Package ‘FeaLect’

October 12, 2022

Type Package
Title Scores Features for Feature Selection
Version 1.20
Date 2020-02-25
Author Habil Zare
Maintainer Habil Zare <zare@u.washington.edu>
Depends lars, rms
Description For each feature, a score is computed that can be useful for feature selection. Several random subsets are sampled from the input data and for each random subset, various linear models are fitted using lars method. A score is assigned to each feature based on the tendency of LASSO in including that feature in the models. Finally, the average score and the models are returned as the output. The features with relatively low scores are recommended to be ignored because they can lead to overfitting of the model to the training data. Moreover, for each random subset, the best set of features in terms of global error is returned. They are useful for applying Bolasso, the alternative feature selection method that recommends the intersection of features subsets.
License GPL (>= 2)
LazyLoad yes
Repository CRAN
Date/Publication 2020-02-25 17:30:06 UTC
NeedsCompilation no
RoxygenNote 6.0.1

R topics documented:

FeaLect-package .................................................. 2
compute.balanced ................................................. 3
Suppose you have a feature matrix with 200 features and only 20 samples and your goal is to build a classifier. You can run the FeaLect() function to compute the scores for your features. Only the relatively high score features (say the top 20) are recommended for further analysis. In this way, one can prevent overfitting by reducing the number of features significantly.
compute.balanced

Balances between negative and positive samples by oversampling.

compute.logistic.score

Fits a logistic regression model using the linear scores.

doctor.validate

Validates a model using validating samples.

ignore.redundant

Refines a feature matrix.

input.check.FeaLect

Checks the inputs to Fealect() function.

mcl_sll

MCL and SLL lymphoma subtypes.

random.subset

Selects a random subset of the input.

train.doctor

Fits various models based on a combination on penalized linear models and logistic regression.

Author(s)

Habil Zare

Maintainer: Habil Zare <zare@u.washington.edu>

References


See Also

FeaLect, train.doctor, doctor.validate, random.subset, compute.balanced, compute.logistic.score, ignore.redundant, input.check.FeaLect, lars-package, and SparseLearner-package

Examples

library(FeaLect)
data(mcl_sll)
F <- as.matrix(mcl_sll[,,-1]) # The Feature matrix
L <- as.numeric(mcl_sll[,1]) # The labels
names(L) <- rownames(F)
message(dim(F)[1], " samples and ",dim(F)[2], " features.")

## For this data, total.num.of.models is suggested to be at least 100.
FeaLect.result.1 <-FeaLect(F=F,L=L,maximum.features.num=10,total.num.of.models=20,talk=TRUE)

If negative samples are less than positive ones, more copies of the negative cases are added and vice versa.
Usage

```r
compute.balanced(F_, L_)
```

Arguments

- `F_` The feature matrix, each column is a feature.
- `L_` The vector of labels named according to the rows of `F`.

Details

Considerably unbalanced classes may be probabilistic for fitting some models.

Value

Returns a list of:

- `F_` The feature matrix, each column is a feature.
- `L_` The vector of labels named according to the rows of `F`.

Author(s)

Habil Zare

References


See Also

`FeaLect`, `train.doctor`, `doctor.validate`, `random.subset`, `compute.balanced`, `compute.logistic.score`, `ignore.redundant`, `input.check.FeaLect`

Examples

```r
library(FeaLect)
data(mcl_sll)
F <- as.matrix(mcl_sll[, -1])  # The Feature matrix
L <- as.numeric(mcl_sll[, 1])  # The labels
names(L) <- rownames(F)
message(L)

balanced <- compute.balanced(F_=F, L_=L)
message(balanced$L_)
```
compute.logistic.score

*Fits a logistic regression model using the linear scores*

**Description**

A logistic regression model is fitted to the linear scores using lrm() function and the logistic scores are computed using the formula: \( \frac{1}{1+\exp(-(a+bX))} \) where a and b are the logistic coefficients.

