Package 'clustMixType'

October 17, 2017

Version 0.1-29

Date 2017-10-15

Title k-Prototypes Clustering for Mixed Variable-Type Data

Author Gero Szepannek

Maintainer Gero Szepannek <gero.szepannek@web.de>

Imports RColorBrewer

Description Functions to perform k-prototypes partitioning clustering for mixed variable-type data according to Z.Huang (1998): Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Variables, Data Mining and Knowledge Discovery 2, 283-304, <DOI:10.1023/A:1009769707641>.

License GPL (>= 2)

RoxygenNote 5.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-10-17 11:19:07 UTC

R topics documented:

clprofiles *profile k prototypes clustering*

Description

Visualization of k prototypes clustering result for cluster interpretation.

Usage

```
clprofiles(object, x, vars = NULL, col = NULL)
```

Arguments

object	Object resulting from a call of resulting kproto. Also other kmeans like objects with object$cluster and object$size are possible.
x	Original data.
vars	Vector of either coloumn indices or variable names.
col	Palette of cluster colours to be used for the plots. As a default RColorBrewer's brewer.pal(max(unique(object$cluster)), "Set3") is used for k > 2 clusters and lightblue and orange else.

Details

For numerical variables boxplots and for factor variables barplots of each cluster are generated.

Author(s)

<gero.szepannek@web.de>

Examples

```
# generate toy data with factors and numerics

n   <- 100
prb <- 0.9
muk <- 1.5
clusid <- rep(1:4, each = n)

x1 <- sample(c("A","B"), 2*n, replace = TRUE, prob = c(prb, 1-prb))
x1 <- c(x1, sample(c("A","B"), 2*n, replace = TRUE, prob = c(1-prb, prb)))
x1 <- as.factor(x1)

x2 <- sample(c("A","B"), 2*n, replace = TRUE, prob = c(prb, 1-prb))
x2 <- c(x2, sample(c("A","B"), 2*n, replace = TRUE, prob = c(1-prb, prb)))
x2 <- as.factor(x2)

x3 <- c(rnorm(n, mean = -muk), rnorm(n, mean = muk), rnorm(n, mean = -muk), rnorm(n, mean = muk))
x4 <- c(rnorm(n, mean = -muk), rnorm(n, mean = muk), rnorm(n, mean = -muk), rnorm(n, mean = muk))
```

```
x <- data.frame(x1,x2,x3,x4)

# apply k prototyps
kpres <- kproto(x, 4)
clprofiles(kpres, x)

# in real world  clusters are often not as clear cut
# by variation of lambda the emphasize is shifted towards factor / numeric variables
kpres <- kproto(x, 2)
clprofiles(kpres, x)

kpres <- kproto(x, 2, lambda = 0.1)
clprofiles(kpres, x)

kpres <- kproto(x, 2, lambda = 25)
clprofiles(kpres, x)
```

`kproto` *k prototypes clustering*

Description

Computes k prototypes clustering for mixed type data.

Usage

```
kproto(x, ...)

## Default S3 method:
kproto(x, k, lambda = NULL, iter.max = 100, nstart = 1,
  keep.data = TRUE, ...)
```

Arguments

`x`	Data frame with both mumerics and factors.
`k`	Either the number of clusters, a vector specifying indices of initial prototypes, or a data frame of prototypes of the same coloumns as `x`.
`lambda`	Parameter > 0 to trade off between Euclidean distance of numeric variables and simple matching coefficient between categorical variables. Also a vector of variable specific factors is possible where the order must correspond to the order of the variables in the data. In this case all variables' distances will be multiplied by their corresponding lambda value.
`iter.max`	Maximum number of iterations if no convergence before.
`nstart`	If > 1 repetetive computations with random initializations are computed and the result with minimum tot.dist is returned.
`keep.data`	Logical whether original should be included in the returned object.
`...`	Currently not used.

Details

The algorithm like k means iteratively recomputes cluster prototypes and reassigns clusters. Clusters are assigned using $d(x,y) = d_{euclid}(x,y) + \lambda d_{simple\,matching}(x,y)$. Cluster prototypes are computed as cluster means for numeric variables and modes for factors (cf. Huang, 1998).

