Online Anomaly Detection in Unlabeled Univariate Time Series Data

A Comparison of Approaches

Malte Zietlow

Department of Computer Science
Nordakademie
Köllner Chaussee 11
25337 Elmshorn

malte.zietlow@nordakademie.de

18th January 2021

Abstract

In this paper, 19 anomaly detection methods (available from open source libraries) are evaluated on the Numenta Anomaly Benchmark (NAB)-dataset. Evaluation metrics include the numenta standard- and F1-Score. Besides metrics, a visual examination and comparison of the algorithms is attempted. At it, the author faced issues inherent with NAB, including a lack of variability. Finally, the author argues that in the future a more dynamic process of visual evaluation should be constructed that might require dynamic generation of time series to more reliably confirm — or falsify claims about discussed algorithms.

Contents

List of Figures

List of Tables

Acronyms

AE Autoencoder. ii, xvi, xxxvii, 13, 14

CBLOF Clustering Based Local Outlier Factor. ii, xvi, xxxii, xxxiii, 13, 14

CDF Cumulative Density Function. viii

DAGMM Deep Autoencoding Gaussian Mixture Model. ii, xvi, xxvii, 14

HTM Hierarchical Temporal Memory. ii, xvi, xxxviii, xxxix, 13–15

kNN k-Nearest Neighbors. ii, xvi, xxxiv, 13, 14

LOF Local Outlier Factor. ii, xvi, xxv, xxvi, 14

NAB Numenta Anomaly Benchmark. x, 1, 5–8, 10, 11, 13, 14, 16

NDT Nonparametric Dynamic Thresholding. ii, xxix, 4, 11–14

OCSVM One-Class Support Vector Machine. ii, vii, xvi, xxiv, 11, 14

RRCF Robust Random Cut Forest. ii, xxviii, 11

1 Introduction

It is not new that companies are using an evergrowing set of computer systems to keep their services online. Monitoring teams are challenged to identify, fix, and prevent breakages to these systems. This involves analysis of continuous streams of system-health data, made up of e.g. natural language logs, hits per minute, or system loads like CPU and RAM. While breakages often manifest visibly to the human spectator in the related system-health data, it is impossible to spectate millions of data streams as often produced in larger companies [cf. ZL17].

To alleviate this complexity, anomaly detection algorithms are introduced, which automate the detection of such system issues. However, the domain of anomaly detection is frequently evolving, and it is unclear, which algorithm to choose.

In this paper, a number of algorithms is evaluated and compared using the Numenta Anomaly Benchmark (NAB)-dataset, consisting of 58 time series [LA15].

The paper is organized as follows. In section 2, definitions for time series, anomalies and anomaly detection are found. In section 3, the NAB metric and dataset are described and a recent controversy surrounding NAB is discussed. Also, reasoning for algorithm-selection is given. In section 4, the produced results are presented, discussed, and recommendations on algorithm selection are given. Finally, in section 5 the paper is wrapped up and recommendations on future work are given.

2 Definitions

Although theoretical examination is not central to this work, the interested reader shall be introduced to the most fundamental concepts in this section.

Namely,

a.) what is the data structure (time series),

b.) what are we trying to retrieve (anomalies), and

c.) how are we retrieving it (anomaly detection).

2.1 What is the data structure?

System-health data is produced in a variety of formats. Most commonly it comprises multiple simultaneously produced streams of either natural language (in form of log-messages [Mar20]) or numerical data.

To reduce complexity, only univariate streams of numerical data will be considered in this paper.

Definition 2.1 (Observation)**.** Observations are numerical values observed at a certain point in time. Mathematically, it is the tuple given in eq. (1)

$$\text{Observation } o := (t, y) \tag{1}$$

where:

$$\begin{aligned} &\text{timestamp } t \in \mathbb{Z} \\ &\text{value } y \quad\ \in \mathbb{R} \end{aligned}$$

Arithmetic operations are executed using value y, s.t. $o_i + o_j = y_i + y_j$, where $+$ is an arbitrary operation.

The timestamp can be further interpreted as POSIX time [The18], coarsely given in eq. (2).

$$\begin{cases} \text{before 01 Jan 1970 00:00:00 GMT,} & \text{if } t < 0 \\ \text{01 Jan 1970 00:00:00 GMT,} & \text{if } t = 0 \\ \text{after 01 Jan 1970 00:00:00 GMT,} & \text{if } t > 0 \end{cases} \tag{2}$$

Definition 2.2 (Time Series)**.** A time series is the ordered set of multiple related observations [cf. Box+16]. It can be indexed in two complementary ways, using either a.) the set of timestamps eq. (3) or b.) serially numbering of the observations (i.e. the set $\{0, 1, \dots, T-1\}$) eq. (4).

$$\text{Timestamp-Indexed TS} := \left\{ y^{(t_0)}, \dots, y^{(t_{T-1})} \right\} \tag{3}$$

$$\text{Count-Indexed TS} := \left\{ o^{(0)}, \dots, o^{(T-1)} \right\} \tag{4}$$

where:

$$\begin{aligned} T \quad &:= \text{Number of observations} \\ t_i \quad &:= \text{Timestamp of ith Observation} \\ o^{(i)} \quad &:= \text{The ith observation} \\ y^{(t_i)} \quad &:= \text{Value of the observation with timestamp } t_i \end{aligned}$$

From a practical standpoint, count-indexed time series are useful for e.g. construction of sliding-windows or neighborhoods (definition 2.3), while timestamp-indexed time series are useful for e.g. inference of cyclicity (definition 2.4)

The observation o_t is then usually related to/influenced by several other observations within the same time series. Most obviously, o_t is related to values from its neighborhood (definition 2.3). Furthermore, it might be influenced by multiple levels of cyclicity (definition 2.4).

Definition 2.3 (Neighborhood)**.** The neighborhood of an observation in a time series is the set of its n-surrounding values. Mathematically given in eq. (5).

$$\{o_{t-n/2}, \dots, o_{t+n/2}\} \setminus o_t,\ n \in \mathbb{N} \tag{5}$$

Definition 2.4 (Cyclicity)**.** A time series has some kind of cyclicity if it contains a pattern that repeats approximately every P steps, where P can be some random variable. In simple terms, the time series TS is said to be (additively-constant) cyclic if $\text{TS}(i + P) \approx \text{TS}(i) + e$, $e \in \mathbb{R}$, where e is some error. This definition could easily be extended to more complex cyclic patterns. For detailed examination see [Fal+12].

2.2 What are we trying to retrieve?

Breakages can manifest visually in different forms. Common forms are

1. a drop/peak significantly below/above *all* previously observed values,
2. a drop/peak below/above values from the neighborhood,
3. a drop/peak below/above expected cyclic dependencies,
4. a change point (adjustment of an existing or introduction of a new cyclic pattern).

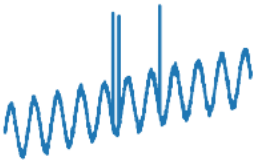
An influential taxonomy of such breakages has been introduced by Chandola, Banerjee and Kumar [CBK09] as follows:

Definition 2.5 (Point Anomaly)**.** Point Anomalies are **singular** observation that deviate significantly from the general distribution [cf. CBK09]. This comprises common breakage form 1. A visualization is given in fig. 1a.

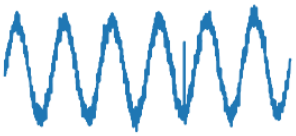
Definition 2.6 (Contextual Anomaly)**.** Contextual Anomalies are **singular** observations which do not deviate strongly from the general distribution but only from their neighborhood or cyclic dependencies [cf. CBK09]. This comprises common breakage forms 2 and 3. A visualization is given in fig. 1b.

Definition 2.7 (Collective Anomaly)**.** Collective Anomalies are multiple data points which, in isolation, do not appear anomalous. When observed as a group however, they show unusual characteristics [cf. CBK09]. This comprises common breakage forms 3 and 4. A visualization is given in fig. 1c.