**Usage**

```r
compute.logistic.score(F_, L_, considered.features, training.samples, validating.samples, linear.scores, report.fitting.failure = TRUE)
```

**Arguments**

- `F_` The feature matrix, each column is a feature.
- `L_` The vector of labels named according to the rows of `F`.
- `training.samples` The names of rows of `F` that should be considered as training samples.
- `validating.samples` The names of rows of `F` that should be considered as validating samples.
- `considered.features` The names of columns of `F` that determine the features of interest.
- `linear.scores` A vector that contains for each training or validating sample, a linear score predicted by the linear method.
- `report.fitting.failure` If TRUE, any failure in fitting the linear of logistic models will be printed.

**Details**

The logistic regression will be fitted to all training and validating samples.

**Value**

Returns a list of:

- `logistic.scores` A vector of predicted logistic values for all samples.
- `logistic.cofs` The coefficients that are computed by logistic regression.

**Note**

Logistic regression is also done on top of fitting the linear models.
Author(s)
Habil Zare

References

See Also
FeaLect, train.doctor, doctor.validate, random.subset, compute.balanced, compute.logistic.score, ignore.redundant, input.check.FeaLect

Examples

library(FeaLect)
data(mcl_sll)
F <- as.matrix(mcl_sll[, -1])  # The Feature matrix
L <- as.numeric(mcl_sll[, 1])  # The labels
names(L) <- rownames(F)
all.samples <- rownames(F); ts <- all.samples[5:10]; vs <- all.samples[c(1, 22)]
L <- L[c(ts, vs)]

asymptotic.scores <- c(1, 0.9, 0.8, 0.2, 0.1, 0.1, 0.7, 0.2)

compute.logistic.score(F_=F, L_=L, training.samples=ts, validating.samples=vs,
considered.features=colnames(F), linear.scores=asymptotic.scores)

doctor.validate

Validates a model using validating samples.

Description
A model fitted on the training samples, can be validated on a separate validating set. The recall, precision, and accuracy of the model are computed.

Usage
doctor.validate(true.labels, predictions)

Arguments
true.labels  A vector of 0 and 1.
predictions  A vector of 0 and 1.

Details
F-measure is equal to: 2 times precision times recall / (precision + recall).
FeaLect

Value

F-measure, precision, and recall are calculated. Also, the mis-labeled cases are reported.

Author(s)

Habil Zare

References


See Also

FeaLect, train.doctor, doctor.validate, random.subset, compute.balanced, compute.logistic.score, ignore.redundant, input.check.FeaLect

Examples

```
tls <- c(1,1,1,0,0)
ps <- c(1,1,0,1,0)
names(tls) <- 1:5; names(ps) <- 1:5

doctor.validate(true.labels=tls, predictions=ps)
```

FeaLect

Computes the scores of the features.

Description

Several random subsets are sampled from the input data and for each random subset, various linear models are fitted using lars method. A score is assigned to each feature based on the tendency of LASSO in including that feature in the models. Finally, the average score and the models are returned as the output.

Usage

```
FeaLect(F, L, maximum.features.num = dim(F)[2], total.num.of.models, gamma = 3/4,
persistence = 1000, talk = FALSE, minimum.class.size = 2,
report.fitting.failure = FALSE, return_linear.models = TRUE, balance = TRUE,
replace = TRUE, plot.scores = TRUE)
```

Arguments

F The feature matrix, each column is a feature.
L The vector of labels named according to the rows of F.
maximum.features.num Upto this number of features are allowed to contribute to each linear model.
total.num.of.models The total number of models that are fitted.
gamma A value in range 0-1 that determines the relative size of sample subsets.
persistence Maximum number of tries for randomly choosing samples. If we try this many times and the obtained labels are all the same, we give up (maybe the whole labels are the same) with the error message: "Not enough variation in the labels...".
talk If TRUE, some messages are printed during the computations.
minimum.class.size The size of both positive and negative classes should be greater than this threshold after sampling.
report.fitting.failure If TRUE, any failure in fitting the linear of logistic models will be printed.
return_linear.models The models are memory intensive, so for if they more than 1000, we may decide to ignore them to prevent memory outage.
balance If TRUE, the cases will be balanced for the same number of positive vs. negatives by oversampling before fitting the linear model.
replace If TRUE, the subsets are sampled with replacement.
plot.scores If TRUE, the scores are plotted in logarithmic scale after each iteration.

