fludashboard Documentation

Release 0.4.0

FluVigilanciaBR

Jun 21, 2018

Contents

Contents:

CHAPTER 1

fludashboard

Flu Dashboard

Flu Dashboard is an initiative to monitor and provide alert levels for Severe Acute Respiratory Infections (SARI) notifications registered at SINAN, the Brazilian Notifiable Diseases Information System (www.saude.gov.br/sinan). Data is provided for each Brazilian State as well as by influenza-like illnessess (ILI) regions.

This is a product of the joint work between researchers at the Scientific Computation Program at Oswaldo Cruz Foundation (Fiocruz, PROCC), School of Applied Mathematics at Fundação Getúlio Vargas (EMAp-FGV), both from Rio de Janeiro, Brazil, and the Inluenza Work Force at the Health Surveillance Secretariat of the Brazilian Ministry of Health (GT-Influenza, SVS, MS).

- Free software: GNU General Public License v3
- Documentation: https://fludashboard.readthedocs.io.

1.1 Features

- Nowcast of weekly incidence;
- Activity thresholds;
- Age distribution of notified cases;
- Seasonal activity level;
- Historical incidence curves.

CHAPTER 2

Deployment

To deploy FluDashboard Prototype use conda package (that will create a conda environment called *fludashboard*):

```
conda config --set always_yes yes --set changeps1 no

conda config --add channels conda-forge

conda update --all

conda env create -f requirements -n fludashboard
```

Change to the new environment created:

```
source activate fludashboard
```

Optionally, the deployment can be done into a docker container. To create a new container with conda environment:

```
docker pull continuumio/anaconda3
```

CHAPTER 3

Running the app

To run the app just type in the terminal (into fludashboard/fludashboard directory):

```
python app.py
```

The application will be available on the port 5000. To set a custom port, use -p argument with the port number:

```
python app.py -p 9000
```

CHAPTER 4

Functionality

FluDashboard presents activity levels and incidence information time series by epidemiological week as well as by epidemiological season. Those two can be accessed by the **"Detalhado (semana)" (i.e, detailed)** and the **"Resumido (ano)" (i.e, summary)** view. Each view is composed by 4 panels:

- Country map
- Incidence chart
- Incidence table
- Age and gender distribution

In each, information can be displayed by State or by Region.

4.1 Detalhado (weekly activity information) view:

- Country map (upper left)

  Each State/Region is colored according to selected week activity level:

  - Low activity (green): incidence below epidemic threshold;
  - Epidemic activity (yellow): incidence above epidemic threshold and below high incidence threshold;
  - High activity (orange): incidence above high and below very high incidence threshold;
  - Very high activity (red): incidence above very high incidence threshold.

- Incidence chart (upper right)

  This panel presents the reported incidence time series (black solid line) for the corresponding season, with an horizontal marker indicating the selected epidemiological week. Incidence is reported per 100 thousand individuals. Incidence estimation, when possible, is shown as a red solid line along with 95% confidence interval as dotted red lines. The activity level probability is also presented as text on the upper left corner. Map color correspond to activity level with highest probability. Along with reported and estimated incidence, the system presents the following activity thresholds, estimated based on historical activity at each State/Region:

  - Pre-epidemic threshold (blue dashed line): activity level which indicates, when crossed, the beginning of sustained transmission for the current season. After crossing this threshold, incidence is expected to present steady increase (subject to fluctuations);
  - High activity threshold (green dashed line): activity level above which incidence is considered high for that location. Calculated based on the estimated 90 percentile of historical activity distribution.
  - Very high activity threshold (red dashed line): activity level above which incidence is considered high for that region. Calculated based on the estimated 97.5 percentile of historical activity distribution.

- Incidence chart background color scheme

  The background color of the incidence time series represent the typical activity level per week. That is, the historical incidence distribution per week. It allows for identification of typical seasonal pattern, making easier to identify the period of higher activity (epidemic period).

  - Weekly low activity (green shade): activity below the 10% percentile in each week;
  - Weekly low to average activity (yellow shade): activity between the 10% and 50% percentiles in each week;
  - Weekly average to high activity (oragne shade): activity between the 50% and 90% percentile in each week;
  - Weekly high activity (red shade): activity above 90% percentile in each week.

  When the incidence in a given week is within the high activity region (red background), even if below the incidence thresholds, it indicates that for that particular week the incidence is unusually high. This information is useful for detecting seasons where the epidemic period starts earlier than usual, for instance. Check activity for season 2016 for example.

