Package ‘qualmap’

September 12, 2018

Type Package

Title Opinionated Approach for Digitizing Semi-Structured Qualitative GIS Data

Version 0.1.1

Description Provides a set of functions for taking qualitative GIS data, hand drawn on a map, and converting it to a simple features object. These tools are focused on data that are drawn on a map that contains some type of polygon features. For each area identified on the map, the id numbers of these polygons can be entered as vectors and transformed using qualmap.

Depends R (>= 3.3)

License GPL-3

URL https://github.com/slu-openGIS/qualmap

BugReports https://github.com/slu-openGIS/qualmap/issues

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

Imports dplyr, glue, leaflet, purrr, rlang, sf

Suggests covr, ggplot2, testthat, tigris, tidycensus, knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Christopher Prener [aut, cre] (<https://orcid.org/0000-0002-4310-9888>)

Maintainer Christopher Prener <chris.prener@slu.edu>

Repository CRAN

Date/Publication 2018-09-12 15:10:14 UTC

R topics documented:

qm_combine .................................................. 2
qm_create .................................................... 3
qm_define ..................................................... 4
**qm_combine**

**Description**

A wrapper around `dplyr::bind_rows` for combining cluster objects created with `qm_create` into a single tibble. Input data for `qm_combine` are validated using `qm_is_cluster` as part of the cluster object creation process.

**Usage**

```r
qm_combine(...)```

**Arguments**

`...`  
A list of cluster objects to be combined.

**Value**

A single tibble with all observations from the listed cluster objects. This tibble is stored with a custom class of `qm_cluster` to facilitate data validation.

**See Also**

`qm_create`, `qm_is_cluster`

**Examples**

```r
# load and format reference data
stl <- stLouis
stl <- dplyr::mutate(stl, TRACTCE = as.numeric(TRACTCE))

# create clusters
cluster1 <- qm_define(118600, 119101, 119300)
cluster2 <- qm_define(119300, 121200, 121100)

# create cluster objects
cluster_obj1 <- qm_create(ref = stl, key = TRACTCE, value = cluster1,
                          rid = 1, cid = 1, category = "positive")
cluster_obj2 <- qm_create(ref = stl, key = TRACTCE, value = cluster2,
                          rid = 1, cid = 2, category = "positive")```

---

**qm_is_cluster**

**Description**

A function to determine if an object conforms to the `qm_cluster` class.

**Usage**

```r
qm_is_cluster(x)```

**Arguments**

`x`  
An object to test.

**Value**

A logical value indicating if `x` is a `qm_cluster`.

**Examples**

```r
# create a qm_cluster object
cluster_obj <- qm_create...

# check if it is a qm_cluster
qm_is_cluster(cluster_obj)
```
# combine cluster objects
clusters <- qm_combine(cluster_obj1, cluster_obj2)

q_create

Create cluster object

Description

Each vector of input values is converted to a tibble organized in a "tidy" fashion.

Usage

qm_create(ref, key, value, rid, cid, category, ...)

Arguments

ref
An sf object that serves as a master list of features

key
Name of geographic id variable in the ref object to match input values to

value
A vector of input values created with qm_define

rid
Respondent identification number; a user defined integer value that uniquely identifies respondents in the project

cid
Cluster identification number; a user defined integer value that uniquely identifies clusters

category
Category type; a user defined value that describes what the cluster represents

... An unquoted list of variables from the sf object to include in the output

Details

A cluster object contains a row for each feature in the reference data set. The key variable values are included in a variable named identically to the key. Additionally, a variable called COUNT is created. This will always be equal to 1 for each observation. It is intended to be used for summarization later in the data analysis process. Finally, three pieces of metadata are also included as arguments to provide data for subsetting later: a respondent identification number (rid), a cluster identification number (cid), and a category for the cluster type (category). These arguments are converted into values for the output variables RID, CID, and CAT respectively. Input data for qm_create are validated using qm_validate as part of the cluster object creation process.

Value

A tibble with the cluster values merged with elements of the reference data. This tibble is stored with a custom class of qm_cluster to facilitate data validation.

See Also

qm_define, qm_validate
Examples

# load and format reference data
stl <- stLouis
stl <- dplyr::mutate(stl, TRACTCE = as.numeric(TRACTCE))

# create cluster
cluster <- qm_define(118600, 119101, 119300)

# create simple cluster object
cluster_obj1 <- qm_create(ref = stl, key = TRACTCE, value = cluster,
                          rid = 1, cid = 1, category = "positive")

# create cluster object with additional variables added from reference data
cluster_obj2 <- qm_create(ref = stl, key = TRACTCE, value = cluster,
                          rid = 1, cid = 1, category = "positive", NAME, NAMELSAD)

qm_define

Define input values

Description

A wrapper around base::c that is used for constructing vectors of individual feature values. Each
output should correspond to a single cluster on the respondent’s map.

Usage

qm_define(...) 

Arguments

... A comma separated list of individual features

Value

A vector list each feature.

Examples

cluster <- qm_define(118600, 119101, 119300)
validate_cluster

Description

This function tests to see whether an object is of class qm_cluster. It is used as part of the qm_combine and qm_summarize functions, and is exported so that it can be used interactively as well.

Usage

qm_is_cluster(obj)

Arguments

obj Object to test

Value

A logical scalar that is TRUE is the given object is of class qm_cluster; it will return FALSE otherwise.

See Also

qm_combine, qm_summarize

Examples

# load and format reference data
stl <- stLouis
stl <- dplyr::mutate(stl, TRACTCE = as.numeric(TRACTCE))

# create cluster
cluster <- qm_define(118600, 119101, 119300)

# create simple cluster object
cluster_obj <- qm_create(ref = stl, key = TRACTCE, value = cluster,
                        rid = 1, cid = 1, category = "positive")

# test cluster object
qm_is_cluster(cluster_obj)
Description

This function renders the input vector as a polygon shapefile using the leaflet package.

Usage

