Package ‘chicagocrime’

January 19, 2020

Type Package

Title Organises and Explores the Chicago Crime Dataset

Version 0.1.0

Author Daniel Williams

Maintainer The package maintainer <fs19144@bristol.ac.uk>

Description Organises and Explores the Chicago Crime Dataset.

License CC0

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

Depends R (>= 3.5.0)

Imports dplyr, tidyr, geosphere, utils, Matrix, stats, graphics, grDevices, stringr, KernSmooth, leaflet, sp, tibble, magrittr

Suggests testthat (>= 2.1.0)

R topics documented:

`add_density_attributes_to_data`
Add density attributes to data

Description

Add density attributes to data

Usage

```
add_density_attributes_to_data(crime_data, crime_kde, population_kde)
```

Arguments

`crime_data`	full crime dataset
`crime_kde`	kernel density estimate of crimes
`population_kde`	kernel density estimate of population

Value

crime dataset with three additional columns:

`Crime Density`	crime density at the location of crime
`Population Density`	population density at the location of crime
`Ratio Density`	crime/population density at the location of crime

`add_location_to_population_data` *Add Location Data*

Description

Adds locations information to population data from census block id

Usage

```
add_location_to_population_data(block_boundaries_data, population_data)
```

Arguments

`block_boundaries_data`
: dataset of census block boundaries

`population_data`
: dataset of population by census block

Value

population dataset with location data

`all_crime` *Chicago Crime data from 2001 to 2019*

Description

A dataset containing reported crimes in the city of Chicago in the period 2001 to 2019.

Usage

```
data(all_crime)

data(sub_all_crime)
```

Format

A data frame with 6,336,525 rows and 14 variables:

Date date of which the crime was reported

Primary Type type of crime that was reported, one of a finite amount of options

Description description of the reported crime

Location Description type of place where the crime was committed

Arrest logical; if `TRUE`, then the crime resulted in an arrest

Domestic logical; if `TRUE`, then the incident was domestic-related

Beat the beat where the crime occurred (smallest police geographical area)

District police district where the crime occurred

Ward the ward (City Council district) in Chicago where the crime occurred

Community Area the community area where the incident occurred

Year year when the crime happened

Latitude latitude co-ordinate (in degrees) of the block where the crime occurred

Longitude longitude co-ordinate (in degrees) of the block where the crime occurred

Hour estimated hour of the day when the crime happened

Source

https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2

`auc_in` *AUC Calculator*

Description

See `roc.lr` for details. This function is used to only output the AUC value for use in other functions, such as `cv.lr`.

Usage

```
auc_in(p, y)
```

Arguments

`p`	a vector of predictions
`y`	a vector of the true observed response variable

Value

the AUC value

block_boundaries *Chicago Census Block Boundaries*

Description

A dataset containing the locations of census blocks in Chicago

Usage

```
block_boundaries
```

Format

A data frame with 46291 rows and 3 variables:

the_geom string indicating polygon co-ordinates of the census block

STATEFP10 two digit number referring to state code of census block

COUNTYFP10 three digit number refferring to county code of census block

TRACTCE10 six digit number referring to area in chicago of census block

BLOCKCE10 four digit number referring to block in census block

GEOID10 full identification number of census block, same as `CENSUS BLOCK FULL` in `block_boundaries`

NAME10 name of census block

TRACT_BLOC shortened identification number of the census block, without the first 5 numbers (relating to area in country), same as `CENSUS BLOCK` in `block_boundaries`

Source

https://data.cityofchicago.org/Facilities-Geographic-Boundaries/Boundaries-Census-Blocks-2010/mfzt-js4n

block_populations *Chicago Population data by Census block*

Description

A dataset containing the population totals of census blocks in the city of Chicago

Usage

```
block_populations
```

Format

A data frame with 46291 rows and 3 variables:

CENSUS BLOCK shortened identification number of the census block, without the first 5 numbers (relating to area in country), same as `TRACT_BLOC` in `block_populations`

CENSUS BLOCK FULL full identification number of the census block, same code as `GEOID10` in `block_populations`