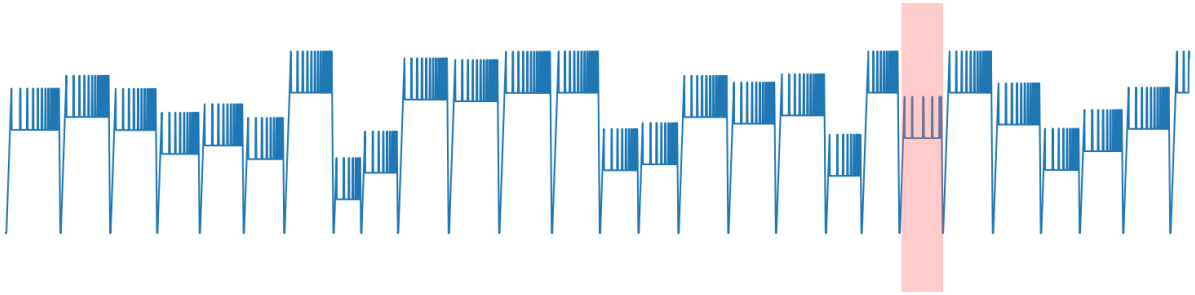
All common breakage forms are included in one of the three anomaly definitions. In anomaly detection we therefore try to retrieve all data points which fall under one of the definitions 2.5 to 2.7.



(a) Point anomalies from Yahoo!S5: A2Benchmark.synthetic_11.



(b) Contextual anomaly from Yahoo!S5: A2Benchmark.synthetic_15.



(c) Collective anomalies from Gasoil-Dataset [FLV16]: 48_Lev_corr_Temp_fault_seed_199_vars_23["RT_level"].

Figure 1: Examples of the three anomaly types [CBK09] from different datasets. Illustrations by the author.

2.3 How are we retrieving it?

Retrieval of anomalies as defined in definitions 2.5 to 2.7 can be done via different paradigms. Extensive taxonomies have been proposed in [CBK09; Mar20]. Because theoretical examination is not central to this paper, the taxonomy will be presented simplified and condensed.

Two major groups of algorithms can be considered: either Forecasting- or Boundary-based:

Definition 2.8 (Forecasting-Based-Algorithm)**.** Forecasting-based algorithms consist of two major phases:

a.) In the first phase, the algorithm tries to predict the next n-observations given a learning history of previous observations. A predicted observation is denoted $\hat{o}$. Deviations from the ground truth are recorded. A common approach would be to calculate the squared error $e = (o - \hat{o})^2$.

b.) In the second phase the current-error e is put in context with previously recorded error-terms. This is often done via parametric [Mal+15; Ahm+17; Guo+16; Mal+16; Shi+17; CV15] or non-parametric statistical tests [ZL17; Hun+18; Mai+18; Su+19]. For details confer [Mar20].

Phase b.) is not part of most forecasting algorithms. It has therefore been implemented by the author. Nonparametric Dynamic Thresholding (NDT) by Hundman et al. [Hun+18] was able to outperform other approaches and was therefore used throughout this paper. For details see appendix A.

Definition 2.9 (Boundary-Based-Algorithm)**.** Boundary-Based algorithms consist of only a single major phase: they try to calculate some boundary to which most observations conform. Observations beyond that boundary are deemed anomalous.

Definition 2.10 (Detection)**.** An observation that is deemed anomalous by the algorithm is called a *detection*.

3 Experimental Setup

In this section, details on the dataset section 3.1 and the selection of algorithms section 3.2 are given.

3.1 Datasets

The algorithms evaluated in this paper are compared on a variety of synthetic and real-world univariate time series, taken from the NAB-collection. While the NAB datasets were chosen for this paper, a selection of additional univariate datasets is given in table 2. Additional multivariate datasets are given in table 3.

In this subsection, the recent controversy surrounding several anomaly detection datasets will be discussed. Further, NAB and its metric will be introduced, also a short visual examination is performed.

3.1.1 Controversy

Recent research inspired by [Nak+20] has raised doubts about the adequacy of multiple anomaly detection datasets [RE20]. Prominently included are Yahoo!S5, NAB, NASA [Hun+18], and OMNI [Su+19] datasets.

It is objected that these datasets suffer from one or multiple of the following flaws [RE20]:

Triviality A dataset is considered trivial if it can be *solved* by a simple combination of statistical operations such as *mean*, *max*, *std*, *diff*.

Examples that demonstrate this flaw are taken mainly from the Yahoo!S5 dataset for which also a code listing of a brute force approach is provided. Three time series from OMNI are examined and only a single one from NAB.

For NASA, Renjie and Eamonn [RE20] claim that about 90% of included anomalies are trivial and show > 10 examples. For OMNI, 50% are claimed to be trial, but only three examples are given. For NAB, the authors claim that *most* time series are trivial, but omit evidence by giving only a single example.

Unrealistic Density A dataset is considered unrealistically dense, if either a.) a large portion (e.g. > 0.33) of observations are labeled anomalous, b.) many (e.g. > 5) proximal regions are marked as separate anomalies, c.) separate anomalies are proximal (e.g. *sandwiching* a single normal observation) [cf. RE20].

Examples that demonstrate this flaw are taken from Yahoo!S5 and NASA.

Mislabeled Ground Truth A dataset is mislabeled if its labels contain any false negatives or false positives.

Examples that demonstrate this flaw are taken mainly from Yahoo!S5. A single time series from NAB is given.

Run-to-failure Bias A dataset suffers from the run-to-failure bias if anomalies appear only towards its end without normal observations following them.

Examples that demonstrate this flaw are taken from Yahoo!S5 and NASA.

In summary, the majority of flaws are to be found within Yahoo!S5 and NASA datasets. From [RE20] it can be concluded that **Anomaly detection is a visual domain and researchers are responsible for providing not just scores, but visual examination of their results.** While the authors do not to provide evidence for the triviality of NAB, this conclusion holds here as well.

As a side note, Renjie and Eamonn [RE20] mention the creation of a novel benchmark dataset. Unfortunately, its release is still delayed.

3.1.2 Numenta Anomaly Benchmark

The Numenta Anomaly Benchmark (NAB) was created as a compilation of 11 synthetic and 47 real, domain specific time series. Of the 11 synthetic time series, 5 do not contain anomalous values. The time series consist of both a timestamp and a single numeric value (see Observation (definition 2.1)) and contain between 1000–22.000 such observations. The complete benchmark contains a total of 365.551 observations [LA15].

The time series (except for the *realKnownCause*-subset) were labeled according to a unified procedure [Num15] by a team of multiple researchers. It was released in October 2015, as a reaction to the lack of publicly available benchmarks for univariate anomaly detection [LA15] (although by that time the Yahoo!S5 benchmark had already been released).

The upcoming paragraph describes the scoring function introduced with NAB. This is followed (in vein of [RE20]) by a short visual examination of the dataset.

Numenta Anomaly Metric

Numenta [LA15] defines a custom scoring function. They choose to do so rather than using F-Score [Mur12, p. 183] or similar well-known metric, because "traditional scoring methods [...] do not incorporate time and do not reward early detection [and are therefore] not applicable to [real-time anomaly detection]" [LA15].

The metric is defined using *anomaly windows*.

Definition 3.1 (Anomaly Window)**.** An anomaly window is a set of two timestamps (in other words a *timerange*), centered around an anomalous point [cf. LA15]. All points inside this window are considered anomalous. An example of this is shown in fig. 2. Window length is defined per time series as:

$$\frac{\text{len(time series)} * 0.1}{\text{number of anomalies} \in \text{time series}}$$

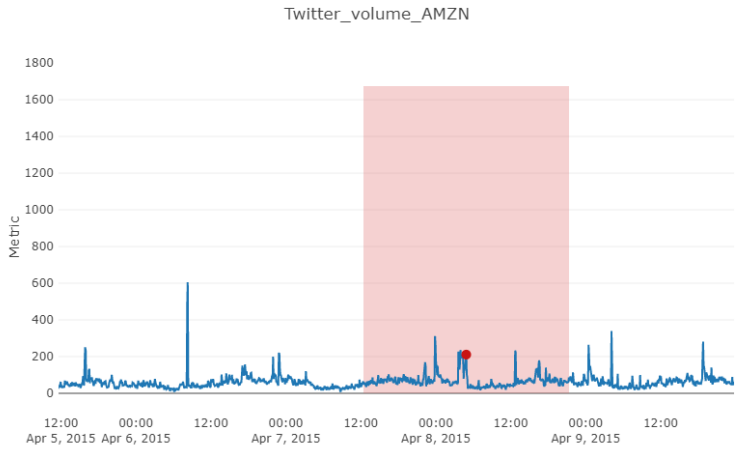


Figure 2: Example of an Anomaly Window (definition 3.1). Illustration by the author.