Details

See the reference for more details.

Value

Returns a list of:

log.scores A vector containing the logarithm of final scores.
feature.matrix The input feature matrix.
labels The input labels
total.num.of.models The total number of models that are fitted.
maximum.features.num Upto this number of features are allowed to contribute to each linear model.
feature.scores.history The matrix of history of feature scores where column i contains the scores after i runs.
num.of.features.score
A vector, entry i contains the number of times that i has been the best number of
features.

best.feature.num
The i’th value of this vector is the best number of features for the i’th model.

mislabling.record
A vector that keeps track of the frequency of mislabelling for each cases.

doctors
List of all models which are created by train.doctor() function.

best.features.intersection
Best features are computed for each sampling and their intersection is reported
as this vector of features names

features.with.best.global.error
A list containing the sets of features. The set i was the best for i’th sampling.

time.taken
Total time used for executing this function.

Note
Logistic regression is also done on top of fitting the linear models.

Author(s)
Habil Zare

References

See Also
FeaLect, train.doctor, doctor.validate, random.subset, compute.balanced, compute.logistic.score,
ignore.redundant, input.check.FeaLect

Examples
library(FeaLect)
data(mcl_sll)
F <- as.matrix(mcl_sll[, -1]) # The Feature matrix
L <- as.numeric(mcl_sll[, 1]) # The labels
names(L) <- rownames(F)
message(dim(F)[1], " samples and ",dim(F)[2], " features.")

## For this data, total.num.of.models is suggested to be at least 100.
FeaLect.result <- FeaLect(F=F, L=L, maximum.features.num=10, total.num.of.models=20, talk=TRUE)
ignore.redundant  

Refines a feature matrix

Description

If the value a feature is the same for all points (e.g. =0), it can be ignored.

Usage

ignore.redundant(F, num.of.values = 1)

Arguments

F  
The feature matrix, each column is a feature.

num.of.values  
A feature should have more than this threshold non-zero values not to be ignored.

Value

The refined feature matrix.

Author(s)

Habil Zare

References


See Also

FeaLect, train.doctor, doctor.validate, random.subset, compute.balanced, compute.logistic.score, ignore.redundant, input.check.FeaLect

Examples

library(FeaLect)
data(mcl_sll)
F <- as.matrix(mcl_sll[, -1]) # The Feature matrix
#F <- cbind(F, rep(1, times=dim(F)[1]))
message(dim(F)[1], " samples and ",dim(F)[2], " features.")

G <- ignore.redundant(F)
message("for ",dim(G)[1], " samples, ",dim(G)[2], " features are left.")
input.check.FeaLect  Checks the inputs to Fealect() function.

Description

We should have: F as a matrix, L as a vector, and length of L be equal to number of rows of F. They should have names accordingly.

Usage

input.check.FeaLect(F_, L_, maximum.features.num, gamma)

Arguments

F_  The feature matrix, each column is a feature.
L_  The vector of labels named according to the rows of F.
maximum.features.num  Upto this number of features are allowed to contribute to each linear model.
gamma  A value in range 0-1 that determines the relative size of sample subsets.

Details

If the input is not appropriate, error or warning message will be produced.

Value

Returns a list of:

F_  The feature matrix, each column is a feature.
L_  The vector of labels named according to the rows of F.
maximum.features.num  Upto this number of features are allowed to contribute to each linear model.

Author(s)

Habil Zare

References


See Also

FeaLect, train.doctor, doctor.validate, random.subset, compute.balanced, compute.logistic.score, ignore.redundant, input.check.FeaLect
Examples

```r
library(FeaLect)
data(mcl_sll)
F <- as.matrix(mcl_sll[, -1]) # The Feature matrix
L <- as.numeric(mcl_sll[, 1]) # The labels
names(L) <- rownames(F)

checked <- input.check.FeaLect(F_=F, L_=L, maximum.features.num=10, gamma=3/4)
```

mcl_sll

MCL and SLL lymphoma subtypes

Description

A total of 237 features are identified for 22 lymphoma patients.

