Package ‘lpRegPath’

September 8, 2008

Type Package

Title lpRegPath

Version 2.0

Date 2008-09-07

Author Yonggang Yao, Department of Statistics, The Ohio State University

Maintainer Yonggang Yao <yao.53@osu.edu>

Depends R (>= 2.5.0)

Description Parametric Linear Programming Package for Regularization Methods

License GPL (>= 2)

R topics documented:

SolPath ......................................................... 2
fit.qr.L1 ...................................................... 3
fit.svm.L1 ..................................................... 5
Package ......................................................... 7
path.Dantzig .................................................. 7
path.qr.GL1 ................................................... 9
path.qr.L1 ..................................................... 11
path.svm.L1 .................................................. 12
draw.num.select ............................................. 14
plot.path ...................................................... 15
plot.risk ...................................................... 17
predict.qr.L1 ............................................... 19
predict.svm.L1 ............................................... 20
risk.qr.L1 .................................................... 21
risk.svm.L1 .................................................. 23

Index 25
SolPath  
An Internal Function to Generating Solution Path

Description

SolPath is an internal function used to generate solution paths for a class of parametric Linear Programming (LP) problems, which is the core function of package lpRegPath. All regularization methods in this package are recast as special cases of the LP problems and call SolPath to compute their solution paths.

Usage

```r
SolPath(b, c, A=NULL, a=NULL, cp=NULL, Ap=NULL, ap=NULL,
        no.repeat=TRUE, precision=1e-07, no.display=TRUE)
```

Arguments

- `no.repeat`: a logical value. If no.repeat is true, the repeated joint solution will not be stored.
- `precision`: a numeric value identifying the computation precision. Any number smaller than precision will be treated as zero.
- `no.display`: a logical value identifying whether the iteration information will be visible.
- `b, c, A, a, cp, Ap`:
  See the following standard form of parametric linear programming.

Details

The function SolPath generates the solution path for the following family of parametric LP problems with respect to $z$:

\[
\begin{align*}
\min & \quad [(c, cp) + \lambda (a, ap)]'z \\
\text{subject to} & \quad Hz = b \\
& \quad z \geq 0 \\
\end{align*}
\]

or

\[
\begin{align*}
\min & \quad (c, cp)'z \\
\text{subject to} & \quad Hz = b \\
& \quad (a, ap)'z \leq s \\
& \quad z \geq 0,
\end{align*}
\]

where $H := [I, A, -I, -A, Ap]$. The two family of parametric LP problems are equivalent, and their solution can be characterized with the same sequency of the joint solutions.

Value

The output of function SolPath is a list containing the solution-path information of the parametric-cost LP problem.

- `z_slt`: a matrix containing the active elements of joint solutions. Each row of the matrix is corresponding to a joint solution vector.
- `idxB_all`: a matrix containing the indices of the active elements corresponding to the joint solutions. Its last three columns are respectively the number of iterations, lambda, and s values.
- `lambda`: a numerical vector containing the lambda values corresponding to joint solutions.
- `s_penalty`: a numerical vector containing the s values corresponding to joint solutions.
Warning

Due to the limited computational precision, the solution path might not be correctly calculated.

Author(s)

Yonggang Yao, Department of Statistics, The Ohio State University. (yao.53@osu.edu)

References


See Also

calibrate.svm.L1, path.qr.L1

Examples

```
# Simulate the training dataset.
set.seed(80)
require(lpRegPath)
n=200; p=10; sd=sqrt(50);
beta=c(2,0,2,0,2,0,0,0,0,2);
x=array(rnorm(p*n),c(n,p));
y=as.vector(sign(x%*%beta+rnorm(n,0,sd)));

# Configure the parametric LP problem.
A=rbind(y,t(x)%*%diag(c(y)));
b = rep(1,n);
c=rbind(c(rep(1,n)/n,rep(0,p+1)),rep(0,(n+p+1)));
a=rbind(c(rep(0,n+1),rep(1,p)),c(rep(0,n+1),rep(1,p)));
cp=NULL; Ap=NULL; ap=NULL;

# Calculate and plot the solution path.
sp=SolPath(b, c, A=A, a=a);
names(sp)
```

Description

fit.qr.L1 searches the optimal models by using L1-norm Quantile Regression method. It calls path.qr.L1 and risk.qr.L1 to generate the solution path and search the optimal tuning parameter (or penalty bound), and picks the optimal models in the solution path.
Usage

```r
## S3 method for class 'qr.L1':
fit(y, x, tau=.5, intercept=TRUE, fold=4, repetition=2,
    w=rep(1,length(y)), new.window=FALSE, precision=1e-8)
```

Arguments

- `y`: a vector of class labels in training dataset. Elements in `y` must be either 1 or -1.
- `x`: a matrix containing covariates in training dataset.
- `tau`: a numeric value specifying the quantile parameter.
- `w`: a weight vector for observation pairs in training dataset in calculating empirical risk. Default `w` is an one-vector of the same size as `y`, such that all the observations are equally weighted.
- `intercept`: a logical value indicating whether the intercept term is included in the classifiers. Default `intercept` is true, such that the classifiers take the form of \( y = \text{sign} \{ \beta_0 + x^* \beta \} \). Otherwise, \( y = \text{sign} \{ x^* \beta \} \).
- `fold`: an integer specifying the fold of the CV method.
- `repetition`: an integer specifying the number of repetition for the CV method.
- `new.window`: a logical value indicating whether a new graphical window should be launched.
- `precision`: an internal control parameter.