If multiple detections exist within a window, only the first detection is scored. Additional detections within the window are ignored [cf. LA15]. Detections earlier in the window give higher positive scores than later detections. Detections outside the anomaly window give negative scores.

Details on calculation of the scoring function are given in appendix B and an exemplary application is depicted in fig. 6.

Visual Examination

In this paragraph, an overview of the most common anomaly types within NAB is given. From visual observation only, the vast majority (about 30) of time series from the dataset contain point anomalies fig. 3. Other more common types include contextual anomalies fig. 4. Collective anomalies

only appear in form of change points. A very good example for collective anomalies however, as the one from fig. 5, is not found within NAB.

Regarding triviality allegations from Lavin and Ahmad [LA15], $\sim 1/3$ of point anomalies are so far from their neighborhood, that $\mu_t + 3 * \sigma_t$ (μ_t, σ_t are the moving mean and the moving standard deviation respectively) would suffice to detect them, e.g. fig. 3. This implies triviality of the dataset. However, such trivial solution would also detect many false positives. While, from short examination, it appears possible to solve many of the time series presented in NAB with comparatively simple statistical operations, such a solution would probably require a unique set of such operations for every time series. Thereby defeating the purpose of NAB: scoring unified algorithms that must *not* be adjusted for every such time series.

Additional plots from the dataset can be found in appendix C.

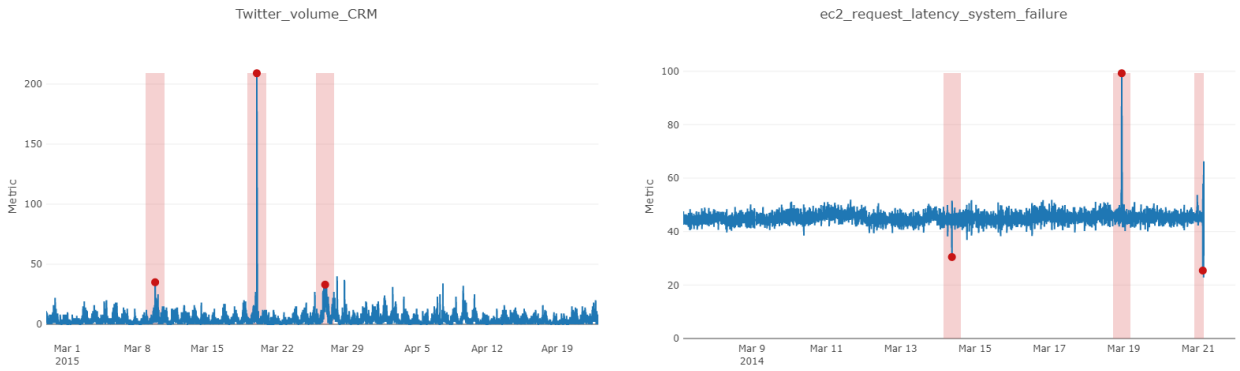


Figure 3: Two examples of Point Anomalies. Illustrations by the author.

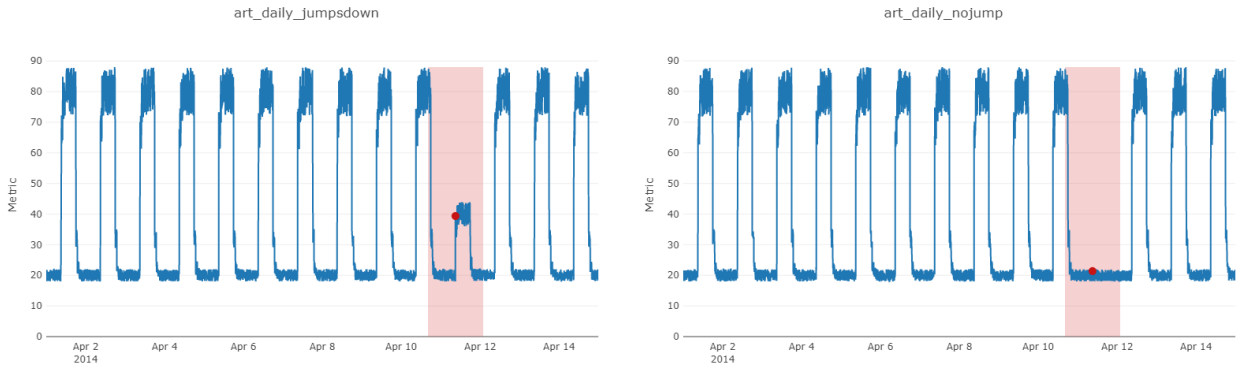


Figure 4: Two examples of (artificial) cyclicity violations (contextual anomalies). Illustrations by the author.

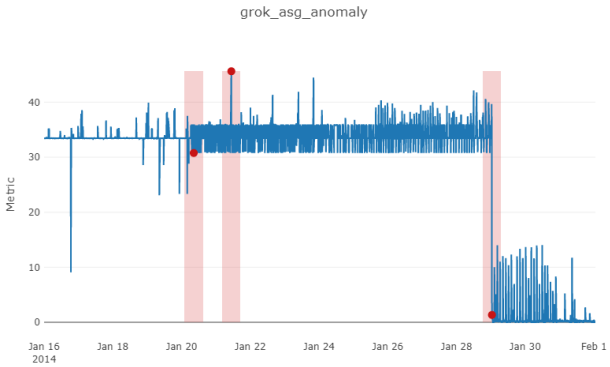


Figure 5: An example of two change points (collective anomalies) and a point anomaly. Illustrations by the author.

3.2 Algorithm overview

In an attempt to reduce the implementation complexity required for this paper, the majority of algorithms has been adopted from open source libraries and adjusted slightly for application to the NAB-dataset.[1] On GitHub, several repositories can be found which curate lists of such libraries. All repositories used for this process are listed in table 4.

From these repositories, all libraries that offer algorithms belonging to either definition 2.8 (Forecasting-Based-Algorithm) or definition 2.9 (Boundary-Based-Algorithm) are listed in the appendix (forecasting-based: table 5, boundary-based: table 6). Both tables contain three columns: an (almost exhaustive) list of models offered by the library, the latest released version, and the date of the latest commit.

Then, a subset of ten libraries is chosen. Five of them with a focus on forecasting and the remaining five with a focus on boundaries. Libraries that are found within another library are dropped from the list (e.g. Prophet, GluonTS $\in$ AtsPy). Finally, a set of libraries is picked by the author, whereby recently updated libraries, libraries with sufficient documentation and libraries with a high number of stars (measuring to an extent the appreciation of the community) are slightly preferred. The chosen libraries are presented in table 7.

From these ten libraries, a subset of algorithms is selected. The selected forecasting-algorithms are shown in table 8, boundary-algorithms are shown in table 9.

If not stated otherwise, default parameters from within the libraries are adopted.

4 Results and Discussion

In this section we will examine the results accomplished by every algorithm (section 4.1) and close with a summary and discussion (section 4.2). The final metrics are presented in table 1.

4.1 Brief discussion of result per algorithm

In this subsection, the results from every algorithm are briefly discussed.

Inapplicable Algorithms

An algorithm is considered inapplicable, if it produces results that are hardly usable for anomaly detection.

TBAT/S suffered from frequent crashes due to numerical instabilities. In conjunction with its long runtime and subpar performance it was sorted out.

[1] Customization most notably included the partition of time series into multiple subsets that were alternatingly used for training and inference.

OCSVM OCSVMs output is predominantly noisy. In fig. 15, two examples (out of many) are shown, in which OCSVM was unable to extract meaningful insights from the data. Its output might be improved by adapting the method from e.g. Gómez-Verdejo, Arenas-Garcázaro-Gredilla and Navia-Vázquez [GAN11]. However, this is beyond the scope of this paper.

