@author: Fred Coerver – 2022 <https://github.com/FredC123/Fast-Anomaly-Detection>

**Description how this model works**

Anomaly detection is a concept, which discussed in many broad area’s, disciplines and mathematical concepts. Various ways can be used to approach the data and find anomalies. In most cases, just like this method the dataset can be huge. This model however is used to find anomalies in a fast way and with a flexible sentitivity. This method is very successful in smaller datasets due to the fact that is is very fast and very precise. The input data is x and y and the model return the x values and indentifies this value whether it is an anomaly or not, given the hyperparameters set by the user.

Thus the model is flexible, precise, can be used for a huge set of x,y values as well al shorter datasets and a dataset with x, y1, y2,….,yn can be processed in the loop of a python script very easy and remains still very fast.

The concept of this algorithm is totally not like machine learning like used in scikit.

The concept of testset and trainingset is applied though. The testset is the evaluated value y[j] and the trainingset is y[j-w] + y[j+w], where w is the trainingset window parameter.

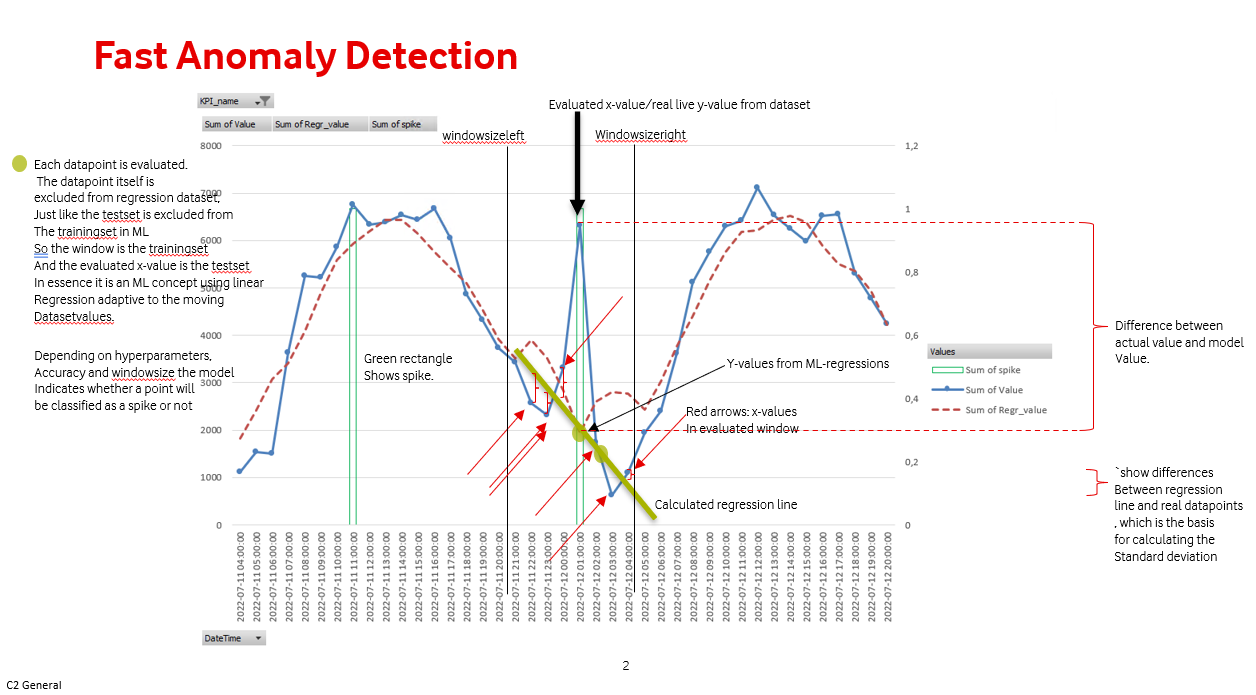
The trainingset is calculated via linear regression and the testset is calculated via slope/interception.

Each x[j] is calculated, resulting that the regressed polynom is close to the originale. See below graph.

