Time series toolskit

Contents

[1. Introduction 1](#_Toc49160211)

[2. Technical description 1](#_Toc49160212)

[2.1. Functional Isolation Forest 1](#_Toc49160213)

[2.2. ADTK 2](#_Toc49160214)

[2.3. Technologies used 2](#_Toc49160215)

[3. Installation 2](#_Toc49160216)

[4. Use of component 3](#_Toc49160217)

[5. Next steps 6](#_Toc49160218)

[References 6](#_Toc49160219)

# Introduction

We present in this document a component currently developed by Thales for IoTwins project. The component objectives consist in proposing some tools to study time series in batch way:

* Detect abnormal time series in a corpus of time series, using Functional Isolation forest (Staerman, Mozharovskyi, Clémençon, & d'Alché-Buc, 2019)
* Detect outliers and change points in a time series with ADTK Python module (ADTK: https://github.com/arundo/adtk, 2020)
* Forecast time series with prophet (Taylor​ & Letham​, 2017)

In next Table, we provide the state of the implementation of the different objectives in the current version of the component that is provided for the Deliverable 3.2.

|  |  |
| --- | --- |
| Objective | Status of implementation |
| Detect abnormal time series in a corpus of time series | 90% |
| Detect outliers and change points in a time series with ADTK | 70% |
| Forecast time series with prophet | 0% |

In this document, we first give some technical description about the approaches used in the current version of the component. Then we describe how deploy the component. The next section is dedicated to an example of usage of the component. In the last section, we detail the next steps for improving the component.

# Technical description

## Functional Isolation Forest

Isolation Forest (Liu, Ting, & Zhou, 2008) is a powerful unsupervised anomaly detection algorithm. It is based on tree ensemble and is built on the basis of decision trees. In these trees, partitions are created by first randomly selecting a feature and then selecting a random split value between the minimum and maximum value of the selected feature. In principle, abnormal points are less frequent than regular observations and are different from them in terms of values. That is why by using such random partitioning they should be identified closer to the root of the tree (shorter average path length, i.e., the number of edges an observation must pass in the tree going from the root to the terminal node), with fewer splits necessary. To gain in robustness, we train several isolation tree and compute the anomaly score : 𝒔(𝒙, 𝒏)=𝟐^(−(𝑬(𝒉(𝒙)))/(𝒄(𝒏))), with x an observation, n the number of instance, E(h(x)) the expectation of the path length h(x) of x which is measured by the number of edges x traverses an Isolation Tree from the root node until the traversal is terminated at an external node and c(n) a normalization constant. This score is between 0 (normal point) and 1 (abnormal point). If the score of all observation are nearly 0.5, then there is no anomaly. The hyperparameters are the number of isolation tree and a subsampling parameter, because above a threshold, it is useless to keep all data. For each Isolation Tree a new subsample is randomly chosen. The evaluation stage consist in passing a new observation x in all the Isolation Tree and make an empirical estimation of E(h(x)).

(Staerman, Mozharovskyi, Clémençon, & d'Alché-Buc, 2019) propose an adaptation to time series of Isolation Forest, and call this adaptation Functional Isolation Forest. The idea behind their article is to project the time series in a well-chosen dictionary according to a chosen scalar product, and follow the same idea that Isolation Forest. The dictionary is chosen to be rich enough to explore diﬀerent properties of data and well appropriate to be sampled in a representative manner. The projection of a time series on each member of the dictionary deﬁnes a feature that partially describes the time series. When considering all the functions of the dictionary, one gets a set of candidate Split variables that provides a rich representation of the time series, depending on the nature of the dictionary. Dictionaries have been throughly studied in the signal processing community to achieve sparse coding of signals. They provide a way to incorporate a priori information about the nature of the data, and so expert knowledge. The choice of a scalar products is driving by the anomalies we search. L2 scalar product allows for detection of location anomalies and the L2 scalar product of derivatives (or slopes) allows to detect anomalies regarding shape. It is possible combination of these two scalars products with Sobolev scalar product.

## ADTK

ADTK is a package for unsupervised/rule-based models of time series anomaly detection. It offers a set of common components that can be combined into various types of anomaly detection models for different scenarios. It allows to detect some usale anomalies type in time series:

* Outliers: data point whose value is significantly different from others
* Spike and Level Shift: An abrupt increase or decrease of value is called a spike if the change is temporary, or a level shift if the change is permanent
* Pattern change: change in volatility, in seasonality, in autoregressive relationship, etc.

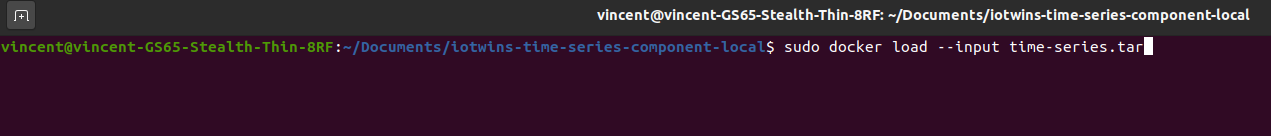
## Technologies used

The component is a Docker container which contains the following requirement:

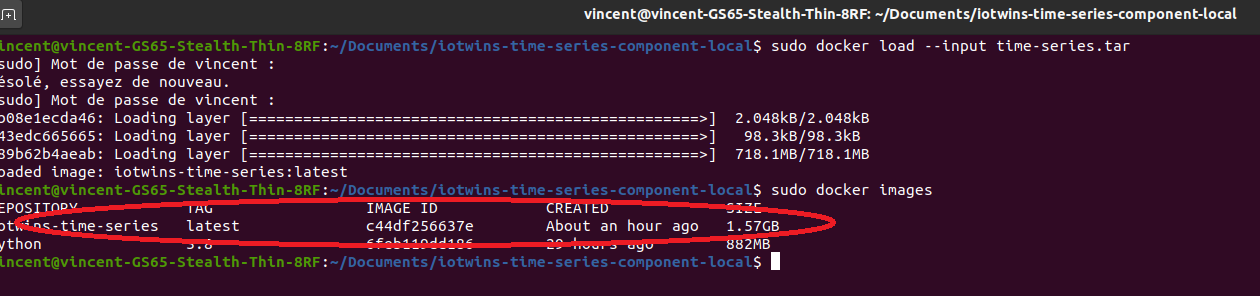
* Python 3.8
* Python module:
  + Plotly>=4.6.0
  + dash==1.14
  + dash-table==4.9
  + dash-upload-components==0.0.2
  + dash-core-components==1.10.1
  + dash-html-components==1.0.3
  + Flask==1.1.2
  + boto3==1.14.36
  + botocore==1.17.36
  + numpy==1.19.1
  + pandas==1.1.0
  + scipy==1.5.2
  + gunicorn==19.9.0
  + scikit-learn==0.23.2
  + urllib3==1.25.10
  + dill==0.3.2
  + adtk==0.6.2
  + cufflinks==0.17.3

