Package ‘africabirdmap’

April 8, 2019

Type Package

Title Functions to clean, analyze and plot Africa Bird Map data.

Version 0.1.0

Author Les Underhill, Michael Brooks, Marc Burman, Colin Jackson

Maintainer David Clarance <dclarance@gmail.com>

Description Enables citizen science by providing easy to use tools so researches can focus on questions rather than struggling with the data.

License To be populated

Encoding UTF-8

LazyData true

Depends dplyr, readr, tidyr, ggplot2, scales, RCurl, digest, RMySQL,
RJSONIO, digest

RoxygenNote 6.1.1

R topics documented:

`days_since_month` *Calculate days from a given calendar month*

Description

An a-historical method to calculate pentades by calculating distance from a given calendar month.

Usage

```
days_since_month(date, start_month = 7)
```

Arguments

`date` A ‘date‘ object. This can be a single date or a vector of dates.

`start_month` An integer between 1 and 12 (inclusive) that indicates the month you want to measure distance in days from. Default is 7.

Details

This functions assumes that all years are 365 days. Necessary corrections for leap years are made.

Value

This function returns an integer between 1 and 365 indicating the number of days elapsed since the first of the month indicated in ‘start_month‘

Examples

```
# single date conversion example
days_since_month(date = as.Date('2018-09-14'), start_month = 7)

# multiple dates conversion examples
sequence_of_dates <- c(as.Date("2019-01-01"), as.Date("2019-03-19"))
days_since_month(date = sequence_of_dates, start_month = 6)
```

`extract_data` *Export data from the Africa Bird Map*

Description

Pull data from the African Bird Map fit for reporting rate and other analyses. This function is temporary and will be made redundant. It is unsafe and is only for demonstrative purposes.

Usage

```
extract_data(username, user_id, password, species_id)
```

Arguments

`username` The citizen scientist’s bird map email

`user_id` The citizen scientist’s bird map user number (usually 5 digits)

`password` The citizen scientist’s password.

`species_id` The species_id for which data is extracted. A complete list of species name and ids are available on the Kenya Bird Map website.

Details

The function requires a username and password in plain text. This is not how the final function will look. The final function will call the RStudio API for a safe transaction. This is just for demonstrative purposes.

Value

A dataframe with every list submitted to Africa Bird Map (includes all countries).

Examples

```
## Not run:

# Extract data for the African Black-shouldered kite.

extract_data(username = "dclarance@gmail.com" ,user_id = 40664, password = "abcd", species_id = 103s)


## End(Not run)
```

filter_pentads *Select pentads based on features*

Description

Select pentads lying within a defined geographic region. Used as a helper function in the reporting rate functions.

Usage

```
filter_pentads(selected_area, selection_type)
```

Arguments

`selected_area` Either a pentad (eg: 0105_3930), country (eg: Kenya), county (eg: Kitui) or province (eg: rift valley). Lists of the same can be applied as well. For instance, for multiple pentads you could use, c('0105_3930', '0110c3620').

`selection_type` Can take either of the four values: 'Pentad', 'Country', 'County' or 'Province'.

Details

‘filter_pentads‘ is used to generate a character list of pentads associated with a given feature. Currently all features are political (counties, provices, countries). Over time we might consider adding habitat and climate features.

The pentad assignments are correct for all pentads that lie strictly within the national boundaries of Kenya. However since the algorithm uses the midpoint of a pentad to discover its assignment, there are cases where the midpoint lies outside Kenya and so the pentads do not have a classification. In the data these show up as ‘unclassified‘.

Ideally we would want to use an assignment based on area. This will likely show up in the next release.

Value

A dataframe where each row is a pentade with the ‘fit‘ values, the standard errors (‘se‘), the ‘reporting_rates‘ and its associated inputs. Ignore the various date columns as those are only meant to be used in graphs.

Examples

```
# get list of pentads in Rift Valley
filter_pentads(selected_area = "Rift Valley", selection_type = "County")
```

hello *Hello, World!*

Description

Prints ’Hello, world!’.

Usage

```
hello()
```

Examples

```
hello()
```

reporting_rate *Calculate the reporting rate of a species*

Description

Obtain a reporting rate of a species over a specified time period and spatial configuration.

Usage

```
reporting_rate(df, species_id, start_date = "1970-01-01",
  end_date = Sys.Date(), selected_area = "Kenya",
  selection_type = "Country")
```

Arguments

`df`	A dataframe obtained from the species API call. Use ‘extract_data‘ if required.
`species_id`	An integer id. The KBM website has a list of all species with their ids.
`start_date`	A character or date object denoting the starting point from which reporting rate is to be measured.
`selected_area`	Either a pentad (eg: 0105_3930), country (eg: Kenya), county (eg: Kitui) or province (eg: rift valley). Lists of the same can be applied as well. For instance, for multiple pentads you could use, c(’0105_3930’, ’0110c3620’).
`selection_type`	Can take either of the four values: ’Pentad’, ’Country’, ’County’ or ’Province’.

