Package 'superpc'

October 12, 2020

Type Package

Title Supervised Principal Components

Version 1.10

Date 2020-10-12

Author Eric Bair [aut], Jean-Eudes Dazard [cre, ctb], Rob Tibshirani [ctb]

Maintainer Jean-Eudes Dazard <jean-eudes.dazard@case.edu>

Description
Supervised principal components for regression and survival analsysis. Especially useful for high-dimnesional data, including microarray data.

Depends R (>= 3.5.0)

Imports survival, stats, graphics, grDevices

NeedsCompilation no

URL http://www-stat.stanford.edu/~tibs/superpc, https://github.com/jedazard/superpc

Repository CRAN, GitHub, Inc.

Date/Publication 2004-09-16

License GPL (>= 3) | file LICENSE

Archs i386, x64

R topics documented:

superpc.cv *Cross-validation for supervised principal components*

Description

This function uses a form of cross-validation to estimate the optimal feature threshold in supervised principal components

Usage

```
superpc.cv(fit,
           data,
           n.threshold=20,
           n.fold=NULL,
           folds=NULL,
           n.components=3,
           min.features=5,
           max.features=nrow(data$x),
           compute.fullcv= TRUE,
           compute.preval=TRUE,
           xl.mode=c("regular","firsttime","onetime","lasttime"),
           xl.time=NULL,
           xl.prevfit=NULL)
```

Arguments

`fit`	Object returned by superpc.train
`data`	Data object of form described in superpc.train documentation
`n.threshold`	Number of thresholds to consider. Default 20.
`n.fold`	Number of cross-validation folds. default is around 10 (program pick a convenient value based on the sample size
`folds`	List of indices of cross-validation folds (optional)
`n.components`	Number of cross-validation components to use: 1,2 or 3.
`min.features`	Minimum number of features to include in determining range for threshold. Default 5.
`max.features`	Maximum number of features to include in determining range for threshold. Default is total number of features in the dataset
`compute.fullcv`	Should full cross-validation be done?
`compute.preval`	Should full pre-validation be done?
`xl.mode`	Used by Excel interface only
`xl.time`	Used by Excel interface only
`xl.prevfit`	Used by Excel interface only

Details

This function uses a form of cross-validation to estimate the optimal feature threshold in supervised principal components. To avoid prolems with fitting Cox models to samll validation datastes, it uses the "pre-validation" approach of Tibshirani and Efron (2002)

Value

`threshold`	Vector of thresholds considered
`nonzero`	Number of features exceeding each value of the threshold
`scor.preval`	Likelihood ratio scores from pre-validation
`scor`	Full CV scores
`folds`	Indices of CV folds used
`featurescores.folds`	Feature scores for each fold
`v.preval`	The pre-validated predictors
`type`	problem type
`call`	calling sequence

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data.*" PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components.*" J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
## Not run:
set.seed(332)

#generate some data
x <- matrix(rnorm(50*30), ncol=30)
y <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
censoring.status <- sample(c(rep(1,20), rep(0,10)))

featurenames <- paste("feature", as.character(1:50), sep="")
data <- list(x=x,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)

a <- superpc.train(data, type="survival")
aa <- superpc.cv(a, data)

## End(Not run)
```

superpc.decorrelate *Decorrelate features with respect to competing predictors*

Description

Fits a linear model to the features as a function of some competing predictors. Replaces the features by the residual from this fit. These "decorrelated" features are then used in the superpc model building process, to explicitly look for predictors that are independent of the competing predictors. Useful for example, when the competing predictors are clinical predictors like stage, grade etc.

Usage

```
superpc.decorrelate(x,
                    competing.predictors)
```

Arguments

`x`	matrix of features. Different features in different rows, one observation per column
`competing.predictors`	List of one or more competing predictors. Discrete predictors should be factors

Value

Returns lm (linear model) fit of rows of x on compeiting predictors.

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data.*" PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components.*" J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
set.seed(332)

#generate some data
x <- matrix(rnorm(50*30), ncol=30)
y <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
censoring.status <- sample(c(rep(1,20), rep(0,10)))

featurenames <- paste("feature", as.character(1:50), sep="")
competing.predictors <- list(pred1=rnorm(30),
                                pred2=as.factor(sample(c(1,2),
                                                replace=TRUE,
                                                size=30)))

#decorrelate x. Remember to decorrelate test data in the same way, before making predictions.
foo <- superpc.decorrelate(x, competing.predictors)
xnew <- t(foo$res)

#now use xnew in superpc
data <- list(x=xnew,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)
a <- superpc.train(data, type="survival")

#etc.
```