TOTAL POPULATION total population per census block

Source

https://data.cityofchicago.org/Facilities-Geographic-Boundaries/Population-by-2010-Census-Block/5yjb-v3mj

---

`census` *Chicago Census data from 2008-2012*

Description

A dataset containing selected socioeconomic indicators in the city of Chicago from 2008 to 2012

Usage

```
census
```

Format

A data frame with 78 rows and 9 variables:

Community Area Number identification number of the community area, matches with that in `all_crime`

COMMUNITY AREA NAME name of the community area

PERCENT OF HOUSING CROWDED Percentage of occupied housing units, in the community area, with more than one person per room

PERCENT HOUSEHOLDS BELOW POVERTY Percentage of households, in the community area, that are living below the federal poverty level

PERCENT AGED 16+ UNEMPLOYED Percentage of persons, in the community area, that are over the age of 16 years that are unemployed

PERCENT AGED 25+ WITHOUT HIGH SCHOOL DIPLOMA Percentage of persons in the community area that are over the age of 25 years without a high school education

PERCENT AGED UNDER 18 OR OVER 64 Percent of the population of the community area under 18 or over 64 years of age

PER CAPITA INCOME Community Area Per capita income; estimated as the sum of tract-level aggregate incomes divided by the total population

HARDSHIP INDEX Score that incorporates each of the six selected socioeconomic indicators

Source

https://data.cityofchicago.org/Health-Human-Services/Census-Data-Selected-socioeconomic-indicators-
kn9c-c2s2

`cleanData` *Clean Data*

Description

Clean Data

Usage

```
cleanData(crime_data)
```

Arguments

`crime_data`	a data frame to be cleaned

Value

A cleaned dataset with NA entries removed

Examples

```
## Not run:
data(sub_all_crime)
sub_all_crime = cleanData(sub_all_crime)
## End(Not run)
```

`create_spatial_attributes` *Create spatial attributes*

Description

Create spatial attributes

Usage

```
create_spatial_attributes(
  crime_data,
  block_boundaries_data,
  population_data,
  h = 0.01,
  grid.size = 100L
)
```

Arguments

- `crime_data` full crime dataset
- `block_boundaries_data` dataset of census block boundaries
- `population_data` dataset of population by census block
- `h` bandwidth for kernel density estimation
- `grid.size` grid size for kernel density estimation

Value

dataset with attributes added

---

cv.lr — *Cross-Validation of Logistic Regression Model*

Description

Implementation of cross-validation for a `lr` object, calculation of error across a number of subsets of the inputted data set.

Usage

```
cv.lr(
  lrfit,
  metric = "mse",
  leave_out = nrow(lrfit$data)/10,
  verbose = TRUE,
  seed = 1
)
```

Arguments

- `lrfit` an object of class "`lr`", the output to `lr`
- `metric` which metric to calculate, one of "mse", "auc" or "both". See 'Details'.
- `leave_out` number of points to leave out for cross-validation.
- `verbose` logical; whether to print information about number of iterations completed.
- `seed` optional; number to be passed to `set.seed` before shuffling the data set

Details

k-fold cross-validation, where k is the input to the `leave_out` argument. This can be used to judge the out-of-sample predictive power of the model by subsetting the original data set into two partitions; fitting the model for the (usually larger) one, and testing the predictions of that model on the (usually smaller) partition. The position of the k points separated from the data set are selected uniformly at random.

The error metrics available are that of mean squared error, AUC, or log score; selected by the `metric` argument being one of "mse", "auc", "log" or "all". See `roc.lr` for details on AUC. If `metric` is "all", then a vector will be output containing all three metrics.

Note that the output from `metric = "auc"` has non-deterministic elements due to the shuffling of the data set. To mitigate this, include a number to the `seed` argument.

Value

error value or vector consisting of the average of the chosen `metric`

`fit_kde` *Fit Kernel Density Estimator*

Description

Fits Kernel density estimates using `bkde2D`

Usage

```
fit_kde(data, h, grid.size)
```

Arguments

`data`	Data frame containing columns `Longitude` and `Latitude`.
`h`	Bandwidth of KDE.
`grid.size`	Number of rows/columns of grid on which density estimate is returned.