LOF produced only a low number of detections, most of which (89) were false positives that (visually) seem unreasonable (see fig. 16). Interestingly, it proved largely resistant to high value spikes. For real-world use however, the number of false negatives is too high.

DAGMM was able to produce 32 true positive detections, which however seem likely coincidental (see fig. 17), making the algorithm inadequate.

RRCF While in simple, consistent time series, RRCF is able to adept to and detect violations of cyclicity, complex data appears to confuse the method and therefore makes it inadequate in large part. Although grid-search has been performed for hyperparameter optimization, it is possible that results could be improved in future research.

Forecasting Algorithms

DeepAnT In simple cases, DeepAnT is able to learn cyclicity of the data, and thereby uncover cyclicity violation (see fig. 10a). In more complex cases however, its output error-terms e more closely *reconstruct* the time series[2] (see fig. 10b). Neither increasing the history window (that is used to predict the upcoming observation) nor increasing the number of training iterations were able to alleviate the problem. An interesting exception from this are change points, where predictions slowly adapt to the new mean of observations (see fig. 10c).

N-BEATS Whether N-BEATS is able to infer cyclicity depends on the type of layers chosen [cf. Ore+20]. From experiments, generic layers were chosen, which are (in general) unable to infer cyclicity (see fig. 11a). For more complex time series, N-BEATS' error-terms also resemble[2] the original time series data (see fig. 11b). Interestingly, value spikes are followed by a short decrease in prediction accuracy (see fig. 11c). The output is therefore often more chaotic than that of DeepAnT. While it may seem counterintuitive that N-BEATS achieves higher scores than DeepAnT, this is related to three conditions: a.) as noted, value spikes cause more chaotic outputs, b.) many anomalies within NAB are point anomalies with high value spikes, and c.) NDT [cf. Hun+18] (used in phase b. of forecasting algorithms, see definition 2.8) is sensitive to relative changes in mean and standard deviation.

We therefore observe more detections in total which are also more likely to be true positives, because NAB contains many such anomalies.

Unfortunately, this comes at the cost of increasing the number of false positives significantly.

[2]When error-terms seem to reconstruct the input time series, this implies that the method is unable to make valuable predictions beyond e.g. inferring the mean.

Holt-Winters Holt-Winters proved to be even more unstable than N-BEATS, resulting in large prediction spikes (see fig. 12a). Predictions also often included some trend that was not to be found within the actual data (see fig. 12b). As Holt-Winters was adopted from AtsPy, it is possible that numerical instabilities resulted from an issue within the library.

Auto-ARIMA Like N-BEATS, Auto-ARIMA is unable to detect long-term cyclicity (see fig. 13b), also resulting in high number of false positive on time series with many spikes (see fig. 13a). This might be improved by adding a seasonal component to ARIMA [cf. Box+16]. Unfortunately, the seasonal ARIMA model proved computationally demanding for the task of real-time anomaly detection.

On more simple time series, ARIMA is able to detect even subtle irregularities (see fig. 13c). However, this is also a downside, because (from observation) real world time series contain many such irregularities that are *not* considered anomalous.

On more chaotic examples the algorithms error-terms (again) more closely resemble[2] the input time series (see fig. 13d).

As a side note to the interested reader, predictions were generated as in-sample 1-step-ahead forecasts using `predict_in_sample(*args, **kwargs)` [Smi+17]. Out-sample n-step-ahead forecasts produced worse results for the task.

Prophet While Prophet scored highest from the forecasting algorithms, it also produced the highest number of false positives. Although a large number (57) of them can be attributed to only a single time series (see fig. 14a), Prophet still did not perform well. Similar to other forecasting algorithms for complex examples, Prophets error-terms reproduced[2] the original time series — yet with a quirk: Prophet assumed seasonality within the data where seasonality was absent. This resulted in a curvy output (see fig. 14b). Besides that, Prophet was able to understand cyclicity in *simple* time series very well (see fig. 14c).

Boundary Algorithms

Nonparametric Dynamic Thresholding [Hun+18] Although not considered as a stand-alone method from the beginning, NDT proved to be very effective at extracting anomalies from forecasting algorithms.

As often observed in this section, the output of most forecasting algorithms closely resembles the shape of the ground truth time series.

Therefore, NDT was added to the set of methods.

NDT tries to detect values in a dataset that cause the greatest change in mean and standard deviation when they are removed. In theory, it should therefore be able to identify regions not only with a high/low max/min but also with unusually high/low mean. From visual evaluation, NDT reacts most strongly to value spikes. In some time series, this leads to a high number of false positives (see fig. 19a).

Unfortunately, the dataset yields not a single anomaly that is detected by NDT obviously due to its unusual mean.

LSTM-AD Like most algorithms discussed so far, LSTM-AD is vulnerable to time series that produce high value spikes and then jump back to approximately zero (see fig. 20a). It can be observed that LSTM-AD is unable to detect cyclicity violations (see fig. 20b). Besides that, all detections from LSTM-AD are closely related to some spiking change in value.

CBLOF From visual examination, CBLOFs results are surprisingly good. It was able to adapt to most regularly spiking time series (see fig. 21b), albeit having trouble with *ir*regularly spiking time series (see fig. 21a). Most of the time however, good results are achieved (see e.g. fig. 21c).

kNN was slightly more robust to value spikes, producing less false positives but also less true positives (see fig. 23a). Although kNN has no notion of time, it was able to detect several (simple) cyclicity violations (see fig. 23b). Presumably, because of their unusual value ranges. Besides that, kNN was also able to detect some of the more challenging anomalies (see fig. 23c).

Threshold Detector The threshold detector was originally a **cheat** within the HTM-code[3] by Numenta [LA15; Ahm+17]. It exploits the many point anomalies (spiking values) within NAB to achieve higher scores.

The detector works by remembering both the highest/lowest observed values. If a new point with higher/lower value is observed (and it is larger/smaller by some margin), an anomaly is reported.

This simple method was able to achieve (by far) the highest F1-Score (see table 1) due to its very low false positive rate. Representative examples are given in fig. 24.

LSTM-ED From visual examination, results look very similar to those achieved from CBLOF. LSTM-ED suffers similarly from irregular spike values (see fig. 22a). LSTM-ED is able to improve upon CBLOF by detecting some of the more intricate anomalies (see fig. 22b). The higher number of false positives from LSTM-ED can be linked to the set of artificial time series data in NAB that does not contain any anomalies. There, LSTM-ED produced 21 anomalies more than CBLOF (see fig. 22c).

Autoencoder The output of the AE was often identical to that of LSTM-ED. The two algorithms differed only on about three of the 58 time series (see fig. 26).

Skyline Skyline was unable to detect cyclicity dropouts (see fig. 25a). It is robust to frequently spike values, as long as the spikes are so common, that they are within a $\mu_t + 3 \times \sigma_t$, where μ_t, σ_t are the moving mean and the moving standard deviation respectively. If observations deviate more (however regular that may be), a detection is recorded. This produced a significant number of false positives in some domains (see fig. 25b).

[3] https://github.com/numenta/NAB/blob/757b586b35a939645dd92f6f7c1b126ebb3614f7/nab/detectors/numenta/numenta_detector.py

Numenta HTM[4] was able to score highest out of all algorithms, and slightly higher than Skyline, because it produced less false positives. In simple time series, HTM was able to detect very simple cyclicity violations (see figs. 27a and 27b). Also, HTM was more sensitive to small deviations than most competitors, enabling it to detect more subtle (see fig. 27c) anomalies, but also producing a slightly larger number of false positives. The algorithm was not robust to datasets where spikes were strong but regular (see fig. 27e).