Details

- This functions assumes that all years are 365 days. Necessary corrections for leap years are made.
- For a full list of all possible values for selection_area, run ‘View(africabirdmap::pentads_geographical_features)‘ in the console.

Value

A numeric object between 0 and 1 (inclusive) denoting the reporting rate of a species. The reporting rate is equal to the number of times a species has been recorded divided by the number of full protocol cards in the specified time and area.

Examples

```
## Not run:

# the reporting rate of African Paradise Flycatchers in the coastal region in Kenya.
reporting_rate(df,
species_id = 682,
start_date = '1970-01-01',
end_date = Sys.Date(),
selected_area = "Coast" ,
selection_type = "Province")


## End(Not run)
```

underhill_multiple_curves

Plot the underhill smoother and associated confidence intervals for multiple species.

Description

Fit a line through the reporting rates and its associated 2 sigma confidence intervals for multiple species.

Usage

```
underhill_multiple_curves(multiple_underhill_smoothers)
```

Arguments

`multiple_underhill_smoothers`
A list of the dataframes returned by the ‘underhill_smoother function‘.

Details

To utilize this function first you need to create a list of dataframe. Do this by calling the ‘list()‘ function and passing the dataframes as arguments

Value

A ggplot plot

Examples

```
## Not run:

list_of_interesting_species = list(underhill_smoother1 , underhill_smoother2, underhill_smoother3)
underhill_curves(list_of_interesting_species)


## End(Not run)
```

---

underhill_single_curves

Plot the underhill smoother and associated confidence intervals for a species.

Description

Fit a line through the reporting rates and its associated 1 and 2 sigma confidence intervals. 1 and 2 sigma roughly correspond to 67 percent and 95 percent confidence intervals. The line is over Pentades defined in the ‘underhill_smoother()‘ function.

Usage

```
underhill_single_curves(underhill_smoother)
```

Arguments

`underhill_smoother`
The dataframe returned by the ‘underhill_smoother function‘.

Details

The dark, thinner interval is the 1 sigma interval and corresponds to 67

This function only takes in one dataframe and is intended for deep study of one species. To make multi species comparisons use ‘underhill_multiple_curves‘

Value

A ggplot plot

Examples

```
## Not run:

underhill_curves(underhill_smoother)

## End(Not run)
```

`underhill_smoother` *Fit the underhill smoother to a reporting rate distribution*

Description

Fit the underhill smoother (put reference here) to reporting rate distribution over pentades.The underhill smoother is a locally weighted binomial general linear model using a logit link, where the weights are generated using a exponential distribution.

Usage

```
underhill_smoother(raw_data, species_id, start_month = 7,
  selected_area = "Kenya", selection_type = "Country", pentade_window,
  first_pentade = 1, last_pentade = 73)
```

Arguments

`raw_data`	Data for a species extracted using ‘extract_data()‘. The StartDate column should be of type ‘date‘.
`species_id`	The species_id for which data is extracted. A complete list of species name and ids are available on the Kenya Bird Map website.
`start_month`	The month to start the pentades from. Default is 7.
`selected_area`	Either a pentad (eg: 0105_3930), country (eg: Kenya), county (eg: Kitui) or province (eg: rift valley). Lists of the same can be applied as well. For instance, for multiple pentads you could use, c(’0105_3930’, ’0110c3620’).
`selection_type`	Can take either of the four values: ’Pentad’, ’Country’, ’County’ or ’Province’.
`pentade_window`	The number of pentades on either side of the target day to give the weights to.
`first_pentade`	The starting value of the pentades. Default is 1.
`last_pentade`	The last value of the pentades. Default is 73.

Details

- Pentades are blocks of 5 consecutive days. The implementation measures pentades from a defined month ‘start_month‘. This implementation needs review and should not be considered final.
- For a full list of all possible values for selection_area, run ‘View(africabirdmap::pentads_geographical_features)‘ in the console.

Value

A dataframe where each row is a pentade with the ‘fit‘ values, the standard errors (‘se‘), the ‘reporting_rates‘ and its associated inputs. Ignore the various date columns as those are only meant to be used in graphs.

Examples

```
## Not run:

underhill_smoother(raw_data, species_id, start_month = 7, pentade_window, first_pentade = 1, last_pentade = 73

## End(Not run)
```

Index