Usage

```r
data(mcl_sll)
```

Format

A matrix. Each of the 237 columns represents a features except the first column which contains the label vector. Each of the 22 rows represents a patients.

Details

7 cases diagnosed with Mantel Cell Lymphoma (MCL) and 15 cases with Small Lymphocytic Lymphoma (SLL). The presented features are computed based on flow cytometry data. The fist column contains the label vector which has value 1 for MCL cases and 0 for SLL cases.

Source

British Columbia Cancer Agency

References


See Also

FeaLect, train.doctor, doctor.validate, random.subset, compute.balanced, compute.logistic.score, ignore.redundant, input.check.FeaLect
random.subset

Examples

library(FeaLect)
data(mcl_sll)
F <- as.matrix(mcl_sll[, -1])  # The Feature matrix
L <- as.numeric(mcl_sll[, 1])  # The labels
names(L) <- rownames(F)
message(dim(F)[1], " samples and ",dim(F)[2], " features.")
L

random.subset

Selects a random subset of the input.

Description

If a subset of samples are selected randomly, the navigate of positive classes might be too sparse or even empty. This function will repeat sampling until the classes are appropriate in this sense.

Usage

random.subset(F_, L_, gamma, persistence = 1000, minimum.class.size=2, replace)

Arguments

F_  The feature matrix, each column is a feature.
L_  The vector of labels named according to the rows of F.
gamma A value in range 0-1 that determines the relative size of sample subsets.
persistence Maximum number of tries for randomly choosing samples. If we try this many times and the obtained labels are all the same, we give up (maybe the whole labels are the same) with the error message: " Not enough variation in the labels...".
minimum.class.size A lower bound on the number of samples in each class.
replace If TRUE, sampling is done by replacement.

Details

The function also returns a refined feature matrix by ignoring too sparse features after sampling.

Value

Returns a list of:

X_  The sampled feature matrix, each column is a feature after ignoring the redundant ones.
Y_  The vector of labels named according to the rows of X_.
remainder.samples The names of the rows of F_ which do not appear in X_, later on can be used for validation.
Author(s)
Habil Zare

References

See Also
FeaLect, train.doctor, doctor.validate, random.subset, compute.balanced, compute.logistic.score, ignore.redundant, input.check.FeaLect

Examples
library(FeaLect)
data(mcl_sll)
F <- as.matrix(mcl_sll[,-1]) # The Feature matrix
L <- as.numeric(mcl_sll[,1]) # The labels
names(L) <- rownames(F)
message(dim(F)[1], " samples and ",dim(F)[2], " features.")

XY <- random.subset(F_=F, L_=L, gamma=3/4, replace=TRUE)
XY$remainder.samples

train.doctor

 Fits various models based on a combination on penalized linear models and logistic regression.

Description
Various linear models are fitted to the training samples using lars method. The models differ in the number of features and each is validated by validating samples. A score is also assigned to each feature based on the tendency of LASSO in including that feature in the models.

Usage
train.doctor(F_, L_, training.samples, validating.samples, considered.features, maximum.features.num, balance = TRUE, return_linear.models = TRUE, report.fitting.failure = FALSE)

Arguments
F_ The feature matrix, each column is a feature.
L_ The vector of labels named according to the rows of F.
training.samples The names of rows of F that should be considered as training samples.
validating.samples
  The names of rows of F that should be considered as validating samples.

considered.features
  The names of columns of F that determine the features of interest.

maximum.features.num
  Up to this number of features are allowed to contribute to each linear model.

balance
  If TRUE, the cases will be balanced for the same number of positive vs. negatives by oversampling before fitting the linear model.

return_linear.models
  The models are memory intensive, so for if they more than 1000, we may decide to ignore them to prevent memory outage.

report.fitting.failure
  If TRUE, any failure in fitting the linear of logistic models will be printed.

