Package ‘sdcSpatial’

July 19, 2019

Title Statistical Disclosure Control for Spatial Data

Version 0.1.1

Description Privacy protected raster maps can be created from spatial point data. Protection methods include smoothing of dichotomous variables by de Jonge and de Wolf (2016) <doi:10.1007/978-3-319-45381-1_9>, continuous variables by de Wolf and de Jonge (2018) <doi:10.1007/978-3-319-99771-1_23>, suppressing revealing values and a generalization of the quad tree method by Suñé, Rovira, Ibáñez and Farré (2017) <doi:10.2901/EUROSTAT.C2017.001>.

License GPL-2

Encoding UTF-8

LazyData true

URL `https://github.com/edwindj/sdcSpatial`

BugReports `https://github.com/edwindj/sdcSpatial/issues`

RoxygenNote 6.1.1

Suggests testthat, knitr, rmarkdown, sp, sf

Imports raster, methods

Depends R (>= 2.10)

VignetteBuilder knitr

NeedsCompilation no

Author Edwin de Jonge [aut, cre] (<https://orcid.org/0000-0002-6580-4718>),
Peter-Paul de Wolf [aut],
Sapphire Han [ctb]

Maintainer Edwin de Jonge `<edwindjonge@gmail.com>`

Repository CRAN

Date/Publication 2019-07-19 17:50:02 UTC

R topics documented:

`sdcSpatial-package` *Privacy Protected maps*

Description

`sdcSpatial` contains functions to create spatial distribution maps, assess the risk of disclosure on a location and to suppress or adjust revealing values at certain locations.

Details

`sdcSpatial` working horse is the `sdc_raster()` object upon which the following methods can be applied:

Sensitivity assessment

- `plot.sdc_raster()`,`plot_sensitive()`
- `print`
- `is_sensitive()`

Protection methods

- `remove_sensitive()`
- `protect_smooth()`
- `protect_quadtree()`

Extraction

- `sum`, extract the `sum` layer from a `sdc_raster` object
- `mean`, extract the `mean` layer from a `sdc_raster` object

Author(s)

Maintainer: Edwin de Jonge <edwindjonge@gmail.com> (0000-0002-6580-4718)

Authors:

- Peter-Paul de Wolf

Other contributors:

- Sapphire Han [contributor]

References

de Jonge, E., & de Wolf, P. P. (2016, September). Spatial smoothing and statistical disclosure control. In International Conference on Privacy in Statistical Databases (pp. 107-117). Springer, Cham.

de Wolf, P. P., & de Jonge, E. (2018, September). Safely Plotting Continuous Variables on a Map. In International Conference on Privacy in Statistical Databases (pp. 347-359). Springer, Cham.

Suñé, E., Rovira, C., Ibáñez, D., Farré, M. (2017). Statistical disclosure control on visualising geocoded population data using a structure in quadtrees, NTTS 2017

See Also

Useful links:

- `https://github.com/edwindj/sdcSpatial`
- Report bugs at `https://github.com/edwindj/sdcSpatial/issues`

`disclosure_risk` *Calculate disclosure risk for raster cells*

Description

The disclosure risk function is used by `is_sensitive()` to determine the risk of a `raster` cell. It returns a score between 0 and 1 for cells that have a finite value (otherwise `NA`).

Usage

```
disclosure_risk(x, risk_type = x$risk_type)
```

Arguments

`x`	`sdc_raster` object.
`risk_type`	`character`: "external", "internal", "discrete".

Details

Different risk functions include:

- external (numeric variable), calculates how much the largest value comprises the total sum within a cell
- internal (numeric variable), calculates how much the largest value comprises the sum without the second largest value
- discrete (logical variable), calculates the fraction of `TRUE` vs `FALSE`

Value

raster::raster object with the disclosure risk.

See Also

Other sensitive: `is_sensitive`, `plot_sensitive`, `remove_sensitive`, `sdc_raster`, `sensitivity_score`

`dwellings` *Simulated dwellings data set*

Description

The data are generated with residence/household locations from the Dutch open data BAG register[1]. The locations are realistic, but the associated data is simulated.

