Package 'crimelinkage'

September 19, 2015

Title Statistical Methods for Crime Series Linkage

Version 0.0.4

Description Statistical Methods for Crime Series Linkage. This package provides code for criminal case linkage, crime series identification, crime series clustering, and suspect identification.

Depends R (>= 3.1.0)

License GPL-3

LazyData true

Date 2015-09-18

BugReports <mporter@cba.ua.edu>

Imports igraph, geosphere, grDevices, graphics, stats, utils

Suggests fields, knitr, gbm

VignetteBuilder knitr

NeedsCompilation no

Author Michael Porter [aut, cre],
Brian Reich [aut]

Maintainer Michael Porter <mporter@cba.ua.edu>

Repository CRAN

Date/Publication 2015-09-19 19:50:30

R topics documented:

`crimelinkage-package` `crimelinkage` *package: Statistical Methods for Crime Series Linkage*

Description

Code for criminal case linkage, crime series identification, crime series clustering, and suspect identification.

Details

The basic inputs will be a data.frame of crime incidents and an offenderTable data.frame that links offenders to (solved) crimes.

The crime incident data must have one column named `crimeID` that provides a unique crime identifier. Other recognized columns include: spatial information: `X`, `Y` which can be in metric or long/lat; `DT.FROM`, `DT.TO` for the event times (these must be of class POSIXct). Other columns containing information about the crime, crime scene, or suspect can be included as well.

The offenderTable must have columns: `crimeID` (unique crime identifier) and `offenderID` (unique offender identifier).

See the vignettes for more details.

bayesPairs *Extracts the crimes with the largest probability of being linked.*

Description

Extracts the crimes (from crimeClust_bayes) with the largest probability of being linked.

Usage

```
bayesPairs(p.equal, drop = 0)

bayesProb(prob, drop = 0)
```

Arguments

p.equal	the posterior probability matrix produced by crimeClust_bayes
drop	only return crimes with a posterior linkage probability that exceeds drop. Set to NA to return all results.
prob	a column (or row) of the posterior probability matrix produced by crimeClust_bayes

Details

This is a helper function to easily extract the crimes with a high probability of being linked from the output of crimeClust_bayes. bayesPairs searches the full posterior probability matrix and bayesProb only searches a particular column (or row).

Value

data.frame of the indices of crimes with estimated posterior probabilities, ordered from largest to smallest

See Also

crimeClust_bayes

clusterPath *Follows path of one crime up a dendrogram*

Description

The sequence of groups that a crime belongs to.

Usage

```
clusterPath(crimeID, tree)
```

Arguments

crimeID	the crime ID for a crime used in hierarchical clustering
tree	an object produced from crimeClust_hier

Details

Agglomerative hierarchical clustering form clusters by sequentially merging the most similar groups at each iteration. This function is designed to help trace the sequence of groups an individual crime is a member of. And it shows at what score (log Bayes factor) the merging occurred.

Value

data.frame of the additional crimes and the log Bayes factor at each merge.

See Also

crimeClust_hier, plot_hcc

Examples

```
# See vignette: "Crime Series Identification and Clustering" for usage.
```

---

compareCrimes *Creates evidence variables by calculating 'distance' between crime pairs*

Description

Calculates spatial and temporal distance, difference in categorical, and absolute value of numerical crime variables

Usage

```
compareCrimes(Pairs, crimedata, varlist, binary = TRUE, longlat = FALSE,
  show.pb = FALSE, ...)
```

Arguments

Pairs	(n x 2) matrix of crimeIDs
crimedata	data.frame of crime incident data. There must be a column named crimedata that refers to the crimeIDs given in Pairs. Other column names must correspond to what is given in varlist list.
varlist	a list with elements named: crimeID, spatial, temporal, categorical, and numerical. Each element should be a vector of the column names of crimedata corresponding to that feature: • crimeID: crime ID for the crimedata that is matched to Pairs

- spatial: X,Y coordinates (in long,lat or Cartesian) of crimes
- temporal: DT.FROM, DT.TO of crimes. If times are uncensored, then only DT.FROM needs to be provided.
- categorical: (optional) categorical crime variables
- numerical: (optional) numerical crime variables