Value

Kernel density estimate in the form:

`x1`	Longitudes of grid
`x2`	Latitudes of grid
`fhat`	Matrix of density estimates at grid points

formatData *Format crime data to include census and police station data*

Description

Format crime data to include census and police station data

Usage

```
formatData(crime_data, densities = TRUE, verbose = TRUE, ...)
```

Arguments

crime_data	dataframe from data(all_crime)
densities	logical; if TRUE, population densities, crime densities, as well as the ratio between the two will also be joined to crime_data
verbose	logical; if TRUE, formatting procedures will be displayed as they are completed
...	further arguments to be passed to create_spatial_attributes

Value

data of the same form of crime_data, with appended census and police station data

Examples

```
## Not run:
data(sub_all_crime)
sub_all_crime = cleanData(sub_all_crime)
sub_all_crime = formatData(sub_all_crime)
## End(Not run)
```

get_error *Match Error Function*

Description

Used to get error function in cv.lr.

Usage

```
get_error(type = "mse")
```

Arguments

type	type of error function to be output, either "mse", "auc" or "log"

Value

function of either mean squared error or auc_in

---

irls.lr *Iteratively Re-weighted Least Squares*

Description

Optimisation procedure based on iteratively re-weighted least squares, called by lr.

Usage

```
irls.lr(y, x, init = NULL, tol = 1e-06, maxiter = 100)
```

Arguments

y	response vector
x	model matrix of covariates
init	initial estimate of theta
tol	tolerance parameter, default 1e-6
maxiter	optional number of iterations

Details

Iterative method used to find parameter estimates of β from the least squares problem

$$\beta = argmin(z - X\beta)^T W(z - X\beta),$$

where W is a diagonal matrix of weights with i-th diagonal element being

$$\sigma(x_i; \beta)(1 - \sigma(x_i; \beta)),$$

and z is the vector

$$z = X\beta + W^{-1}(y - \sigma(x_i; \beta)).$$

The parameter vector β is updated iteratively with a Newton update of the form

$$\beta = (X^T WX)^{-1} X^T Wz.$$

Value

a list containing

par	a vector of estimates of theta, the parameters being optimised
val	the value of the log-likelihood at the final theta estimate
iters	the number of iterations needed to converge

joinLonLat *Join by Longitude/Latitude*

Description

Append one data frame to another by nearest longitude and latitude

Usage

```
joinLonLat(
  x,
  y,
  f = NULL,
  lonname = "Longitude",
  latname = "Latitude",
  outname = "distance",
  inc_lonlat = TRUE,
  verbose = TRUE
)
```

Arguments

x	data frame to be appended to; should be the larger of x and y
y	data frame to join onto x
f	function to define what column is added to x, see 'Details' and 'Examples'
lonname	name of the longitude column shared by both data frames
latname	name of the latitude column shared by both data frames
outname	name of the column added to x by f
inc_lonlat	logical; if TRUE, longitude and latitude co-ordinates from y will be added to x
verbose	logical; if TRUE, progress bars will be displayed as data sets are joining

Details

This function first finds the distance between two sets of co-ordinates, for each data frame, using the haversine distance function (from the geosphere package),

$$d = 2rarcsin(\sqrt{sin^2((\theta_2 - \theta_1)/2) + cos(\theta_1)cos(\theta_2)sin^2((\phi_2 - \phi_1)/2)}),$$

where (ϕ_1, θ_1) is the longitude and latitude of point 1, and (ϕ_2, θ_2) is the longitude and latitude of point 2, both in radians, and r is the radius of the Earth (6378137m). Once the distances are found, the smallest distances are calculated and those rows in the data frame y that are closest to each row in x are appended

The function f decides what column will be appended to the original data frame x, it takes two arguments (but both do not need to be used), where the first argument uses data frame x, and the second uses y.

Value

the original data frame x, with one appended column defined by f, and the longitudes/latitudes if inc_lonlats=TRUE.