Usage

```
dwellings
```

Format

An object of class `data.frame` with 90603 rows and 4 columns.

Details

- `x`, `integer`, x coordinate of dwelling (crs 28992)
- `y`, `integer`, y coordinate of dwelling (crs 28992)
- `consumption`, `numeric`, simulated continuous value
- `unemployed`, `logical`, simulated discrete value

References

Basisregistratie Adressen en Gebouwen `https://zakelijk.kadaster.nl/bag-producten`

[1] `https://zakelijk.kadaster.nl/bag-producten`

Examples

```
# dwellings is a data.frame, the best way is to first turn it
# into a sf or sp object.

# create an sf object from our data
if (requireNamespace("sf")){
  dwellings_sf <- sf::st_as_sf(dwellings, coords=c("x", "y"), crs=28992)

  unemployed <- sdc_raster( dwellings_sf
                          , "unemployed"
                          , r=200
                          , max_risk = 0.9
                          )

  plot(unemployed)
  sensitivity_score(unemployed)

  unemployed_smoothed <- protect_smooth(unemployed, bw = 0.4e3)
  plot(unemployed_smoothed, main="Employment rate")
  plot(unemployed_smoothed, "sum", main = "Employment")
} else {
  message("Package 'sf' was not installed.")
}

dwellings_sp <- dwellings
# or change a data.frame into a sp object
sp::coordinates(dwellings_sp) <- ~ x + y
tryCatch(
  # not working on some OS versions.
  sp::proj4string(dwellings_sp) <- "+init=epsg:28992"
)
consumption <- sdc_raster(dwellings_sp, dwellings_sp$consumption, r = 500)
consumption

plot(consumption)

# but we can also create a raster directly from a data.frame
unemployed <- sdc_raster( dwellings[c("x","y")], dwellings$unemployed)
```

`enterprises` *Simulated data set with enterprise locations.*

Description

`enterprises` is generated from the dutch open data BAG register[2]. The locations are realistic, but the associated data is simulated.

[2] https://zakelijk.kadaster.nl/bag-producten

Usage

```
enterprises
```

Format

An object of class `SpatialPointsDataFrame` with 8348 rows and 2 columns.

Details

- `production` `numeric` simulated production (lognormal).
- `fined` `logical` simulated variable if an enterprise is fined or not.

References

Basisregistratie Adressen en Gebouwen: `https://zakelijk.kadaster.nl/bag-producten`

Examples

```
library(sdcSpatial)
library(raster)

data("enterprises")

production <- sdc_raster(enterprises, "production", min_count = 10)
print(production)

# show the average production per cell
plot(production, "mean")
production$min_count <- 2 # adjust norm for sdc
plot(production)

production_safe <- remove_sensitive(production)
plot(production_safe)
```

`is_sensitive` *Return raster with sensitive locations.*

Description

Create a binary raster with sensitive locations.

Usage

```
is_sensitive(x, max_risk = x$max_risk, min_count = x$min_count,
  risk_type = x$risk_type)
```

Arguments

x	sdc_raster object.
max_risk	a risk value higher than max_risk will be sensitive.
min_count	a count lower than min_count will be sensitive.
risk_type	what kind of measure should be used (see details).

Details

By default the risk settings are taken from x, but they can be overriden.

Different risk functions can be used:

- external (numeric variable), calculates how much the largest value comprises the total sum
- internal (numeric variable), calculates how much the largest value comprises the sum without the second largest value
- discrete (logical variable), calculates the fraction of sensitive values.

See Also

Other sensitive: disclosure_risk, plot_sensitive, remove_sensitive, sdc_raster, sensitivity_score

Examples

```
dwellings_sp <- dwellings
sp::coordinates(dwellings_sp) <- ~ x + y
tryCatch(
  # does not work on some OS versions
  sp::proj4string(dwellings_sp) <- "+init=epsg:28992"
)
# create a 1km grid
unemployed <- sdc_raster(dwellings_sp, dwellings_sp$unemployed, r = 1e3)
print(unemployed)

# retrieve the sensitive cells
is_sensitive(unemployed)
```

---

plot.sdc_raster *Plot a sdc_raster object*

Description

Plot a sdc_raster object together with its sensitivity.

