Package 'kpeaks'

September 19, 2017

Type Package

Title Determination of K Using Peak Counts of Features for Clustering

Version 0.1.0

Date 2017-09-17

Author Zeynel Cebeci [aut, cre], Cagatay Cebeci [aut]

Maintainer Zeynel Cebeci <zcebeci@cukurova.edu.tr>

Description The input argument k which is the number of clusters is needed to start all of the partitioning clustering algorithms. In unsupervised learning applications, an optimal value of this argument is widely determined by using the internal validity indexes. Since these indexes suggest a k value which is computed on the clustering results after several runs of a clustering algorithm they are computationally expensive. On the contrary, 'kpeaks' enables to estimate k before running any clustering algorithm. It is based on a simple novel technique using the descriptive statistics of peak counts of the features in a data set.

Depends R (>= 2.10.0)

License GPL (>= 2)

LazyData true

Imports graphics, stats, utils

NeedsCompilation no

Repository CRAN

Date/Publication 2017-09-19 09:37:37 UTC

R topics documented:

kpeaks-package *Determination of K Using Peak Counts of Features for Clustering*

Description

The input argument k that represents the number of clusters is needed to start all of the partitioning clustering algorithms. In unsupervised learning applications, an optimal value of this argument is widely determined by using the internal validity indexes. Since these indexes suggest a k value which is computed on the clustering results after several runs of a clustering algorithm, they are computationally expensive. On the contrary, '**kpeaks**' enables to estimate k before running any clustering algorithm. It is based on a simple novel technique using the descriptive statistics of peak counts of the features in a dataset.

Details

The package '**kpeaks**' contains five functions and one synthetically created dataset for testing purposes. In order to suggest an estimate of k, the function `findk` internally calls the functions `genpolygon` and `findpolypeaks`, respectively. The frequency polygons can be visually inspected by using the function `plotpolygon`. Using `rmshoulders` is recommended to flatten or remove the the shoulder peaks around the main peaks of a frequency polygon, if any.

Author(s)

Zeynel Cebeci, Cagatay Cebeci

See Also

`findk`, `findpolypeaks`, `genpolygon`, `plotpolygon`, `rmshoulders`

---

findk *Estimate the Number of Clusters in a Data Set*

Description

Based on some of descriptive statistics of the peak counts in the frequency polygon of a feature, this function proposes a list of estimates of the number of clusters in a data set.

Usage

```
findk(x, binrule, nbins, tcmethod, tc, trmethod, tv, rms=FALSE, rcs=FALSE, tpc=1)
```

Arguments

x	a numeric data frame or matrix.
binrule	a string specifying the binning rule to compute the number of classes of a frequency polygon.
nbins	an integer specifying the number of classes (bins). It is internally computed according to the selected binning rule except 'usr'. See all available options in genpolygon.
tcmethod	a string representing a threshold method to compute a threshold distance value to discard the small or empty bins of a frequency polygon. See all available options in findpolypeaks.
tc	an integer for threshold frequency value assigned by tcmethod.
trmethod	a string used to specify a removal method to discard the shoulders around the main peaks in a frequency polygon. See all available options in rmshoulders.
tv	a numeric threshold distance value assigned by trmethod.
rms	a logical value whether the shoulders removal is applied or not. Default value is 'FALSE'.
rcs	a logical value whether the estimates of k computed on the reduced counts set instead of the full set. Default value is 'FALSE', and set to TRUE in order to use the reduced counts set.
tpc	an integer threshold value for creating the reduced set of the peak counts. Default value is 1.

Details

The function findk returns a list of k values which are proposed as the estimates of number of clusters in a given data set. The estimation is based on various descriptive statistics of the peak counts in the frequency polygon of the features. Firstly, the classes of frequency polygons of the features are generated by using the function genpolygon. Then, the main peaks in frequency polygons are determined by using the function findpolypeaks. If desired, with the function rmshoulders the shoulder peaks are removed from the peaks matrix returned by the function findpolypeaks. In the returned peaks matrix, the peaks are counted for each feature, and a list of estimates of k is produced by using various descriptive statistics of the peak counts.