Value

The output of function `fit.qr.L1` is a list containing the information about the selected optimal classifiers.

- `opt.model`: a matrix with two rows. The first row contains the coefficients of the optimal classifier in penalty bound, and the second row contains the coefficients of the optimal classifier in tuning parameter. The classifiers are optimal in terms of check loss.
- `sp`: a list containing information about the associated solution path. See `path.qr.L1` for more details.
- `est.risk.path`: a list containing the information of the estimated risk curves with respect to tuning parameter or penalty. See `risk.qr.L1` for more details.

Author(s)

Yonggang Yao, Department of Statistics, The Ohio State University. (yao.53@osu.edu).

References


See Also

path.qr.L1, plot.path, risk.qr.L1, plot.risk.

Examples

```r
############################
# Simulate the training dataset.
set.seed(980)
require(lpRegPath)
n=400; p=10; sd=sqrt(50);
beta=c(2,0,2,0,2,0,0,0,0,2);
x=array(rnorm(p*n),c(n,p));
y=as.vector(x%*%beta+rnorm(n,0,sd));
############################
# Calculate and plot the solution path.
sp=fit.qr.L1(y, x)

# S3 method
class(y)="qr.L1";
sp2=fit(y,x);
############################
```

fit.svm.L1  

**L1-norm Support Vector Machine**

fit.svm.L1 searches the optimal models by using L1-norm SVM method. It calls path.svm.L1 and risk.svm.L1 to generate the solution path and search the optimal tuning parameter (or penalty bound), and picks the optimal models in the solution path.

Usage

```r
## S3 method for class 'svm.L1':
fit(y, x, intercept=TRUE, fold=4, repetition=2,
    w=rep(1,length(y)), new.window=FALSE, precision=1e-8)
```

Arguments

- **y**: a vector of class labels in training dataset. Elements in y must be either 1 or -1.
- **x**: a matrix containing covariates in training dataset.
- **w**: a weight vector for observation pairs in training dataset in calculating empirical risk. Default w is an one-vector of the same size as y, such that all the observations are equally weighted.
- **intercept**: a logical value indicating whether the intercept term is included in the classifiers. Default intercept is true, such that the classifiers take the form of y=sign[beta0+x*beta]. Otherwise, y=sign[x*beta].
- **fold**: an integer specifying the fold of the CV method.
- **repetition**: an integer specifying the number of repetitions for the CV method.
- **new.window**: a logical value indicating whether a new graphical window should be launched.
- **precision**: an internal control parameter.
The output of function fit.svm.L1 is a list containing the information about the selected optimal classifiers.

- **opt.model**: a matrix with two rows. The first row contains the coefficients of the optimal classifier in penalty bound, and the second row contains the coefficients of the optimal classifier in tuning parameter. The classifiers are optimal in terms of 0-1 loss.

- **sp**: a list containing information of the solution path. See path.svm.L1 for more details.

- **est.risk.path**: a list containing the information of the estimated risk curves with respect to tuning parameter or penalty. See risk.svm.L1 for more details.

**Author(s)**

Yonggang Yao, Department of Statistics, The Ohio State University. (yao.53@osu.edu).

**References**


**See Also**

path.svm.L1, plot.path, risk.svm.L1, plot.risk.

**Examples**

```r
# Simulate the training dataset.
set.seed(980)
require(lpRegPath)
n=400; p=10; sd=sqrt(50);
beta=c(2,0,2,0,2,0,0,0,0,2);
x=array(rnorm(p*n),c(n,p));
y=as.vector(sign(x%*%beta+rnorm(n,0,sd)));
# Calculate and plot the solution path.
sp=fit.svm.L1(y, x)
# S3 method
class(y)="svm.L1";
sp2=fit(y,x);
```
Description

lpRegPath is an R package designed for solving a family of regularization problems that satisfy certain conditions on their loss and penalty. Instead of solving the regularization problem for a fixed value of the tuning parameter, lpRegPath implements an algorithm that generates the entire solution path as a function of the tuning parameter by utilizing simplified tableau simplex algorithm. Such a solution path offers rich information about how constrained models evolve with features and what features are persistent in the data, which are of general interest in data analysis. In addition to facilitating computation and tuning, the path-finding algorithm for feature selection can equip the data analyst with a useful tool for visualizing a path of the fitted models instead of a single one. With the aids of risk measures, such a path can portray a full spectrum of potentially good models for selection and averaging.

Details

The current version of lpRegPath provides the solution-path algorithms for L1-norm support vector machines and L1-norm quantile regression.