Usage

```
## S3 method for class 'sdc_raster'
plot(x, value = "mean", sensitive = TRUE, ...,
  main = paste(substitute(x)), col)
```

Arguments

`x`	`sdc_raster` object to be plotted
`value`	`character` which value layer to be used for plotting, e.g. "sum", "count", "mean" (default).
`sensitive`	`logical` show the sensitivity in the plot?
`...`	passed on to `raster::plot()`
`main`	title of plot
`col`	color palette to be used, passed on to `raster::plot()`.

Details

When `sensitive` is set to `TRUE`, a side-by-side plot will be made of the `value` and its `sensitivity`.

See Also

Other plotting: `plot_sensitive`

`plot_sensitive` *Plot the sensitive cells of the sdc_raster.*

Description

Plots t the sensitive cells of the sdc_raster. The sensitive cells are plotted in red. The sensitive cells are determined using `is_sensitive`.

Usage

```
plot_sensitive(x, value = "mean", main = "sensitive", col, ...)
```

Arguments

`x`	sdc_raster object
`value`	character which value layer to be used for values, e.g. "sum", "count", "mean" (default).
`main`	character title of map.
`col`	color palette to be used, passed on to `raster::plot()`.
`...`	passed on to `plot.sdc_raster`.

See Also

Other plotting: `plot.sdc_raster`

Other sensitive: `disclosure_risk`, `is_sensitive`, `remove_sensitive`, `sdc_raster`, `sensitivity_score`

protect_quadtree *Protect a raster with a quadtree method.*

Description

protect_quadtree reduces sensitivy by aggregating sensisitve cells with its three neighbors, and does this recursively until no sensitive cells are left or when the maximum zoom levels has been reached.

Usage

```
protect_quadtree(x, max_zoom = Inf, ...)
```

Arguments

x	sdc_raster object to be protected.
max_zoom	numeric, restricts the number of zoom steps and thereby the max resolution for the blocks. Each step will zoom with a factor of 2 in x and y so the max resolution = resolution * 2^max_zoom.
...	Arguments passed on to is_sensitive **x** sdc_raster object. **max_risk** a risk value higher than max_risk will be sensitive. **min_count** a count lower than min_count will be sensitive. **risk_type** what kind of measure should be used (see details).

Details

This implementation generalizes the method as described by Suñé et al., in which there is no risk function, and only a min_count to determine sensitivity. Furthermore the method the article only handles count data (x$value$count), not mean or summed values. Currently the translation feature of the article is not (yet) implemented, for the original method does not take the disclosure_risk into account.

Value

a sdc_raster object, in which sensitive cells have been recursively aggregated until not sensitive or when max_zoom has been reached.

References

Suñé, E., Rovira, C., Ibáñez, D., Farré, M. (2017). Statistical disclosure control on visualising geocoded population data using a structure in quadtrees, NTTS 2017

See Also

Other protection methods: protect_smooth, remove_sensitive

Examples

```
library(raster)

fined <- sdc_raster(enterprises, enterprises$fined)
plot(fined)
fined_qt <- protect_quadtree(fined)
plot(fined_qt)

fined <- sdc_raster(enterprises, enterprises$fined, r=50)
plot(fined)
fined_qt <- protect_quadtree(fined)
plot(fined_qt)
```

`protect_smooth` *Protect a sdc_raster by smoothing*

Description

`protect_smooth` reduces the sensitivity by applying a Gaussian smoother, making the values less localized.

Usage

```
protect_smooth(x, bw = raster::res(x$value), ...)
```

Arguments

`x`	raster object
`bw`	bandwidth
`...`	passed through to `focal`.

