Package: focustools (via r-universe)

September 13, 2024

Title Forecasting COVID-19 in the United States (FOCUS) tools

Version 0.2.0

Description Miscellaneous functions for retrieving data, creating and evaluating time series forecasting models for COVID-19 cases and deaths in the United States. Built for participation in the COVID-19 Forecast Hub.

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 2.10)

LazyData true

RoxygenNote 7.1.2

Imports magrittr, dplyr, readr, tidyr, yaml, glue, fable, fabletools, tsibble, MMWRweek, httr, lubridate, purrr, reticulate, tibble, stringr, ggplot2, shiny, scales, feasts, urca

Suggests here, lifecycle, shinyWidgets, DT, knitr, rmarkdown

VignetteBuilder knitr

Repository https://stephenturner.r-universe.dev

RemoteUrl https://github.com/signaturescience/focustools

RemoteRef HEAD

RemoteSha eb0fb54bc24abc70fe81390878a2837ad1a468d0

Contents

`extract_arima_params` *Extract ARIMA parameters*

Description

Extracts ARIMA model parameters, including p, d, q, P, D, Q, and results from tidy and glance on an ARIMA model object.

Usage

```
extract_arima_params(arimafit)
```

Arguments

`arimafit`	A single-row mable (`mdl_df`) from `fabeltools::model(arima=ARIMA(...))`.

Value

A single-row `tibble` containing ARIMA model parameter and diagnostic information.

`focustools` `focustools` *package*

Description

Tools for forecasting COVID-19 in the United States

focus_explorer *Function to launch FOCUS Explorer shiny app*

Description

The explorer app allows a user to view plots of forecasts, inspect tabular output of submission files, and download subsets of forecast submission data. The app includes an interface to interactively select locations to include in the plots, table, and download. This function wraps `shiny::runApp` and accepts arguments for the data against which the forecasts should be plotted, as well as the directory containing submission files, both of which are temporarily attached to the global environment for use during the app session. Additional arguments passed to `...` will be inherited by runApp.

Usage

```
focus_explorer(.data, submission_dir, ...)
```

Arguments

`.data`	Tibble with historical data for trend leading up to forecast
`submission_dir`	Full path to directory of submission files containing forecast submissions to explore
`...`	Additional arguments to be passed to runApp

Value

This function starts a shiny app. On exit it removes objects (see ".data" and "submission_dir") that are temporarily attached and used by the app session.

format_for_submission *Format forecast for COVID-19 Forecast Hub submission entry*

Description

The submission file for the COVID-19 Forecast Hub must adhere to requirements for file format, column names, target identifiers, and date ranges for horizons. This function takes output from a `focustools` forecasting function (e.g. ts_forecast) and prepares an appropriately formatted object that can be written to a file. Formatting steps include constructing a valid string for horizon and target name (e.g. '3 wk ahead inc case'), computing the 'target_end_date' value based on the epidemiological week for the horizon, filtering distributional cutpoints for certain targets ('inc case' only needs 7 of the quantiles), converting all estimates to integers, and bounding all predicted values at minimum of 0.

Usage

```
format_for_submission(.forecast, target_name)
```

Arguments

`.forecast`	Forecast object
`target_name`	Name of the target for the forecast; must be one of `'inc case'`, `'inc death'`, or `'cum death'`

Value

A `tibble` with target names and quantiles/point estimates formatted per the COVID-19 Forecast Hub submission guidelines.

References

https://covid19forecasthub.org/

`is_monday` *Check Monday*

Description

This is a helper function to see if today is Monday.

Usage

```
is_monday()
```

Value

Logical indicating whether or not today is Monday

`make_tsibble` *Make* `tsibble`

Description

This function converts an input `tibble` with columns for epiyear and epiweek into a tsibble object. The `tsibble` has columns specifying indices for the time series as well as a date for the Monday of the epiyear/epiweek combination at each row. Users can optionally ignore the current week when generating the `tsibble` via the "chop" argument.

Usage

```
make_tsibble(df, chop = TRUE)
```

Arguments

df	A tibble containing columns epiyear and epiweek.
chop	Logical indicating whether or not to remove the most current week (default TRUE).

Value

A tsibble containing additional columns monday indicating the date for the Monday of that epiweek, and yweek (a yearweek vctr class object) that indexes the tsibble in 1 week increments.

---

plot_forecast *Visualize forecast output*

Description

This function serves as a plotting mechanism for prepped forecast submission data (see format_for_submission). Using truth data supplied, the plots show the historical trajectory of each outcome along with the point estimates for forecasts. Optionally, the user can include 50% prediction interval as well. Plots include trajectories of incident cases, incident deaths, and cumulative deaths faceted by location.