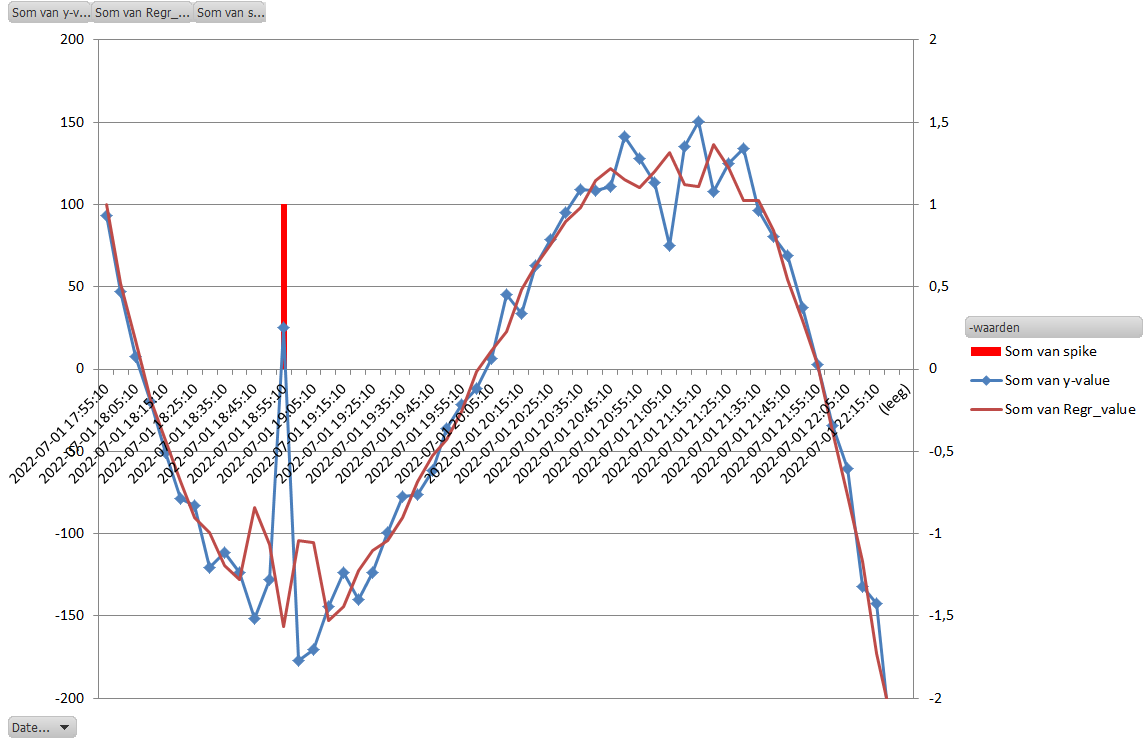
Blue are the original value, and the red dotted line is the model polynom. From there the spikes are indentified via the hyperparameters.

The overall similarity between ML-algorithms in scikit and this model, is that the setting of the hyperparameters is extremely important and must be optimize for each x,y[j] combination.

A good example of applying this model is fast anomaly detection of recent data, from a live data-feed

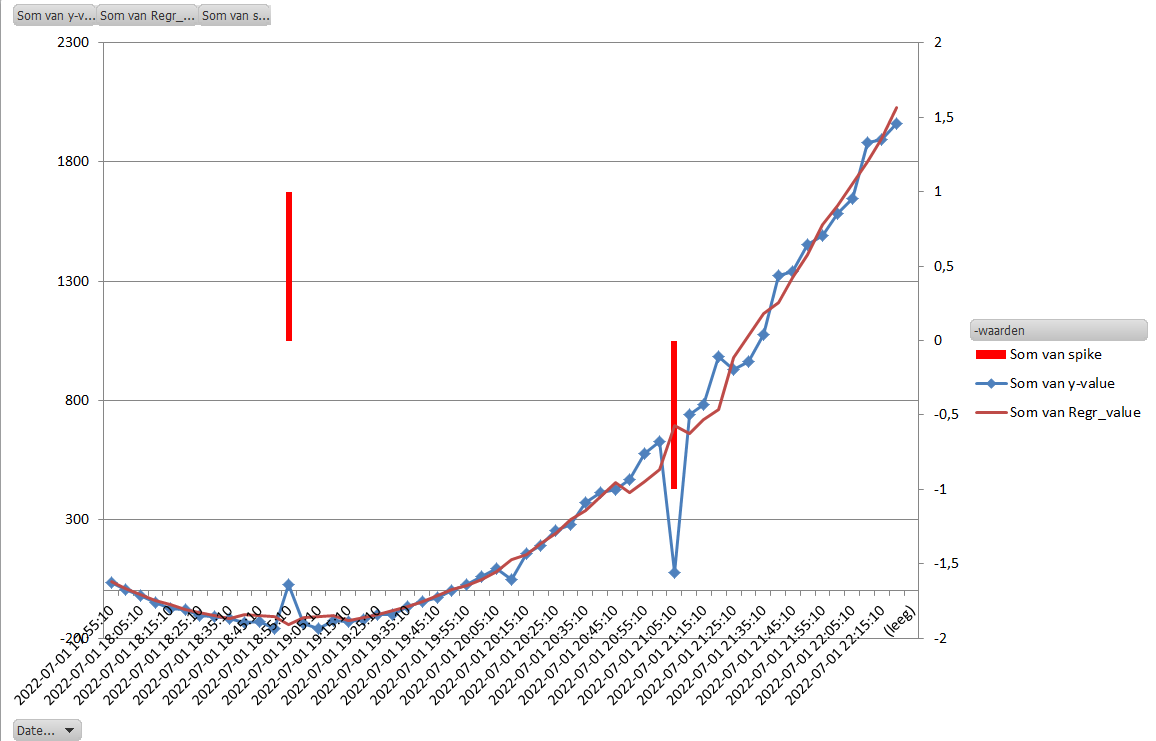


An indication of the sensitivity is shown below:



Below graph shows that the model is very good in following increasing trends, which is giving you a hard time with scikit models.