`binary`	(logical) match/no match or all combinations for categorical data
`longlat`	(logical) are spatial coordinates in (long,lat)?
`show.pb`	(logical) show the progress bar
`...`	other arguments passed to hidden functions

Value

data.frame of various proximity measures between the two crimes

- If `spatial` data is provided: the euclidean distance (if `longlat = FALSE`) or Haversine great circle distance (`distHaversine` if `longlat = TRUE`) is returned (in kilometers).
- If `temporal` data is provided: the expected absolute time difference is returned:
  - temporal - overall difference (in days) [0,max]
  - tod - time of day difference (in hours) [0,12]
  - dow - fractional day of week difference (in days) [0,3.5]
- If `categorical` data is provided: if `binary = TRUE` then a 1 if the categories of each crime match and a 0 if they do not match. If `binary = FALSE`, then a factor of merged values (in form of f1:f2)
- If `numerical` data is provided: the absolute difference is returned.

References

Porter, M. D. (2014). A Statistical Approach to Crime Linkage. *arXiv preprint arXiv:1410.2285.*. http://arxiv.org/abs/1410.2285

Examples

```
data(crimes)
 pairs = t(combn(crimes$crimeID[1:4],m=2))   # make some crime pairs

 varlist = list(
   spatial = c("X", "Y"),
   temporal = c("DT.FROM","DT.TO"),
   categorical = c("MO1",  "MO2", "MO3"))    # crime variables list

 compareCrimes(pairs,crimes,varlist,binary=TRUE)
```

`crimeClust_bayes` *Bayesian model-based partially-supervised clustering for crime series identification*

Description

Bayesian model-based partially-supervised clustering for crime series identification

Usage

```
crimeClust_bayes(crimeID, spatial, t1, t2, Xcat, Xnorm, maxcriminals = 1000,
  iters = 10000, burn = 5000, plot = TRUE, update = 100, seed = NULL,
  use_space = TRUE, use_time = TRUE, use_cats = TRUE)
```

Arguments

`crimeID`	n-vector of criminal IDs for the n crimes in the dataset. For unsolved crimes, the value should be `NA`.
`spatial`	(n x 2) matrix of spatial locations, represent missing locations with `NA`
`t1`	earliest possible time for crime
`t2`	latest possible time for crime. Crime occurred between `t1` and `t2`.
`Xcat`	(n x q) matrix of categorical crime features. Each column is a variable, such as mode of entry. The different factors (window, door, etc) should be coded as integers 1,2,. . . ,m.
`Xnorm`	(n x p) matrix of continuous crime features.
`maxcriminals`	maximum number of clusters in the model.
`iters`	Number of MCMC samples to generate.
`burn`	Number of MCMC samples to discard as burn-in.
`plot`	(logical) Should plots be produced during run.
`update`	Number of MCMC iterations between graphical displays.
`seed`	seed for random number generation
`use_space`	(logical) should the spatial locations be used in clustering?
`use_time`	(logical) should the event times be used in clustering?
`use_cats`	(logical) should the categorical crime features be used in clustering?

Value

(list) p.equal is the (n x n) matrix of probabilities that each pair of crimes are committed by the same criminal.

if `plot=TRUE`, then progress plots are produced.

Author(s)

Brian J. Reich

References

Reich, B. J. and Porter, M. D. (2015), Partially supervised spatiotemporal clustering for burglary crime series identification. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*. 178:2, 465–480. http://www4.stat.ncsu.edu/~reich/papers/CrimeClust.pdf

