Package ‘predmixcor’ documentation

of

February 21, 2008

Version 1.1-1

Title Classification rule based on Bayesian mixture models with feature selection bias corrected

Author Longhai Li <longhai@math.usask.ca>

Maintainer Longhai Li <longhai@math.usask.ca>

Depends R (>= 1.5.0)

Description “train_predict_mix” predicts the binary response with binary features

License GPL (>=2)

URL http://www.r-project.org, http://math.usask.ca/~longhai

R topics documented:

`gendata.mix` *Generate binary data with Bayesian mixture models*

Description

`gendata.mix` generates data (both training and test data) from Bayesian mixture model. The prior distribution of "theta" is uniform(0,1). The value of "alpha" is given by argument `alpha`, which controls the the overall relationship between the response and the predictor variables.

Usage

```
gendata.mix (n1,n2,m1,m2,p,alpha,prob.y=c(0.9,0.1))
```

Arguments

`n1`	the number of class 1 in training data
`n2`	the number of class 2 in training data
`m1`	the number of class 1 in test data
`m2`	the number of class 2 in test data
`p`	the number of features
`alpha`	a parameter controlling the dependency between the features and the response
`prob.y`	a vector of two elements specifying the probabilities of the response being 1 in each group

Value

`train`	the training data, with the row standing for the cases and the first column being the response
`test`	the test data, of the same format as "train"

See Also

`train_predict_mix`

`internal` *Internal functions used in package 'predmixcor'*

Description

this function is internal. Type the function name to see its definition

`train_predict_mix` *Classification rule based on Bayesian mixture models with feature selection bias corrected*

Description

`train_predict_mix` predicts the binary response based on high dimemsional binary features modeled with Bayesian mixture models. The model is trained with Gibbs sampling. A smaller number of features can be selected based on the correlations with the response. The bias due to the selection procedure can be corrected. The software is written entirely with R language.

Usage

```
train_predict_mix(
         test,train,k,
         theta0=0,alpha.shape=0.5,alpha.rate=5,no.alpha=30,
         common.alpha=FALSE,no.alpha0=100,
         mc.iters=200,iters.labeltheta=10,
         iters.theta=20,width.theta=0.1,
         correction=TRUE,no.theta.adj=30,approxim=TRUE,
         pred.start=100)
```

Arguments

`test`	a binary test data, a matrix, i.e. the data for which we want to predict the responses. The row stands for the cases. The first column is the binary response, which could be NA if they are missing.
`train`	a training data, of the same format as `test`
`k`	the number of features retained
`theta0`	the prior of "theta" is uniform over (`theta0`, `1-theta0`)
`alpha.shape`	the shape parameter of the Inverse Gamma, which is the prior distribution of "alpha"
`alpha.rate`	the rate parameter of the Inverse Gamma, as above
`no.alpha`	the number of "alpha"'s used in mid-point rule, which is used to approximate the integral with respect to "alpha".
`common.alpha`	Indicator whether the parameter "alpha" for the response (i.e "alpha0" in the reference) and the parameter "alpha" for the features are the same. By default they are two independent paramters with the same prior distribution, i.e, `common.alpha=FALSE`.
`no.alpha0`	the number of "alpha0"'s used in mid-point rule, which is used to approximate the integral with respect to "alpha0".. This parameter takes effect only when `common.alpha=FALSE`. Otherwise “alpha” and “alpha0” are the same.

`mc.iters`	iterations of Gibbs sampling used to train the model.
`iters.labeltheta`	In each Gibbs iteration, the combination of updating the “labels” once and updating the “theta” is repeated `iters.labeltheta` times, and then “alpha” and “alpha0” are updated once.
`iters.theta`	iterations of updating "theta" using M-H method.
`width.theta`	the proposal distribution used to update "theta" with Metropolis-Hastings method is uniform over the interval (current "theta" +- `width.theta`).
`correction`	Indicator whether the correction method shall be applied
`no.theta.adj`	the parameter in Simpson’s rule used to evaluate the integration w.r.t. "theta", which is needed in calculating the adjustment factor. The integrant is evaluated at 2*(`no.theta.adj`)+1 points.
`approxim`	Indicator whether the adjustment factor is ignored in updating the labels (laten values). In theory it should be considered. However, it has little actual effect, but costs much computation, since we need to recompute the adjustment factor when updating the label of each case. By default, `approxim=TRUE`
`pred.start`	The Markov chain iterations after `pred.start` will be used to make Monte Carlo estimation

Value

`prediction`	a matrix showing the detailed prediction result: the 1st column being the true responses, the 2nd being the predicted responses, the 3rd being the predictive probabilities of class 1 and the 4th being the indicator whether wrong prediction is made.
`aml`	the average minus log probabilities
`error.rate`	the ratio of wrong prediction
`mse`	the average square error of the predictive probabilities
`summary.pred`	tabular display of the predictive probabilities and the actual fraction of class 1.
`features.selected`	The features selected using correlation criterion
`label`	the Markov chain samples of latent values, with each column for an iteration. The number of rows of `label` is equal to the number of training cases.
`I1`	the number of “1”s of features (columns) in those cases labeled by “1”, counted for each Markov chain iterations (row).
`I2`	Similar as `I1`, but for those cases labeled by “2”.
`N1`	a vector recording the number of cases labeled by “1” for each Markov chain iteration.
`N2`	a vector recording the number of cases labeled by “2” for each Markov chain iteration.
`theta`	Markov chain samples of “theta". Each row is an iteration.
`alpha`	a vector storing the Markov chain samples of “alpha”.
`alpha0`	a vector storing the Markov chain samples of “alpha0”.

`alpha_set`	all the possible values the "alpha" can take. The prior of "alpha" is approximated by the uniform over this set.
`alpha0_set`	all the possible values the "alpha0" can take. The prior of "alpha0" is approximated by the uniform over this set.

References

http://math.usask.ca/ longhai/publication.html

See Also

`gendata.mix`

Examples

```
#simulating data set from a Bayesian mixture model
data <- gendata.mix(20,20,50,50,101,10,c(0.9,0.1))

#training the model using Gibbs sampling, without correcting for the feature
#selection bias, then testing on predicting the responses of the test cases,

predict.uncor <- train_predict_mix(
            test=data$test,train=data$train,k=5,
            theta0=0,alpha.shape=0.5,alpha.rate=5,no.alpha=5,
            common.alpha=FALSE,no.alpha0=100,
            mc.iters=30,iters.labeltheta=1,
            iters.theta=10,width.theta=0.1,
            correction=FALSE,no.theta.adj=5,approxim=TRUE,
            pred.start=10)

#As above, but with the feature selection bias corrected
predict.cor <-   train_predict_mix(
            test=data$test,train=data$train,k=5,
            theta0=0,alpha.shape=0.5,alpha.rate=5,no.alpha=5,
            common.alpha=FALSE,no.alpha0=100,
            mc.iters=30,iters.labeltheta=1,
            iters.theta=10,width.theta=0.1,
            correction=TRUE,no.theta.adj=5,approxim=TRUE,
            pred.start=10)
```

Index