Details

The sensitivity of a raster can be decreased by applying a kernel density smoother as argued by de Jonge et al. (2016) and de Wolf et al. (2018). Smoothing spatially spreads localized values, reducing the risk for location disclosure. Note that smoothing often visually enhances detection of spatial patterns. The kernel applied is a Gaussian kernel with a bandwidth `bw` supplied by the user. The smoother acts upon the `x$value$count` and `x$value$sum` from which a new `x$value$mean` is derived.

References

de Jonge, E., & de Wolf, P. P. (2016, September). Spatial smoothing and statistical disclosure control. In International Conference on Privacy in Statistical Databases (pp. 107-117). Springer, Cham.

de Wolf, P. P., & de Jonge, E. (2018, September). Safely Plotting Continuous Variables on a Map. In International Conference on Privacy in Statistical Databases (pp. 347-359). Springer, Cham.

See Also

Other protection methods: `protect_quadtree`, `remove_sensitive`

Examples

```
library(sdcSpatial)
library(raster)

data(enterprises)

# create a sdc_raster from point data with raster with
# a resolution of 200m
production <- sdc_raster(enterprises, variable = "production"
                        , r = 200, min_count = 3)

print(production)

# plot the raster
zlim <- c(0, 3e4)
# show which raster cells are sensitive
plot(production, zlim=zlim)

# but we can also retrieve directly the raster
sensitive <- is_sensitive(production, min_count = 3)
plot(sensitive, col = c('white', 'red'))

# what is the sensitivy fraction?
sensitivity_score(production)
# or equally
cellStats(sensitive, mean)

# let's smooth to reduce the sensitivity
smoothed <- protect_smooth(production, bw = 400)
plot(smoothed)

# what is the sensitivy fraction?
sensitivity_score(smoothed)

# let's remove the sensitive data.
smoothed_safe <- remove_sensitive(smoothed, min_count = 3)
plot(smoothed_safe, zlim=zlim)

# let's communicate!
production_mean <- mean(smoothed_safe)
production_total <- sum(smoothed_safe)

# and create a contour plot
raster::filledContour(production_mean, nlevels = 6, main = "Mean production")


# generated with R 3.6 >=
#col <- hcl.colors(10, rev=TRUE)
```

```
col <- c("#FDE333", "#BBDD38", "#6CD05E", "#00BE7D",
         "#00A890", "#008E98",  "#007094", "#185086", "#422C70", "#4B0055")
raster::filledContour(production_total, nlevels = 10
             , col = col
             , main="Total production")
```

---

remove_sensitive *Remove sensitive cells from raster*

Description

`remove_sensitive` removes sensitive cells from a `sdc_raster`. The sensitive cells, as found by `is_sensitive()` are set to NA.

Usage

```
remove_sensitive(x, max_risk = x$max_risk, min_count = x$min_count,
  ...)

mask_sensitive(x, max_risk = x$max_risk, min_count = x$min_count, ...)
```

Arguments

`x`	`sdc_raster` object.
`max_risk`	a risk value higher than `max_risk` will be sensitive.
`min_count`	a count lower than `min_count` will be sensitive.
`...`	passed on to `is_sensitive`.

Details

Removing sensitive cells is a protection method, which often is useful to finalize map protection after other protection methods have been applied. `mask_sensitive` and `remove_sensitive` are synonyms, to accommodate both experienced `raster` users as well as sdc users.

Value

sdc_raster object with sensitive cells set to `NA`.

See Also

Other sensitive: `disclosure_risk`, `is_sensitive`, `plot_sensitive`, `sdc_raster`, `sensitivity_score`

Other protection methods: `protect_quadtree`, `protect_smooth`

Examples

```
library(raster)

unemployed <- sdc_raster(dwellings[1:2], dwellings$unemployed, r=200)

# plot the normally rastered data
plot(unemployed, zlim=c(0,1))
plot_sensitive(unemployed)

unemployed_safe <- remove_sensitive(unemployed, risk_type="discrete")
plot_sensitive(unemployed_safe, zlim=c(0,1))
print(unemployed)
unemployed$value
```

`sdc_raster` *Create a raster map with privacy awareness*

Description

`sdc_raster` creates multiple `raster::raster` objects ("count", "mean", "sum") from supplied point data `x` and calculates the sensitivity to privacy disclosure for each location.