```r
qm_preview(ref, key, value)
```

Arguments

- **ref**: An sf object that serves as a master list of features
- **key**: Name of geographic id variable in the ref object to match input values to
- **value**: A vector of input values created with qm_define

Value

An interactive leaflet map with the features from the defined vector specified in value highlighted in red.

See Also

- qm_define

Examples

```r
## Not run:
# load and format reference data
stl <- stLouis
stl <- dplyr::mutate(stl, TRACTCE = as.numeric(TRACTCE))

# create cluster
cluster <- qm_define(118600, 119101, 119300)

# preview cluster
qm_preview(ref = stl, key = TRACTCE, value = cluster)

## End(Not run)
```
**qm_summarize**

**Summarize Clusters**

**Description**

This function creates a column that contains a single observation for each unique value in the key variable. For each feature, a count corresponding to the number of times that feature is identified in a cluster for the given category is also provided.

**Usage**

```r
qm_summarize(ref, key, clusters, category, geometry = TRUE, use.na = FALSE)
```

**Arguments**

- `ref`: An sf object that serves as a master list of features.
- `key`: Name of geographic id variable in the ref object to match input values to.
- `clusters`: A tibble created by `qm_combine` with two or more clusters worth of data.
- `category`: Value of the cat variable to be analyzed.
- `geometry`: A logical scalar that returns the full geometry and attributes of ref when TRUE (default). If FALSE, only the key and count of features is returned after validation.
- `use.na`: A logical scalar that returns NA values in the count variable if a feature is not included in any clusters when TRUE. If FALSE (default), a 0 value is returned in the count variable for each feature that is not included in any clusters. This parameter only impacts output if the geometry argument is TRUE.

**Value**

A tibble or a sf object (if geometry = TRUE) that contains a count of the number of clusters a given feature is included in. The tibble option (when geometry = FALSE) will only return valid features. The sf option (default; when geometry = TRUE) will return all features with either zeros (when use.na = FALSE) or NA values (when use.na = TRUE) for features not included in any clusters.

**See Also**

`qm_combine`

**Examples**

```r
# load and format reference data
stl <- stLouis
stl <- dplyr::mutate(stl, TRACTCE = as.numeric(TRACTCE))