# Installation

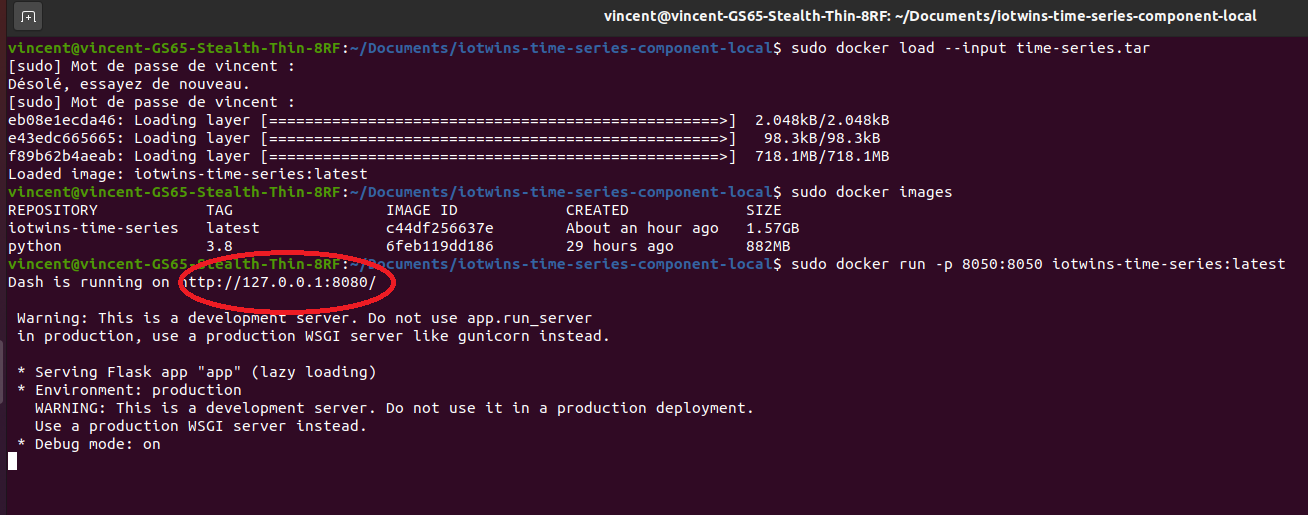
The component is saved in a tar folder, time-series.tar. To install, the component open a Terminal, go in the folder with the time-series.tar (basically with a cd path/to/time-series-component command). Then load the docker as presented in the next Figure.



Then use a sudo docker image to check if the image of time-series component is well loaded, like in the following Figure.



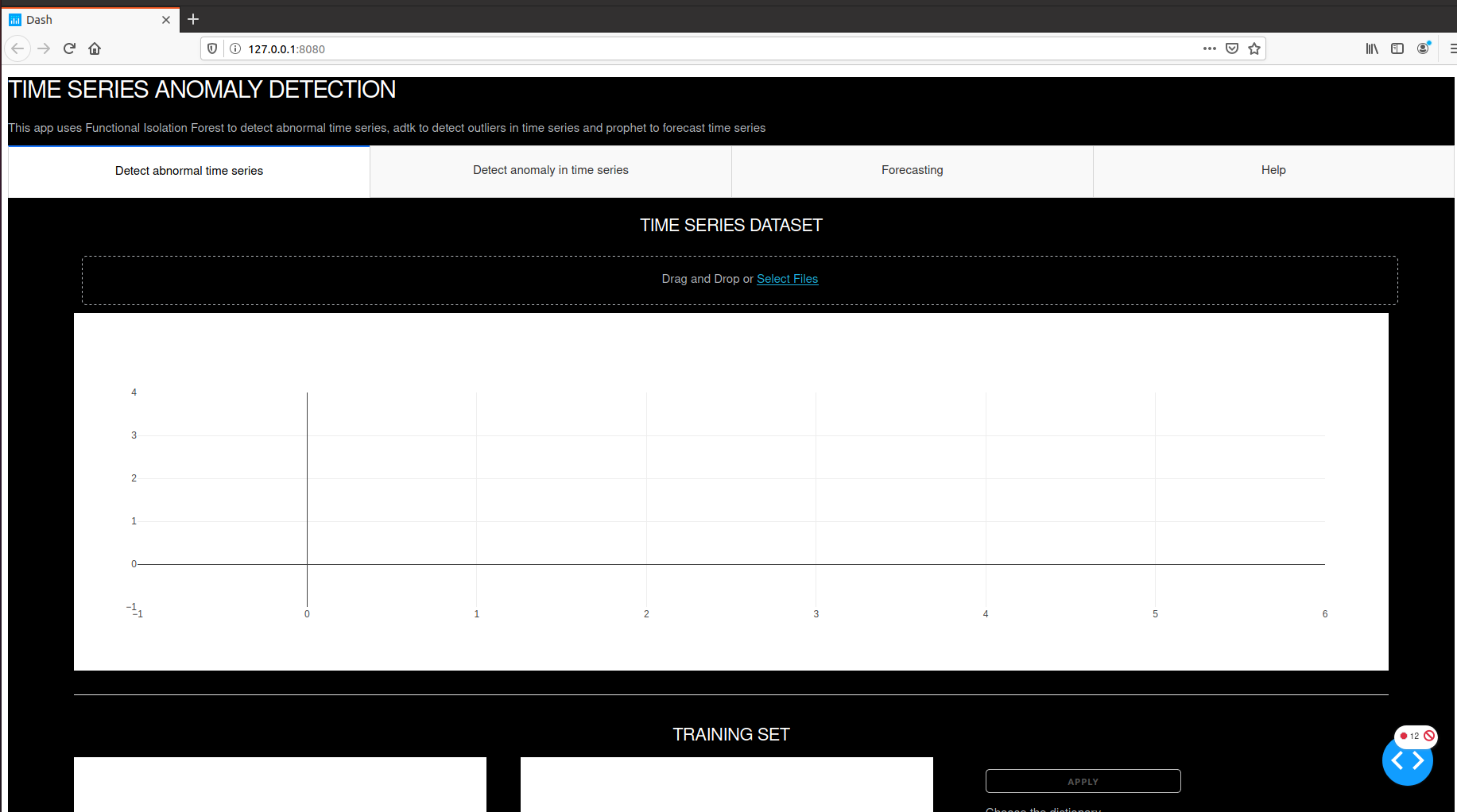
Then launch the container with a docker run command, like in the following Figure. The component is now running on <http://127.0.0.1:8080> port, which can be opened in a browser.