Examples

```
coords1 = data.frame("Longitude" = c(-20, -25, -23, -40),
                     "Latitude" = c(10, 15, 23, 13))
coords2 = data.frame("Longitude" = c(-17, -15, -26),
                "Latitude"= c(11, 31, 42),
                "Names" = c("Frasier", "Niles", "Martin"))
joinLonLat(coords1, coords2, f = function(z1, z2) z2$Names, outname="Closest_Names")
```

---

loglik_lr — *Bernoulli distribution log-likelihood*

Description

Log-likelihood of the Bernoulli distribution, commonly used in optimisation procedures for maximisation.

Usage

```
loglik_lr(theta, x, y)
```

Arguments

theta	parameters relating to x
x	explanatory variables in model matrix
y	binary response variable

Details

The log-likelihood is written

$$\sum_{i=1}^{n} y \log(\sigma(X\beta)) + (1 - y) \log(1 - \sigma(X\beta))$$

Value

single value, the log-likelihood of the Bernoulli distribution with fixed x and y

Examples

```
theta = c(1, 0.1)
X = matrix(rnorm(4), 50, 2)
y = sample(0:1, 50, replace=TRUE)
loglik_lr(theta, X, y)
```

log_score *Log Score*

Description

Log score diagnostic based on probabilities of predicting the correct class

Usage

```
log_score(p, y)
```

Arguments

p	a vector of probabilities
y	a vector of the true observed response variable

Details

The log score is defined as

$$LS = 1/n \sum -logp(z)$$

where $p(z)$ is the probability of predicting the correct value z. This is averaged over all data points.

The log score penalises probabilities that the model assigns to the correct class that are low, and rewards those that are high in the correct place.

Value

a single value, the mean of the negative log of the probabilities for predicting the correct class

lr *Logistic Regression*

Description

`lr` is used to fit a logistic regression model for a binary response variable.

Usage

```
lr(formula, data, init = NULL)
```

Arguments

formula	an object of class `formula`, a symbolic description of the model to be fitted. The specified names need to also be in `data`.
data	a required data frame containing the variables in the model
init	optional initial conditions to be passed to optimisation of the log-likelihood

Details

The form of the formula argument will be of the form response ~ predictor1 + predictor2 + ..., with predictor1 and predictor2 being named columns of the data frame in data.

The log-likelihood (from loglik_lr) is maximised using irls.lr with initial estimates given by init. If no initial values are supplied, this uses a vector of zeros instead.

Value

An S3 object of class 'lr', which is a list containing

coefficients	a vector of coefficients corresponding to covariates specified in formula
data	the data input to the function
formula	the formula input to the function
X	the model matrix X
val	value of the final log-likelihood at the values of coefficients, given by loglik_lr
its	number of iterations performed to retrieve the maximised log-likelihood

Examples

```
y = sample(0:1, 50, replace=TRUE)
d = data.frame(y = y, x = rnorm(10*y + 15))
fit = lr(y ~ x, data = d)
```

---

make_population_data_for_kde
Format population data

Description

Make population data suitable for fit_kde function

Usage

```
make_population_data_for_kde(population_data)
```

Arguments

population_data
: dataset of populations at locations

Value

dataset of home locations with one row per person

plot_kde *Plot Kernel Density Estimate*

Description

Plots kernel density estimates on map using `leaflet`

Usage

```
plot_kde(kde)
```

Arguments

kde	Kernel density estimate object generated using `fit_kde`

police_stations *Chicago Police Station locations*

Description

A dataset containing the longitudes and latitudes of police stations in the city of Chicago, updated June 2016

Usage

```
police_stations
```

Format

A data frame with 23 rows and 2 variables:

lon longitude of a police station

lat latitude of a police station

Source

https://data.cityofchicago.org/Public-Safety/Police-Stations/z8bn-74gv

predict.lr — *Predict from a Logistic Regression model*

Description

Predicted values of a certain type based on a lr object.