See Also

bayesPairs

Examples

```
# Toy dataset with 12 crimes and three criminals.

 # Make IDs: Criminal 1 committed crimes 1-4, etc.
 id <- c(1,1,1,1,
         2,2,2,2,
                 3,3,3,3)

 # spatial locations of the crimes:
 s <- c(0.8,0.9,1.1,1.2,
        1.8,1.9,2.1,2.2,
        2.8,2.9,3.1,3.2)
 s <- cbind(0,s)

 # Categorical crime features, say mode of entry (1=door, 2=other) and
 # type of residence (1=apartment, 2=other)
 Mode <- c(1,1,1,1,  #Different distribution by criminal
           1,2,1,2,
           2,2,2,2)
 Type <- c(1,2,1,2,  #Same distribution for all criminals
           1,2,1,2,
           1,2,1,2)
 Xcat <- cbind(Mode,Type)

 # Times of the crimes
 t <- c(1,2,3,4,
        2,3,4,5,
        3,4,5,6)

 # Now let's pretend we don't know the criminal for crimes 1, 4, 6, 8, and 12.
 id <- c(NA,1,1,NA,2,NA,2,NA,3,3,3,NA)

 # Fit the model (nb: use much larger iters and burn on real problem)
 fit <- crimeClust_bayes(crimeID=id, spatial=s, t1=t,t2=t, Xcat=Xcat,
                   maxcriminals=12,iters=500,burn=100,update=100)

 # Plot the posterior probability matrix that each pair of crimes was
 # committed by the same criminal:
```

```
if(require(fields,quietly=TRUE)){
fields::image.plot(1:12,1:12,fit$p.equal,
          xlab="Crime",ylab="Crime",
          main="Probability crimes are from the same criminal")
}

# Extract the crimes with the largest posterior probability
bayesPairs(fit$p.equal)
bayesProb(fit$p.equal[1,])
```

`crimeClust_hier` *Agglomerative Hierarchical Crime Series Clustering*

Description

Run hierarchical clustering on a set of crimes using the log Bayes Factor as the similarity metric.

Usage

```
crimeClust_hier(crimedata, varlist, estimateBF, linkage = c("average",
  "single", "complete"), ...)
```

Arguments

`crimedata`	data.frame of crime incidents. Must contain a column named `crimeID`.
`varlist`	a list of the variable names (columns of `crimedata`) used to create evidence variables with `compareCrimes`.
`estimateBF`	function to estimate the log bayes factor from evidence variables
`linkage`	the type of linkage for hierarchical clustering • "average" uses the average bayes factor • "single" uses the largest bayes factor (most similar) • "complete" uses the smallest bayes factor (least similar)
`...`	other arguments passed to `compareCrimes`

Details

This function first compares all crime pairs using `compareCrimes`, then uses `estimateBF` to estimate the log Bayes factor for every pair. Next, it passes this information into `hclust` to carry out the agglomerative hierarchical clustering. Because `hclust` requires a dissimilarity, this uses the negative log Bayes factor.

The input `varlist` is a list with elements named: crimeID, spatial, temporal, categorical, and numerical. Each element should be a vector of the column names of `crimedata` corresponding to that feature. See `compareCrimes` for more details.

Value

An object of class `hclust` (from `hclust`).

References

Porter, M. D. (2014). A Statistical Approach to Crime Linkage. *arXiv preprint arXiv:1410.2285.*. http://arxiv.org/abs/1410.2285

See Also

clusterPath, plot_hcc

Examples

```
data(crimes)
 #- cluster the first 10 crime incidents
 crimedata = crimes[1:10,]
 varlist = list(spatial = c("X", "Y"), temporal = c("DT.FROM","DT.TO"),
     categorical = c("MO1",  "MO2", "MO3"))
 estimateBF <- function(X) rnorm(NROW(X))   # random estimation of log Bayes Factor
 HC = crimeClust_hier(crimedata,varlist,estimateBF)
 plot_hcc(HC,yticks=-2:2)

 # See vignette: "Crime Series Identification and Clustering" for more examples.
```

---

crimes *Ficticious dataset of crime events*

Description

Some realistic, but fictious, crime incident data.

Usage

```
data(crimes)
```

Format

490 crime events

crimeID The crime ID number

X, Y Spatial coordinates

MO1 A categorical MO variable that takes values 1,. . . ,31

MO2 A categorical MO variable that takes values a,. . . ,h

MO3 A categorical MO variable that takes values A,. . . ,O

DT.FROM The earliest possible Date-time of the crime.

DT.TO The latest possible Date-time of the crime

Source

Ficticious data, but hopefully realistic

Examples

```
head(crimes)
```

---

getBF *Estimates the bayes factor for continous and categorical predictors.*

Description

This adds pseudo counts to each bin count to give df effective degrees of freedom. Must have all possible factor levels and must be of factor class.