Value

kmeans like object of class kproto:

cluster	Vector of cluster memberships.
centers	Data frame of cluster prototypes.
lambda	Distance parameter lambda.
size	Vector of cluster sizes.
withinss	Vector of summed distances to the cluster prototype per cluster.
tot.withinss	Target function: sum of all distances to cluster prototype.
dists	Matrix with distances of observations to all cluster prototypes.
iter	Prespecified maximum number of iterations.
trace	List with two elements (vectors) tracing the iteration process: tot.dists and moved number of observations over all iterations.

Author(s)

<gero.szepannek@web.de>

References

Z.Huang (1998): Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Variables, Data Mining and Knowledge Discovery 2, 283-304.

Examples

```
# generate toy data with factors and numerics

n   <- 100
prb <- 0.9
muk <- 1.5
clusid <- rep(1:4, each = n)

x1 <- sample(c("A","B"), 2*n, replace = TRUE, prob = c(prb, 1-prb))
x1 <- c(x1, sample(c("A","B"), 2*n, replace = TRUE, prob = c(1-prb, prb)))
x1 <- as.factor(x1)

x2 <- sample(c("A","B"), 2*n, replace = TRUE, prob = c(prb, 1-prb))
x2 <- c(x2, sample(c("A","B"), 2*n, replace = TRUE, prob = c(1-prb, prb)))
x2 <- as.factor(x2)

x3 <- c(rnorm(n, mean = -muk), rnorm(n, mean = muk), rnorm(n, mean = -muk), rnorm(n, mean = muk))
x4 <- c(rnorm(n, mean = -muk), rnorm(n, mean = muk), rnorm(n, mean = -muk), rnorm(n, mean = muk))
```

```
x <- data.frame(x1,x2,x3,x4)

# apply k prototypes
kpres <- kproto(x, 4)
clprofiles(kpres, x)

# in real world  clusters are often not as clear cut
# by variation of lambda the emphasize is shifted towards factor / numeric variables
kpres <- kproto(x, 2)
clprofiles(kpres, x)

kpres <- kproto(x, 2, lambda = 0.1)
clprofiles(kpres, x)

kpres <- kproto(x, 2, lambda = 25)
clprofiles(kpres, x)
```

`lambdaest` *compares variance of all variables*

Description

Investigation of variances to specify lambda for k prototypes clustering.

Usage

```
lambdaest(x, num.method = 1, fac.method = 1, outtype = "numeric")
```

Arguments

`x`	Original data.
`num.method`	Integer 1 or 2. Specifies the heuristic used for numeric variables.
`fac.method`	Integer 1 or 2. Specifies the heuristic used for factor variables.
`outtype`	Specidfies the desired output: either 'numeric', 'vector' or 'variation'.

Details

Variance (`num.method = 1`) or standard deviation (`num.method = 2`) of numeric variables and $1 - \sum_i p_i^2$ (`fac.method = 1/3`) or $1 - \max_i p_i$ (`fac.method = 2/4`) for categorical variables is computed.

Value

`lambda`	Ratio of averages over all numeric/factor variables is returned. In case of `outtype = "vector"` the separate lambda for all variables is returned as the inverse of the single variables' variation as specified by the `method` argument. `outtype = "variation"` returns these values and is not ment to be passed directly to `kproto()`.

Author(s)

<gero.szepannek@web.de>

Examples

```
# generate toy data with factors and numerics

n   <- 100
prb <- 0.9
muk <- 1.5
clusid <- rep(1:4, each = n)

x1 <- sample(c("A","B"), 2*n, replace = TRUE, prob = c(prb, 1-prb))
x1 <- c(x1, sample(c("A","B"), 2*n, replace = TRUE, prob = c(1-prb, prb)))
x1 <- as.factor(x1)

x2 <- sample(c("A","B"), 2*n, replace = TRUE, prob = c(prb, 1-prb))
x2 <- c(x2, sample(c("A","B"), 2*n, replace = TRUE, prob = c(1-prb, prb)))
x2 <- as.factor(x2)

x3 <- c(rnorm(n, mean = -muk), rnorm(n, mean = muk), rnorm(n, mean = -muk), rnorm(n, mean = muk))
x4 <- c(rnorm(n, mean = -muk), rnorm(n, mean = muk), rnorm(n, mean = -muk), rnorm(n, mean = muk))

x <- data.frame(x1,x2,x3,x4)

lambdaest(x)
res <- kproto(x, 4, lambda = lambdaest(x))
```

---

predict.kproto *k prototypes clustering*

Description

Predicts k prototypes cluster memberships and distances for new data.