Value

a list of the estimates of k consists of the following items which are computed from the peak counts of the features in a given data set:

am	arithmetic mean of peak counts.
med	median of peak counts.
mod	mode of peak counts.
cr	center of the range of peak counts.
ciqr	center of the interquartile range (IQR) of peak counts.
mppc	overall mean of the pairwise means of peak counts.

mq3m	mean of the third quartile (Q3) and maximum of peak counts.
mtl	mean of two largest value of peak counts.
avgk	proposed k as the mean of all the estimates.
modk	proposed k as the mode of all the estimates.
mtlk	proposed k as the mean of two largest estimates.
dst	a string representing the type of counts set which is used in computations.
pcounts	an integer vector containing the peak counts of the features.

Note

As the input arguments, findk normally uses the outputs from the functions findpolypeaks and rmshoulders.

Author(s)

Zeynel Cebeci, Cagatay Cebeci

See Also

findpolypeaks, rmshoulders

Examples

```
# Estimate the number of clusters in x5p4c data set
data(x5p4c)
estk <- findk(x5p4c, binrule="sturges")
print(estk)
summary(estk$pcounts)
cat("Estimated the number of clusters as the mean of Q3 and max peak count:", estk$mq3m, fill=TRUE)
cat("Proposed number of clusters based on the mean of two largest estimates:", estk$mtlk, fill=TRUE)

# Estimate the number of clusters in x5p4c data set by using threshold frequency method 'avg'
# and shoulders removal method 'q1'
estk <- findk(x5p4c, binrule="usr", nbins=15, tcmethod="usr", tc=1, trmethod="avg", rms=TRUE)
print(estk)
summary(estk$pcounts)
cat("Proposed number of clusters based on the mean of two largest estimates:", estk$mtlk, fill=TRUE)

# Estimate the number of clusters in iris data set
data(iris)
estk <- findk(iris[,1:4], binrule="bc", rcs=FALSE)
print(estk)
summary(estk$pcounts)
cat("Proposed number of clusters based on the mean of estimates:", estk$avgk, fill=TRUE)
cat("Proposed number of clusters based on the mode of estimates:", estk$modk, fill=TRUE)
cat("Proposed number of clusters based on the mean of two largest estimates:", estk$mtlk, fill=TRUE)
```

`findpolypeaks` *Find the Peaks of a Frequency Polygon*

Description

Frequency polygons are graphics to reveal the shapes of data distributions as histograms do. The peaks of frequency polygons are required in several data mining applications. `findpolypeaks` finds the peaks in a frequency polygon by using the frequencies and middles values of the classes of it.

Usage

```
findpolypeaks(xm, xc, tcmethod, tc)
```

Arguments

`xm`	a numeric vector contains the middle values of the classes of the frequency polygon (or the bins of a histogram).
`xc`	an integer vector contains the frequencies of the classes of the frequency polygon.
`tcmethod`	a string represents the threshold method to discard the empty and the small bins whose frequencies are smaller than a threshold frequency value. Default method is '`usr`'. Alternatively, the methods given below can be used to compute a threshold frequency value using the descriptive statistics of the frequencies in `xc`. • '`sd1`' and '`sd2`' use the standard deviation. • '`q1`' uses the first quartile (Q1). • '`iqr`' uses the interquartile range (IQR). • '`avg`' uses the arithmetic mean. • '`min`' and '`min2`' use the minimum. • '`log2`' uses the two-base logarithm of n, vector size. • '`usr`' uses a user-specified value.
`tc`	an integer which is used as the threshold frequency value for discarding the empty and small height classes in the frequency polygon. Default value is 1 if the threshold option '`usr`' is chosen. Depending on the selected methods, the value of `tc` equals to: • one standart deviation with the method '`sd1`', • one quarter of the standart deviation with the method '`sd2`', • the first quartile with the method '`q1`', • one quarter of the interquartile range with method '`iqr`', • 10% of the arithmetic mean with the method '`avg`', • the minimum value with the method '`min`', • two times of minimum with method '`min2`', • two-base logarithm of the number of classes divided by ten with the method '`log2`', • an arbitrary number specified with the method '`usr`'.