Usage

```
getBF(x, y, weights, breaks = NULL, df = 5)
```

Arguments

x	predictor vector (continuous or categorical/factors)
y	binary vector indicating linkage (1 = linked, 0 = unlinked) or logical vector (TRUE = linked, FALSE = unlinked)
weights	a vector of observation weights or the column name in data that corresponds to the weights.
breaks	set of break point for continuous predictors or NULL for categorical or discrete
df	the effective degrees of freedom for the cetegorical density estimates

Details

Continous predictors are first binned, then estimates shrunk towards zero.

Value

data.frame containing the levels/categories with estimated Bayes factor

Note

Give linked and unlinked a different prior according to sample size

Examples

```
# See vignette: "Statistical Methods for Crime Series Linkage" for usage.
```

getCrimes *Generate a list of crimes for a specific offender*

Description

Generate a list of crimes for a specific offender

Usage

```
getCrimes(offenderID, crimedata, offenderTable)
```

Arguments

offenderID	an offender ID that is in offenderTable
crimedata	data.frame of crime incident data. crimedata must be a data.frame with a column named: crimeID
offenderTable	offender table that indicates the offender(s) responsible for solved crimes. offenderTable must have columns named: offenderID and crimeID.

Value

The subset of crimes in crimedata that are attributable to the offender named offenderID

See Also

getCrimeSeries

Examples

```
data(crimes)
 data(offenders)

 getCrimes("O:40",crimes,offenders)
```

getCrimeSeries *Generate a list of offenders and their associated crime series.*

Description

Generate a list of offenders and their associated crime series.

Usage

```
getCrimeSeries(offenderID, offenderTable, restrict = NULL, show.pb = FALSE)
```

Arguments

offenderID	vector of offender IDs
offenderTable	offender table that indicates the offender(s) responsible for solved crimes. offenderTable must have columns named: offenderID and crimeID.
restrict	if vector of crimeID, then only include those crimeIDs in offenderTable. If NULL, then return all crimes for offender.
show.pb	(logical) should a progress bar be displayed

Value

List of offenders with their associated crime series.

See Also

makeSeriesData, getCriminals, getCrimes

Examples

```
data(offenders)

 getCrimeSeries("O:40",offenders)
 getCrimeSeries(c("O:40","O:3"),offenders)  # list of crime series from multiple offenders
```

---

getCriminals *Lookup the offenders responsible for a set of solved crimes*

Description

Generates the IDs of criminals responsible for a set of solved crimes using the information in offenderTable.

Usage

```
getCriminals(crimeID, offenderTable)
```

Arguments

crimeID	crimeID(s) of solved crimes.
offenderTable	offender table that indicates the offender(s) responsible for solved crimes. offenderTable must have columns named: offenderID and crimeID.

Value

Vector of offenderIDs responsible for crimes labeled crimeID.

See Also

getCrimeSeries

Examples

```
data(offenders)

 getCriminals("C:1",offenders)

 getCriminals("C:78",offenders)                      # shows co-offenders

 getCriminals(c("C:26","C:78","85","110"),offenders) # all offenders from a crime series
```

---

getROC *Cacluate ROC like metrics.*

Description

Orders scores from largest to smallest and evaluates performance for each value. This assumes an analyst will order the predicted scores and start investigating the linkage claim in this order.

Usage

```
getROC(f, y)
```

Arguments

f	predicted score for linkage
y	truth; linked=1, unlinked=0

Value

data.frame of evaluation metrics:

- FPR - false positive rate - proportion of unlinked pairs that are incorrectly assessed as linked
- TPR - true positive rate; recall; hit rate - proportion of all linked pairs that are correctly assessed as linked
- PPV - positive predictive value; precision - proportion of all pairs that are predicted linked and truely are linked
- Total - the number of cases predicted to be linked
- TotalRate - the proportion of cases predicted to be linked
- threshold - the score threshold that produces the results

Examples

```
f = 1:10
y = rep(0:1,length=10)
getROC(f,y)
```

linkage *Hierarchical Based Linkage*

Description

Groups the Bayes Factors by crime group and calculates the linkage score for each group.