Version: 2.0
Date: 2008-09-07

Author(s)

Yonggang Yao, Department of Statistics, The Ohio State University. (yao.53@osu.edu).
Maintainer: Yonggang Yao

References


Description

path.Dantzig is used to generate the entire solution path for a Dantzig Regression. It can output a set of the optimal linear models as a function of the tuning parameter.

Usage

```r
## S3 method for class 'Dantzig':
path(y, x, w=rep(1,length(y)), intercept=TRUE,
     precision=1e-8)
```
Arguments

- **y**: a vector of responses in training dataset.
- **x**: a matrix containing covariates in training dataset.
- **w**: a weight vector for observation pairs in training dataset in calculating empirical risk. Default w is an one-vector, such that the weights for all observation pairs are the same.
- **intercept**: a logical value indicating whether the intercept term is included in the model. Default intercept is true, such that the classifiers take the form of $y.tau = beta_0 + x^*beta$. Otherwise, $y.tau = x^*beta$.
- **precision**: an internal control parameter.

Value

The output of function `path.qr.L1` is a list containing the solution-path information for a L1-norm QR problem. The list subsumes the following variables.

- **n_iter**: an integer denoting the number of joint solutions in the solution path.
- **coeff**: a matrix containing all the joint solutions in the solution path. Every row vector of coeff contains the coefficients for one joint solution. If intercept is True, the first column of coeff contains the estimated intercepts.
- **s_penalty**: a vector specifying the L1-norm penalty. Its entries are the L1-norms of the corresponding coefficient vectors for the joint solutions. Note that intercepts are not considered in calculating the L1-norm penalty.
- **lambda**: a vector specifying the tuning parameters. The entries of lambda are the tuning parameters of the corresponding joint solutions.
- **intercept**: a logical value that is the same as the variable intercept used in arguments.

Author(s)

Yonggang Yao, Department of Statistics, The Ohio State University. ⟨yao.53@osu.edu⟩.

References


See Also

- `plot.path`

Examples

```r
# Simulate the training dataset.
set.seed(80)
require(lpRegPath)
n=400; p=10; sd=sqrt(50);
beta=c(2,3,2,0,0,0,0,-3,2,-2);
```
### path.qr.GL1

Solution Path for Quantile Regression with Grouped Variables

#### Description

path.qr.GL1 is used to generate the entire solution path for a Quantile Regression (QR) problem with grouped variables. It can output a set of the optimal QR linear models as a function of the tuning parameter.

#### Usage

```r
## S3 method for class 'qr.GL1':
path(y, x, g.idx=(1:dim(x)[2]), tau=0.5, w=rep(1,length(y)),
     intercept=TRUE, precision=1e-08)
```

#### Arguments

- `y` a vector of responses in training dataset.
- `x` a matrix containing covariates in training dataset.
- `g.idx` a integer vector specifying the group label for each covariates in x.
- `tau` a number between 0 and 1 specifying the quantile parameter.
- `w` a weight vector for observation pairs in training dataset in calculating empirical risk. Default w is an one-vector, such that the weights for all observation pairs are the same.
- `intercept` a logical value indicating whether the intercept term is included in the model. Default intercept is true, such that the classifiers take the form of y.tau = beta0 + x*beta. Otherwise, y.tau = x*beta.
- `precision` an internal control parameter.
path.qr.GL1

Value

The output of function path.qr.L1 is a list containing the solution-path information for a L1-norm QR problem. The list subsumes the following variables.

- **n_iter**: an integer denoting the number of joint solutions in the solution path.
- **coeff**: a matrix containing all the joint solutions in the solution path. Every row vector of coeff contains the coefficients for one joint solution. If intercept is True, the first column of coeff contains the estimated intercepts.
- **s_penalty**: a vector specifying the L1-norm penalty. Its entries are the L1-norms of the corresponding coefficient vectors for the joint solutions. Note that intercepts are not considered in calculating the L1-norm penalty.
- **lambda**: a vector specifying the tuning parameters. The entries of lambda are the tuning parameters of the corresponding joint solutions.
- **intercept**: a logical value that is the same as the variable intercept used in arguments.

Author(s)

Yonggang Yao, Department of Statistics, The Ohio State University. ⟨yao.53@osu.edu⟩.

References


See Also

plot.path.risk.qr.L1.

Examples

```
# Simulate the training dataset.
set.seed(80)
require(lpRegPath)
n=400; p=10; sd=sqrt(50);
beta=c(2,3,2,0,0,0,-3,2,-2);
g.idx=c(1,1,1,2,2,2,3,3,3,3);
X.names=c('11','12','13','14','21','22','23','31','32','33');
x=array(rnorm(p*n),c(n,p));
y=as.vector(x%*%beta+rnorm(n,0,sd));
# Calculate and plot the solution path.
sp=path.qr.GL1(y,x,g.idx=g.idx);
plot.path(sp, X.names=X.names, col=g.idx+1);
# Plot the solution path for absolute coefficients.
sp$coeff=abs(sp$coeff);
plot.path(sp, X.names=X.names, col=g.idx+1);
```
path.qr.L1

Solution Path for L1-norm Quantile Regression

Description

path.qr.L1 is used to generate the entire solution path for an L1-norm Quantile Regression (QR) problem. It can output a set of the optimal QR linear models as a function of the tuning parameter.