---

superpc.fit.to.outcome
Fit predictive model using outcome of supervised principal components

---

Description

Fit predictive model using outcome of supervised principal components, via either coxph (for surival data) or lm (for regression data)

Usage

```
superpc.fit.to.outcome(fit,
                       data.test,
                       score,
                       competing.predictors=NULL,
                       print=TRUE,
                       iter.max=5)
```

Arguments

`fit`	Object returned by superpc.train.
`data.test`	Data object for prediction. Same form as data object documented in superpc.train.
`score`	Supervised principal component score, from superpc.predict.
`competing.predictors`	Optional - a list of competing predictors to be included in the model.
`print`	Should a summary of the fit be printed? Default TRUE.
`iter.max`	Max number of iterations used in predictive model fit. Default 5. Currently only relevant for Cox PH model.

Value

Returns summary of coxph or lm fit.

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data*." PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components*." J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
set.seed(332)

#generate some data
x <- matrix(rnorm(50*30), ncol=30)
y <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
ytest <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
censoring.status <- sample(c(rep(1,20), rep(0,10)))
censoring.status.test <- sample(c(rep(1,20), rep(0,10)))

featurenames <- paste("feature", as.character(1:50), sep="")
data <- list(x=x,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)
data.test <- list(x=x,
                  y=ytest,
                  censoring.status=censoring.status.test,
                  featurenames=featurenames)

a <- superpc.train(data, type="survival")
fit <- superpc.predict(a,
                       data,
                       data.test,
                       threshold=1.0,
                       n.components=1,
                       prediction.type="continuous")
superpc.fit.to.outcome(a,
                       data,
                       fit$v.pred)
```

superpc.listfeatures *Return a list of the important predictors*

Description

Return a list of the important predictor

Usage

```
superpc.listfeatures(data,
                     train.obj,
                     fit.red,
                     fitred.cv=NULL,
                     num.features=NULL,
                     component.number=1)
```

Arguments

`data`	Data object
`train.obj`	Object returned by superpc.train
`fit.red`	Object returned by superpc.predict.red, applied to training set
`fitred.cv`	(Optional) object returned by superpc.predict.red.cv
`num.features`	Number of features to list. Default is all features.
`component.number`	Number of principal component (1,2, or 3) used to determine feature importance scores

Value

Returns matrix of features and their importance scores, in order of decreasing absolute value of importance score. The importance score is the correlation of the reduced predictor and the full supervised PC predictor. It also lists the raw score- for survival data, this is the Cox score for that feature; for regression, it is the standardized regression coefficient. If fitred.cv is supplied, the function also reports the average rank of the gene in the cross-validation folds, and the proportion of times that the gene is chosen (at the given threshold) in the cross-validation folds.

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data.*" PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components.*" J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
set.seed(332)

#generate some data
x <- matrix(rnorm(50*30), ncol=30)
y <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
ytest <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
censoring.status <- sample(c(rep(1,20), rep(0,10)))
censoring.status.test <- sample(c(rep(1,20), rep(0,10)))

featurenames <- paste("feature", as.character(1:50), sep="")
data <- list(x=x,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)
data.test <- list(x=x,
                  y=ytest,
                  censoring.status=censoring.status.test,
                  featurenames=featurenames)

a <- superpc.train(data, type="survival")
fit.red <- superpc.predict.red(a,
                               data,
                               data.test,
                               .6)
superpc.listfeatures(data,
                     a,
                     fit.red,
                     num.features=10)
```

---

superpc.lrtest.curv *Compute values of likelihood ratio test from supervised principal components fit*

Description

Compute values of likelihood ratio test from supervised principal components fit

Usage

```
superpc.lrtest.curv(object,
                    data,
                    newdata,
                    n.components=1,
                    threshold=NULL,
                    n.threshold=20)
```

Arguments

`object`	Object returned by superpc.train.
`data`	List of training data, of form described in superpc.train documentation.
`newdata`	List of test data; same form as training data.
`n.components`	Number of principal components to compute. Should be 1,2 or 3.
`threshold`	Set of thresholds for scores; default is n.threshold values equally spaced over the range of the feature scores.
`n.threshold`	Number of thresholds to use; default 20. Should be 1,2 or 3.