Usage

```
linkage(BF, group, method = c("average", "single", "complete"))
```

Arguments

BF	vector of Bayes Factors
group	crime group
method	the type of linkage for comparing a crime to a set of crimes • "average" uses the average bayes factor • "single" uses the largest bayes factor (most similar) • "complete" uses the smallest bayes factor (least similar)

Details

If methods is a vector of linkages to use, then the all linkages are calcualted and ordered according to the first element.

Value

a data.frame of the Bayes Factor scores ordered (highest to lowest).

Examples

```
# See vignette: "Crime Series Identification and Clustering" for usage.
```

makeGroups *Generates crime groups from crime series data*

Description

This function generates crime groups that are useful for making unlinked pairs and for agglomerative linkage.

Usage

```
makeGroups(X, method = 1)
```

Arguments

X	crime series data (generated from makeSeriesData) with offender ID (offenderID), crime ID (crimeID), and the event datetime (TIME)
method	Method=1 (default) forms groups by finding the maximal connected offender subgraph. Method=2 forms groups from the unique group of co-offenders. Method=3 forms from groups from offenderIDs

Details

Method=1 forms groups by finding the maximal connected offender subgraph. So if two offenders have ever co-offended, then all of their crimes are assigned to the same group. Method=2 forms groups from the unique group of co-offenders. So for two offenders who co-offended, all the co-offending crimes are in one group and any crimes committed individually or with other offenders are assigned to another group. Method=3 forms groups from the offender(s) responsible. So a crime that is committed by multiple people will be assigned to multiple groups.

Value

vector of crime group labels

Examples

```
data(crimes)
 data(offenders)
 seriesData = makeSeriesData(crimedata=crimes,offenderTable=offenders)
 groups = makeGroups(seriesData,method=1)
 head(groups,10)
```

---

makePairs *Generates indices of linked and unlinked crime pairs (with weights)*

Description

These functions generate a set of crimeIDs for linked and unlinked crime pairs. Linked pairs are assigned a weight according to how many crimes are in the crime series. For unlinked pairs, m crimes are selected from each *crime group* and pairs them with crimes in other *crime groups*.

Usage

```
makePairs(X, thres = 365, m = 40, show.pb = FALSE, seed = NULL)

makeLinked(X, thres = 365)

makeUnlinked(X, m, thres = 365, show.pb = FALSE, seed = NULL)
```

Arguments

X	crime series data (generated from makeSeriesData) with offender ID (offenderID), crime ID (crimeID), and the event datetime (TIME)
thres	the threshold (in days) of allowable time distance
m	the number of samples from each crime group (for unlinked pairs)
show.pb	(logical) should a progress bar be displayed
seed	seed for random number generation

Details

makePairs is a Convenience function that calls makeLinked and makeUnlinked and combines the results. It is unlikely that the latter two functions will need to be called directly.

For *linked* crime pairs, the weights are such that each crime series contributes a total weight of no greater than 1. Specifically, the weights are $W_{ij} = \min\{1/N_m : V_i, V_j \in C_m\}$, where C_m is the crime series for offender m and N_m is the number of crime pairs in their series (assuming V_i and V_j are together in at least one crime series). Due to co-offending, the sum of weights will be smaller than the number of series with at least two crimes.

To form the *unlinked* crime pairs, *crime groups* are identified as the maximal connected offender subgraphs. Then m indices are drawn from each crime group (with replacment) and paired with crimes from other crime groups according to weights that ensure that large groups don't give the most events.

Value

matrix of indices of crime pairs with weights. For makePairs, The last column type indicates if the crime pair is linked or unlinked.

Examples

```
data(crimes)
 data(offenders)
 seriesData = makeSeriesData(crimedata=crimes,offenderTable=offenders)
 allPairs = makePairs(seriesData,thres=365,m=40)
```

---

makeSeriesData *Make crime series data*

Description

Creates a data frame with index to crimedata and offender information. It is used to generate the linkage data.

Usage

```
makeSeriesData(crimedata, offenderTable, time = c("midpoint", "earliest",
  "latest"))
```

Arguments

crimedata	data.frame of crime incident data. crimedata must have columns named: crimeID, DT.FROM, and DT.TO. Note: if crime timing is known exactly (uncensored) than only DT.FROM is required.
offenderTable	offender table that indicates the offender(s) responsible for solved crimes. offenderTable must have columns named: offenderID and crimeID.
time	the event time to be returned: 'midpoint', 'earliest', or 'latest'

Details

The creates a crimeseries data object that is required for creating linkage data. It creates a crime series ID (CS) for every offender. Because of co-offending, a single crime (crimeID) can belong to multiple crime series.