Usage

```
plot_forecast(
  .data,
  submission,
  target = c("Incident Cases", "Incident Deaths", "Cumulative Deaths"),
  location = "US",
  pi = TRUE
)
```

Arguments

.data	Historical truth data for all locations and outcomes in submission targets
submission	Formatted submission
target	Vector specifying target(s) to plot; default is c('Incident Cases','Incident Deaths','Cumulative Deaths')
location	Vector specifying locations to filter to; 'US' by default.
pi	Logical as to whether or not the plot should include 50% prediction interval; default is TRUE

Value

A ggplot2 plot object with line plots for outcome trajectories faceted by location

`retrieve` *Retrieve data*

Description

The package includes functions to retrieve observed data from two canonical sources: the New York Times and the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University. Both organizations administer data aggregation efforts that post daily COVID-19 case and death to GitHub. The data retrieval functions allow the user to specify "source" and "granularity" of data. Internally each function builds a path to the appropriate `.csv` file on GitHub, then reads the data into memory. The returned object is data aggregated weekly (using epiweek and epiyear designations) for available locations at the granularity specified (national, state, or county level).

Usage

```
get_cases(source = "jhu", granularity = "national")

get_deaths(source = "jhu", granularity = "national")
```

Arguments

`source`	Data source to query; must be one of `'jhu'` or `'nyt'`; default is `'jhu'`
`granularity`	Data aggregation level; must be one of `'national'`, `'state'`, or `'county'`; if data source is `'nyt'` then only `'national'` can be used currently; default is `'national'`

Value

A `tibble` with (at minimum) the following columns:

- **epiyear**: Epidemiological year (see epiyear for more details)
- **epiweek**: Epidemiological week (see epiweek for more details)
- **icases/ideaths**: Incident counts (cases or deaths)
- **ccases/cdeaths**: Cumulative counts (cases or deaths)

If `source = 'jhu'` and `granularity = 'state'` then the **location** column will include the full name of the state. If `source = 'jhu'` and `granularity = 'county'` then the **location** column will include fips (county code).

References

https://github.com/CSSEGISandData/COVID-19

https://github.com/nytimes/covid-19-data

spread_value *Reshape data for submission summary*

Description

This unexported helper function is used in submission_summary. It spreads forecast targets to a wide format and forces "US" locations to be at the top of the resulting `tibble`.

Usage

```
spread_value(.data, ...)
```

Arguments

`.data`	Tibble with submission data
`...`	Additional arguments passed to spread

Value

A `tibble` with wide summary data.

submission_summary *Summarize submission*

Description

This function summarizes and reformats submission data as 4-week ahead counts and percent change. The summaries are stratified by location and target (incident cases, incident deaths, and cumulative deaths).

Usage

```
submission_summary(.data, submission, location = NULL)
```

Arguments

`.data`	Tibble with historical data for trend leading up to forecast
`submission`	Formatted submission
`location`	Vector specifying locations to filter to; `NULL` by default meaning all locations will be used

Value

Named list with summarized count and percent change data. Each summary is stratified by target and returned in the list as a `tibble` with columns for "location", "Previous" (value week prior to forecast), "1w ahead", "2w ahead", 3w ahead, and "4w ahead".

`this_monday` *Get Monday*

Description

This function is a helper to get the date for the Monday of the current week.

Usage

```
this_monday()
```

Value

Date for the Monday of the current week. For more details see floor_date.

`ts_cumulative_forecast` *Helper used in* `ts_forecast()` *to get cumulative forecast from incident*

Description

This unexported helper is used internally in ts_forecast to generate cumulative forecasts from incident. The function cumulatively sums incident estimates (quantile and point) at each location. Note that if used outside of ts_forecast one must be sure that the ".data" argument matches object used to generate the incident forecast object ("inc_forecast").

Usage

```
ts_cumulative_forecast(.data, outcome = "cdeaths", inc_forecast)
```

Arguments

`.data`	Data from which the cumulative forecast should get recent counts; *CAUTION* for best results make sure that the data passed to this argument is the same object as used to generate the model/forecast that is specified in "inc_forecast"
`outcome`	Name of the outcome; should be be one of `'cdeaths'` or `'ccases'`
`inc_forecast`	A `tibble` with incident forecast data generated using ts_forecast; should *only* be incident cases corresponding to outcome for which cumulative count is to be aggregated

Value

A `tibble` with forecast results, including the name of the model, year and week, value of the forecast estimate, type of estimate (quantile or point), and bin of the quantile (if applicable) for the estimate.

ts_fit *Wrapper to fit time series models*

Description

[Experimental]

The time series forecasting pipeline depends on time series models fit with the model function. This function provides a wrapper that allows the user to pass in a list of function definitions and return a list of model outputs (`mable` objects) corresponding to each fit. The function also allows the user to pass in a vector of multiple outcome variable names (i.e. "ideaths" and "icases").