Details

The peaks are determined after removing the empty and small height classes whose frequencies are below the chosen threshold frequency. Default threshold value is 1 that means that all the classes which have frequencies of 0 and 1 are removed in the input vectors `xm` and `xc`.

Value

`pm`	a data frame with two columns which are named *pvalues* and *pfreqs* containing the middle values and frequencies of the peaks which determined in the frequency polygon, respectively.
`np`	an integer representing the number of peaks in the frequency polygon.

Author(s)

Zeynel Cebeci, Cagatay Cebeci

See Also

`findk`, `genpolygon`, `rmshoulders`

Examples

```
data(x5p4c)
# Using a user-specified number of bins, build the frequency polygon of p2 in the data set x5p4c
hvals <- genpolygon(x5p4c$p2, binrule="usr", nbins=20)
plotpolygon(x5p4c$p2, nbins=hvals$nbins, ptype="ph")

# Find the peaks in the frequency polygon by using the threshold method min
resfpp1 <- findpolypeaks(hvals$mids, hvals$freqs, tcmethod="min")
print(resfpp1)

# Find the peaks in the frequency polygon by using the threshold equals to 5
resfpp2 <- findpolypeaks(hvals$mids, hvals$freqs, tcmethod="usr", tc=5)
print(resfpp2)

data(iris)
# By using Doane rule, build the frequency polygon of the 4th feature in the data set iris
hvals <- genpolygon(iris[,4], binrule="doane")
plotpolygon(iris[,4], nbins=hvals$nbins, ptype="p")

#Find the peaks in the frequency polygon by using the threshold method avg
resfpp3 <- findpolypeaks(hvals$mids, hvals$freqs, tcmethod="avg")
print(resfpp3)
```

genpolygon *Generate the Classes to Build a Frequency Polygon*

Description

Constructs the histogram of a feature by using a selected binning rule, returns the middle values and frequencies of classes for further works on the frequency polygon.

Usage

```
genpolygon(x, binrule, nbins, disp = FALSE)
```

Arguments

x	a numeric vector containing the observations for a feature.
binrule	name of the rule in order to compute the number of bins to build the histogram.
nbins	an integer representing the number of bins which is computed by using the selected binning rule. Default rule is 'sturges'. Depending on the selected rule nbins equals to (In the formulae, n is the number of observations.):

- `floor(sqrt(n))` if the rule is 'sqr',
- `ceiling(1+log(n, 2))` if the rule is 'sturges',
- `ceiling(1+3.332*log(n, 10))` if the rule is 'huntsberger',
- `ceiling(5*log(n, 10))` if the rule is 'bc',
- `ceiling(n^(1/3))` if the rule is 'cencov',
- `ceiling(2*n^(1/3))` if the rule is 'rice',
- `ceiling((2*n)^(1/3))` if the rule is 'ts',
- `ceiling(((max(x)-min(x))/(3.5*sqrt(var(x))*n^(-1/3))))` if the rule is 'scott',
- `ceiling(((max(x)-min(x))/(2*IQR(x)*n^(-1/3)))` if the rule is 'fd',
- `ceiling(1+log(n,2)+log(1+abs(skewness(x))/(6*(n-2)/((n+1)*(n+3))^0.5),2))` if the rule is 'doane',
- `ceiling(log(n)/2*pi)` if the rule is 'cebeci',
- a user-specified integer if the rule is 'usr'.

disp	a logical value should be set to TRUE to display the histogram.