Usage

```
sdc_raster(x, variable, r = 200, max_risk = 0.95, min_count = 10,
  risk_type = c("external", "internal", "discrete"), ...,
  field = variable)
```

Arguments

`x`	sp::SpatialPointsDataFrame, sf::sf or a two column matrix or data.frame that is used to create a raster map.
`variable`	name of data column or `numeric` with same length as `x` to be used for the data in the raster map.
`r`	either a desired resolution or a pre-existing raster object. In the first case, the crs of `x` (if present) will be used, in the latter the properties of the `r` will be kept.
`max_risk`	`numeric`, the maximum_risk score (`disclosure_risk`) before a cell in the map is considered sensitive.
`min_count`	`numeric`, a raster cell with less then `min_count` observations is considered sensitived.
`risk_type`	passed on to `disclosure_risk()`.
`...`	passed through to `raster::rasterize()`
`field`	synonym for `variable`. If both supplied, `field` has precedence.

Details

A sdc_raster object is the vehicle that does the book keeping for calculating sensitivity. Protection methods work upon a sdc_raster and return a new sdc_raster in which the sensitivity is reduced. The sensitivity of the map can be assessed with sensitivity_score, plot.sdc_raster(), plot_sensitive() or print. Reducing the sensitivity can be done with protect_smooth(), protect_quadtree() and remove_sensitive(). Raster maps for mean, sum and count data can be extracted from the $value (brick()).

Value

object of class "sdc_raster":

- $value: raster::brick() object with differenct layers e.g. count, sum, mean.
- $max_risk: see above.
- $min_count: see above.
- $scale: used together with min_count to determine sensitivity: result of protection operation protect_smooth() or protect_quadtree().
- $type: data type of variable, either numeric or logical
- $risk_type, "external", "internal" or "discrete" (see disclosure_risk())

See Also

Other sensitive: disclosure_risk, is_sensitive, plot_sensitive, remove_sensitive, sensitivity_score

Examples

```
library(raster)
prod <- sdc_raster(enterprises, field = "production", r = 500)
print(prod)

prod <- sdc_raster(enterprises, field = "production", r = 1e3)
print(prod)

# get raster with the average production per cell averaged over the enterprises
prod_mean <- mean(prod)
summary(prod_mean)

# get raster with the total production per cell
prod_total <- sum(prod)
summary(prod_total)
```

sensitivity_score *Mean sensitivity for raster*

Description

sensitivity_score calculates the fraction of cells (with a value) that are considered sensitive according to the used disclosure_risk

Usage

```
sensitivity_score(x, max_risk = x$max_risk, min_count = x$min_count,
  ...)
```

Arguments

x	sdc_raster object.
max_risk	a risk value higher than max_risk will be sensitive.
min_count	a count lower than min_count will be sensitive.
...	passed on to is_sensitive

See Also

Other sensitive: disclosure_risk, is_sensitive, plot_sensitive, remove_sensitive, sdc_raster

Examples

```
consumption <- sdc_raster(dwellings[1:2], variable = dwellings$consumption, r = 500)

sensitivity_score(consumption)
# same as
print(consumption)

# change the rules! A higher norm generates more sensitive cells
sensitivity_score(consumption, min_count = 20)
```

smooth_raster *Create kde density version of a raster*

Description

Create kde density version of a raster

Usage

```
smooth_raster(x, bw = raster::res(x), smooth_fact = 5,
  keep_resolution = TRUE, na.rm = TRUE, pad = TRUE,
  threshold = NULL, ...)
```

Arguments

`x`	raster object
`bw`	bandwidth
`smooth_fact`	`integer`, disaggregate factor to have a better smoothing
`keep_resolution`	`integer`, should the returned map have same resolution as `x` or keep the disaggregated raster resulting from `smooth_fact`?
`na.rm`	should the `NA` value be removed from the raster?
`pad`	should the data be padded?
`threshold`	cells with a lower (weighted) value of this threshold will be removed.
`...`	passed through to `focal`.