Usage

```r
## S3 method for class 'qr.L1':
path(y, x, tau=0.5, w=rep(1,length(y)), intercept=TRUE,
    precision = 1e-08)
```

Arguments

- `y`: a vector of responses in training dataset.
- `x`: a matrix containing covariates in training dataset.
- `tau`: a number between 0 and 1 specifying the quantile parameter.
- `w`: a weight vector for observation pairs in training dataset in calculating empirical risk. Default `w` is an one-vector, such that the weights for all observation pairs are the same.
- `intercept`: a logical value indicating whether the intercept term is included in the model. Default intercept is true, such that the classifiers take the form of `y.tau = beta0 + x*beta`. Otherwise, `y.tau = x*beta`.
- `precision`: an internal control parameter.

Value

The output of function `path.qr.L1` is a list containing the solution-path information for a L1-norm QR problem. The list subsumes the following variables.

- `n_iter`: an integer denoting the number of joint solutions in the solution path.
- `coeff`: a matrix containing all the joint solutions in the solution path. Every row vector of `coeff` contains the coefficients for one joint solution. If intercept is True, the first column of `coeff` contains the estimated intercepts.
- `s_penalty`: a vector specifying the L1-norm penalty. Its entries are the L1-norms of the corresponding coefficient vectors for the joint solutions. Note that intercepts are not considered in calculating the L1-norm penalty.
- `lambda`: a vector specifying the tuning parameters. The entries of `lambda` are the tuning parameters of the corresponding joint solutions.
- `intercept`: a logical value that is the same as the variable `intercept` used in arguments.
path.svm.L1

Solution Path for L1-norm SVM

Description

path.svm.L1 is used to generate the entire solution path for an L1-norm Support Vector Machine (SVM) problem. It can output a set of the optimal linear classifiers as a function of the tuning parameter.

Usage

```r
## S3 method for class 'svm.L1':
path(y, x, w=rep(1,length(y)), intercept=TRUE, precision=1e-08)
```

Author(s)

Yonggang Yao, Department of Statistics, The Ohio State University. (yao.53@osu.edu).

References


See Also

plot.path.risk.qr.L1.

Examples

```r
#################################
# Simulate the training dataset.
set.seed(980)
require(lpRegPath)
n=400; p=10; sd=sqrt(50);
beta=c(2,0,2,0,2,0,0,0,0,2);
x=array(rnorm(p*n),c(n,p));
y=as.vector(x%*%beta+rnorm(n,0,sd));
#################################
# Calculate and plot the solution path.
sp=path.qr.L1(y, x)
plot.path(sp)
# S3 method
class(y)="qr.L1";
sp2=path(y,x);
plot(sp2);
#################################
```

`plot.path`
Arguments

y  a vector of class labels in training dataset. Elements in y must be either 1 or -1.
x  a matrix containing covariates in training dataset.
w  a weight vector for observation pairs in training dataset in calculating empirical risk. Default w is an one-vector of the same size as y, such that all the observations are equally weighted.
intercept  a logical value indicating whether the intercept term is included in the classifiers. Default intercept is true, such that the classifiers take the form of y=sign[beta0+x*%*Beta]. Otherwise, y=sign[x*%*Beta].
precision  an internal control parameter.

Value

The output of function path.svm.L1 is a list containing the solution-path information. The list subsumes the following variables.

n_iter  an integer denoting the number of joint solutions in the solution path.
coeff  a matrix containing all the joint solutions in the solution path. Every row vector of coeff contains the coefficients for one joint solution. If intercept is True, the first column of coeff contains the estimated intercepts.
s_penalty  a vector specifying the L1-norm penalty. Its entries are the L1-norms of the corresponding coefficient vectors for the joint solutions. Note that intercepts are not considered in calculating the L1-norm penalty.
lambda  a vector specifying the tuning parameters. The entries of lambda are the tuning parameters of the corresponding joint solutions.
intercept  a logical value that is the same as the variable intercept used in arguments.

Author(s)

Yonggang Yao, Department of Statistics, The Ohio State University. ⟨yao.53@osu.edu⟩.

References


See Also

plot.path, risk.svm.L1.

Examples

#################################
# Simulate the training dataset.
set.seed(980)
require(lpRegPath)
n=400; p=10; sd=sqrt(50);
draw.num.select

Empirical Risk Plotting

draw.num.select produces a plot of the number of selected variable or functional components for a
regularization problem as a function of its penalty or tuning parameter.

Description

plot.num.select produces a plot of the number of selected variable or functional components for a
regularization problem as a function of its penalty or tuning parameter.