Usage

```
## S3 method for class 'lr'
predict(object, newdata = NULL, thresh = 0, type = "preds", ...)
```

Arguments

object	an object of class "lr", the output from lr
newdata	an optional data frame to predict from. If omitted, predictions will be from the original data frame used to fit the model.
thresh	an optional threshold parameter, see 'Details'.
type	type of predictions, can be one of "preds", "probs" or "vals", see 'Details'.
...	further arguments

Details

Predictions from a logistic regression model are formed by first multiplying the model matrix by the parameter vector, i.e.

$$f(x; \beta) := X\beta.$$

These values are outputted when the `type` argument is "vals". Predictions for the positive class can be obtained when these values are above a certain threshold value, the argument `thresh`, i.e.

$$f(x; \beta) \geq t,$$

where the threshold t is usually equal to zero. This is how the predictions are made when the `type` argument is "preds". To get probabilities, the sigmoid function is used on the values, i.e.

$$p(y = 1) = 1/(1 + e^{-X\beta}).$$

Value

a vector of predictions, probabilities or values, depending on the input to `type`.

Examples

```
y = sample(0:1, 50, replace=TRUE)
d = data.frame(y = y, x = rnorm(10*y + 15))
fit = lr(y ~ x, data = d)
predict(fit, type="preds")
```

`print.lr` *Print Logistic Regression Model*

Description

Displays important output from a `lr` object; the parameter estimates and the maximised log-likelihood value.

Usage

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

Arguments

`x`	an object of class "`lr`", the output from `lr`
`...`	further arguments

Value

a printed, named vector of coefficients and log-likelihood value at these estimates

Examples

```
y = sample(0:1, 50, replace=TRUE)
d = data.frame(y = y, x = rnorm(10*y + 15))
fit = lr(y ~ x, data = d)
print(fit)
```

`process_geom_string` *Geom string process Process* `the_geom` *string to return block center*

Description

Geom string process Process `the_geom` string to return block center

Usage

```
process_geom_string(x, long = TRUE)
```

Arguments

`x`	`the_geom` string from block boundaries data
`long`	if TRUE returns longitude, else returns latitide

Value

center of block (longitude or latitude)

roc.lr *Receiver Operating Characteristic (ROC) curve*

Description

A method of judging the predictive performance of a model by plotting and/or averaging the probability of predicting the positive class correctly, over multiple thresholds. See 'Details'.

Usage

```
roc.lr(lrfit, newdata = NULL, plot = TRUE, len = 50)
```

Arguments

`lrfit`	an object of class "`lr`", the output from `lr`
`newdata`	an optional data frame to predict from. If ignored, the default data frame is that used to fit the original model.
`plot`	logical; if `TRUE`, then the ROC curve is plotted
`len`	optional; number of different thresholds to use.

Details

A positive prediction from a logistic regression model is made when

$$f(x; \beta) := X\beta \geq t.$$

where t is some threshold. See `predict.lr` for details. A different threshold t_0 will yield a different set of predictions. For a given sequence $t_j in[min(t), max(t)]$, for $j = 1, \ldots, J$, the True Positive Rate (TPR) and False Positive Rate (FPR) can be calculated as

$$TPR(j) = \sum I(f(x_i; \beta) \geq t_j) / \sum I(y_i = 1),$$

$$FPR(j) = \frac{\sum I(f(x_i; \beta) < t_j)}{\sum I(y_i = 1)}.$$

The ROC curve is plotted from the pairs $(FPR(j), TPR(j))$, and the AUC is calculated as the area under this curve, i.e.

$$AUC = \int_{j=1}^{J} TPR(FPR(j)) dj.$$

Value

the AUC value, and a plot of the ROC curve if `plot=TRUE` `sigmoid` *Sigmoid function*

Description

Sigmoid function, otherwise known as a logistic function

Usage

```
sigmoid(z)
```

Arguments

z	input

Details

The sigmoid (or logistic) function,

$$\sigma(z) = 1/(1 + e^{-z})$$

is used in logistic regression to model probabilities, commonly the probability of predicting the positive class, i.e. $p(y = 1)$.

Value

output value of the function

Examples

```
theta = c(1, 0.1)
X = matrix(rnorm(4), 2, 2)
sigmoid(X %*% theta)
```

Index