Value

data frame representation of the crime series present in the crimedata. It includes the crime ID (crimeID), index of that crimeID in the original crimedata (Index), the crime series ID (CS) corresponding to each offenderID, and the event time (TIME).

See Also

getCrimeSeries

Examples

```
data(crimes)
 data(offenders)

 seriesData = makeSeriesData(crimedata=crimes,offenderTable=offenders)
 head(seriesData)

 nCrimes = table(seriesData$offenderID)  # length of each crime series
 table(nCrimes)                  # distribution of crime series length
 mean(nCrimes>1)                 # proportion of offenders with multiple crimes

 nCO = table(seriesData$crimeID) # number of co-offenders per crime
 table(nCO)                      # distribution of number of co-offenders
 mean(nCO>1)                     # proportion of crimes with multiple co-offenders
```

---

naiveBayes *Naive bayes classifier using histograms and shrinkage*

Description

After binning, this adds pseudo counts to each bin count to give df approximate degrees of freedom. If partition=quantile, this does not assume a continuous uniform prior over support, but rather a discrete uniform over all (unlabeled) observations points.

Usage

```
naiveBayes(formula, data, weights, df = 20, nbins = 30,
  partition = c("quantile", "width"))

naiveBayes.fit(X, y, weights, df = 20, nbins = 30,
  partition = c("quantile", "width"))
```

Arguments

formula	an object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. Only main effects (not interactions) are allowed.
data	data.frame of predictors, can include continuous and categorical/factors along with a response vector (1 = linked, 0 = unlinked), and (optionally) observation weights (e.g., weight column). The column names of data need to include the terms specified in formula.
weights	a vector of observation weights or the column name in data that corresponds to the weights.
df	the degrees of freedom for each component density. if vector, each predictor can use a different df
nbins	the number of bins for continuous predictors
partition	for binning; indicates if breaks generated from quantiles or equal spacing
X	data frame of categorical and/or numeric variables
y	binary vector indicating linkage (1 = linked, 0 = unlinked) or logical vector (TRUE = linked, FALSE = unlinked)

Details

Fits a naive bayes model to continous and categorical/factor predictors. Continous predictors are first binned, then estimates shrunk towards zero.

Value

BF a bayes factor object; list of component bayes factors

See Also

predict.naiveBayes, plot.naiveBayes

Examples

```
# See vignette: "Statistical Methods for Crime Series Linkage" for usage.
```

offenders *Ficticious offender data*

Description

Offender table relating crimes (crimeID) to offenders (offenderID)

Usage

```
data(offenders)
```

Format

1357 offenders committed 1377 crimes

offenderID ID number of offender

crimeID ID number of crime

Source

Ficticious data, but hopefully realistic

Examples

```
head(offenders)
```

plot.naiveBayes *Plots for Naive Bayes Model*

Description

This function attempts to plot all of the component plots in one window by using the mfrow argument of par. If more control is desired then use plotBF to plot individual Bayes factors.

Usage

```
## S3 method for class 'naiveBayes'
plot(x, vars, log.scale = TRUE, show.legend = 1,
  cols = c(color("darkred", alpha = 0.75), color("darkblue", alpha = 0.75)),
  ...)
```

Arguments

x	a naiveBayes object
vars	name or index of naive Bayes components to plot. Will plot all if blank.
log.scale	(logical)
show.legend	either a value or values indicating which plot to show the legend, or TRUE/FALSE to show or not show the legend on all plots.
cols	Colors for plotting. First element is for linkage, second unlinked
...	arguemnts passed into plotBF

Details

Plots (component) bayes factors from naiveBayes()

Value

plots of Bayes factor from a naive Bayes model

See Also

plotBF, naiveBayes, predict.naiveBayes

Examples

```
# See vignette: "Statistical Methods for Crime Series Linkage" for usage.
```