Usage

```
## S3 method for class 'kproto'
predict(object, newdata, ...)
```

Arguments

object	Object resulting from a call of resulting kproto.
newdata	New data frame (of same structure) where cluster memberships are to be predicted.
...	Currently not used.

Details

The algorithm like k means iteratively recomputes cluster prototypes and reassigns clusters. Clusters are assigned using $d(x,y) = d_{euclid}(x,y) + \lambda d_{simple\,matching}(x,y)$. Cluster prototypes are computed as cluster means for numeric variables and modes for factors (cf. Huang, 1998).

Value

kmeans like object of class kproto:

cluster	Vector of cluster memberships.
dists	Matrix with distances of observations to all cluster prototypes.

Author(s)

<gero.szepannek@web.de>

Examples

```
# generate toy data with factors and numerics

n   <- 100
prb <- 0.9
muk <- 1.5
clusid <- rep(1:4, each = n)

x1 <- sample(c("A","B"), 2*n, replace = TRUE, prob = c(prb, 1-prb))
x1 <- c(x1, sample(c("A","B"), 2*n, replace = TRUE, prob = c(1-prb, prb)))
x1 <- as.factor(x1)

x2 <- sample(c("A","B"), 2*n, replace = TRUE, prob = c(prb, 1-prb))
x2 <- c(x2, sample(c("A","B"), 2*n, replace = TRUE, prob = c(1-prb, prb)))
x2 <- as.factor(x2)

x3 <- c(rnorm(n, mean = -muk), rnorm(n, mean = muk), rnorm(n, mean = -muk), rnorm(n, mean = muk))
x4 <- c(rnorm(n, mean = -muk), rnorm(n, mean = muk), rnorm(n, mean = -muk), rnorm(n, mean = muk))

x <- data.frame(x1,x2,x3,x4)

# apply k prototyps
kpres <- kproto(x, 4)
predicted.clusters <- predict(kpres, x)
```

summary.kproto *Summary method for kproto cluster result*

Description

Investigation of variances to specify lambda for k prototypes clustering.

Usage

```
## S3 method for class 'kproto'
summary(object, data = NULL, pct.dig = 3, ...)
```

Arguments

object	Object of class kproto.
data	Optional data set to be analyzed. If !(is.null(data)) clusters for data are assigned by predict(object, data). If not specified the clusters of the original data ara analyzed. Only possible if kproto has been called using keep.data = TRUE.
pct.dig	Number of digits for rounding percentages of factor variables.
...	Further arguments to be passed to internal call of summary() for numeric variables.

Details

For numeric variables statistics are computed for each clusters using summary(). For categorical variables distribution percent are computed.

Value

List where each element corresponds to one variable. Each row of any element corresponds to one cluster.

Author(s)

<gero.szepannek@web.de>

Examples

```
# generate toy data with factors and numerics

n   <- 100
prb <- 0.9
muk <- 1.5
clusid <- rep(1:4, each = n)

x1 <- sample(c("A","B"), 2*n, replace = TRUE, prob = c(prb, 1-prb))
x1 <- c(x1, sample(c("A","B"), 2*n, replace = TRUE, prob = c(1-prb, prb)))
x1 <- as.factor(x1)
```

```
x2 <- sample(c("A","B"), 2*n, replace = TRUE, prob = c(prb, 1-prb))
x2 <- c(x2, sample(c("A","B"), 2*n, replace = TRUE, prob = c(1-prb, prb)))
x2 <- as.factor(x2)

x3 <- c(rnorm(n, mean = -muk), rnorm(n, mean = muk), rnorm(n, mean = -muk), rnorm(n, mean = muk))
x4 <- c(rnorm(n, mean = -muk), rnorm(n, mean = muk), rnorm(n, mean = -muk), rnorm(n, mean = muk))

x <- data.frame(x1,x2,x3,x4)

res <- kproto(x, 4)
summary(res)
```

Index