Usage

draw.num.select(sp, plot.title=NULL, intercept=FALSE, xtype='s',
                file.name=NULL, col=NULL, lty=NULL, X.names=NULL,
                x.standardized=FALSE, vline=NULL, vline.col=NULL)

Arguments

sp a list containing the solution path information. sp is often an output of a solution-path function (e.g., path.svm.L1 for L1-norm SVM or path.qr.L1 for L1-norm quantile regression).

xtype a character specifying the x-axis of the plot. There are three options for xtype: 's', 'lam' (or 'lam.nlog'), and 'lam.inv'.

plot.title a string as the title of the plot.

intercept a logical value. If intercept is true, the intercept curve will be drawn.

file.name a string specifying the directory/name of an EPS (Encapsulated PostScript) file which saves the solution-path plot. If file.name is null, the plot will be shown in a graphic window.

col a character vector. Each element in col specifies the color of its corresponding coefficient curve in the solution-path plot. See par for the options of the elements in col.

lty a integer vector specifying the line types for the coefficient curves. See par for the options of the elements in lty.

X.names a character vector specifying the names of the coefficient curves.

x.standardized a logical value. If true, the x-axis will be standardized, such that the x-axis of the plot will be unitized.

vline, vline.col one or more numbers and colors locating some models in the solution-path plot. The default vline.col is c('green', 'blue', 'skyblue', ...).
Details

The settings for the three `xtype` of the plots:

<table>
<thead>
<tr>
<th>xtype</th>
<th>x-axis</th>
<th>type of plot</th>
<th>xlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>'s'</td>
<td>numeric</td>
<td>line</td>
<td>[|\beta|]</td>
</tr>
<tr>
<td>'lam'</td>
<td>[\log(\lambda)]</td>
<td>step</td>
<td>[\log(\lambda)]</td>
</tr>
<tr>
<td>'lam.inv'</td>
<td>[\frac{1}{\lambda}]</td>
<td>step</td>
<td>[\lambda^{-1}]</td>
</tr>
</tbody>
</table>

Author(s)

Yonggang Yao, Department of Statistics, The Ohio State University. ⟨yao.53@osu.edu⟩.

See Also

`path.svm.L1`, `risk.svm.L1`, `path.qr.L1`, `risk.qr.L1`.

Examples

```r
# L1-norm Median Regression and L1-norm SVM examples
# Simulate the training dataset
set.seed(88)
n=200; p=10; sd=sqrt(50);
beta=c(2,0,2,0,2,0,0,0,0,2);
x=array(rnorm(p*n),c(n,p));
y.qr=as.vector(x%*%beta+rnorm(n,0,sd));
y.svm=sign(y.qr);

# Calculate and plot the solution paths
sp.qr=path.qr.L1(y.qr, x)
sp.svm=path.svm.L1(y.svm, x)

par(mfrow=c(2,2))
draw.num.select(sp.qr, plot.title='L1-norm Median Regression')
draw.num.select(sp.qr, xtype='lam', x.standardized=TRUE, plot.title='L1-norm Median Regression')
draw.num.select(sp.svm,xtype='lam', plot.title='L1-norm SVM')
draw.num.select(sp.svm,xtype='s', x.standardized=TRUE, plot.title='L1-norm SVM')
```

Description

`plot.path` produces a solution-path plot for a regularization problem as a function of its penalty or tuning parameter.
Usage

```r
## S3 method for class 'path':
plot(sp, plot.title=NULL, intercept=FALSE, xtype='s',
     file.name=NULL, col=NULL, lty=NULL, X.names=NULL,
     x.standardized=FALSE, vline=NULL, vline.col=NULL)
```

Arguments

- `sp` a list containing the solution path information. sp is often an output of a solution-path function (e.g., `path.svm.L1` for L1-norm SVM or `path.qr.L1` for L1-norm quantile regression).
- `xtype` a character specifying the x-axis of the plot. There are three options for xtype: 's', 'lam' (or 'lam.nlog'), and 'lam.inv'.
- `plot.title` a string as the title of the plot.
- `intercept` a logical value. If intercept is true, the intercept curve will be drawn.
- `file.name` a string specifying the directory/name of an EPS (Encapsulated PostScript) file which saves the solution-path plot. If file.name is null, the plot will be shown in a graphic window.
- `col` a character vector. Each element in col specifies the color of its corresponding coefficient curve in the solution-path plot. See `par` for the options of the elements in col.
- `lty` a integer vector specifying the line types for the coefficient curves. See `par` for the options of the elements in lty.
- `X.names` a character vector specifying the names of the coefficient curves.
- `x.standardized` a logical value. If true, the x-axis will be standardized, such that the x-axis of the plot will be unitized.
- `vline`, `vline.col` one or more numbers and colors locating some models in the solution-path plot. The default vline.col is c('green', 'blue', 'skyblue', ...)