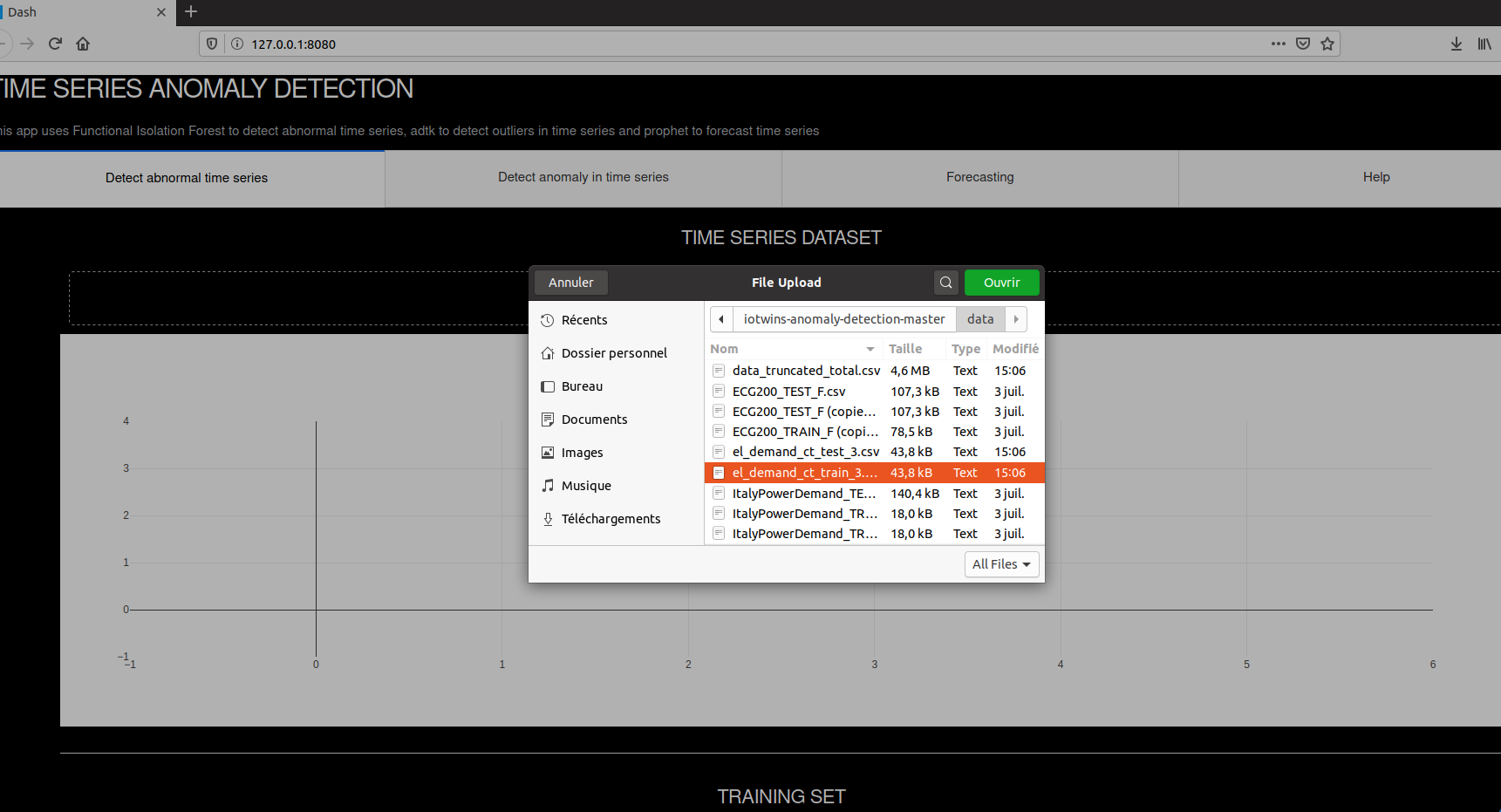
# Use of component

## Detect abnormal time series in a corpus of time series

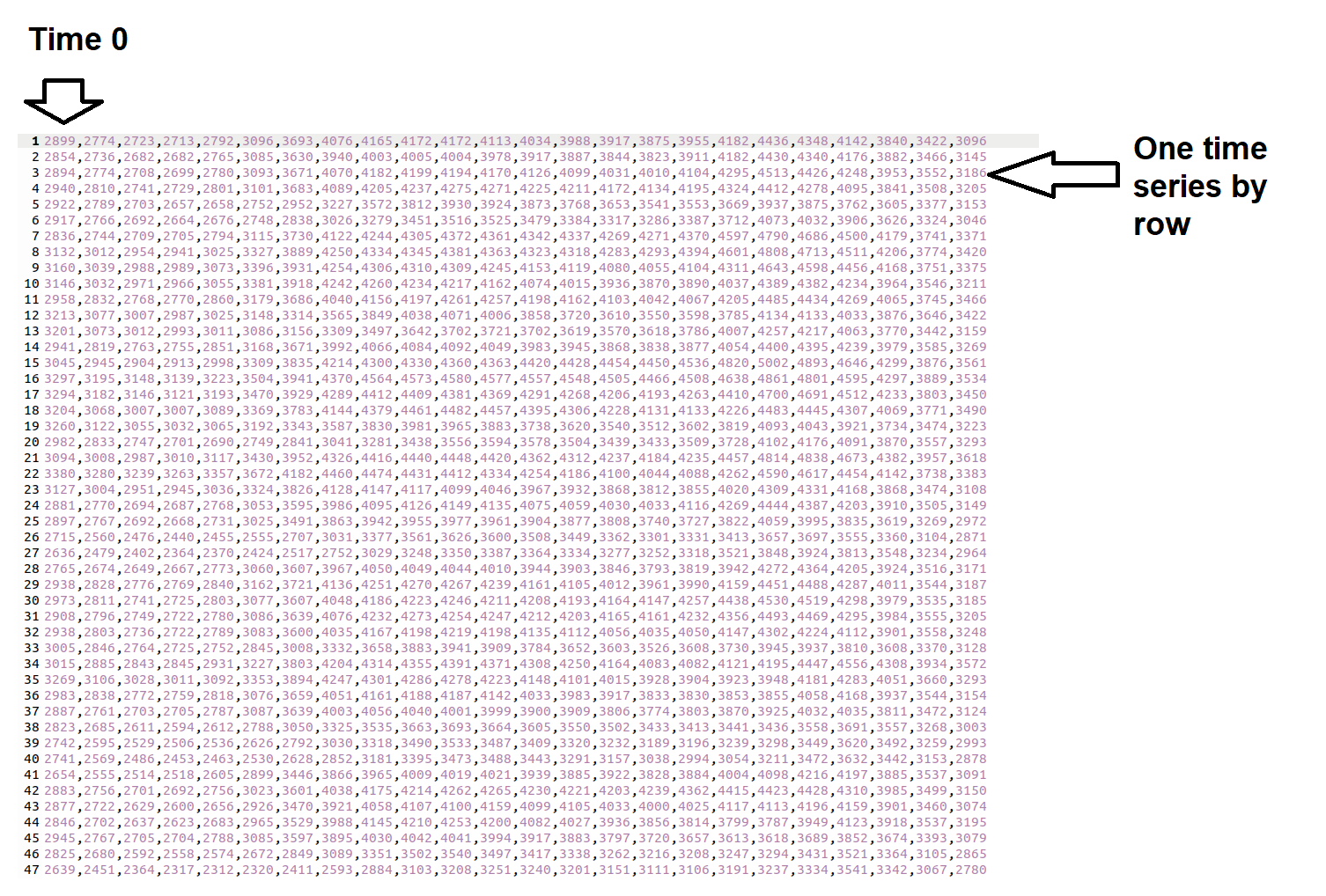
When the application is running, the user can open it in a browser. On the first tab, there is