# create clusters
cluster1 <- qm_define(118600, 119101, 119300)
cluster2 <- qm_define(119300, 121200, 121100)
```
# create cluster objects
cluster_obj1 <- qm_create(ref = stl, key = TRACTCE, value = cluster1, 
    rid = 1, cid = 1, category = "positive")
cluster_obj2 <- qm_create(ref = stl, key = TRACTCE, value = cluster2, 
    rid = 1, cid = 2, category = "positive")

# combine cluster objects
clusters <- qm_combine(cluster_obj1, cluster_obj2)

# summarize cluster objects
positive1 <- qm_summarize(ref = stl, key = TRACTCE, clusters = clusters, category = "positive")
class(positive1)
mean(positive1$positive)

# summarize cluster objects with NA's instead of 0's
positive2 <- qm_summarize(ref = stl, key = TRACTCE, clusters = clusters, category = "positive", 
    use.na = TRUE)
class(positive2)
mean(positive2$positive, na.rm = TRUE)

# return tibble of valid features only
positive3 <- qm_summarize(ref = stl, key = TRACTCE, clusters = clusters, category = "positive", 
    geometry = FALSE)
class(positive3)
mean(positive3$positive)

---

**qm_validate**

**Validate input vector**

**Description**

This function ensures that the input vector values match valid values in a source shapefile.

**Usage**

qm_validate(ref, key, value)

**Arguments**

- **ref**: An sf object that serves as a master list of features
- **key**: Name of geographic id variable in the ref object to match input values to
- **value**: A vector of input values created with qm_define

**Value**

A logical scalar that is TRUE is all input values match values in the key variable.
qualmap

See Also
qm_define

Examples

# load and format reference data
stl <- stLouis
stl <- dplyr::mutate(stl, TRACTCE = as.numeric(TRACTCE))

# create clusters
clusterValid <- qm_define(118600, 119101, 119300)
clusterError <- qm_define(118600, 119101, 800000)

# validate clusters
qm_validate(ref = stl, key = TRACTCE, value = clusterValid)
qm_validate(ref = stl, key = TRACTCE, value = clusterError)

qualmap: Opinionated Approach for Digitizing Semi-structured Qualitative GIS Data

Description

Provides a set of functions for taking qualitative GIS data, hand drawn on a map, and converting it to a simple features object. These tools are focused on data that are drawn on a map that contains some type of polygon features. For each area identified on the map, the id numbers of these polygons can be entered as vectors and transformed using qualmap.

Details

Qualitative GIS outputs are notoriously difficult to work with because individuals’ conceptions of space can vary greatly from each other and from the realities of physical geography themselves. qualmap builds on a semi-structured approach to qualitative GIS data collection. Respondents use a specially designed basemap that allows them free reign to identify geographic features of interest and makes it easy to convert their annotations into digital map features. This is facilitated by including on the basemap a series of polygons, such as neighborhood boundaries or census geography, along with an identification number that can be used by qualmap. A circle drawn on the map can therefore be easily associated with the features that it touches or contains.

qualmap provides a suite of functions for entering, validating, and creating sf objects based on these hand drawn clusters and their associated identification numbers. Once the clusters have been created, they can be summarized and analyzed either within R or using another tool.

This approach provides an alternative to either unstructured qualitative GIS data, which are difficult to work with empirically, and to digitizing respondents’ annotations as rasters, which require a sophisticated workflow. This semi-structured approach makes integrating qualitative GIS with
existing census and administrative data simple and straightforward, which in turn allows these data to be used as measures in spatial statistical models.

There are six key verbs introduced in qualmap:

- **define**: create a vector of feature id numbers that constitute a single "cluster"
- **validate**: check feature id numbers against a reference data set to ensure that the values are valid
- **preview**: plot cluster on an interactive map to ensure the feature ids have been entered correctly (the preview should match the map used as a data collection instrument)
- **create**: create a single cluster object once the data have been validated and visually inspected
- **combine**: combine multiple cluster objects together into a single tibble data object
- **summarize**: summarize the combined data object based on a single qualitative construct to prepare for mapping

**Tidy Evaluation**

qualmap makes use of tidy evaluation, meaning that key and category references may be either quoted or unquoted (i.e. bare).

**Author(s)**

**Author and Maintainer**: Christopher Prener, Ph.D. chris.prener@slu.edu

**See Also**

Useful links:

- Package Website and Documentation
- Source Code on GitHub
- Bug Reports and Feature Requests
- Chris Prener’s Open GIS Project at Saint Louis University

---

**stLouis**

*St. Louis Census Tracts, 2016*

**Description**

A simple features data set containing the geometry and associated attributes for the 2016 City of St. Louis census tracts.

**Usage**

```r
data(stLouis)
```
Format

A data frame with 106 rows and 7 variables:

- **STATEFP** state FIPS code
- **COUNTYFP** county FIPS code
- **TRACTCE** tract FIPS code
- **GEOID** full GEOID string
- **NAME** tract FIPS code, decimal
- **NAMELSAD** tract name
- **geometry** simple features geometry

Note

These data have been modified from the full version available from the Census Bureau - some variables related to geometry and geography type have been removed.

Source

- U.S. Census Bureau

```R
#' @examples str(stLouis) head(stLouis)
```
Index

*Topic datasets
  stLouis, 10

qm_combine, 2
qm_create, 3
qm_define, 4
qm_is_cluster, 5
qm_preview, 6
qm_summarize, 7
qm_validate, 8
qualmap, 9
qualmap-package (qualmap), 9

stLouis, 10