Details

The settings for the three xtype of the plots:

<table>
<thead>
<tr>
<th>xtype</th>
<th>x-axis</th>
<th>type of plot</th>
<th>xlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>'s'</td>
<td>numeric</td>
<td>line</td>
<td>$</td>
</tr>
<tr>
<td>'lam'</td>
<td>$-\log(\lambda)$</td>
<td>step</td>
<td>$-\log(\lambda)$</td>
</tr>
<tr>
<td>'lam.inv'</td>
<td>$\frac{1}{\lambda}$</td>
<td>step</td>
<td>$\lambda^{-1}$</td>
</tr>
</tbody>
</table>

Author(s)

Yonggang Yao, Department of Statistics, The Ohio State University. (yao.53@osu.edu).

See Also

`path.svm.L1`, `risk.svm.L1`, `path.qr.L1`, `risk.qr.L1`. 
Examples

######################################################
# L1-norm Median Regression and L1-norm SVM examples #
######################################################
# Simulate the training dataset
set.seed(88)
n=200; p=10; sd=sqrt(50);
beta=c(2,0,2,0,2,0,0,0,0,2);
x=array(rnorm(p*n),c(n,p));
y.qr=as.vector(x%*%beta+rnorm(n,0,sd));
y.svm=sign(y.qr);

# Calculate and plot the solution paths
sp.qr=path.qr.L1(y.qr, x)
sp.svm=path.svm.L1(y.svm, x)

par(mfrow=c(2,2))
plot.path(sp.qr, plot.title='L1-norm Median Regression')
plot.path(sp.qr, xtype='lam', x.standardized=TRUE, plot.title='L1-norm Median Regression')
plot.path(sp.svm,xtype='lam', plot.title='L1-norm SVM')
plot.path(sp.svm,xtype='s', x.standardized=TRUE, plot.title='L1-norm SVM')

plot.risk

Empirical Risk Plotting

Description

plot.risk produces an empirical risk plot for a regularization problem as a function of its penalty or tuning parameter.

Usage

## S3 method for class 'risk':
plot(ER.est, criteria=NULL, xtype='s', file.name=NULL,
     plot.title=NULL, x.standardized=FALSE, both=TRUE,
     x.lab=NULL, y.lab=NULL, x.range=c(0,1))

Arguments

ER.est
  a list containing the empirical risk information. ER.est is often an output of a
  risk-estimation function (e.g., risk.svm.L1 for L1-norm SVM or risk.qr.L1
  for L1-norm quantile regression).

criteria
  a string identifying on which loss functions the empirical risk curves are calculated.

xtype
  a character specifying the x-axis of the plot. There are three options for xtype:
  's', 'lam' (or 'lam.nlog'), and 'lam.inv'.

file.name
  a string specifying the directory/name of an EPS (Encapsulated PostScript) file
  which saves the empirical risk plot. If file.name is null, the plot will be shown
  in a graphic window.
plot.risk

plot.title a string as the title of the plot.

x.standardized

a logical value. If true, the x-axis will be standardized, such that the x-axis of the plot will be unitized.

both

a logical value specifying whether two empirical risk curves should be drawn in the same framework.

x.lab a string specifying the label of the x-axis.

y.lab a string specifying the label of the y-axis.

x.range a vector of two numbers between 0 and 1. x.range specifies which portion of empirical risk curves should be drawn.

Details

The settings for the three xtype of the plots:

<table>
<thead>
<tr>
<th>xtype</th>
<th>x-axis type of plot</th>
<th>xlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>'s'</td>
<td>numeric</td>
<td>$\beta$</td>
</tr>
<tr>
<td>'lam'</td>
<td>$-\log(\lambda)$</td>
<td>$-\log(\lambda)$</td>
</tr>
<tr>
<td>'lam.inv'</td>
<td>$\frac{1}{\lambda}$</td>
<td>$\lambda^{-1}$</td>
</tr>
</tbody>
</table>

Author(s)

Yonggang Yao, Department of Statistics, The Ohio State University. {yao.53@osu.edu}.

See Also

path.svm.L1, risk.svm.L1, path.qr.L1, risk.qr.L1.

Examples

```r
# Simulate the training dataset
set.seed(88)  
n=200; p=10; sd=sqrt(50);  
beta=c(2,0,2,0,2,0,0,0,0,2);  
x=array(rnorm(p*n),c(n,p));  
y.qr=as.vector(x%*%beta+rnorm(n,0,sd));  
y.svm=sign(y.qr);  
# Calculate and plot the empirical risk curves
svm.ER.cv=risk.svm.L1(y.svm, x)  
qr.ER.cv=risk.qr.L1(y.qr, x)  

par(mfrow=c(2,2))  
plot.risk(svm.ER.cv, criteria='hinge loss')  
plot.risk(svm.ER.cv, criteria='0-1 loss')  
plot.risk(qr.ER.cv, x.range=c(.38,.97), x.standardized=TRUE)  
plot.risk(qr.ER.cv, xtype='lam')
```
predict.qr.L1  Make Predictions for New data from an L1-norm Quantile Regression Solution Path

Description
predict.qr.L1 is used to make predictions for a new dataset from a L1-norm Quantile Regression (QR) solution path.

Usage
## S3 method for class 'qr.L1':
predict(sp, x.new)

Arguments
sp a list containing the solution-path information. See path.qr.L1 for more information.
x.new a matrix containing new data of interest.