- Incidence table (lower left)

  Incidence for the corresponding State/Region at selected epidemiological week, along with 90% confidence interval when based on estimation. Along with the name of the State/Region and incidence, this table also presents selected data current status:

  - Stable: reported data is considered to be sufficiently close to total number of notifications. Reported values are expected to suffer minor updates in the future, if any;
  - Estimated: reported data is based on estimation of the digitization opportunity. That is, based on the number of notifications already entered in the system (incomplete) and typical delay between notification at health unit and digitization in the system. Reported values are expected to change in the future, becoming stable after a few weeks;
  - Incomplete: reported data is not yet stable due to digitization opportunity pattern in the selected State/Region and our system is not able to provide reliable estimates. Data is subject to significant changes in the future, becoming stable after a few weeks.

- Age and gender distribution

  Reported incidence (without estimation) bar chart by gender and age bracket.

  - Females (blue);
  - Males (orange);
  - Total population (green).

  Distributions are subject to future updates as described in the incidence table. Distribution in this panel does not use estimations, being always the currently reported distribution, either stable or incomplete.

4.2 Resumido (seasonal activity) view:

This view uses detailed activity levels to report the seasonal one.

- Country map (upper left)

  Each State/Region is colored according to selected week activity level:

  - Low activity (green): incidence below epidemic threshold during the whole season
  - Epidemic activity (yellow): incidence has crossed the epidemic threshold at least once, but never crossed high incidence threshold;
  - High activity (orange): weekly incidence has been reported above high or very high incidence threshold between 1 to 4 weeks;
  - Very high activity (red): weekly incidence has been reported above high or very high incidence threshold for 5 weeks or more.

- Incidence chart (upper right)

  This panel presents the reported incidence time series (black solid line) for the corresponding season. Incidence is reported per 100 thousand individuals. Incidence estimation, when possible, is shown as a red solid line along with 95% confidence interval as dotted red lines. The activity level probability is also presented as text on the upper left corner. Map color correspond to activity level with highest probability. Along with reported and estimated incidence, the system presents the following activity thresholds, estimated based on historical activity at each State/Region:

  - Pre-epidemic threshold (blue dashed line): activity level which indicates, when crossed, the beginning of sustained transmission for the current season. After crossing this threshold, incidence is expected to present steady increase (subject to fluctuations);
  - High activity threshold (green dashed line): activity level above which incidence is considered high for that location. Calculated based on the estimated 90 percentile of historical activity distribution.
  - Very high activity threshold (red dashed line): activity level above which incidence is considered high for that region. Calculated based on the estimated 97.5 percentile of historical activity distribution.

- Incidence table (lower left)

  Incidence for the corresponding State/Region for selected season up to latest report. Along with the name of the State/Region and incidence, this table also presents selected data current status:

  - Stable: reported data is considered to be sufficiently close to total number of notifications. Reported values are expected to suffer minor updates in the future, if any;
  - Incomplete: reported data is not yet stable due to digitization opportunity pattern in the selected State/Region. Data is subject to significant changes in the future, becoming stable after a few weeks.

- Age and gender distribution

  Reported incidence bar chart by gender and age bracket for the selected season.

  - Females (blue);
  - Males (orange);
  - Total population (green).

  Distributions are subject to future updates as described in the incidence table. Distribution in this panel does not use estimations, being always the currently reported distribution, either stable or incomplete.

CHAPTER 5

Installation

5.1 Stable release

To install fludashboard, run this command in your terminal:

```
$ pip install fludashboard
```

This is the preferred method to install fludashboard, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

5.2 From sources

The sources for fludashboard can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/FluVigilanciaBR/fludashboard
```

Or download the tarball:

```
$ curl  -OL https://github.com/FluVigilanciaBR/fludashboard/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER 6

Usage

To use fludashboard in a project:

```
import fludashboard
```

CHAPTER 7

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

7.1 Types of Contributions

7.1.1 Report Bugs

Report bugs at https://github.com/FluVigilanciaBR/fludashboard/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

7.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

7.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

7.1.4 Write Documentation

fludashboard could always use more documentation, whether as part of the official fludashboard docs, in docstrings, or even on the web in blog posts, articles, and such.

To update sphinx python documentation, at /docs/ directory, do:

```
$ sphinx-apidoc -o . ../fludashboard
```

To update sphinx javascript documentation, first install jsdoc and jsdoc-sphinx:

```
$ npm install jsdoc
$ npm install jsdoc-sphinx
```

Next, enter the following command:

```
$ jsdoc -t <path_template> -d <DOC_ROOT_OF_PRJECT>/jsdoc/ ../fludashboard/static/js/
```

To update both python and javascript documentation code just run:

```
$ python setup.py doc
```

7.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/FluVigilanciaBR/fludashboard/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

7.2 Coding Style

For JavaScript, see: https://developer.mozilla.org/en-US/docs/Mozilla/Developer_guide/Coding_Style

For Python: see: https://www.python.org/dev/peps/pep-0008/

7.3 Get Started!

Ready to contribute? Here's how to set up *fludashboard* for local development.

