in R.

Methods

**predict** signature(object = "Rcpp_s2net"): See predict_Rcpp_s2net

Fields

- beta: Object of class matrix. The fitted model coefficients.
- intercept: The model intercept.

Class-Based Methods

- **initialize**(data, loss): data s2Data object
  - loss Loss function: 0 = linear, 1 = logit
- **setupFista** (s2Fista): Configures the FISTA internal algorithm.
- **predict**(newX, type): newX New data matrix to make predictions.
  - type 0 = default, 1 = response, 2 = probs, 3 = class
- **fit**(params, frame, proj): params s2Params object
  - frame 0 = "JT", 1 = "ExtJT"
  - proj 0 = no, 1 = yes, 2 = auto

Author(s)

Juan C. Laria

Examples

```r
data("auto_mpg")
train = s2Data(xL = auto_mpg$P1$xL, yL = auto_mpg$P1$yL, xU = auto_mpg$P1$xU)

# We create the C++ object calling the new method (constructor)
obj = new(s2net, train, 0) # 0 = regression
obj

# We call directly the $fit method of obj,
obj$fit(s2Params(lambda1 = 0.01, lambda2 = 0.01, gamma1 = 0.05, gamma2 = 100,
```


```r
# fitted model
obj$beta

# We can test the results using the unlabeled data
test = s2Data(xL = auto_mpg$P1$xU, yL = auto_mpg$P1$yU, preprocess = train)
ypred = obj$predict(test$xL, 0)

## Not run:
if(require(ggplot2)){
  ggplot() +
  aes(x = ypred, y = test$yL) + geom_point() +
  geom_abline(intercept = 0, slope = 1, linetype = 2)
}
## End(Not run)
```

---

**s2Data**  
*Data wrapper for s2net.*

**Description**

This function preprocess the data to fit a semi-supervised linear joint trained model.

**Usage**

```r
s2Data(xL, yL, xU = NULL, preprocess = T)
```

**Arguments**

- **xL**: The labeled data. Could be a matrix or data.frame.
- **yL**: The labels associated with xL. Could be a vector, matrix or data.frame, of factor or numeric types.
- **xU**: The unlabeled data (optional). Could be a matrix or data.frame.
- **preprocess**: Should the input data be pre-processed? Possible values are:
  - TRUE (default) The data is converted to a matrix. Factor variables are automatically coded using `model.matrix`. The data is scaled, and constant columns are removed.
  - FALSE Do nothing. Keep in mind that the theoretical framework assumes that xL is centered. Unless you are absolutely sure, avoid this.

Another object of class s2Data that was obtained from similar data (same original variables). This is useful when using train/validation sets, to apply the validation data the same transformation as train data.
Value

Returns an object of S3 class s2Data with fields

- **xL**: Transformed labeled data
- **yL**: Transformed labels. If `yL` was a factor, it is converted to numeric, and the base category is kept in `base`
- **xU**: Transformed unlabeled data
- **type**: Type of task. This one is inferred from the response labels.
- **base**: Base category for classification (0 = base)

In addition the following attributes are stored.

- **pr:rm.cols**: logical vector of removed columns
- **pr:center**: column center
- **pr:scale**: column scale
- **pr:ycenter**: `yL` center. Regression
- **pr:y.scale**: `yL` scale. Regression

Author(s)

Juan C. Laria

See Also

- `s2Fista`

Examples

data("auto_mpg")

```r
train = s2Data( xL = auto_mpg$P1$xL,
               yL = auto_mpg$P1$yL,
               xU = auto_mpg$P1$xU,
               preprocess = TRUE )

show(train)

# Notice how ordered factor variable $cylinders is handled
# .L (linear) .Q (quadratic) .C (cubic) and .^4
head(train$xL)

# if you want to do validation with the unlabeled data
idx = sample(length(auto_mpg$P1$yU), 200)

train = s2Data(xL = auto_mpg$P1$xL, yL = auto_mpg$P1$yL, xU = auto_mpg$P1$xU[idx, ],
               preprocess = TRUE )

valid = s2Data(xL = auto_mpg$P1$xU[-idx, ], yL = auto_mpg$P1$yU[-idx], preprocess = train)
```
test = s2Data(xL = auto_mpg$xU[idx,], yL = auto_mpg$yU[idx], preprocess = train)

train
valid
test

---

**s2Fista**

*Hyper-parameter wrapper for FISTA.*

**Description**

This is a very simple function that supplies the hyper-parameters for the Fast Iterative Soft-Threshold Algorithm (FISTA) that solves the s2net minimization problem.

**Usage**

```r
s2Fista(MAX_ITER_INNER = 5000, TOL = 1e-07, t0 = 2, step = 0.1, use_warmstart = FALSE)
```

**Arguments**

- **MAX_ITER_INNER**: Number of iterations of FISTA
- **TOL**: The relative tolerance. The algorithm stops when the objective does not improve more than TOL*the null model’s objective function evaluation, after two successive iterations.
- **t0**: The initial stepsize for backtracking.
- **step**: The scale factor in the stepsize to backtrack until a valid step is found.
- **use_warmstart**: Should we use a warm beta to fit the model? This is useful to speed-up hyper-parameter searching methods.

**Value**

Returns an object of S3 class `s2Fista` with the input arguments as fields.

**References**


**See Also**

`s2Params, s2Data`
s2netR

Trains a generalized extended linear joint trained model using semi-supervised data.

Description

This function is a wrapper for the class s2net. It creates the C++ object and fits the model using input data.