NOTE: The functionality in `ts_fit()` is experimental. Users may find more flexibility using the model function to fit models to be used downstream in `ts_forecast()`.

Usage

```
ts_fit(.data, outcomes, .fun, single = TRUE)
```

Arguments

`.data`	Data to use for modeling
`outcomes`	Character vector specifying names of the column to use as the outcome
`.fun`	List of modeling functions to use
`single`	Boolean indicating whether or not a "shortcut" should be used to return a single `tibble`; only works if there is one outcome ("outcomes") and one model function (".fun"); default is `TRUE`

Value

A single `mable` (model table) if (`single = TRUE`) or a named list of `mables` (if `single = FALSE`). For more details on data structure see mable.

ts_forecast *Generate time series forecasts including quantile estimates*

Description

This function can convert models fit with model or the ts_fit wrapper to forecasted values. The user specifies the horizon out to which forecasts should be generated, as well as any optional covariate data needed for forecasting (e.g. when using a model of incident deaths based on lagged incident cases, the forecast function needs incident cases moving into the forecast horizons; see "new_data" argument). The forecasts generated will include point estimates as well as 23 quantiles: 0.01, 0.025, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95, 0.975, 0.99. By default these quantiles are calculated using the hilo.

Usage

```
ts_forecast(mable, outcome, horizon = 4, new_data = NULL, ...)
```

Arguments

mable	A mable (model table); for more information see mable
outcome	Name of the outcome; must be one of 'icases','ideaths', 'cdeaths', 'ccases'
horizon	Optional horizon periods through which the forecasts should be generated; default is 4
new_data	Optional covariate data for forecasts using models that were fit using other variables; should be generated using new_data; default is NULL
...	Additional parameters passed to the ts_cumulative_forecast helper; only used if the forecast is cumulative

Value

A tibble with forecast results, including the name of the model, year and week, value of the forecast estimate, type of estimate (quantile or point), and bin of the quantile (if applicable) for the estimate.

---

ts_futurecases *Helper to generate the estimate of incident cases from an icases forecast object*

Description

This function takes a time series forecast and extracts the point estimate for incident cases out to a specified horizon. This is necessary to generate the "new_data" to be passed into the ts_forecast incident death models that are based on lagged cases.

Usage

```
ts_futurecases(.data, .forecast, horizon = 4)
```

Arguments

.data	Data from which the new_data should be generated; *CAUTION* for best results make sure that the data passed to this argument is the same object as used to generate the model/forecast that is specified in ".forecast"
.forecast	A tibble with forecast data generated using ts_forecast; should *only* be a forecast of incident cases
horizon	Horizon periods through which the new_data should be generated; default is 4

Value

A tsibble with horizon periods and respective forecasted incident cases.

validate_forecast *Check for valid quantile csv file input*

Description

The submission file for the COVID-19 Forecast Hub must adhere to requirements for file format, column names, target identifiers, and date ranges for horizons. The organizers include Python scripts to validate weekly submission data. This function provides an R wrapper for one of the validation methods from the `zoltpy` Python module. In order to wrap the Python functionality, the function calls `reticulate` internally to attach the Python environment with `zoltpy` installed. Any changes made upstream (in `zoltpy` release on PyPi repository) will be propagated to this function given a fresh module installation (see "install" argument).

Usage

```
validate_forecast(filename, verbose = TRUE, install = FALSE, envname = NULL)
```

Arguments

`filename`	Full path to the forecast file to be checked
`verbose`	Logical indicating whether or not the output from this function should include validation message; default `TRUE`
`install`	Logical as to whether or not the python dependencies should be installed; if `TRUE` the module will be installed to the virtual environment specified in "envname"; default is `FALSE`
`envname`	Character vector specifying the name of the virtualenv to which the python dependencies should be installed if `install = TRUE`; default is `NULL` which will install the module to a virtualenv named `r-reticulate`

Value

If `verbose = FALSE`, the returned value will be a boolean with `TRUE` for valid submission file and `FALSE` for invalid file. If `verbose = FALSE`, the function will return a named list with two elements: "valid" (boolean with the `TRUE`/`FALSE` validation code) and "message" (the output from the `zoltpy valid_quantile_csv_file()` function).

References

https://pypi.org/project/zoltpy/

https://covid19forecasthub.org/

Index