Details

According to Hyndman (1995), Sturges's rule was the first rule to calculate k, the number of classes to build a histogram. Most of the statistical packages use this simple rule for determining the number of classes in constructing histograms. Brooks & Carruthers (1953) proposed a rule using log_{10} instead of log_2 giving always larger k when compared to Sturges's rule. The rule by Huntsberger (1962) yields nearly equal result to those of Sturges's rule. These two rules work well if n is less than 200. Scott (1992) argued that Sturges's rule leads to generate oversmoothed histograms in case of large number of n. In his rule, Cencov (1962) used the cube root of n simply. This rule was

followed by its extensions, i.e., Rice rule and Terrell & Scott (1985) rule. When compared to the others, the square root rule produces larger k (Davies & Goldsmith, 1980).

Most of the rules simply include only n as the input argument. On the other hand, the rules using variation and shape of data distributions can provide more optimal k values. For instance, Doane (1976) extended the Sturges's rule by adding the standardized skewness in order to overcome the problem with non-normal distributions need more classes. In order to estimate optimal k values, Scott (1979) added the standard deviation to his formula. Freedman and Diaconis (1981) proposed to use the interquartile range (IQR) statistic which is less sensitive to outliers than the standard deviation. In a study on unsupervised discretization methods, Cebeci & Yildiz (2017) tested a binning rule formula based on the ten-base logarithm of n divided by `2*pi`. They also argued that the rules Freedman-Diaconis and Doane were slightly performed better than the other rules based on the training model accuracies on a chicken egg quality traits dataset. Therefore, using the above mentioned rules may be more effective in determining the peaks of a frequency polygon.

Value

`xm`	a numeric vector containing the middle values of bins.
`xc`	an integer vector containing the frequencies of the bins.
`nbins`	an integer containing the number of bins to build the histogram.

Author(s)

Zeynel Cebeci, Cagatay Cebeci

References

Brooks C E P & Carruthers N (1953). Handbook of statistical methods in meteorology. H M Stationary Office, London.

Cebeci Z & Yildiz F (2017). Unsupervised discretization of continuous variables in a chicken egg quality traits dataset. *Turk. J Agriculture-Food Sci. & Tech.* 5(4): 315-320.

Cencov N N (1962). Evaluation of an unknown distribution density from observations. *Soviet Mathematics* 3: 1559-1562.

Davies O L & Goldsmith P L (1980). Statistical methods in research and production. 4th edn, Longman: London.

Doane D P (1976). Aesthetic frequency classification. *American Statistician* 30(4):181-183.

Freedman D & Diaconis P (1981). On the histogram as a density estimator: L2 Theory. *Zeit. Wahr. ver. Geb.* 57(4):453-476.

Hyndman R J (1995). The problem with Sturges rule for constructing histograms. http://robjhyndman.com/papers/sturges.pdf.

Huntsberger D V (1962). Elements of statistical inference. London: Prentice-Hall.

Scott D W (1992). Multivariate density estimation: Theory, Practice and Visualization. John Wiley & Sons: New York.

Sturges H (1926). The choice of a class-interval. *J Amer. Statist. Assoc.* 21(153):65-66.

Terrell G R & Scott D W (1985). Oversmoothed nonparametric density estimates. *J Amer. Statist. Assoc.* 80(389):209-214.

See Also

findk, findpolypeaks, plotpolygon

Examples

```
x <- rnorm(n=100, mean=5, sd=0.5)
# Construct the histogram of x according to the Sturges rule with no display
hvals <- genpolygon(x, binrule = "sturges")
print(hvals)

# Plot the histogram of x by using the user-specified number of classes
hvals <- genpolygon(x, binrule = "usr", nbins = 20, disp = TRUE)
print(hvals)

# Plot the histogram of the second feature in iris dataset
# by using the Freedman-Diaconis (fd) rule
data(iris)
hvals <- genpolygon(iris[,2], binrule = "fd", disp = TRUE)
print(hvals)
```

---

plotpolygon *Plot Frequency Polygons*

Description

Plots the frequency polygon and histogram of a feature with some options.