Boundary Algorithms	Standard Profile	Punish high FP	F1	TP	FP	FN
Numenta HTM [Ahm+17]	59.3	21.9	0.47	82	153	34
Skyline	57.8	-6.4	0.37	87	273	29
AE [GBC16, pp. 499 sqq.]	53.5	22.9	0.47	76	129	40
LSTM-ED [Mal+16]	51.2	18.3	0.45	74	141	42
Numenta Threshold Detector [Ahm+17]	50.1	44.9	0.65	65	19	51
kNN [Mur12, pp. 16 sqq.]	47.9	17.6	0.44	68	123	48
CBLOF [HXD03]	47.7	19.6	0.46	68	114	48
NDT [Hun+18]	46.2	9.69	0.42	68	140	48
LSTM-AD [Mal+15]	36.3	17	0.42	51	78	65
Robust Random Cut Forest [BMT19]	34.3	-14	0.29	54	207	62
DAGMM[Zon+18]	17.9	-17.1	0.22	32	147	84
LOF [Bre+00]	14.2	-7.9	0.21	24	89	91
OCSVM [Sch+99; TD04]	1.2	-0.5	0.02	1	5	115
Forecasting Algorithm						
Prophet [TL17]	48.0	-26.5	0.3	78	319	38
Auto-ARIMA [Smi+17]	47.6	-19.7	0.32	77	287	39
Holt-Winters (additive model) [Win60]	46.2	-18.2	0.31	74	281	42
N-BEATS [Ore+20]	44.7	-13.6	0.31	71	272	45
DeepAnT [Mun+19]	36.9	-2.5	0.34	57	166	59
TBAT/S	/	/	/	/	/	/

Table 1: (left) overview of scores achieved by two NAB-metric profiles. (right) F1-Score [Mur12, p. 183], true positives, false positives and false negatives respectively.

4.2 Discussion

Visual evaluation of algorithms on NAB has proven harder than expected, because

a.) the dataset comprises mainly point anomalies,

b.) most time series are of very similar shape,

c.) few time series contain easily digestible cyclicity,

[4]HTM was implemented without the Threshold Detector cheat by Numenta.

d.) many time series contain too few examples of their cyclicity for it to be learnt,

e.) anomalies are sometimes very similar to normal data,

f.) the dataset was not compiled to *answer such questions*.

While Renjie and Eamonn [RE20] are currently constructing a supposedly better benchmark dataset, it remains to be seen, whether it will improve visual interpretability and comparability of approaches. The examples currently available [RE21] unfortunately do not suggest that, coming mostly from very steady biological processes, characterized by strong spikes and no long term cyclicities.

As a consequence, drawing a clear conclusion and giving advices when to use which algorithm is difficult.

Based on the observations above (section 4.1), we conclude for forecasting methods:

1. Forecasting methods (DeepAnT, Prophet) perform good on simple time series with obvious cyclicities, where they are able to infer dropouts. (figs. 10a and 14c)
2. Forecasting methods break down on more complex time series, where their error terms only approximate the original time series — often with additionally introduced noise, significantly increasing their false positive rate. (figs. 10b, 13d and 14b)

And for boundary methods:

1. Boundary methods are often unable to detect dropouts, unless they lie within previously unpopulated value ranges (figs. 23b and 27a).
2. Boundary methods are often sensitive to point anomalies and sometimes sensitive to contextual anomalies, unless they depend on cyclicity.
3. From all methods HTM and Skyline were the most sensitive, where HTM produced 120 false positives less than Skyline.
4. LSTM-AD produced the least false positives, however, all true positives were related to value spikes.
5. The vulnerability (producing false positives) varies between time series. Even between time series that *look* very similar.
6. Most algorithms are vulnerable (producing false positives) to the same time series.

5 Conclusion

In this paper, a larger number of diverse algorithms was applied to the Numenta Numenta Anomaly Benchmark (NAB). Algorithms were divided into two families, *forecasting*- and *boundary*-based algorithms. It was observed, that forecasting-based algorithms were often unable to handle complex time series, for which their error-terms merely reconstructed the input time series. On more simple time series, forecasting-based methods were able to infer cyclicity and detect anomalies regarding it. Boundary-based algorithms, on the other hand, were (usually) unable to infer cyclicity but were often more robust to complex time series. However, they mostly detected anomalies revolving around a value spike (either definition 2.5 (Point Anomaly) or definition 2.6 (Contextual Anomaly)) and were often insensitive to more complex anomalies. Especially value dropouts were very hard to detect. Also, small value fluctuations were often overlooked. For such fluctuations, discord-detection algorithms would presumably improve results [cf. Nak+20].

Regarding the dataset, time series from NAB are often unable to answer specific questions on the behavior of algorithms. A new supposedly improved dataset is currently under construction by Renjie and Eamonn [RE20]. From experience however, visual evaluation lives by building, confirming, and destroying hypotheses. A static dataset might not be able to provide such versatility.

And so unfortunately, the task of evaluating the applicability of any algorithm remains the task of the practitioner.

5.1 Future Work

Future work derived from this paper is rather vague.

Obviously, one could evaluate additional algorithms. The author would have also liked to examine e.g. Luminol [Lin15], MERLIN [RE20], and the forecasting package GluonTS [Ale+20] — but compatibility issues were faced that could not have been solved within time.

A more important contribution however would be the clear definition of an extended evaluation process. This process might build upon a more diverse dataset (hopefully [RE20]) and require the generation of new time series to verify and fence off claims about discussed methods.

6 Bibliography

[Ahm+17] Subutai Ahmad et al. 'Unsupervised real-time anomaly detection for streaming data'. In: *Neurocomputing* 262 (2017), pp. 134–147.

[Ale+20] Alexander Alexandrov et al. 'GluonTS: Probabilistic and Neural Time Series Modeling in Python'. In: *J. Mach. Learn. Res.* 21 (2020), 116:1–116:6. URL: http://jmlr.org/papers/v21/19-820.html.

[BMT19] Matthew Bartos, Abhiram Mullapudi and Sara C. Troutman. 'rrcf: Implementation of the Robust Random Cut Forest algorithm for anomaly detection on streams'. In: *The Journal of Open Source Software* 4.35 (2019), p. 1336.

[Box+16] George E. P. Box et al. *Time series analysis: Forecasting and control*. 5th ed. Wiley Series in Probability and Statistics. Hoboken: Wiley, 2016. ISBN: 1118674928.

[Bre+00] Markus M. Breunig et al. 'LOF: Identifying Density-Based Local Outliers'. In: *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA*. Ed. by Weidong Chen, Jeffrey F. Naughton and Philip A. Bernstein. ACM, 2000, pp. 93–104.

[BW20] Mohammad Braei and Sebastian Wagner. 'Anomaly Detection in Univariate Time-series: A Survey on the State-of-the-Art'. In: *CoRR* abs/2004.00433 (2020).

[CBK09] Varun Chandola, Arindam Banerjee and Vipin Kumar. 'Anomaly Detection: A survey'. In: *ACM computing surveys (CSUR)* 41.3 (2009), pp. 1–58.

[CV15] Sucheta Chauhan and Lovekesh Vig. 'Anomaly detection in ECG time signals via deep long short-term memory networks'. In: *2015 IEEE International Conference on Data Science and Advanced Analytics, DSAA 2015, Campus des Cordeliers, Paris, France, October 19-21, 2015*. IEEE, 2015, pp. 1–7.

[Du+17] Min Du et al. 'DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning'. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*. Ed. by Bhavani M. Thuraisingham et al. ACM, 2017, pp. 1285–1298.

[Fal+12] Michael Falk et al. *A First Course on Time Series Analysis: Examples with SAS*. Berlin: epubli GmbH, 2012. ISBN: 3844228454.

[FLV16] Pavel Filonov, Andrey Lavrentyev and Artem Vorontsov. 'Multivariate Industrial Time Series with Cyber-Attack Simulation: Fault Detection Using an LSTM-based Predictive Data Model'. In: *CoRR* abs/1612.06676 (2016).

[FSG17] Valentin Flunkert, David Salinas and Jan Gasthaus. 'DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks'. In: *CoRR* abs/1704.04110 (2017).

[GAN11] Vanessa Gómez-Verdejo, Jerónimo Arenas-Garcázaro-Gredilla and Ángel Navia-Vázquez. 'Adaptive One-Class Support Vector Machine'. In: *IEEE Trans. Signal Process.* 59.6 (2011), pp. 2975–2981.

[GBC16] Ian J. Goodfellow, Yoshua Bengio and Aaron Courville. *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[Guh+16] Sudipto Guha et al. 'Robust Random Cut Forest Based Anomaly Detection on Streams'. In: *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. Ed. by Maria-Florina Balcan and Kilian Q. Weinberger. Vol. 48. JMLR Workshop and Conference Proceedings. JMLR.org, 2016, pp. 2712–2721. URL: http://proceedings.mlr.press/v48/guha16.html.