1. Fork the *fludashboard* repo on GitHub.
2. Clone your fork locally:

   ```
   $ git clone git@github.com:your_name_here/fludashboard.git
   ```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv fludashboard
$ cd fludashboard/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 fludashboard tests
$ python setup.py test or py.test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

7.4 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/FluVigilanciaBR/fludashboard/pull_requests and make sure that the tests pass for all supported Python versions.

7.5 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_fludashboard
```

CHAPTER 8

Front-end contents:

8.1 Class: `Dashboard`

Local Navigation

8.1.1 Children

8.1.2 Description

8.1.3 Function: `init`

starts up the charts

`init`()

8.1.4 Function: `load_graphs`

load data and build charts

`load_graphs`()

8.1.5 Function: changeWeek

When a week mark is changed, this function should update the charts and SRAG data table.

changeWeek()

8.1.6 Function: makeGraphs

Trigger the functions to create the charts and the data table.

makeGraphs(*error*)

Arguments

- **error** (*object*) – data about any error.

Member: group_by:

Member: delimitation:

Member: lastWeek:

Member: lastWeekYears:

Member: sragData:

Member: sragTable:

Member: sragMap:

Member: sragIncidenceChart:

Member: sragAgeChart:

8.2 Class: SRAGIncidenceChart

Local Navigation

8.2.1 Children

8.2.2 Description

8.2.3 Function: displayInfo

Shows activity information about the criteria established on the chart.

displayInfo(*year*, *week*, *stateName*)

Arguments

- **year** (*number*) – SRAG incidence year (e.g. 2013).
- **week** (*number*) – SRAG incidence week (e.g. 2).
- **stateName** (*string*) – Federal state name (e.g. "Acre").

8.2.4 Function: plot

Plots SRAG incidence chart

plot (*year*, *week*, *stateName-*)

Arguments

- **year** (*number*) – SRAG incidence year (e.g. 2013).
- **week** (*number*) – SRAG incidence week (e.g. 2).
- **stateName-** (*string*) – Federal state name (e.g. "Acre").

Return object

- Chart object.

Member: bindTo:

Member: lastWeekYears:

8.3 Class: SRAGAgeChart

Local Navigation

8.3.1 Children

8.3.2 Description

8.3.3 Function: plot

Plots SRAG incidence chart by age

plot (*year*, *week*, *stateName-*)

Arguments

- **year** (*number*) – SRAG incidence year.
- **week** (*number*) – SRAG incidence week.
- **stateName-** (*string*) – Federal state name (e.g. "Acre").

Member: bindTo:

8.4 Class: SRAGMap

Local Navigation

8.4.1 Children

8.4.2 Description

8.4.3 Function: makeMap

Builds a Brazilian map

makeMap(*geoJsonBr*, *sragData*)

Arguments

- **geoJsonBr** (*dict*) – geoJson data about Brazilian territory
- **sragData** (*dict*) – SRAG data

8.4.4 Function: getAlertLevelForWholeYear

Gets the alert level color using the follow criteria:

- Red (level 4) if the incidence was above the high threshold for at least 5 weeks;
- Orange (level 3) if above the high threshold from 1 to 4 weeks;
- Yellow (level 2) if crossed the epidemic threshold but not the high one;
- Green (level 1) if it did not cross the epidemic threshold.

getAlertLevelForWholeYear(*d*)

Arguments

- **d** (*dict*) – Total number of alert occurrence

Return number

- Alert level (1-4)

8.4.5 Function: changeColorMap

Changes the color of the map using the alerts criteria

changeColorMap(*df*)

Arguments

- **df** (*dict*) – Data frame object

Member: `fluColors`:

Member: `map`:

Member: `osm`:

Member: `geojsonLayer`:

Member: `regionIds`:

Member: `regionNames`:

Member: `legend`:

Member: `geojsonLayer`:

Member: `geojsonLayer`:

8.5 Class: `SRAGTable`

Local Navigation

8.5.1 Children

8.5.2 Description

8.5.3 Function: `getDataTableContent`

Returns HTML table that will be created.

`getDataTableContent`()

Return string Returns HTML table that will be created.

8.5.4 Function: `makeTable`

Builds the SRAG incidence table.

`makeTable`(*year*, *week*, *stateName*)

Arguments

- **year** (*number*) – SRAG incidence year (e.g. 2013).
- **week** (*number*) – SRAG incidence week (e.g. 2).

- **stateName** (*string*) – Federal state name (e.g. “Acre”).

Member: dataTable:

Member: dataTable:

CHAPTER 9

Indices and tables

- genindex
- modindex
- search

Index