---

plotBF *plots 1D bayes factor*

Description

plots 1D bayes factor

Usage

```
plotBF(BF, log.scale = TRUE, show.legend = TRUE, xlim, ylim = NULL,
  cols = c(color("darkred", alpha = 0.75), color("darkblue", alpha = 0.75)),
  ...)
```

Arguments

BF	Bayes Factor
log.scale	(logical)
show.legend	(logical)
xlim	range of x-axis
ylim	range of y-axis
cols	Colors for plotting. First element is for linkage, second unlinked
...	arguemnts passed into plotBKG

Value

plot of Bayes factor

See Also

plot.naiveBayes, plotBKG

Examples

```
# See vignette: "Statistical Methods for Crime Series Linkage" for usage.
```

---

plot_hcc *Plot a hierarchical crime clustering object*

Description

Similar to plot.dendrogram.

Usage

```
plot_hcc(tree, yticks = seq(-2, 8, by = 2), hang = -1, ...)
```

Arguments

tree	an object produced from crimeClust_hier
yticks	the location of the tick marks for log Bayes factors
hang	the hang argument of as.dendrogram
...	other arguments passed to plot.dendrogram

Details

This function creates a dendrogram object and then plots it. It corrects the y-axis to give the proper values and adds the number of clusters if the tree were cut at a particular log Bayes factor.

Value

A dendrogram

See Also

crimeClust_hier

Examples

```
# See vignette: "Crime Series Identification and Clustering" for usage.
```

`predict.naiveBayes` *Generate prediction (sum of log bayes factors) from a* `naiveBayes` *object*

Description

This does not include the log prior odds, so will be off by a constant.

Usage

```
## S3 method for class 'naiveBayes'
predict(object, newdata, components = FALSE,
  vars = NULL, ...)
```

Arguments

`object`	a naive bayes object from `naiveBayes`
`newdata`	data frame of new predictors, column names must match NB names
`components`	(logical) return the log bayes factors from each component or return the sum of log bayes factors
`vars`	the names or column numbers of specific predictors. If NULL, then all predictors will be used
`...`	not currently used

Value

BF if `components = FALSE`, the sum of log bayes factors, if `components = TRUE` the component bayes factors (useful for plotting).

It will give a warning, but still produce output if X is missing predictors. The output in this situation will be based on the predictors that are in X.

See Also

`naiveBayes`, `plot.naiveBayes`

Examples

```
# See vignette: "Statistical Methods for Crime Series Linkage" for usage.
```

predictBF *Generate prediction of a component bayes factor*

Description

This does not include the log prior odds, so will be off by a constant

Usage

```
predictBF(BF, x, log = TRUE)
```

Arguments

BF	bayes factor data.frame from getBF
x	vector of new predictor values
log	(logical) if TRUE, return the **log** bayes factor estimate

Value

estimated (log) bayes factor from a single predictor

Examples

```
# See vignette: "Statistical Methods for Crime Series Linkage" for usage.
```

seriesID *Crime series identification*

Description

Performs crime series identification by finding the crime series that are most closely related (as measured by Bayes Factor) to an unsolved crime.

Usage

```
seriesID(crime, solved, seriesData, varlist, estimateBF,
  linkage.method = c("average", "single", "complete"), group.method = 3,
  ...)
```

Arguments

crime	crime incident; vector of crime variables
solved	incident data for the solved crimes. Must have a column named crimeID.
seriesData	table of crimeIDs and crimeseries (results from makeSeriesData)
varlist	a list of the variable names (columns of solved and crime) used to create evidence variables with compareCrimes.
estimateBF	function to estimate the bayes factor from evidence variables
linkage.method	the type of linkage for comparing one crime to a set of crimes • "average" uses the average bayes factor • "single" uses the largest bayes factor (most similar) • "complete" uses the smallest bayes factor (least similar)
group.method	the type of crime groups to form (see makeGroups for details)
...	other arguments passed to compareCrimes

Value

A list with two objects. score is a data.frame of the similarity scores for each element in solved. groups is the data.frame seriesData with an additional column indicating the crime group (using the method specified in group.method).

References

Porter, M. D. (2014). A Statistical Approach to Crime Linkage. *arXiv preprint arXiv:1410.2285.*. http://arxiv.org/abs/1410.2285

Examples

```
# See vignette: "Crime Series Identification and Clustering" for usage.
```

Index