[Guo+16] Tian Guo et al. 'Robust Online Time Series Prediction with Recurrent Neural Networks'. In: *2016 IEEE International Conference on Data Science and Advanced Analytics, DSAA 2016, Montreal, QC, Canada,October 17-19, 2016*. IEEE, 2016, pp. 816–825.

[Hot90] Harold Hotelling. 'Stability in competition'. In: *The Collected Economics Articles of Harold Hotelling*. Springer, 1990, pp. 50–63.

[Hun+18] Kyle Hundman et al. 'Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding'. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*. Ed. by Yike Guo and Faisal Farooq. ACM, 2018, pp. 387–395.

[HXD03] Zengyou He, Xiaofei Xu and Shengchun Deng. 'Discovering cluster-based local outliers'. In: *Pattern Recognition Letters* 24.9 (2003), pp. 1641–1650. ISSN: 0167-8655.

[LA15] Alexander Lavin and Subutai Ahmad. 'Evaluating Real-Time Anomaly Detection Algorithms - The Numenta Anomaly Benchmark'. In: *14th IEEE International Conference on Machine Learning and Applications, ICMLA 2015, Miami, FL, USA, December 9-11, 2015*. Ed. by Tao Li et al. IEEE, 2015, pp. 38–44.

[Lin15] LinkedIn. *linkedin/luminol*. 2015. URL: https://github.com/linkedin/luminol (visited on 18/01/2021).

[Mai+18] Lorenzo Fernández Maimó et al. 'A Self-Adaptive Deep Learning-Based System for Anomaly Detection in 5G Networks'. In: *IEEE Access* 6 (2018), pp. 7700–7712.

[Mal+15] Pankaj Malhotra et al. 'Long Short Term Memory Networks for Anomaly Detection in Time Series'. In: *23rd European Symposium on Artificial Neural Networks, ESANN 2015, Bruges, Belgium, April 22-24, 2015*. 2015. URL: http://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2015-56.pdf.

[Mal+16] Pankaj Malhotra et al. 'LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection'. In: *CoRR* abs/1607.00148 (2016).

[Mar20] Martrikelnummer: 8323. *Survey: Deep Learning for Semi-Supervised and Unsupervised Anomaly Detection on Natural Language Server Logs*. 2020.

[Mun+19] Mohsin Munir et al. 'DeepAnT: A Deep Learning Approach for Unsupervised Anomaly Detection in Time Series'. In: *IEEE Access* 7 (2019), pp. 1991–2005.

[Mur12] Kevin P. Murphy. *Machine learning - a probabilistic perspective*. Adaptive computation and machine learning series. MIT Press, 2012. ISBN: 0262018020.

[Nak+20] Takaaki Nakamura et al. 'MERLIN: Parameter-Free Discovery of Arbitrary Length Anomalies in Massive Time Series Archives'. In: *Proc. 20th IEEE Intl. Conf. Data Mining*. 2020.

[Num15] Numenta. *Anomaly Labeling Instructions*. 2015. (Visited on 08/01/2021).

[Ore+20] Boris N. Oreshkin et al. 'N-BEATS: Neural basis expansion analysis for interpretable time series forecasting'. In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL: https://openreview.net/forum?id=r1ecqn4YwB.

[Ran+18] Syama Sundar Rangapuram et al. 'Deep State Space Models for Time Series Forecasting'. In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*. Ed. by Samy Bengio et al. 2018, pp. 7796–7805. URL: http://papers.nips.cc/paper/8004-deep-state-space-models-for-time-series-forecasting.

[RE20] Wu Renjie and J. Keogh Eamonn. 'Current Time Series Anomaly Detection Benchmarks are Flawed and are Creating the Illusion of Progress'. In: *CoRR* abs/2009.13807 (2020).

[RE21] Wu Renjie and J. Keogh Eamonn. *UCR Time Series Anomaly Contest: Call for Datasets*. 2021. URL: https://www.dropbox.com/sh/gsvm653d0m8tk41/AAD4Swf8d0mKSmGfsdLTh0gia/AnomalydetectionfortimeseriescontestCallfordatasets.pdf (visited on 17/01/2021).

[Sch+99] Bernhard Schölkopf et al. 'Support Vector Method for Novelty Detection'. In: *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*. Ed. by Sara A. Solla, Todd K. Leen and Klaus-Robert Müller. The MIT Press, 1999, pp. 582–588.

[Shi+17] Dominique T. Shipmon et al. 'Time Series Anomaly Detection; Detection of anomalous drops with limited features and sparse examples in noisy highly periodic data'. In: *CoRR* abs/1708.03665 (2017).

[Smi+17] Taylor G. Smith et al. *pmdarima: ARIMA estimators for Python*. 2017. URL: http://www.alkaline-ml.com/pmdarima.

[Su+19] Ya Su et al. 'Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network'. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*. Ed. by Ankur Teredesai et al. ACM, 2019, pp. 2828–2837.

[TD04] David M. J. Tax and Robert P. W. Duin. 'Support Vector Data Description'. In: *Mach. Learn.* 54.1 (2004), pp. 45–66.

[The18] The Open Group. 'IEEE Standard for Information Technology-Portable Operating System Interface (POSIX(TM)) Base Specifications, Issue 7'. In: *IEEE Std 1003.1-2017 (Revision of IEEE Std 1003.1-2008)* (2018).

[TL17] Sean J. Taylor and Benjamin Letham. 'Forecasting at Scale'. In: *PeerJ Prepr* 5 (2017), e3190.

[Wan+19] Yuyang Wang et al. 'Deep Factors for Forecasting'. In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, 2019, pp. 6607–6617. URL: http://proceedings.mlr.press/v97/wang19k.html.

[Wei+05] Li Wei et al. 'Assumption-Free Anomaly Detection in Time Series'. In: *17th International Conference on Scientific and Statistical Database Management, SSDBM 2005, 27-29 June 2005, University of California, Santa Barbara, CA, USA, Proceedings*. Ed. by James Frew. 2005, pp. 237–240.

[Win60] Peter R. Winters. 'Forecasting Sales by Exponentially Weighted Moving Averages'. In: *Manage. Sci.* 6.3 (1960), pp. 324–342. ISSN: 0025-1909.

[Xu+18] Haowen Xu et al. 'Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications'. In: *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*. Ed. by Pierre-Antoine Champin et al. ACM, 2018, pp. 187–196.

[ZL17] Lingxue Zhu and Nikolay Laptev. 'Deep and Confident Prediction for Time Series at Uber'. In: *2017 IEEE International Conference on Data Mining Workshops, ICDM Workshops 2017, New Orleans, LA, USA, November 18-21, 2017*. Ed. by Raju Gottumukkala et al. IEEE Computer Society, 2017, pp. 103–110.

[Zon+18] Bo Zong et al. 'Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection'. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL: https://openreview.net/forum?id=BJJLHbb0-.

A Details on phase b. of forecasting-based algorithms

As noted in section 2.3, the evaluation of forecasting errors is not part of (most) available forecasting packages. Therefore, a variety of approaches was adopted for this paper. Most trivially, a moving mean and standard deviation were used as a dynamic threshold. Every forecasting error exceeding this threshold was remarked as a detection (definition 2.10). Both 1. using an OCSVM and 2. identifying the best fitting distribution, then performing p-tests was disregarded for every forecasting error were disregarded for their computational complexity and subpar results.

More complex scoring methods from [Ahm+17; Hun+18] were adopted. While the method from Ahmad et al. [Ahm+17] is illustrated in this section, non-parametric thresholding from Hundman et al. [Hun+18] was able to outperform it in every application and was therefore used to produce the final results. For details please review [Hun+18].