Usage

s2netR(data, params, loss = "default", frame = "ExtJT", proj = "auto", fista = NULL, S3 = TRUE)

Arguments

data A s2Data object with the (training) data.
params A s2Params object with the model hyper-parameters.
loss Loss function. One of "default" (figure it out from the data), "linear" or "logit".
frame The semi-supervised frame: "ExtJT" (the extended linear joint trained model), "JT" (the linear joint trained model from Ryan and Culp. 2015)
proj Should the unlabeled data be shifted to remove the model’s effect? One of "no","yes","auto" (option auto shifts the unlabeled data if the angle between beta and the center of the data is important)
fista Fista setup parameters. An object of class s2Fista.
S3 Boolean: should the method return an S3 object (default) or a C++ object?

Value

Returns an object of S3 class s2netR or a C++ object of class s2net

Author(s)

Juan C. Laria

References


See Also

s2net
s2Params

Hyper-parameter wrapper for s2net

Description
This is a very simple function that collapses the input parameters into a named vector to supply to C++ methods.

Usage
s2Params(lambda1, lambda2 = 0, gamma1 = 0, gamma2 = 0, gamma3 = 0)

Arguments
lambda1 elastic-net regularization parameter - $l_1$ norm.
lambda2 elastic-net regularization parameter - $l_2$ norm.
gamma1 s2net weight hyper-parameter.
gamma2 s2net covariance hyper-parameter (between 1 and Inf).
gamma3 s2net shift hyper-parameter (between 0 and 1).

Examples
data("auto_mpg")
train = s2Data(xL = auto_mpg$P1$xL, yL = auto_mpg$P1$yL, xU = auto_mpg$P1$xU)

model = s2netR(train,
s2Params(lambda1 = 0.1,
lambda2 = 0,
gamma1 = 0.1,
gamma2 = 100,
gamma3 = 0.1),
loss = "linear",
frame = "ExtJT",
proj = "auto",
fista = s2Fista(5000, 1e-7, 1, 0.8))

valid = s2Data(auto_mpg$P1$xU, auto_mpg$P1$yU, preprocess = train)
ypred = predict(model, valid$xL)

## Not run:
if(require(ggplot2)){
  ggplot() +
  aes(x = ypred, y = valid$yL) + geom_point() +
  geom_abline(intercept = 0, slope = 1, linetype = 2)
}

## End(Not run)
Value
Returns a named vector of S3 class s2Params.

See Also
s2Data, s2Fista

simulate_extra
Simulate extrapolated data

Description
Simulated data scenarios described in the paper from Ryan and Culp (2015).

Usage
simulate_extra(n_source = 100, n_target = 100, p = 1000, shift = 10, scenario = "same", response = "linear", sigma2 = 2.5)

Arguments
n_source Number of source samples (labeled)
n_target Number of target samples (unlabeled)
p Number of variables (p > 10)
shift The shift applied to the first 10 columns of xU.
scenario Simulation scenario. One of "same" (same distribution), "lucky" (extrapolation with lucky $\beta$), "unlucky" (extrapolation with unlucky $\beta$)
response Type of response: "linear" or "logit"
sigma2 The variance of the error term, linear response case.

Value
A list, with

- xL data frame with the labeled (source) data
- yL labels associated with xL
- xU data frame with the unlabeled (target) data
- yU labels associated with xU (for validation/testing)

References
simulate_groups

See Also

simulate_groups

Examples

```r
set.seed(0)
data = simulate_extra()

train = s2Data(data$xL, data$yL, data$xU)
valid = s2Data(data$xU, data$yU, preprocess = train)

model = s2netR(train, s2Params(0.1))
ypred = predict(model, valid$xL)
plot(ypred, valid$yL)
```

simulate_groups  Simulate data (two groups design)

Description

Simulated data scenario described in paper [citation here].

Usage

```r
simulate_groups(n_source = 100, n_target = 100, p = 200, response = "linear")
```

Arguments

- `n_source`: Number of labeled observations
- `n_target`: Number of unlabeled (target) observations
- `p`: Number of variables
- `response`: Type of response: "linear" or "logit"

Value

A list, with

- `xL`: data frame with the labeled (source) data
- `yL`: labels associated with xL
- `xU`: data frame with the unlabeled (target) data
- `yU`: labels associated with xU (for validation/testing)

Author(s)

Juan C. Laria

See Also

simulate_extra
# Index

*Topic **datasets**
  - auto_mpg, 4

*Topic **manip**
  - s2Data, 9

*Topic **methods**
  - predict.s2netR, 5
  - print.s2Data, 7

*Topic **models**
  - Rcpp_s2net-class, 8

*Topic **optimize**
  - Rcpp_s2net-class, 8
  - s2Fista, 11

*Topic **package**
  - s2net-package, 2

*Topic **regression**
  - Rcpp_s2net-class, 8
  - _rcpp_module_boot_Rcpp_s2net_export
    - (Rcpp_s2net-class), 8

- auto_mpg, 4

- model.matrix, 9

- predict,Rcpp_s2net-method
  - (Rcpp_s2net-class), 8
  - predict.s2netR, 5
  - predict_Rcpp_s2net, 6, 8
  - print.s2Data, 7
  - print.s2Fista (print.s2Data), 7

- Rcpp_s2net, 3, 7
- Rcpp_s2net (Rcpp_s2net-class), 8
- Rcpp_s2net-class, 8

- s2Data, 3, 5, 7, 8, 9, 11, 12, 14
- s2Fista, 8, 10, 11, 12, 14
- s2net, 6, 12
- s2net (s2net-package), 2
- s2net-package, 2
- s2netR, 3, 5, 6, 12
- s2Params, 8, 11, 12, 13

- simulate_extra, 14, 15
- simulate_groups, 15, 15