Details
  See the reference for more details.

Value
  Returns a list of:

linear.models
  The result of model fitting computed by lars().

best.number.of.features
  According to best accuracy.

probabilities
  The best computed logistic score.

accuracy
  The best F-measure.

best.logistic.coef
  According to best accuracy.

contribution.to.feature.scores
  This vector should be added to the total feature scores.

contribution.to.feature.scores.frequency
  This vector should be added to the total frequency of features.

training.samples
  Input, the names of rows of F that should be considered as training samples.

validating.samples
  Input, the names of rows of F that should be considered as validating samples.

precision
  Ratio of number of true positives to predicted positives.

recall
  Ratio of number of true positives to real positives.

selected.features.sequence
  A list of sets of features which are selected in different models.

global.errors
  A vector of global error of the linear fits.

features.with.best.global.error
  A vector of names of good features in terms of global error of linear fits.
Note

Logistic regression is also done on top of fitting the linear models.

Author(s)

Habil Zare

References


See Also

FeaLect, train.doctor, doctor.validate, random.subset, compute.balanced, compute.logistic.score, ignore.redundant, input.check.FeaLect

Examples

library(FeaLect)
data(mcl_sll)
F <- as.matrix(mcl_sll[, -1]) # The Feature matrix
L <- as.numeric(mcl_sll[, 1]) # The labels
names(L) <- rownames(F)
message(dim(F)[1], " samples and ",dim(F)[2], " features.")

all.samples <- rownames(F); ts <- all.samples[5:10]; vs <- all.samples[c(1,22)]

doctor <- train.doctor(F_=F, L_=L, training.samples=ts, validating.samples=vs, considered.features=colnames(F), maximum.features.num=10)
Index

* **classif**
  compute.balanced, 3  
  compute.logistic.score, 5  
  doctor.validate, 6  
  FeaLect, 7  
  FeaLect-package, 2  
  ignore.redundant, 10  
  input.check.FeaLect, 11  
  random.subset, 13  
  train.doctor, 14  
* **datasets**
  mcl_sll, 12  
* **debugging**
  input.check.FeaLect, 11  
* **error**
  input.check.FeaLect, 11  
* **misc**
  input.check.FeaLect, 11  
* **models**
  compute.balanced, 3  
  compute.logistic.score, 5  
  doctor.validate, 6  
  FeaLect, 7  
  FeaLect-package, 2  
  ignore.redundant, 10  
  input.check.FeaLect, 11  
  random.subset, 13  
  train.doctor, 14  
* **multivariate**
  compute.balanced, 3  
  compute.logistic.score, 5  
  doctor.validate, 6  
  FeaLect, 7  
  FeaLect-package, 2  
  ignore.redundant, 10  
  input.check.FeaLect, 11  
  random.subset, 13  
  train.doctor, 14  
* **package**
  FeaLect-package, 2  
* **regression**
  compute.balanced, 3  
  compute.logistic.score, 5  
  doctor.validate, 6  
  FeaLect, 7  
  FeaLect-package, 2  
  ignore.redundant, 10  
  input.check.FeaLect, 11  
  random.subset, 13  
  train.doctor, 14  

compute.balanced, 3, 3, 4, 6, 7, 9–12, 14, 16  
compute.logistic.score, 3, 4, 5, 6, 7, 9–12, 14, 16  
doctor.validate, 3, 4, 6, 7, 9–12, 14, 16  
FeaLect, 3, 4, 6, 7, 9–12, 14, 16  
FeaLect-package, 2  
ignore.redundant, 3, 4, 6, 7, 9, 10, 11, 12, 14, 16  
input.check.FeaLect, 3, 4, 6, 7, 9–11, 11, 12, 14, 16  
lars, 3  
mcl_sll, 12  
random.subset, 3, 4, 6, 7, 9–12, 13, 14, 16  
train.doctor, 3, 4, 6, 7, 9–12, 14, 14, 16