Value
The output of predict.qr.L1 is a matrix containing the predicted values for new dataset from an L1-norm QR solution path. Its output is a matrix. Each row vector of the matrix contains the predicted values for its corresponding model in the solution path.

Author(s)
Yonggang Yao, Department of Statistics, The Ohio State University.  ⟨yao.53@osu.edu⟩.

References

See Also
path.qr.L1, plot.path, risk.qr.L1.

Examples
### Simulate the training dataset.
set.seed(980)
require(lpRegPath)
n=400; p=10; sd=sqrt(50);
beta=c(2,0,2,0,2,0,0,0,0,2);
x=array(rnorm(p*n),c(n,p));
predict.svm.L1

Make Predictions for New data from an L1-norm SVM Solution Path

Description

predict.svm.L1 is used to make predictions for a new dataset from a L1-norm SVM solution path.

Usage

## S3 method for class 'svm.L1':
predict(sp, x.new, rtype='label')

Arguments

sp                  a list containing the solution-path information or fitted models. See path.svm.L1 and fit.svm.L1 for more information.
x.new               a matrix containing new data of interest.
rtype               a character specifying the type of predicted values. The default rtype is 'label', which means the predicted values are class labels (1 or -1). If rtype is 'score', the predicted values are computed by x%*%coeff. Note their signs are the predicted class labels.

Value

The output of predict.svm.L1 is a matrix containing the predicted values for new dataset from an L1-norm SVM solution path. Its output is a matrix. Each row vector of the matrix contains the predicted values for its corresponding model in the solution path.

Author(s)

Yonggang Yao, Department of Statistics, The Ohio State University. ⟨yao.53@osu.edu⟩.
**risk.qr.L1**

### Description

risk.qr.L1 estimates the risk curves of an L1-norm Quantile Regression problem as a function of the tuning parameter or the bound of model penalty by using cross-validation (CV) methods. Here, the risk refers to check loss.

### Usage

```r
## S3 method for class 'qr.L1':
risk(y, x, tau=.5, w=rep(1,length(y)), intercept=TRUE, fold=5, repetition=1, precision=1e-08)
```

### References


### See Also

`path.svm.L1`, `fit.svm.L1`, `risk.svm.L1`.

### Examples

```r
# Simulate the training dataset.
set.seed(980)
require(lpRegPath)
n=400; p=10; sd=sqrt(1);
beta=c(2,0,2,0,2,0,0,0,0,2);
x=array(rnorm(p*n),c(n,p));
y=as.vector(sign(x%*%beta+rnorm(n,0,sd)));

# Calculate and plot the solution path.
sp=path.svm.L1(y, x)
plot.path(sp)

# Predict the class labels for new covariates
n.new=30;
x.new = array(rnorm(p*n.new), c(n.new,p));
y.new = as.vector(sign(x.new%*%beta+rnorm(n.new, 0, sd)));
yhat.new = predict.svm.L1(sp, x.new)
er.new=apply(abs(y.new-t(yhat.new))/2,2,mean) # Estimated error rates
plot(sp$s_penalty, er.new, type='l', col='navy')
```

---

**risk.qr.L1**  
Risk Estimation for L1-norm Quantile Regression via Cross Validation

---

### Description

risk.qr.L1 estimates the risk curves of an L1-norm Quantile Regression problem as a function of the tuning parameter or the bound of model penalty by using cross-validation (CV) methods. Here, the risk refers to check loss.

### Usage

```r
## S3 method for class 'qr.L1':
risk(y, x, tau=.5, w=rep(1,length(y)), intercept=TRUE, fold=5, repetition=1, precision=1e-08)
```
Arguments

- **y**: A vector of class labels in training dataset. Elements in `y` must be either 1 or -1.
- **x**: A matrix containing covariates in training dataset.
- **tau**: A numeric value specifying the quantile parameter.
- **w**: A numeric vector specifying the weights of the observation pairs in training dataset in calculating empirical risk. Default `w` is an one-vector of the same size as `y`, such that all the observations are equally weighted.
- **intercept**: A logical value indicating whether the intercept term is included in the classifiers. Default `intercept` is true, such that the classifiers take the form of $y = \text{sign}[\beta_0 + x^T \beta]$. Otherwise, $y = \text{sign}[x^T \beta]$.
- **fold**: An integer specifying the fold of the CV method.
- **repetition**: An integer specifying the number of repetition for the CV method.
- **precision**: An internal control parameter.

Details

Consider a two-repetition-five-fold CV procedure. For each repetition, the CV method randomly partitions the training dataset into five groups with similar size. A risk curve can be estimated by treating one group as the testing dataset, and the rest four groups as the training dataset. Then, the two partition can totally output ten estimated risk curves (five for each repetition). The final estimated risk curve is the average of the ten estimated risk curves.

Value

The output of function `qr.L1.cv` is a list containing the information of the Estimated Risk (ER) curves with respect to tuning parameter or penalty. The list subsumes the two sets of variables. One set is associated with tuning parameter lambda, and the other set is associated with the penalty bound `s`.