the sub-component that allows detecting abnormal time series in a corpus of time series, based on Functional Isolation Forest.



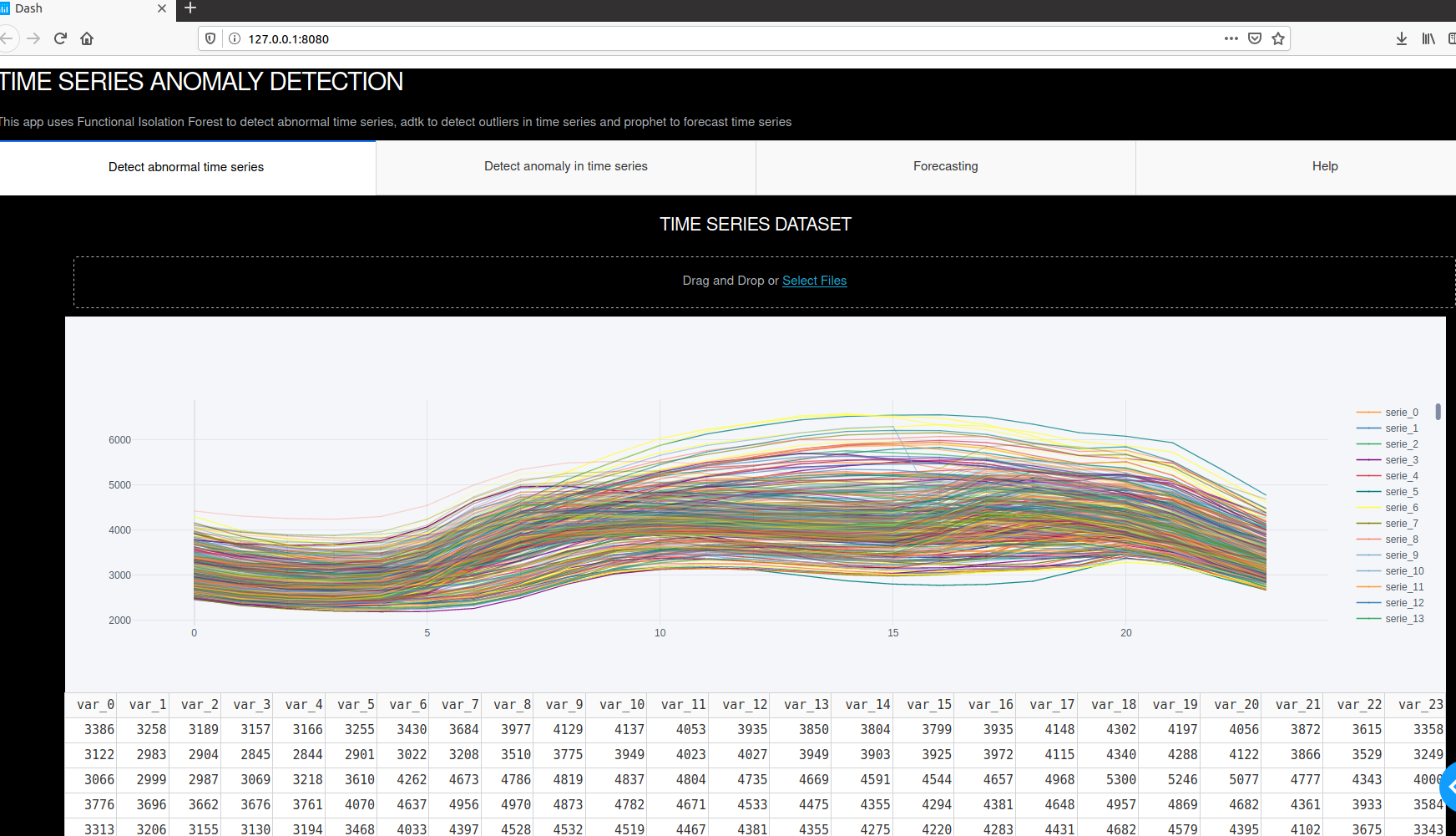
With the button “Drag and Drop data or select files”, the user can load some data.