The downside of this model is prediction, which is really not possible if you understand the concept. Nevertheless for anomaly-detection this is not needed anyhow,



This module takes a pandas frame as input, together with several hyperparameters.

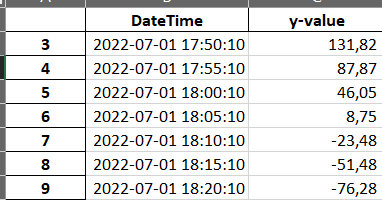
The pandas data frame must have a straight forward x and y column including an index. Not more Not less.

The output are two pandas dataframes. One table with statistical information of the input and whether a value x is identified as a spike or dip in the dataset, based on the input parameters. And the second is a list of spike. More details about the output below. The algorithm makes a window of datapoints from x - windowsizeleft to x + windowsizeright

Linear regression is performed on this x-range with the give y-values. The regression value : y = slope\*x + intercept

This window is iterated over the pandas frame and the regression values are put into the output table, along with the other statistical values.

Input dataframe structure :



1. pandas index as int
2. x or DateTime value format %Y-%m-%d %H:%M:%S {working on an update that also accepts floats as x}. Column name must be "DateTime" if Timescale is true and must be next/right to the index
3. y {float or int}. Columnn name is free.

e.g. Regression is internally in the model executed on :

y=[115730244, 117300778, 116863076, 108493789, 97012607, 95430706]

x=[1657540800, 1657544400, 1657548000, 1657555200, 1657558800, 1657562400]

The data frame can have more columns, but the script only analyse 1 columns at a time.

So in case you want to analyse more columns, you need to call the script for each column and slicing

the data frame in a way that the input frame complies to 1), 2) and 3)

Hyperparameters

ignore\_startsamples = 5 (in case you want to omit starting rows from your calculation of the dataset. This parameter does not delete the rows! = default is 5. In case you increase the windowsizeleft, you might need to increase this value as well)

ignore\_endsamples = 3 (in case you want to omit starting rows from your calculation of the dataset. This parameter does not delete the rows! In case you increase the windowsizeright, you might need to increase this value as well)

P1inc = 10 ---> In case the standard deviation of a regression window is P1inc-times higher as the previous std, then this x value is marked as a spike

accuracy = 4 ---> If a y-value exceeds the regressed value +- accuracy \* std then this x-value is marked as spike

window\_sizeleft = 3 ---> the regresssionwindow is #windowsizeleft x-values from the analysed x-point and #windowsizeright x-values from the analyzed x-point. So suppose the algorythm calculates whether x[j] is a spike, then the window x[j-windowsizeleft] until x[j+windowsizeright] is considered as the window of x-points. From these x- an y-point the regression is calculated, and the regression value is compared with the actuel y-value

window\_sizeright = 3 ---> amount of datapoint right from the x-value, which is subject for analyzing whether it is a spike/dip or not

ignoreendsamples = 3 ---> the Endresult output dataframe is capped with [ignorestartsamples:-ignoreendsamples] before returning to main

Timescale ---> If x-values are different from time then any string is valid, in case the x-values need to be in timeformat then this parameter is True.

OUTPUT:

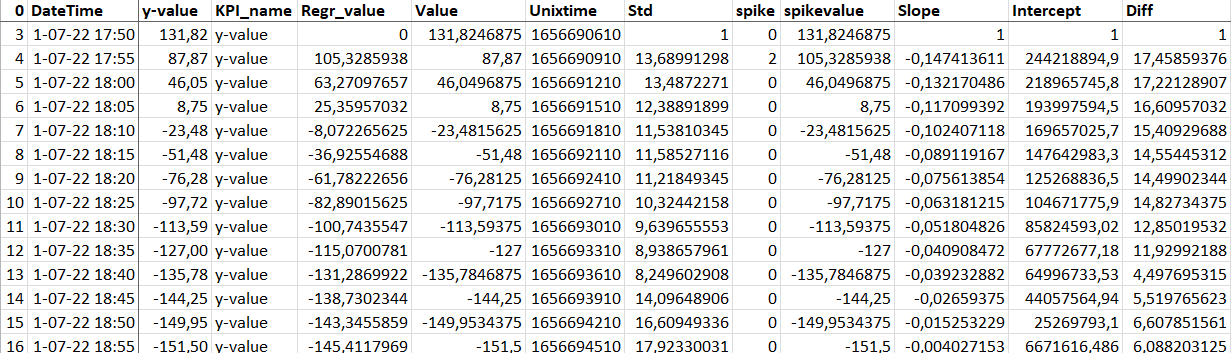
1) Pandas dataframe with format columns:

'DateTime', 'y-value', 'KPI\_name', 'Regr\_value', 'Value', 'Unixtime',

'Std', 'spike', 'spikevalue', 'Slope', 'Intercept', 'Diff',

'Regr\_value\_plus\_std', 'Regr\_value\_min\_std'

Example of output data



2) Pandas dataframe as 1) but then sorted and 'spike' >=1 or <=-1

Same structure as frame in 1) but applied filter >=1 or <=1

"""

Attached script is included in the repository as an example.



The definition GetSandD() is included in the repository