- **ER.lam**: A matrix containing the minimum estimated-risk values in `ER.lam` in terms of check loss, and their associated tuning parameter values.
- **optER.lam**: A matrix containing the minimum estimated-risk values in `ER.lam` in terms of check loss, and its associated penalty value.

Author(s)

Yonggang Yao, Department of Statistics, The Ohio State University. (yao.53@osu.edu).

References

risk.svm.L1

See Also

plot.risk.path.qr.L1.

Examples

```
# Simulate the training dataset.
set.seed(980)
require(lpRegPath)
n=400; p=10; sd=sqrt(50);
beta=c(2,0,2,0,2,0,0,0,0,2);
x=array(rnorm(p*n),c(n,p));
y=as.vector(x%*%beta+rnorm(n,0,sd));
# Calculate and plot the solution path.
Est.risk=risk.qr.L1(y, x, fold=3, repetition=1)
Est.risk$optER.lam;
plot.risk(Est.risk)
```

Description

risk.svm.L1 estimates the risk curves of an L1-norm SVM problem as a function of the tuning parameter or the bound of model penalty by using cross-validation (CV) methods. The risks under consideration are in terms of 0-1 loss or hinge loss.

Usage

```
## S3 method for class 'svm.L1':
risk(y, x, w=rep(1,length(y)), intercept=TRUE, fold=5,
   repetition=1, precision=1e-08)
```

Arguments

- **y**: a vector of class labels in training dataset. Elements in y must be either 1 or -1.
- **x**: a matrix containing covariates in training dataset.
- **w**: a numeric vector specifying the weights of the observation pairs in training dataset in calculating empirical risk. Default w is an one-vector of the same size as y, such that all the observations are equally weighted.
- **intercept**: a logical value indicating whether the intercept term is included in the classifiers. Default intercept is true, such that the classifiers take the form of y=sign[beta0+x*beta]. Otherwise, y=sign[x*beta].
- **fold**: an integer specifying the fold of the CV method.
- **repetition**: an integer specifying the number of repetition for the CV method.
- **precision**: an internal control parameter.
**Details**

Consider a two-repetition-five-fold CV procedure. For each repetition, the CV method randomly partitions the training dataset into five groups with similar size. A risk curve can be estimated by treating one group as the testing dataset, and the rest four groups as the training dataset. Then, the two partition can totally output ten estimated risk curves (five for each repetition). The final estimated risk curve is the average of the ten estimated risk curves.

**Value**

The output of function `svm.L1.cv` is a list containing the information of the Estimated Risk (ER) curves with respect to tuning parameter or penalty. The list subsumes the two sets of variables. One set is associated with tuning parameter lambda, and the other set is associated with the penalty bound s.

- `ER.lam`  
  - `optER.lam` a matrix containing the minimum estimated-risk values in `ER.lam` in terms of 0-1 loss and hinge loss, and their associated tuning parameter values.

- `ER.s`  
  - `optER.s` a matrix containing the minimum estimated-risk values in `ER.lam` in terms of 0-1 loss and hinge loss, and their associated penalty values.

**Author(s)**

Yonggang Yao, Department of Statistics, The Ohio State University. (yao.53@osu.edu).

**References**


**See Also**

`plot.risk`, `path.svm.L1`.

**Examples**

```r
# Simulate the training dataset.
set.seed(980)
require(lpRegPath)
n=400; p=10; sd=sqrt(50);
beta=c(2,0,2,0,2,0,0,0,0,2);
x=array(rnorm(p*n),c(n,p));
y=as.vector(sign(x%*%beta+rnorm(n,0,sd)));

# Calculate and plot the solution path.
Est.risk=risk.svm.L1(y, x, fold=3, repetition=1)
Est.risk$optER.lam;
plot.risk(Est.risk)
```
Index

*Topic regression
  draw.num.select, 14
  fit.qr.L1, 3
  fit.svm.L1, 5
  Package, 6
  path.Dantzig, 7
  path.qr.GL1, 9
  path.qr.L1, 10
  path.svm.L1, 12
  plot.path, 15
  plot.risk, 17
  predict.qr.L1, 18
  predict.svm.L1, 20
  risk.qr.L1, 21
  risk.svm.L1, 23
  SolPath, 1

draw.num.select, 14

  fit.qr.L1, 3
  fit.svm.L1, 5, 20

lpRegPath (Package), 6

  Package, 6
  par, 14, 16
  path.Dantzig, 7
  path.qr.GL1, 9
  path.qr.L1, 3, 4, 10, 14–16, 18, 19, 22
  path.svm.L1, 3, 5, 6, 12, 14–16, 18, 20, 24
  plot.path, 4, 6, 8, 10, 11, 13, 15, 19
  plot.risk, 4, 6, 17, 22, 24
  predict.qr.L1, 18
  predict.svm.L1, 20

  risk.qr.L1, 4, 10, 11, 15–19, 21
  risk.svm.L1, 5, 6, 13, 15–18, 20, 23

  SolPath, 1