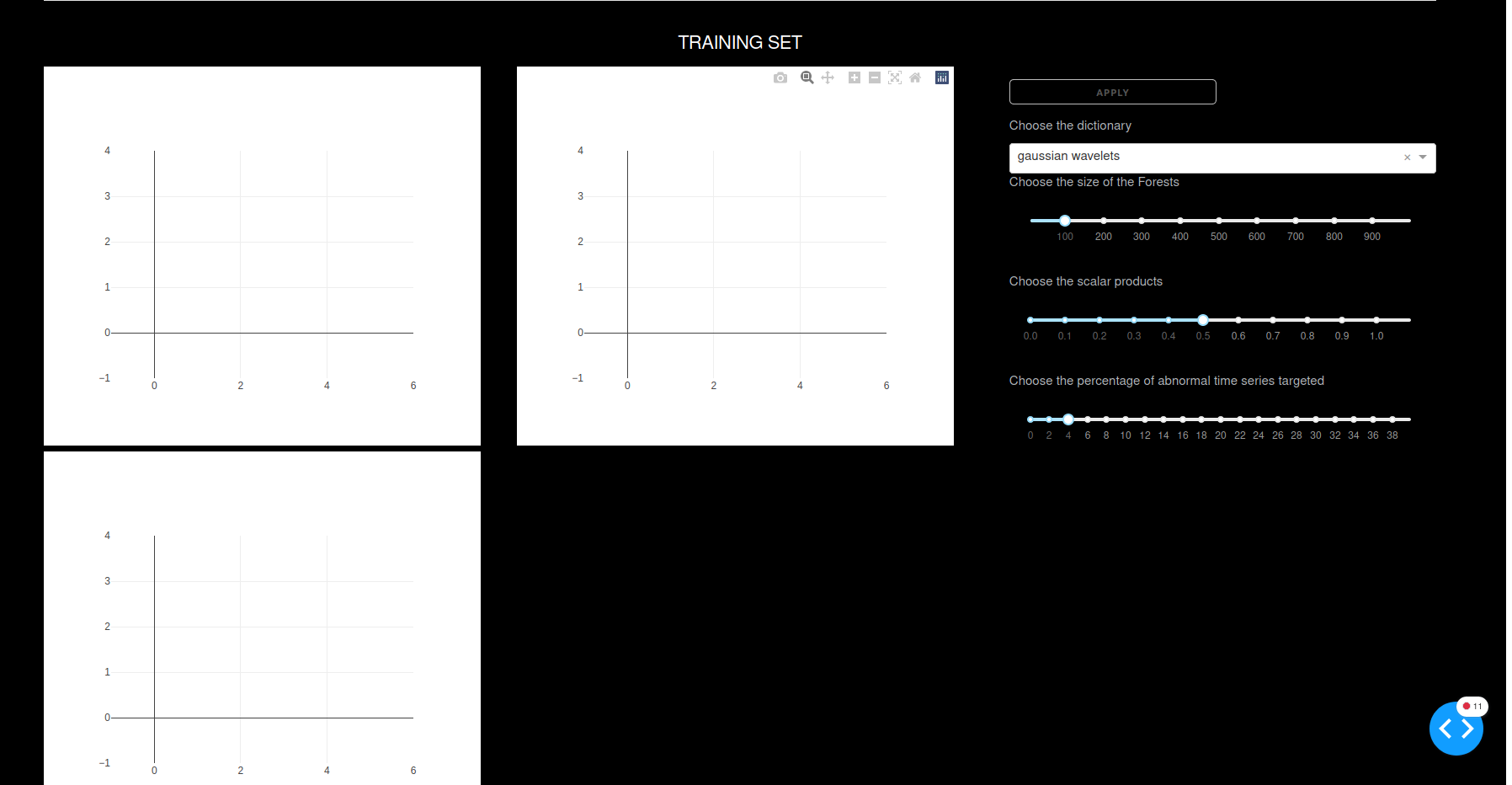
The format of the data should have for rows a time series and for columns the time, with no column header.



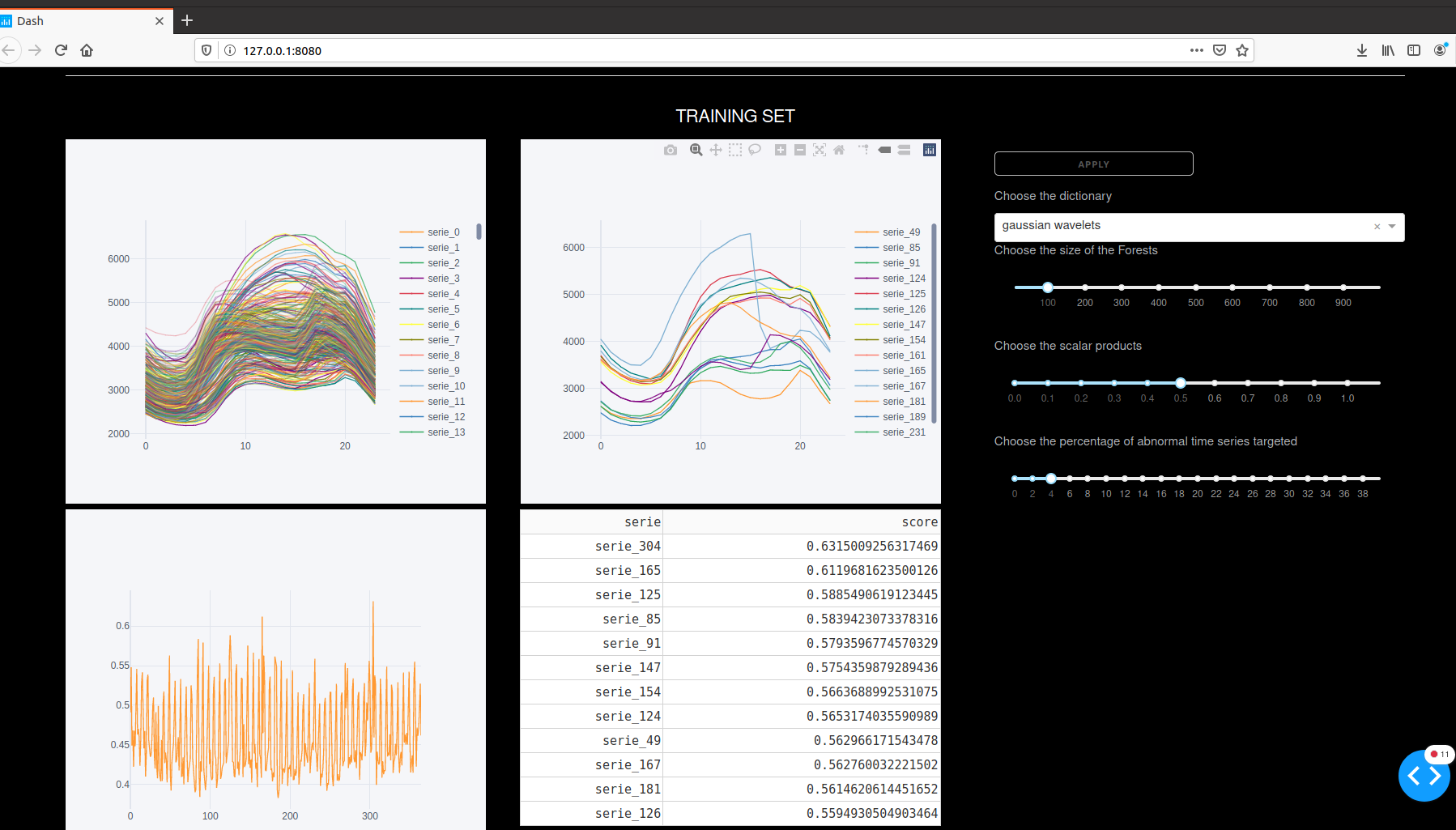
When the data are loaded, the time series of the corpus are plotted on the graph and the first time series values are given in the table.