Value

`lrtest`	Values of likelihood ratio test statistic
`comp2`	Description of 'comp2'
`threshold`	Thresholds used
`num.features`	Number of features exceeding threshold
`type`	Type of outcome variable
`call`	calling sequence

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data.*" PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components.*" J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
set.seed(332)

#generate some data
x <- matrix(rnorm(50*30), ncol=30)
y <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
ytest <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
censoring.status <- sample(c(rep(1,20), rep(0,10)))
censoring.status.test <- sample(c(rep(1,20), rep(0,10)))

featurenames <- paste("feature", as.character(1:50), sep="")
data <- list(x=x,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)
data.test <- list(x=x,
                  y=ytest,
                  censoring.status=censoring.status.test,
                  featurenames=featurenames)

a <- superpc.train(data, type="survival")
aa <- superpc.lrtest.curv(a, data, data.test)
#superpc.plot.lrtest(aa)
```

superpc.plot.lrtest *Plot likelhiood ratio test statistics*

Description

Plot likelhiood ratio test statistics from output of superpc.predict

Usage

```
superpc.plot.lrtest(object.lrtestcurv,
                      call.win.metafile=FALSE)
```

Arguments

`object.lrtestcurv`	Output from superpc.lrtest.curv
`call.win.metafile`	For use by PAM Excel interface

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data*." PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components*." J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
set.seed(332)

#generate some data
x <- matrix(rnorm(50*30), ncol=30)
y <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
ytest <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
censoring.status <- sample(c(rep(1,20), rep(0,10)))
censoring.status.test <- sample(c(rep(1,20), rep(0,10)))

featurenames <- paste("feature", as.character(1:50), sep="")
data <- list(x=x,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)
data.test <- list(x=x,
                  y=ytest,
                  censoring.status=censoring.status.test,
                    featurenames=featurenames)

a <- superpc.train(data, type="survival")
bb <- superpc.lrtest.curv(a,
                          data,
                          data.test)
superpc.plot.lrtest(bb)
```

superpc.plotcv *Plot output from superpc.cv*

Description

Plots pre-validation results from plotcv, to aid in choosing best threshold

Usage

```
superpc.plotcv(object,
                cv.type=c("full","preval"),
                smooth=TRUE,
                smooth.df=10,
                call.win.metafile=FALSE, ...)
```

Arguments

`object`	Object returned by superpc.cv.
`cv.type`	Type of cross-validation used - "full" (Default; this is "standard" cross-validation; recommended) and "preval"- pre-validation.
`smooth`	Should plot be smoothed? Only relevant to "preval". Default FALSE.
`smooth.df`	Degrees of freedom for smooth.spline, default 10. If NULL, then degrees of freedom is estimated by cross-validation.
`call.win.metafile`	Ignore: for use by PAM Excel program.
`...`	Additional plotting args to be passed to matplot.

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data*." PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components*." J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
## Not run:
set.seed(332)

#generate some data
x <- matrix(rnorm(50*30), ncol=30)
y <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
censoring.status <- sample(c(rep(1,20), rep(0,10)))

featurenames <- paste("feature", as.character(1:50), sep="")
data <- list(x=x,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)

a <- superpc.train(data, type="survival")
aa <- superpc.cv(a,data)

superpc.plotcv(aa)

## End(Not run)
```

`superpc.plotred.lrtest` *Plot likelihood ratio test statistics from supervised principal components predictor*

Description

Plot likelihood ratio test statistics from supervised principal components predictor

Usage