Usage

```
plotpolygon(x, nbins, ptype, bcol = "gray", pcol = "blue")
```

Arguments

x	a numeric vector containing the observations of a feature, or a numeric matrix when ptype is 'sp'.
nbins	an integer for the number of classes in the frequency polygon.
bcol	a string for the color of bins. Default is 'gray'.
pcol	a string for the color of polygon lines. Default is 'blue'.
ptype	a string specifying the type of plot. Use 'p' for plotting the polygon only or 'ph' for plotting the polygon with the histogram. Default is 'sp' for the scatterplots between the pairs of features and the polygons on the diagonal panel.

Author(s)

Zeynel Cebeci, Cagatay Cebeci

See Also

genpolygon

Examples

```
# plot the frequency polygon of the 2nd feature in x5p4c data set
data(x5p4c)
hvals <- genpolygon(x5p4c[,2], binrule="usr", nbins=20)

# plot the frequency polygon of the 2nd feature in x5p4c data set
plotpolygon(x5p4c[,2], nbins=hvals$nbins, ptype="p")

# plot the histogram and frequency polygon of the 2nd feature in x5p4c data set
plotpolygon(x5p4c[,2], nbins=hvals$nbins, ptype="ph", bcol="orange", pcol="blue")

# plot the pairwise scatter plots of the features in x5p4c data set
pairs(x5p4c, diag.panel=plotpolygon, upper.panel=NULL, cex.labels=1.5)

# plot the histogram and frequency polygon of Petal.Width in iris data set
data(iris)
hvals <- genpolygon(iris$Petal.Width, binrule="doane")
plotpolygon(iris$Petal.Width, nbins=hvals$nbins, ptype="ph")
```

---

rmshoulders *Shoulders Removal in Frequency Polygons*

Description

Removes the shoulders around the main peaks in a frequency polygon.

Usage

```
rmshoulders(xm, xc, trmethod, tv)
```

Arguments

xm	a numeric vector containing the middle values of peaks of a frequency polygon.
xc	an integer vector containing the frequencies of peaks of a frequency polygon.
trmethod	a string representing the type of shoulders removal option for computing a threshold value. Default method is 'usr'. The alternatives are 'sd', 'q1', 'iqr', 'avg' and 'med'. These methods compute the threshold distance value using some statistics of the distances between the middle values of two successive peaks in the vector xm.

- 'sd' uses the standard deviation.
- 'q1' uses the first quartile (Q1).
- 'q3' uses the third quartile (Q3).
- 'iqr' uses the interquartile range (IQR).

- 'avg' uses the arithmetic mean.
- 'med' uses the median.
- 'usr' uses a user-specified number.

`tv` a numeric value to be used as the threshold distance for deciding the shoulders. Default threshold is 1 if the removal method 'usr' is chosen. Depending on the selected removal method `tv` equals to:

- one standart deviation if `trmethod` is 'sd',
- the first quartile if `trmethod` is 'q1',
- the third quartile if `trmethod` is 'q3',
- one quarter of the interquartile range if `trmethod` is 'iqr',
- the arithmetic mean if `trmethod` is 'avg',
- the median if `trmethod` is 'med',
- a user-specified number if `trmethod` is 'usr'.

Details

Literally speaking, a *shoulder peak* or shortly *shoulder* is a secondary peak in a close location before or after the main peak of a mountain. In a frequency polygon, a shoulder is a smaller peak that is quite close to a higher peak resulting a non-obvious valley between them. Shoulders may occur randomly due to some reasons such as random noises or selecting higher number of classes in histogram building etc. Usually, it is desired to remove them from the peaks vector of a frequency polygon. In '**kpeaks**', a peak considered as a shoulder when its height is smaller than the height of its neighbor peak and its distance to its neighbor is also lower than a threshold distance value. In order to compute a threshold distance value, here, we propose to use seven options as listed in the section 'arguments'. The options `q1` and `iqr` can be applied to remove the minor shoulders that are very near to the main peaks while `q3` is recommended to eliminate the substantial shoulders in the processed frequency polygon. The remaining options may be more efficient for removing the moderate shoulders.