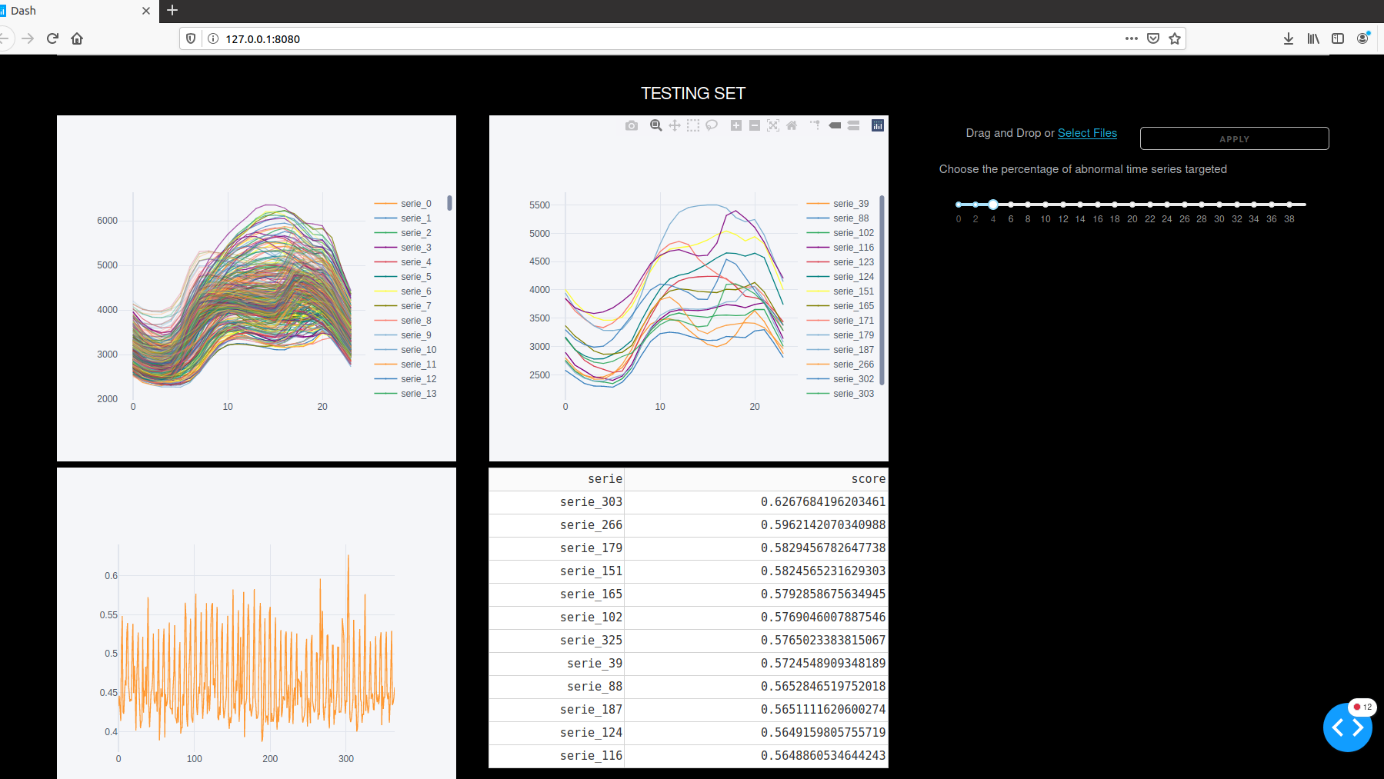
By scrolling down the tab, the user gets to the Functional isolation forest hyperparameters. She can choose between several possible dictionary, the scalar products. This value correspond to the trade-off in the Sobolev scalar products: when selecting 1, it corresponds to the L2 norm for the function, 0 to the L2 norm for the derivate, and between 0 and 1 the weight given to the two L2 norm. The user can choose the percentage of abnormal time series targeted. If she indicated 4% for instance, the 4% time series with the highest anomaly score will be considered as abnormal. When the user has choose the hyperparameter, she can click on apply button.



When the Functional Isolation Forest is learnt, 3 plots and one Table is given. On the top left chart, all the time series considered as normal are plotted. On the top right chat, the abnormal time series are plotted. For this two charts, it is possible to select some series by double clicking on their legend. On the bottom left chart, the anomaly score of each series is represented. In the Table, the series with strongest anomaly score is given. It is possible to wave the charts, zoom some points, etc.

