Package: teda (via r-universe)

August 30, 2024

Type Package

Title An Implementation of the Typicality and Eccentricity Data
Analysis Framework

Version 0.1.1

Maintainer David Ciar <dciar86@ceh.ac.uk>

Description The typicality and eccentricity data analysis (TEDA)
framework was put forward by Angelov (2013)
<DOI:10.14313/JAMRIS_2-2014/16>. It has been further developed
into multiple different techniques since, and provides a
non-parametric way of determining how similar an observation,
from a process that is not purely random, is to other
observations generated by the process. This package provides
code to use the batch and recursive TEDA methods that have been
published.

License GPL (>= 3)

LazyData TRUE

RoxygenNote 5.0.1

Imports graphics, stats

Suggests testthat

NeedsCompilation no

Author David Ciar [cre, aut], James Wright [aut]

Repository CRAN

Date/Publication 2017-01-22 11:25:24

Contents

---

plot.tedab *Plot the tedab object*

Description

Takes a tedab object and plots each metric individually

Usage

```
## S3 method for class 'tedab'
plot(x, ...)
```

Arguments

x	The teda batch (tedab) object with which to create the plot output.
...	additional arguments affecting the summary produced.

Details

Takes a tedab object and creates four plots in order of: eccentricity, typicality, normalised eccentricity, and normalised typicality.

---

print.tedab *Print the tedab object*

Description

Takes a tedab object and prints out the values within

Usage

```
## S3 method for class 'tedab'
print(x, ...)
```

Arguments

x	The teda batch (tedab) object with which to create the printed output.
...	additional arguments affecting the summary produced.

Details

Takes a tedab object and prints out each vector in order of: eccentricity, typicality, normalised eccentricity, and normalised typicality.

---

print.tedar	*Print the tedar object*

Description

#' @description Takes a tedar object and prints out the object values.

Usage

```
## S3 method for class 'tedar'
print(x, ...)
```

Arguments

x	The teda recursive (tedar) object with which to create the print output.
...	additional arguments affecting the summary produced.

Details

Takes a tedar object and prints out the values within (currently the same as summarize).

---

summary.tedab	*Summarise the tedab object*

Description

Summarises the teda batch object using an S3 method

Usage

```
## S3 method for class 'tedab'
summary(object, ...)
```

Arguments

object	The teda batch (tedab) object with which to create the summary output.
...	additional arguments affecting the summary produced.

Details

Takes a tedab object and prints out the following summary details:

- the number of observations
- the number of observations that exceed the normalised eccentricity limit
- the normalised eccentricity threshold

---

summary.tedar   *Summarize the tedar object*

Description

Takes a tedar object and prints out the summary values.

Usage

```
## S3 method for class 'tedar'
summary(object, ...)
```

Arguments

object	The teda recursive (tedar) object with which to create the summary output.
...	additional arguments affecting the summary produced.

Details

Takes a tedar object and prints out the summary values.

---

teda   *teda: An implementation of the Typicality and Eccentricity Data Analysis Framework.*

Description

The package provides functions to calculate both the batch and recursive typicality and eccentricity values of given observations.

Details

TEDA provides a non-parametric technique to determine how eccentric/typical an observation is with respect to the other observations generated by the same process. Available as either a batch function working over a whole dataset, or as a recursive one-time-pass function that needs the current mean and variance values to be passed as arguments.

Both batch and recursive methods return a datatype (tedab or tedar) which provide print and summary generic function implementations. The batch object also provides a generic plot function.

Further work will implement more of the analytical framework built up around TEDA, such as clustering algorithms.

References

Angelov, P., 2014. Outside the box: an alternative data analytics framework. Journal of Automation Mobile Robotics and Intelligent Systems, 8(2), pp.29-35. DOI: 10.14313/JAMRIS_2-2014/16

Bezerra, C.G., Costa, B.S.J., Guedes, L.A. and Angelov, P.P., 2016, May. A new evolving clustering algorithm for online data streams. In Evolving and Adaptive Intelligent Systems (EAIS), 2016 IEEE Conference on (pp. 162-168). IEEE. DOI: 10.1109/EAIS.2016.7502508

---

teda_b   *Create teda batch object from a vector*

Description

Takes a vector of observations and return a teda batch object, which holds the eccentricity and typicality values, both original and normalised versions.

Usage

```
teda_b(observations, dist_type = "Euclidean")
```

Arguments

observations   A vector of numeric observations

dist_type   A string representing the distance metric to use, default value (and currently only supported value) is "Euclidean"

Details

Uses the algorithm from Angelov (2014) to create a teda batch object. This contains a vector for the eccentricity (standard and normalised), typicality (standard and normalised), the outlier threshold, and whether each observation is or is not an outlier. Also provides the original vector of values.

Value

The teda batch object

References

Angelov, P., 2014. Outside the box: an alternative data analytics framework. Journal of Automation Mobile Robotics and Intelligent Systems, 8(2), pp.29-35. DOI: 10.14313/JAMRIS_2-2014/16

See Also

teda_r for the recursive version of the TEDA framework.

Other TEDA.functions: teda_r

Examples

```
vec = c(20, 12, 10)
teda_b(vec)
# same as
a = teda_b(vec,"Euclidean")
summary(a)
plot(a)
```

---

teda_r   *Create teda recursive object from observation (+ state)*

Description

A recursive method that takes the state variables of previous mean, previous variance, and the current timestep position, along with the current observation. It returns a teda recursive object. Currently only a univariate implementation.

Usage

```
teda_r(curr_observation, previous_mean = curr_observation, previous_var = 0,
  k = 1, dist_type = "Euclidean")
```

Arguments

curr_observation

        A single observation, the most recent in a series

previous_mean   The mean value returned by the previous call to this function, if no previous calls, default value is used.

previous_var   The variance value returned by the previous call to this function, if no previous calls, default value is used.

k   The count of observations processed by the recursive function, including the current observation

dist_type   A string representing the distance metric to use, default value (and currently only supported value) is "Euclidean"

Details

The function has two intended ways of use: on the first pass, it only takes the observation value as a paramter and the rest are provided by defaults, on all other passes, it takes the current observation, the previous mean and variance values, and the current k (number of observations) which includes the current observation.

On return, the teda recursive object holds:

- the current observation
- the current mean

- the current variance
- the current observation's eccentricity
- the current observation's typicality
- the current observation's normalised eccentricity
- the current observation's normalised typicality
- whether the current observation is an outlier
- the current outlier threshold
- the next timestep value, k+1

It provides generic functions for print and summary, at this moment both provide the same outout.

Value

The teda recursive object

References

Bezerra, C.G., Costa, B.S.J., Guedes, L.A. and Angelov, P.P., 2016, May. A new evolving clustering algorithm for online data streams. In Evolving and Adaptive Intelligent Systems (EAIS), 2016 IEEE Conference on (pp. 162-168). IEEE. DOI: 10.1109/EAIS.2016.7502508

See Also

Other TEDA.functions: teda_b

Examples

```
vec = c(20, 12, 10, 20)
a = teda_r(vec[1])
b = teda_r(vec[2],
           a$curr_mean,
           a$curr_var,
           a$next_k)
c = teda_r(vec[3],
           b$curr_mean,
           b$curr_var,
           b$next_k)

d = teda_r(vec[4],
           c$curr_mean,
           c$curr_var,
           c$next_k)
summary(d)
```

Index