Package 'PoiClaClu'

January 4, 2019

Type Package

Title Classification and Clustering of Sequencing Data Based on a Poisson Model

Version 1.0.2.1

Date 2013-01-02

Author Daniela Witten

Maintainer Daniela Witten <dwitten@u.washington.edu>

Description Implements the methods described in the paper, Witten (2011) Classification and Clustering of Sequencing Data using a Poisson Model, Annals of Applied Statistics 5(4) 2493-2518.

License GPL-2

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2019-01-04 17:05:57 UTC

R topics documented:

PoiClaClu-package 2
Classify 3
Classify.cv 5
ColorDendrogram 7
CountDataSet 8
FindBestTransform 9
PoissonDistance 10

Index **12**

PoiClaClu-package *Classification and clustering of RNA sequencing data using a Poisson model*

Description

A simple approach for performing classification and clustering of samples for which RNA sequencing data is available. Based upon a simple Poisson model proposed by a number of authors (e.g. Marioni et al Genome Research 2008, Bullard et al BMC Bioinformatics 2010, and others).

Details

Package:	PoiClaClu
Type:	Package
Version:	1.0.2
Date:	2013-01-02
License:	GPL-2
LazyLoad:	yes

Author(s)

Daniela Witten

Maintainer: Daniela Witten <dwitten@u.washington.edu>

References

D. Witten (2011) Classification and clustering of sequencing data using a Poisson model. Annals of Applied Statistics 5(4): 2493-2518.

Examples

```
# Poisson clustering #
set.seed(1)
dat <- CountDataSet(n=20,p=100,sdsignal=.5,K=4,param=10)
dd <- PoissonDistance(dat$x, type="mle")
print(dd)
ColorDendrogram(hclust(dd$dd), y=dat$y)

# Poisson classification #
set.seed(1)
dat <- CountDataSet(n=20,p=100,sdsignal=.1,K=4,param=10)
out <- Classify(x=dat$x,y=dat$y,xte=dat$xte,rhos=c(0,5,10))
print(out)
```

`Classify` *Classify observations using a Poisson model.*

Description

Classify observations using a simple Poisson model. This function implements the "sparse Poisson linear discriminant analysis classifier", which is similar to linear discriminant analysis but assumes a Poisson model rather than a Gaussian model for the data. The classifier soft-thresholds the estimated effect of each feature in order to achieve sparsity.

Usage

```
Classify(x, y, xte=NULL, rho = 0, beta = 1, rhos = NULL, type=c("mle","deseq","quantile"),
prior = NULL, transform=TRUE, alpha=NULL)
```

Arguments

`x`	A n-by-p training data matrix; n observations and p features. Used to train the classifier.
`y`	A numeric vector of class labels of length n: 1, 2, ..., K if there are K classes. Each element of y corresponds to a row of x; i.e. these are the class labels for the observations in x.
`xte`	A m-by-p data matrix: m test observations and p features. The classifier fit on the training data set x will be tested on this data set. If NULL, then testing will be performed on the training set.
`rho`	Tuning parameter controlling the amount of soft thresholding performed, i.e. the level of sparsity, i.e. number of nonzero features in classifier. Rho=0 means that there is no soft-thresolding, i.e. all features used in classifier. Larger rho means that fewer features will be used.
`beta`	A smoothing term. A Gamma(beta,beta) prior is used to fit the Poisson model. Recommendation is to just leave it at 1, the default value.
`rhos`	A vector of tuning parameters that control the amount of soft thresholding performed. If "rhos" is provided then a number of models will be fit (one for each element of "rhos"), and a number of predicted class labels will be output (one for each element of "rhos").
`type`	How should the observations be normalized within the Poisson model, i.e. how should the size factors be estimated? Options are "quantile" or "deseq" (more robust) or "mle" (less robust). In greater detail: "quantile" is quantile normalization approach of Bullard et al 2010 BMC Bioinformatics, "deseq" is median of the ratio of an observation to a pseudoreference obtained by taking the geometric mean, described in Anders and Huber 2010 Genome Biology and implemented in Bioconductor package "DESeq", and "mle" is the sum of counts for each sample; this is the maximum likelihood estimate under a simple Poisson model.

prior	Vector of length equal to the number of classes, representing prior probabilities for each class. If NULL then uniform priors are used (i.e. each class is equally likely).
transform	Should data matrices x and xte first be power transformed so that it more closely fits the Poisson model? TRUE or FALSE. Power transformation is especially useful if the data are overdispersed relative to the Poisson model.
alpha	If transform=TRUE, this determines the power to which the data matrices x and xte are transformed. If alpha=NULL then the transformation that makes the Poisson model best fit the data matrix x is computed. (Note that alpha is computed based on x, not based on xte). Or a value of alpha, 0<alpha<=1, can be entered by the user.