```
superpc.plotred.lrtest(object.lrtestred,
                          call.win.metafile=FALSE)
```

Arguments

`object.lrtestred`	Output from either superpc.predict.red or superpc.predict.redcv
`call.win.metafile`	Used only by PAM Excel interface call to function

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data.*" PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components.*" J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
## Not run:
set.seed(332)

#generate some data
x <- matrix(rnorm(50*30), ncol=30)
y <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
ytest <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
censoring.status <- sample(c(rep(1,20), rep(0,10)))
censoring.status.test <- sample(c(rep(1,20), rep(0,10)))

featurenames <- paste("feature", as.character(1:50), sep="")
data <- list(x=x,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)
data.test <- list(x=x,
                  y=ytest,
                  censoring.status=censoring.status.test,
                  featurenames=featurenames)

a <- superpc.train(data, type="survival")
aa <- superpc.cv(a, data)
fit.red <- superpc.predict.red(a,
                               data,
                               data.test,
                               .6)
fit.redcv <- superpc.predict.red.cv(fit.red,
                                    aa,
                                    data,
                                    .6)
superpc.plotred.lrtest(fit.redcv)

## End(Not run)
```

`superpc.predict` *Form principal components predictor from a trained superpc object*

Description

Computes supervised principal components, using scores from "object"

Usage

```
superpc.predict(object,
                data,
                newdata,
                threshold,
                n.components=3,
                prediction.type=c("continuous","discrete","nonzero"),
                n.class=2)
```

Arguments

`object`	Obect returned by superpc.train
`data`	List of training data, of form described in superpc.train documentation,
`newdata`	List of test data; same form as training data
`threshold`	Threshold for scores: features with abs(score) > threshold are retained.
`n.components`	Number of principal components to compute. Should be 1,2 or 3.
`prediction.type`	"continuous" for raw principal component(s); "discrete" for principal component categorized in equal bins; "nonzero" for indices of features that pass the threshold
`n.class`	Number of classes into which predictor is binned (for prediction.type="discrete"

Value

`v.pred`	Supervised principal componients predictor
`u`	U matrix from svd of feature matrix x
`d`	singual values from svd of feature matrix x
`which.features`	Indices of features exceeding threshold
`n.components`	Number of supervised principal components requested
`call`	calling sequence

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data.*" PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components.*" J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
set.seed(332)

#generate some data
x <- matrix(rnorm(50*30), ncol=30)
y <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
ytest <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
```

```
censoring.status <- sample(c(rep(1,20), rep(0,10)))
censoring.status.test <- sample(c(rep(1,20), rep(0,10)))

featurenames <- paste("feature", as.character(1:50), sep="")
data <- list(x=x,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)
data.test <- list(x=x,
                  y=ytest,
                  censoring.status=censoring.status.test,
                  featurenames=featurenames)

a <- superpc.train(data, type="survival")
fit <- superpc.predict(a,
                       data,
                       data.test,
                       threshold=1.0,
                       n.components=1)
plot(fit$v.pred, ytest)
```

superpc.predict.red *Feature selection for supervised principal components*

Description

Forms reduced models to approximate the supervised principal component predictor.

Usage

```
superpc.predict.red(fit,
                    data,
                    data.test,
                    threshold,
                    n.components=3,
                    n.shrinkage=20,
                    shrinkages=NULL,
                    compute.lrtest=TRUE,
                    sign.wt="both",
                    prediction.type=c("continuous", "discrete"),
                    n.class=2)
```

Arguments

`fit`	Object returned by superpc.train
`data`	Training data object, of form described in superpc.train dcoumentation
`data.test`	Test data object; same form as train
`threshold`	Feature score threshold; usually estimated from superpc.cv
`n.components`	Number of principal components to examine; should equal 1,2, etc up to the number of components used in training

`n.shrinkage`	Number of shrinkage values to consider. Default 20.
`shrinkages`	Shrinkage values to consider. Default NULL.
`compute.lrtest`	Should the likelihood ratio test be computed? Default TRUE
`sign.wt`	Signs of feature weights allowed: "both", "pos", or "neg"
`prediction.type`	Type of prediction: "continuous" (Default) or "discrete". In the latter, superprc score is divided into n.class groups
`n.class`	Number of groups for discrete predictor. Default 2.

Details

Soft-thresholding by each of the "shrinkages" values is applied to the PC loadings. This reduce the number of features used in the model. The reduced predictor is then used in place of the supervised PC predictor.

Value

`shrinkages`	Shrinkage values used
`lrtest.reduced`	Likelihood ratio tests for reduced models
`num.features`	Number of features used in each reduced model
`feature.list`	List of features used in each reduced model
`coef`	Least squares coefficients for each reduced model
`import`	Importance scores for features
`wt`	Weight for each feature, in constructing the reduced predictor
`v.test`	Outcome predictor from reduced models. Array of n.shrinkage by (number of test observations)
`v.test.1df`	Outcome combined predictor from reduced models. Array of n.shrinkage by (number of test observations)
`n.components`	Number of principal components used
`type`	Type of outcome
`call`	calling sequence

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data*." PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components*." J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
set.seed(332)

#generate some data
x <- matrix(rnorm(50*30), ncol=30)
y <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
ytest <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
censoring.status <- sample(c(rep(1,20), rep(0,10)))
censoring.status.test <- sample(c(rep(1,20), rep(0,10)))

featurenames <- paste("feature", as.character(1:50), sep="")
data <- list(x=x,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)
data.test <- list(x=x,
                  y=ytest,
                  censoring.status=censoring.status.test,
                  featurenames=featurenames)

a <- superpc.train(data, type="survival")
fit.red <- superpc.predict.red(a,
                               data,
                               data.test,
                               threshold=.6)
superpc.plotred.lrtest(fit.red)
```