Value

`pm` a data frame with two columns whose names are *pvalues* and *pfreqs* for the middle values and the frequencies of the peaks after removal process, respectively.

`np` an integer representing the number of peaks after removal of the shoulders.

Note

The function `rmshoulders` normally should be called with the input values that are returned by the function `findpolypeaks`.

Author(s)

Zeynel Cebeci, Cagatay Cebeci

See Also

`findpolypeaks`, `plotpolygon`, `genpolygon`

Examples

```
# Build a data vector with three peaks
x1 <-rnorm(100, mean=20, sd=5)
x2 <-rnorm(50, mean=50, sd=5)
x3 <-rnorm(150, mean=90, sd=10)
x <- c(x1,x3,x2)

# generate the frequency polygon and histogram of x by using Doane rule
hvals <- genpolygon(x, binrule="doane")
plotpolygon(x, nbins=hvals$nbins, ptype="p")

# find the peaks in frequency polygon of x by using the default threshold frequency
resfpp <- findpolypeaks(xm=hvals$mids, xc=hvals$freqs)
print(resfpp)

# remove the shoulders with the threshold distance option 'avg'
resrs <- rmshoulders(resfpp$pm[,1], resfpp$pm[,2], trmethod = "avg")
print(resrs)

# remove the shoulders with the threshold distance option 'iqr'
resrs <- rmshoulders(resfpp$pm[,1], resfpp$pm[,2], trmethod = "iqr")
print(resrs)

data(x5p4c)
# plot the frequnecy polygon and histogram of p2 in x5p4c data set
hvals <- genpolygon(x5p4c$p2, binrule="usr", nbins=30)
plotpolygon(x5p4c$p2, nbins=hvals$nbins, ptype="ph")

# find the peaks in frequency polygon of p2
resfpp <- findpolypeaks(xm=hvals$mids, xc=hvals$freqs, tcmethod = "min")
print(resfpp)

# remove the shoulders with threshold distance option 'q1'
resrs <- rmshoulders(resfpp$pm[,1], resfpp$pm[,2], trmethod = "q1")
print(resrs)

## Not run:
data(iris)
# plot the frequency polygon and histogram of Petal.Length in iris data set
# by using a user-defined class number
hvals <- genpolygon(iris$Petal.Length, binrule="usr", nbins=30)
plotpolygon(iris$Petal.Length, nbins=hvals$nbins, ptype="p")

# find the peaks in frequency polygon of Petal.Length with default
# threshold frequency value
resfpp <- findpolypeaks(xm=hvals$mids, xc=hvals$freqs)
print(resfpp)

# remove the shoulders with threshold option 'med'
resrs <- rmshoulders(resfpp$pm[,1], resfpp$pm[,2], trmethod = "med")
print(resrs)
```

```
## End(Not run)
```

---

x5p4c *Synthetic Data Set with 5 Variables and 4 Clusters*

Description

A synthetically created data frame consists of five continous variables that form four clusters.

Usage

```
data(x5p4c)
```

Format

A data frame with 400 rows and 5 numeric variables:

p1 a continous variable with one mode

p2 a continous variable with four modes

p3 a continous variable with two modes

p4 a continous variable with three modes

p5 a continous variable with two modes

Note

The data set x5p4c is recommended to use in comparing the performances of the internal validity indexes in cluster analysis.

Examples

```
data(x5p4c)
# descriptive statistics of the variables
summary(x5p4c)
# plot the histogram of the variable p2
hist(x5p4c$p2, breaks=15)
# scatter plots of the variable pairs
pairs(x5p4c)
```

Index