Value

ytehat	The predicted class labels for each of the test observations (rows of xte).
discriminant	A m-by-K matrix, where K is the number of classes. The (i,k) element is large if the ith element of xte belongs to class k.
ds	A K-by-p matrix indicating the extent to which each feature is under- or over-expressed in each class. The (k,j) element is >1 if feature j is over-expressed in class k, and is <1 if feature j is under-expressed in class k. When rho is large then many of the elemtns of this matrix are shrunken towards 1 (no over- or under-expression).
alpha	Power transformation used (if transform=TRUE).

Author(s)

Daniela Witten

References

D Witten (2011) Classification and clustering of sequencing data using a Poisson model. To appear in Annals of Applied Statistics.

See Also

Classify.cv

Examples

```
set.seed(1)
dat <- CountDataSet(n=40,p=500,sdsignal=.1,K=3,param=10)
cv.out <- Classify.cv(dat$x,dat$y)
print(cv.out)
out <- Classify(dat$x,dat$y,dat$xte,rho=cv.out$bestrho)
print(out)
cat("Confusion matrix for predicted and true test class labels:", fill=TRUE)
print(table(out$ytehat,dat$yte))
```

Classify.cv *Function to do cross-validation for Poisson classification.*

Description

Perform cross-validation for the function that implements the "sparse Poisson linear discriminant analysis classifier", which is similar to linear discriminant analysis but assumes a Poisson model rather than a Gaussian model for the data. The classifier soft-thresholds the estimated effect of each feature in order to achieve sparsity. This cross-validation function selects the proper value of the tuning parameter that controls the level of soft-thresholding.

Usage

```
Classify.cv(x, y, rhos = NULL, beta = 1, nfolds = 5, type=c("mle","deseq","quantile"),
folds = NULL, transform=TRUE, alpha=NULL, prior=NULL)
```

Arguments

x	A n-by-p training data matrix; n observations and p features.
y	A numeric vector of class labels of length n: 1, 2, ..., K if there are K classes. Each element of y corresponds to a row of x; i.e. these are the class labels for the observations in x.
rhos	A vector of tuning parameters to try out in cross-validation. Rho controls the level of shrinkage performed, i.e. the number of features that are not involved in the classifier. When rho=0 then all features are involved in the classifier, and when rho is very large no features are involved. If rhos=NULL then a vector of rho values will be chosen automatically.
beta	A smoothing term. A Gamma(beta,beta) prior is used to fit the Poisson model. Recommendation is to leave it at 1, the default value.
nfolds	The number of folds in the cross-validation; default is 5-fold cross-validation.
type	How should the observations be normalized within the Poisson model, i.e. how should the size factors be estimated? Options are "quantile" or "deseq" (more robust) or "mle" (less robust). In greater detail: "quantile" is quantile normalization approach of Bullard et al 2010 BMC Bioinformatics, "deseq" is median of the ratio of an observation to a pseudoreference obtained by taking the geometric mean, described in Anders and Huber 2010 Genome Biology and implemented in Bioconductor package "DESeq", and "mle" is the sum of counts for each sample; this is the maximum likelihood estimate under a simple Poisson model.
prior	Vector of length equal to the number of classes, representing prior probabilities for each class. If NULL then uniform priors are used (i.e. each class is equally likely).
transform	Should data matrices x and xte first be power transformed so that it more closely fits the Poisson model? TRUE or FALSE. Power transformation is especially useful if the data are overdispersed relative to the Poisson model.

alpha	If transform=TRUE, this determines the power to which the data matrices x and xte are transformed. If alpha=NULL then the transformation that makes the Poisson model best fit the data matrix x is computed. (Note that alpha is computed based on x, not based on xte). Or a value of alpha, 0<alpha<=1, can be entered by the user.
folds	Instead of specifying the number of folds in cross-validation, one can explicitly specify the folds. To do this, input a list of length r (to perform r-fold cross-validation). The rth element of the list should be a vector containing the indices of the test observations in the rth fold.