---

superpc.predict.red.cv

Cross-validation of feature selection for supervised principal components

---

Description

Applies superpc.predict.red to cross-validation folds generates in superpc.cv. Uses the output to evaluate reduced models, and compare them to the full supervised principal components predictor.

Usage

```
superpc.predict.red.cv(fitred,
                       fitcv,
                       data,
                       threshold,
                       sign.wt="both")
```

Arguments

`fitred`	Output of superpc.predict.red
`fitcv`	Output of superpc.cv
`data`	Training data object
`threshold`	Feature score threshold; usually estimated from superpc.cv
`sign.wt`	Signs of feature weights allowed: "both", "pos", or "neg"

Value

`lrtest.reduced`	Likelihood ratio tests for reduced models
`components`	Number of supervised principal components used
`v.preval.red`	Outcome predictor from reduced models. Array of num.reduced.models by (number of test observations)
`type`	Type of outcome
`call`	calling sequence

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data*." PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components*." J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
## Not run:
set.seed(332)

#generate some data
x <- matrix(rnorm(50*20), ncol=20)
y <- 10 + svd(x[1:10,])$v[,1] + .1*rnorm(20)
ytest <- 10 + svd(x[1:10,])$v[,1] + .1*rnorm(20)
censoring.status <- sample(c(rep(1,15), rep(0,5)))
censoring.status.test <- sample(c(rep(1,15), rep(0,5)))

featurenames <- paste("feature", as.character(1:50), sep="")
data <- list(x=x,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)
data.test <- list(x=x,
                  y=ytest,
                  censoring.status=censoring.status.test,
                  featurenames=featurenames)

a <- superpc.train(data, type="survival")
aa <- superpc.cv(a, data)
fit.red <- superpc.predict.red(a,
                               data,
                               data.test,
                               threshold=.6)
fit.redcv <- superpc.predict.red.cv(fit.red,
                                    aa,
                                    data,
                                       threshold=.6)

## End(Not run)
```

superpc.predictionplot *Plot outcome predictions from superpc*

Description

Plots outcome predictions from superpc

Usage

```
superpc.predictionplot(train.obj,
                       data,
                       data.test,
                       threshold,
                       n.components=3,
                       n.class=2,
                       shrinkage=NULL,
                       call.win.metafile=FALSE)
```

Arguments

`train.obj`	Object returned by superpc.train
`data`	List of training data, of form described in superpc.train documentation
`data.test`	List of test data; same form as training data
`threshold`	Threshold for scores: features with abs(score) > threshold are retained.
`n.components`	Number of principal components to compute. Should be 1,2 or 3.
`n.class`	Number of classes for survival stratification. Only applicable for survival data. Default 2.
`shrinkage`	Shrinkage to be applied to feature loadings. Default is NULL, meaning no shrinkage
`call.win.metafile`	Used only by Excel interface call to function

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data.*" PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components.*" J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
set.seed(332)

#generate some data
x <- matrix(rnorm(50*30), ncol=30)
y <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
ytest <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
censoring.status <- sample(c(rep(1,20), rep(0,10)))
censoring.status.test <- sample(c(rep(1,20), rep(0,10)))

featurenames <- paste("feature", as.character(1:50), sep="")
data <- list(x=x,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)
data.test <- list(x=x,
                  y=ytest,
                  censoring.status=censoring.status.test,
                  featurenames=featurenames)

a <- superpc.train(data, type="survival")
superpc.predictionplot(a,
                       data,
                       data.test,
                       threshold=1)
```

`superpc.rainbowplot` *Make rainbow plot of superpc and compeiting predictors*

Description

Makes a heatmap display of outcome predictions from superpc, along with expected survival time, and values of competing predictors.