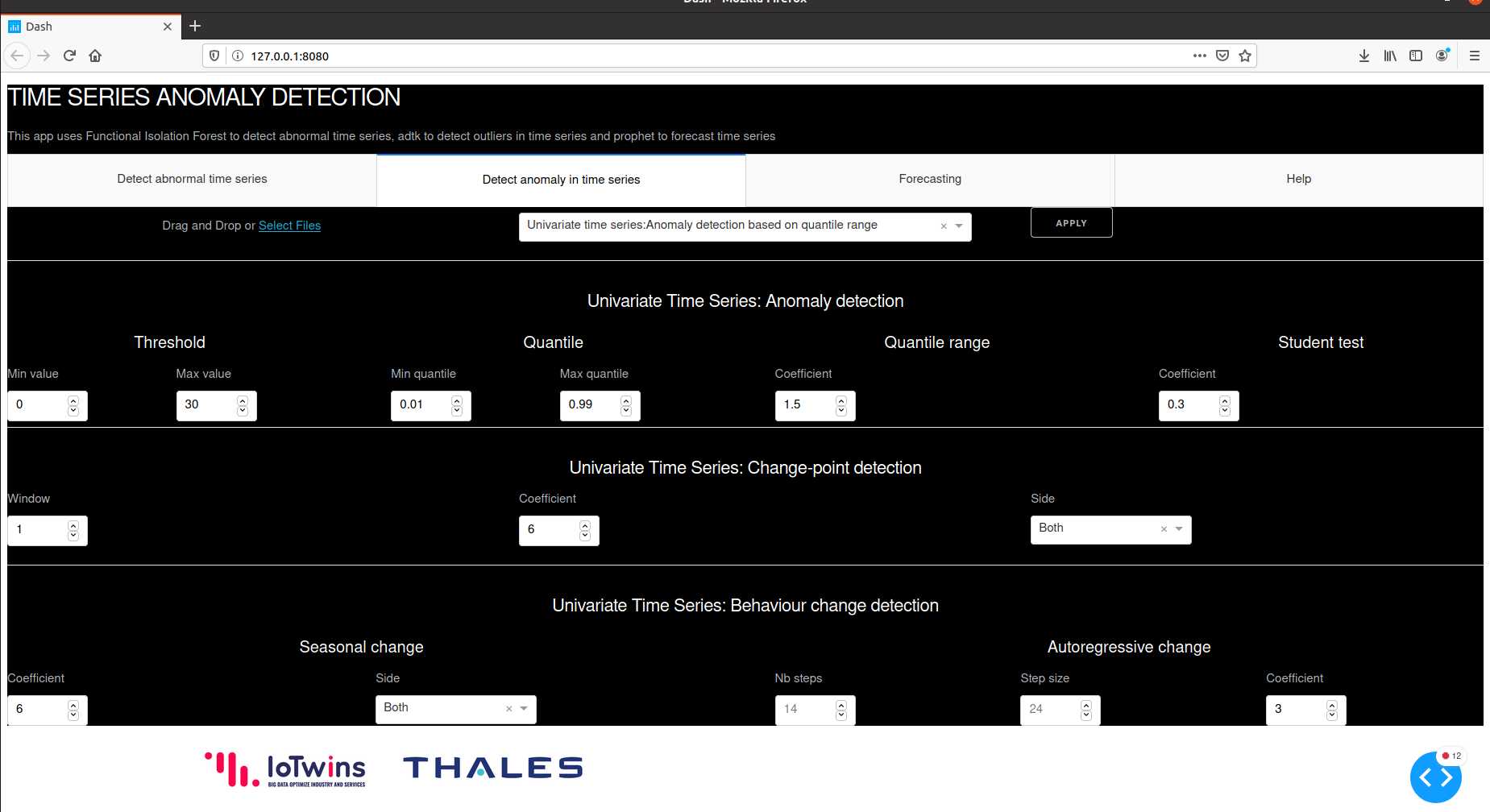


It is possible to use the Functional Isolation Forest which has been learnt on a new dataset of time series. For that, the user has to scrolling down the tab again. The only parameter she has to choose is the percentage of abnormal time series targeted. By clicking on “Drag and Drop or Select Files”, the user can choose the new time series corpus, which has to respect the same format that the previous data format. The given charts and Table are the same that for the training set.



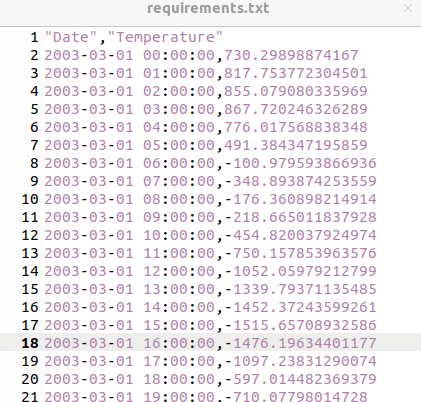
## Detect outliers and change points with ADTK

The second tab offers the dashboard for using ADTK. The objective here is to detect outliers and change points in one time series.