Ahmad et al. [Ahm+17] suggest modelling the distribution of errors as a rolling normal distribution with mean and standard deviation μ_t, σ_t according to eqs. (6) and (7)

$$\mu_t = \frac{\sum_{i=0}^{i=W-1} s_{t-i}}{W} \tag{6}$$

$$\sigma_t = \frac{\sum_{i=0}^{i=W-1} (s_{t-i} - \mu_t)^2}{W-1} \tag{7}$$

where:

W := Window size, set to $W = 8000$ as proposed by [Ahm+17]

and a short term rolling average $\tilde{\mu}_t$ according to eq. (8).

$$\tilde{\mu}_t = \frac{\sum_{i=0}^{i=W'-1} s_{t-i}}{W'} \tag{8}$$

where:

W' := Short-term window size, set to $W' = 10$ as proposed by [Ahm+17]

The likelihood (eq. (9)) is calculated using the Cumulative Density Function (CDF) of a normal gaussian distribution given in eq. (10) as

$$L_t = \text{CDF}\left(\frac{\tilde{\mu}_t - \mu_t}{\sigma_t}\right) \tag{9}$$

$$\text{CDF}(x) = \frac{1}{2}\left(1 + \text{erf}(\frac{x-\mu}{\sqrt{2\sigma}})\right) \tag{10}$$

where:

$\mu \quad = 0$
$\sigma \quad = 1$
erf := the gaussian error function

Finally, a detection is recorded, when $L_t \geq \epsilon \vee L_t \leq 1 - \epsilon$. This is a slight deviation from [Ahm+17], where only an upper p-test is performed.

B Calculation of the Numenta Anonamly Metric Score

The Numenta Anomaly Metric Score for a single time series is the addition of

a.) the sum of scores for every Detection (definition 2.10) (given by eq. (11)) and

b.) the score for missed anomaly windows (false negatives) (given by eq. (12)).

$$\sigma(t) = (W_{TP} - W_{FP}) \left(\frac{1}{1 + e^{5t}} \right) - 1 \tag{11}$$

$$\phi(F) = W_{FN} * |F| \tag{12}$$

$$\text{Score} = \left(\sum_{t \in T} \sigma(t) \right) + \phi(F) \tag{13}$$

where:

t	:= the relative timestamp/position of the detection within the window.
$W_{\{TP,FP,FN\}}$	:= the weights attributed to true and false positives and false negatives respectively.
F	:= the set of false positives (anomaly windows without any detections).
T	:= the set of time stamps associated with the detections.

The final score per time series is calculated by eq. (13) [cf LA15]. An example of the scoring function can be seen in fig. 6.

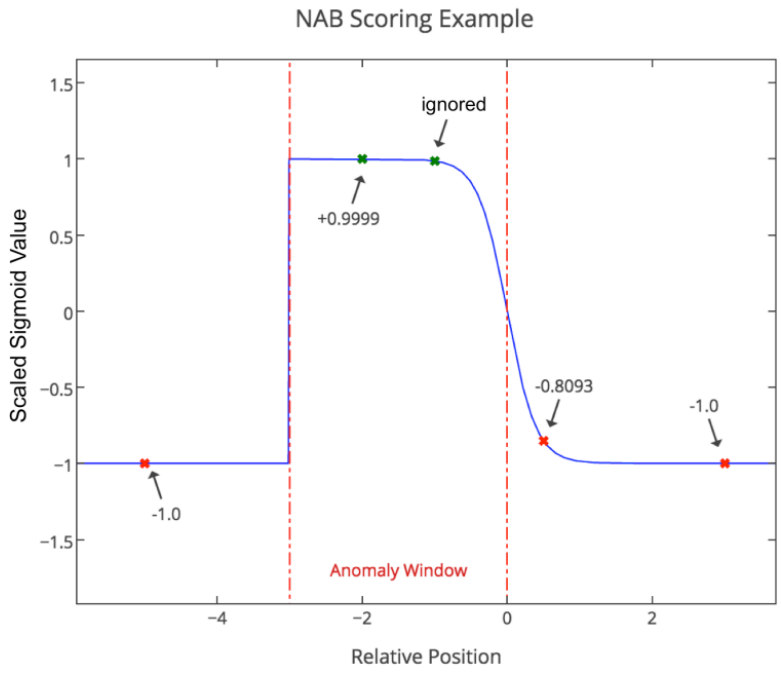
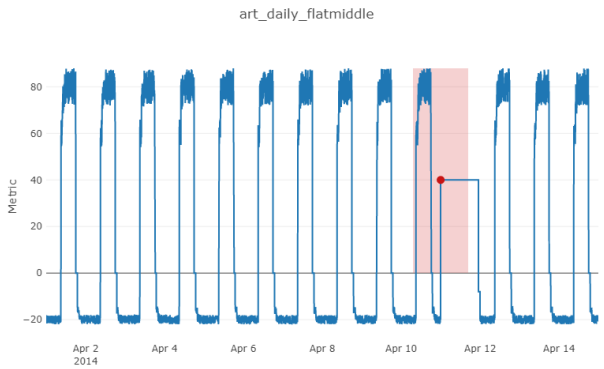


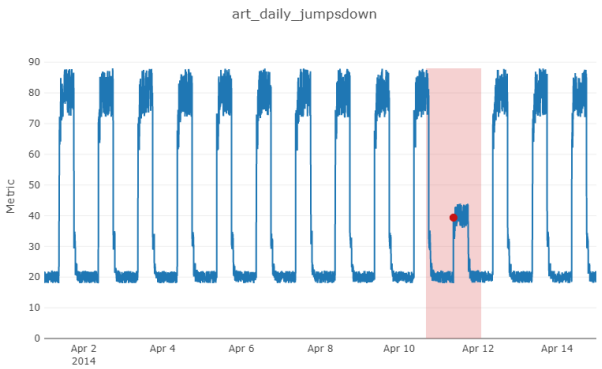
Figure 6: Example application of the anomaly scoring function. Illustration adopted from [LA15]. The following text (blue) is cited from Lavin and Ahmad [LA15]: Scoring example for a sample anomaly window, where the values represent the scaled sigmoid function, the second term in eq. (11). The first point is an FP preceding the anomaly window (red dashed lines) and contributes -1.0 to the score. Within the window we see two detections, and only count the earliest TP for the score. There are two FPs after the window. The first is less detrimental because it is close to the window, and the second yields -1.0 because it's too far after the window to be associated with the true anomaly. TNs make no score contributions. The scaled sigmoid values are multiplied by the relevant application profile weight, as shown in Eq., the NAB score for this example would calculate as: $-1.0W_{FP} + 0.9999W_{TP} - 0.8093W_{FP} - 1.0W_{FP}$. With the standard application profile this would result in a total score of 0.6909.

C Additional Plots of the Datasets

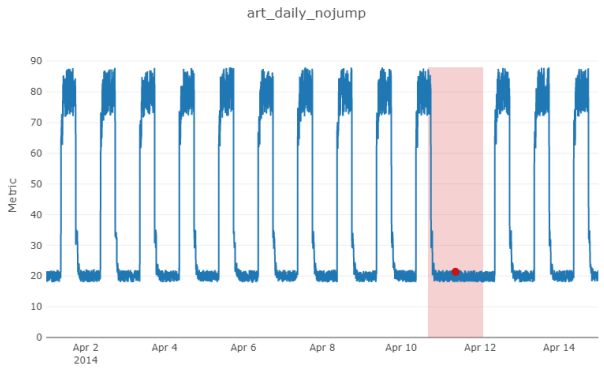
This section provides additional time series plots from NAB.