Usage

```
superpc.rainbowplot(data,
                    pred,
                    sample.labels,
                    competing.predictors,
                    call.win.metafile=FALSE)
```

Arguments

`data`	List of (test) data, of form described in superpc.train documentation
`pred`	Superpc score from superpc.predict or superpc.predict.red
`sample.labels`	Vector of sample labels of test data
`competing.predictors`	List of competing predictors to be plotted
`call.win.metafile`	Used only by Excel interface call to function

Details

Any censored survival times are estimated by E(T|T > C), where $C$ is the observed censoring time and the Kaplan-Meier estimate from the training set is used to estimate the expectation.

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data*." PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components*." J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
set.seed(332)

#generate some data
x <- matrix(rnorm(50*30), ncol=30)
y <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
ytest <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
censoring.status <- sample(c(rep(1,20), rep(0,10)))
censoring.status.test <- sample(c(rep(1,20), rep(0,10)))

featurenames <- paste("feature", as.character(1:50), sep="")
competing.predictors.test <- list(pred1=rnorm(30),
                                  pred2=as.factor(sample(c(1,2),
                                                  replace=TRUE,
                                                  size=30)))

data <- list(x=x,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)
data.test <- list(x=x,
                  y=ytest,
                  censoring.status=censoring.status.test,
                  featurenames=featurenames)
sample.labels <- paste("te", as.character(1:20), sep="")

a <- superpc.train(data, type="survival")
pred <- superpc.predict(a,
                        data,
                        data.test,
                        threshold=.25,
                        n.components=1)$v.pred
superpc.rainbowplot(data,
                    pred,
                    sample.labels,
                    competing.predictors=competing.predictors.test)
```

superpc.train *Prediction by supervised principal components*

Description

Does prediction of a quantitative regression or survival outcome, by the supervised principal components method.

Usage

```
superpc.train(data,
              type=c("survival", "regression"),
              s0.perc=NULL)
```

Arguments

`data`	Data object with components x- p by n matrix of features, one observation per column; y- n-vector of outcome measurements; censoring.status- n-vector of censoring censoring.status (1= died or event occurred, 0=survived, or event was censored), needed for a censored survival outcome
`type`	Problem type: "survival" for censored survival outcome, or "regression" for simple quantitative outcome
`s0.perc`	Factor for denominator of score statistic, between 0 and 1: the percentile of standard deviation values added to the denominator. Default is 0.5 (the median)

Details

Compute wald scores for each feature (gene), for later use in superpc.predict and superpc.cv

Value

`feature.scores`	Score for each feature (gene)
`type`	problem type
`s0.perc`	Factor for denominator of score statistic
`call`	calling sequence

Author(s)

- "Eric Bair, Ph.D."
- "Jean-Eudes Dazard, Ph.D."
- "Rob Tibshirani, Ph.D."

Maintainer: "Rob Tibshirani, Ph.D."

References

- E. Bair and R. Tibshirani (2004). "*Semi-supervised methods to predict patient survival from gene expression data.*" PLoS Biol, 2(4):e108.
- E. Bair, T. Hastie, D. Paul, and R. Tibshirani (2006). "*Prediction by supervised principal components.*" J. Am. Stat. Assoc., 101(473):119-137.

Examples

```
set.seed(332)

#generate some data
x <- matrix(rnorm(50*30), ncol=30)
y <- 10 + svd(x[1:50,])$v[,1] + .1*rnorm(30)
censoring.status <- sample(c(rep(1,20), rep(0,10)))

featurenames <- paste("feature", as.character(1:50), sep="")
data <- list(x=x,
             y=y,
             censoring.status=censoring.status,
             featurenames=featurenames)

a <- superpc.train(data, type="survival")
```

Index