Value

errs	A matrix of dimension (number of folds)-by-(length of rhos). The (i,j) element is the number of errors occurring in the ith cross-validation fold for the jth value of the tuning parameter, i.e. rhos[j].
bestrho	The tuning parameter value resulting in the lowest overall cross-validation error rate.
rhos	The vector of rho values used in cross-validation.
nnonzero	A matrix of dimension (number of folds)-by-(length of rhos). The (i,j) element is the number of features included in the classifier occurring in the ith cross-validation fold for the jth value of the tuning paramer.
folds	Cross-validation folds used.
alpha	Power transformation used (if transform=TRUE).

Author(s)

Daniela Witten

References

D Witten (2011) Classification and clustering of sequencing data using a Poisson model. To appear in Annals of Applied Statistics.

Examples

```
set.seed(1)
dat <- CountDataSet(n=40,p=500,sdsignal=.1,K=3,param=10)
cv.out <- Classify.cv(dat$x,dat$y)
print(cv.out)
out <- Classify(dat$x,dat$y,dat$xte,rho=cv.out$bestrho)
print(out)
cat("Confusion matrix comparing predicted class labels to true class
labels for training observations:", fill=TRUE)
print(table(out$ytehat,dat$yte))
```

ColorDendrogram *Color the leaves in a hierarchical clustering dendrogram*

Description

Pass in the output of "hclust" and a class label for each observation. A colored dendrogram will result, with the leaf colors indicating the classes.

Usage

```
ColorDendrogram(hc, y, main = "", branchlength = 0.7, labels = NULL,
xlab = NULL, sub = NULL, ylab = "", cex.main = NULL)
```

Arguments

hc	The output of running "hclust" on a nxn dissimilarity matrix
y	A vector of n class labels for the observations that were clustered using "hclust". If labels are numeric from 1 to K, then colors will be determine automatically. Otherwise the labels can take the form of colors (e.g. c("red", "red", "orange", "orange")).
main	The main title for the dendrogram.
branchlength	How long to make the colored part of the branches. Adjustment will be needed for each dissimilarity matrix
labels	The labels for the n observations.
xlab	X-axis label.
sub	Sub-x-axis label.
ylab	Y-axis label.
cex.main	The amount by which to enlarge the main title for the figure.

Author(s)

Daniela Witten

Examples

```
set.seed(1)
dat <- CountDataSet(n=20,p=100,sdsignal=2,K=4,param=10)
dd <- PoissonDistance(dat$x,type="mle")
ColorDendrogram(hclust(dd$dd), y=dat$y, branchlength=10)
```

`CountDataSet` *Generate a simulated sequencing data set using a negative binomial model.*

Description

Generate two nxp data sets: a training set and a test set, as well as outcome vectors y and yte of length n indicating the class labels of the training and test observations.

Usage

```
CountDataSet(n, p, K, param, sdsignal)
```

Arguments

`n`	Number of observations desired.
`p`	Number of features desired. Note that 30% of the features will differ between classes, though some of those differences may be small.
`K`	Number of classes desired. Note that the function requires that n be at least equal to 4K – i.e. there must be at least 4 observations per class on average.
`param`	The dispersion parameter for the negative binomial distribution. The negative binomial distribution is parameterized using "mu" and "size" in the R function "rnbinom". That is, Y ~ NB(mu, param) means that E(Y)=mu and Var(Y) = mu+mu^2/param. So when param is very large this is essentially a Poisson distribution, and when param is smaller then there is a lot of overdispersion relative to the Poisson distribution.
`sdsignal`	The extent to which the classes are different. If this equals zero then there are no class differences and if this is large then the classes are very different.

Details

This is based in part on a function in the DESeq Bioconductor package (Anders and Huber 2010 Genome Biology) for generating a simulated RNA sequencing data set.

Value

`x`	nxq data matrix. May have q<p because features with 0 total counts are removed.
`y`	class labels for the n observations in x.
`xte`	nxq data matrix of test observations; the q features are those with >0 total counts in x. So q<=p.
`yte`	class labels for the n observation in xte.

Author(s)

Daniela Witten, based on software written by Anders and Huber in the DESeq Bioconductor package.

Examples

```
set.seed(1)
dat <- CountDataSet(n=20,p=100,sdsignal=2,K=4,param=10)
dd <- PoissonDistance(dat$x,type="mle", transform=TRUE)
```

`FindBestTransform` *Find the power transformation that makes a data set approximately Poisson.*

Description

Find a constant alpha, 0<alpha<=1, such that x raised to the power alpha approximately follows the simple Poisson log linear model that says that the (i,j) element of x is Poisson with mean si times gj, where si is a sample-specific term and gj is a feature-specific term. Alpha is selected via a grid search.