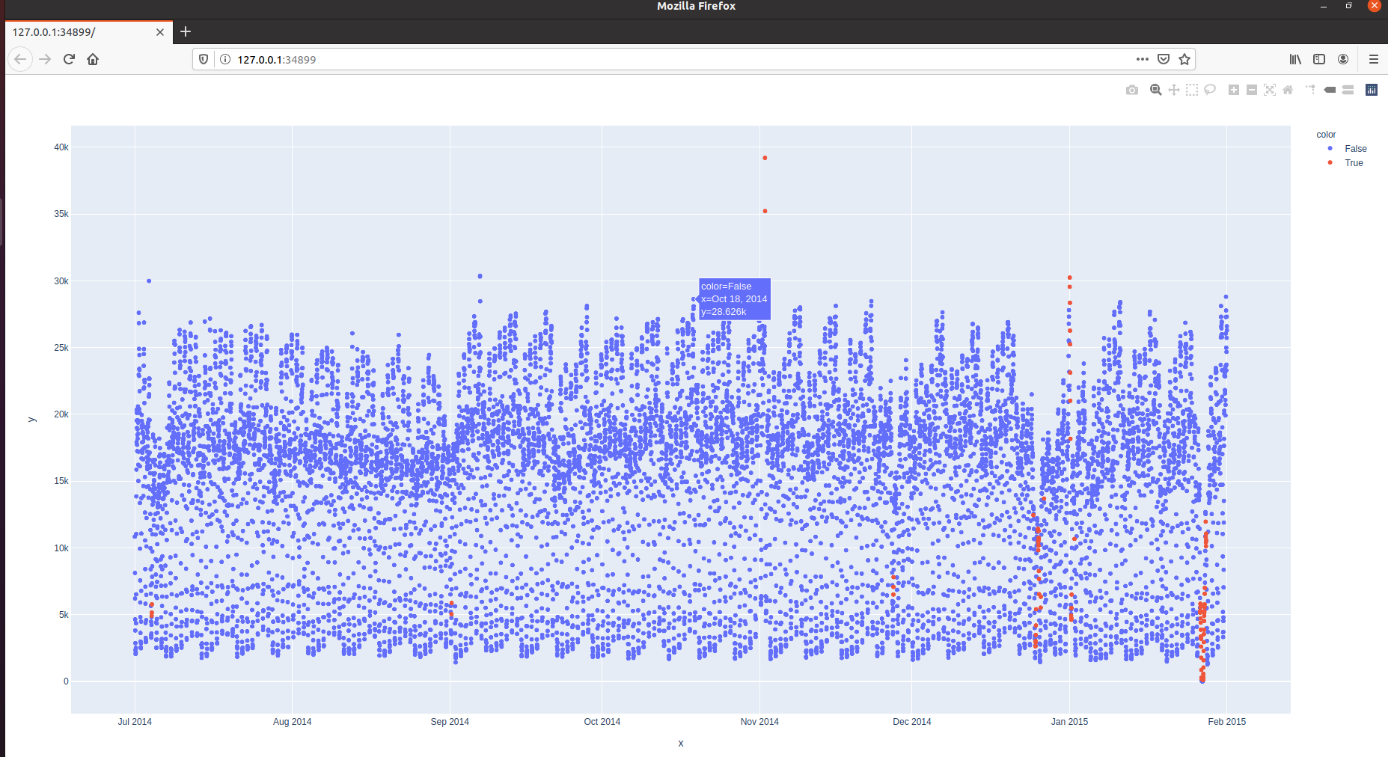


|  |  |  |  |
| --- | --- | --- | --- |
| Objective | Description | Hyperparameter | Hyperparameter Description |
| Anomaly detection based on threshold | The value smaller and greater than two thresholds are considered as abnormal | Threshold min value | Value for which the lower values are considered abnormal |
| Threshold max value | Value for which the stronger values are considered abnormal |
| Anomaly detection based on quantile | The value smaller and greater than two chosen quantiles are considered as abnormal | Quantile min value | Quantile for which the lower values are considered abnormal |
| Quantile max value | Quantile for which the stronger values are considered abnormal |
| Anomaly detection based on quartile range | A value is said abnormal when it is out of the range defined by [𝑄1−𝑐×𝐼𝑄𝑅, 𝑄3+𝑐×𝐼𝑄𝑅] where 𝐼𝑄𝑅=𝑄3−𝑄1 is the difference between 25% and 75% quantiles | Quantile range coefficient | Coefficient c of objective description |
| Anomaly detection based on student test | Detects anomaly based on generalized extreme Studentized deviate (ESD) test. Note a key assumption of generalized ESD test is that normal values follow an approximately normal distribution | Student test coefficient | Significance level. Classical value is 0.05. |
| Change point detection based on recent observation | Compares each time series value with its previous values | Univariate time series: change point detection window | Size of the preceding time window. Default is 1. |
| Univariate time series: change point detection coefficient | Factor used to determine the bound of normal range based on historical interquartile range. |
| Univariate time series: change point detection side | If "both", to detect anomalous positive and negative changes.  If "positive", to only detect anomalous positive changes. If "negative", to only detect anomalous negative changes. |
| Change point detection based on level shift | Detects shift of value level by tracking the difference between median values at two sliding time windows next to each other. | Univariate time series: change point detection window | Size of the preceding time window. Default is 1. |
| Univariate time series: change point detection coefficient | Factor used to determine the bound of normal range based on historical interquartile range. |
| Univariate time series: change point detection side | If "both", to detect anomalous positive and negative changes.  If "positive", to only detect anomalous positive changes. If "negative", to only detect anomalous negative changes. |
| Change point detection based on variance shift | Detects a shift of volatility level by tracking the difference between standard deviations at two sliding time windows next to each other. | Univariate time series: change point detection window | Size of the preceding time window. Default is 1. |
| Univariate time series: change point detection coefficient | Factor used to determine the bound of normal range based on historical interquartile range. |
| Univariate time series: change point detection side | If "both", to detect anomalous positive and negative changes.  If "positive", to only detect anomalous positive changes. If "negative", to only detect anomalous negative changes. |
| Change in seasonal behavior | Detects anomalous violations of seasonal pattern | Seasonal change coefficient | Factor used to determine the bound of normal range based on historical interquartile range. |
| Seasonal change side | If "both", to detect anomalous positive and negative changes.  If "positive", to only detect anomalous positive changes. If "negative", to only detect anomalous negative changes. |
| Change in autoregressive behavior | Detects anomalous changes of autoregressive behavior in time series. | Autoregressive change Nb steps | Number of steps (previous values) to include in the model |
| Autoregressive change Step size | Length of a step. For example, if n\_steps=2, step\_size=3, X\_[t-3] and X\_[t-6] will be used to predict X\_[t]. |
| Autoregressive change Coefficient | Factor used to determine the bound of normal range based on historical interquartile range. |

Given data should be on the following format, with two columns: first one is the time and the second is the series:



Next Figure gives an example of the results. Here, we detect some abnormal seasonal behavior.



# Next steps

* Robustify component implementation
* Improve IHM
* Abnormal time series detection in a time series corpus tab:
  + Improve representation
  + Robustify internal code
* Detect anomaly in time series
  + Implement multivariate case
  + Improve representation and dashboard
  + Robustify internal code
* Forecasting
  + Implement the use of prophet
  + Dashboard implementation
* Help writing in HTML

# References

ADTK: https://github.com/arundo/adtk. (2020).

Liu, Ting, & Zhou. (2008). Isolation forest. *In Data Mining, 2008. ICDM’08. Eighth IEEE International Conference on*.

Staerman, Mozharovskyi, Clémençon, & d'Alché-Buc. (2019). Functional Isolation Forest.

Taylor​, & Letham​. (2017). Forecasting at scale.