Usage

```
FindBestTransform(x)
```

Arguments

x	A n-by-p matrix of sequencing data, with n observations and p features.

Value

Returns alpha, the power to which x should be raised.

Author(s)

Daniela Witten

References

D Witten (2011) Classification and clustering of sequencing data using a Poisson model. To appear in Annals of Applied Statistics.

Examples

```
set.seed(1)
dat <- CountDataSet(n=20,p=100,sdsignal=2,K=4,param=10)
alpha <- FindBestTransform(dat$x)
# This is the best transformation!
dd <- PoissonDistance(dat$x^alpha,type="mle", transform=FALSE)
# OR we could get the samething automatically:
dd2 <- PoissonDistance(dat$x,type="mle",transform=TRUE)
# or like this:
dd3 <- PoissonDistance(dat$x,type="mle",transform=TRUE,alpha=alpha)
ColorDendrogram(hclust(dd$dd), y=dat$y, branchlength=10)
```

PoissonDistance *Given a n-by-p data matrix, compute the corresponding n-by-n Poisson dissimilarity matrix.*

Description

This function computes a Poisson dissimilarity matrix as described in the paper referenced below, and is intended to be applied to a data matrix of counts resulting from a sequencing experiment. The (i,k) element of the Poisson dissimilarity matrix is the dissimilarity between observations i and k of the data matrix x: that is, the log likelihood ratio statistic under a simple Poisson model.

Usage

```
PoissonDistance(x, beta = 1, type=c("mle","deseq","quantile"),
transform=TRUE, alpha=NULL, perfeature=FALSE)
```

Arguments

x	A n-by-p data matrix with observations on the rows, and p features on the columns. The (i,j) element of x is the number of reads in observation i that mapped to feature (e.g. gene or exon) j.
beta	A smoothing term; essentially the parameter beta in a Gamma(beta,beta) prior used to estimate the log likelihood ratio statistic for computing the dissimilarity between a pair of observations. Recommended to leave it at 1, the default value.
type	How should the observations be normalized within the Poisson model, i.e. how should the size factors be estimated? Options are "quantile" or "deseq" (more robust) or "mle" (less robust). In greater detail: "quantile" is quantile normalization approach of Bullard et al 2010 BMC Bioinformatics, "deseq" is median of the ratio of an observation to a pseudoreference obtained by taking the geometric mean, described in Anders and Huber 2010 Genome Biology and implemented in Bioconductor package "DESeq", and "mle" is the sum of counts for each sample; this is the maximum likelihood estimate under a simple Poisson model.
transform	Should data matrix x first be power transformed so that it more closely fits the Poisson model? TRUE or FALSE. Power transformation is especially useful if the data are overdispersed relative to the Poisson model.
alpha	If transform=TRUE, this determines the power to which the data matrix x is transformed. If alpha=NULL then the transformation that makes the Poisson model best fit the data is computed. Or a value of alpha, 0<alpha<=1, can be entered by the user.
perfeature	If perfeature=TRUE, then in addition to the nxn dissimilarity matrix, a nxnxp array will be returned. Its elements will be the contributions of each of the p features to the nxn dissimilarity matrix; summing over the 3rd index will simply give back the nxn dissimilarity matrix.

Details

More details can be found in the paper referenced below.

Value

dd	A nxn Poisson dissimilarity matrix, containing pairwise dissimilarities between observations based on the original nxp data matrix x input by the user.
alpha	Power to which data was transformed before computing dissimilarity matrix, if transform was TRUE. This was either input by the user, or computed automatically if not specified.
x	Data used to compute dissimilarity matrix, this will be x raised to the power alpha.
ddd	If perfeature=TRUE, then this is the nxnxp array containing the contribution of each feature to the nxn dissimilarity matrix.

Author(s)

Daniela Witten

References

D Witten (2011) Classification and clustering of sequencing data using a Poisson model. To appear in Annals of Applied Statistics.

See Also

FindBestTransform

Examples

```
set.seed(1)
dat <- CountDataSet(n=20,p=100,sdsignal=2,K=4,param=10)
dd <- PoissonDistance(dat$x,type="mle")
print(dd)
ColorDendrogram(hclust(dd$dd), y=dat$y, branchlength=10)
```

Index

Classify, 3
Classify.cv, *4*, 5
ColorDendrogram, 7
CountDataSet, 8

FindBestTransform, 9, *11*

PoiClaClu *(*PoiClaClu-package*)*, 2
PoiClaClu-package, 2
PoissonDistance, 10