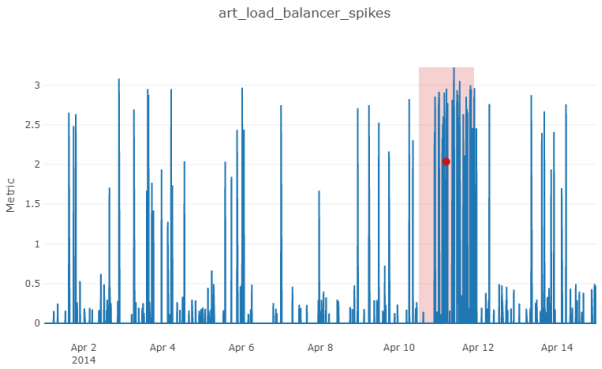
(a) art_daily_flatmiddle.csv



(b) art_daily_jumpsdown.csv

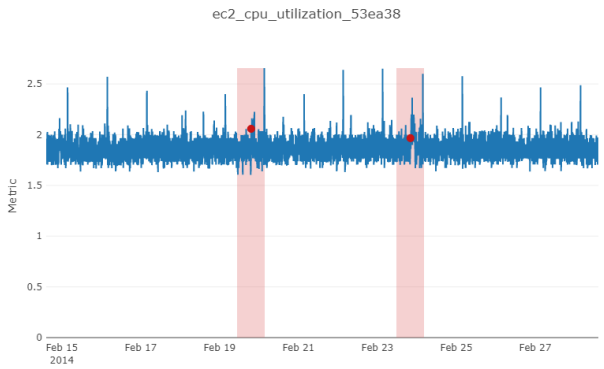


(c) art_daily_nojump.csv

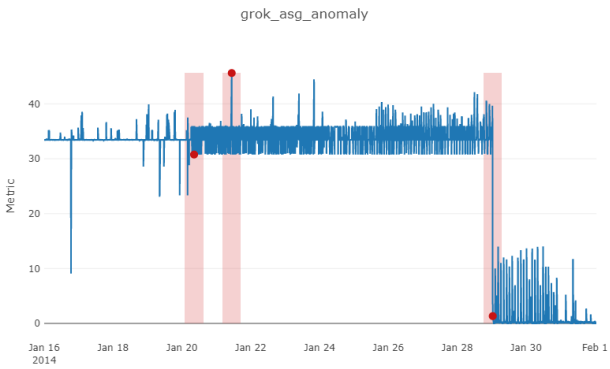


(d) art_load_balancer_spikes.csv

Figure 7: Artificial anomaly samples. Illustrations by the author.

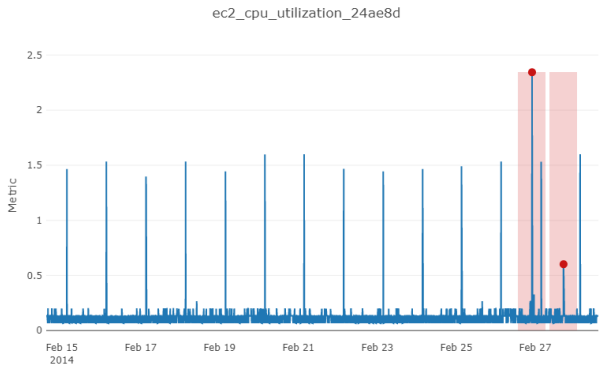


(a) ec2_cpu_utilization_53ea38.csv

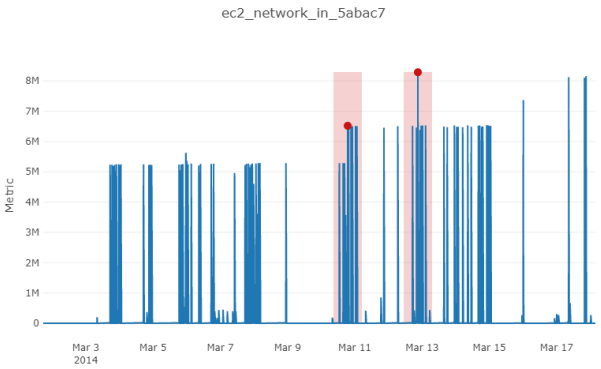


(b) grok_asg_anomaly.csv

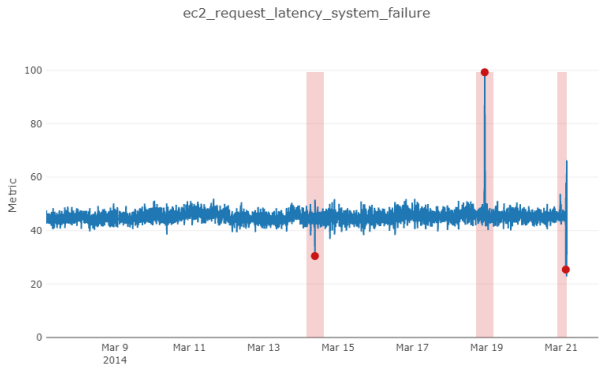
Figure 8: (a) shows contextual anomalies, (b) shows two change points and a point-anomaly. Illustrations by the author.



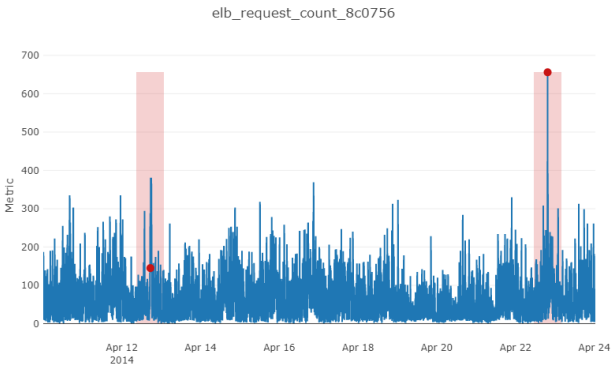
(a) ec2_cpu_utilization_24ae8d.csv



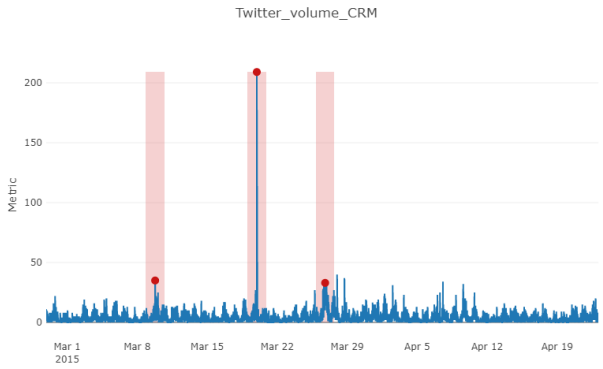
(b) ec2_network_in_5abac7.csv



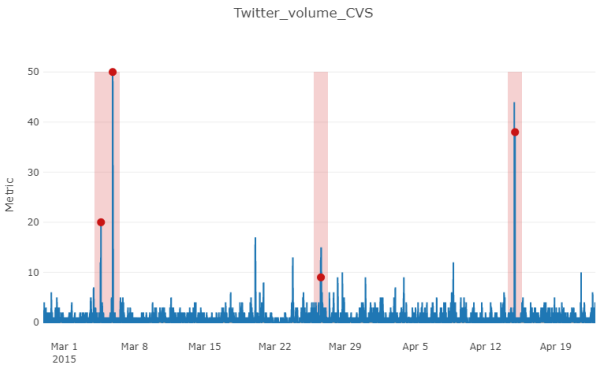
(c) ec2_request_latency_system_failure.csv



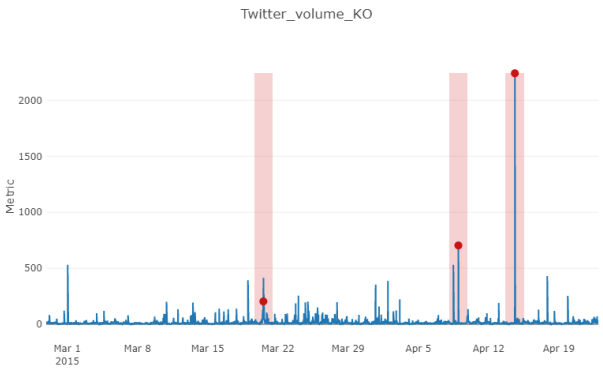
(d) elb_request_count_8c0756.csv



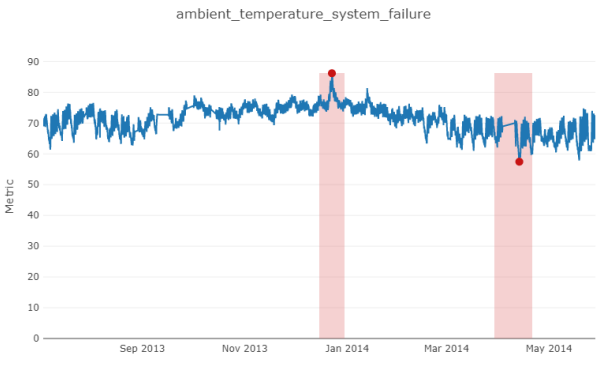
(e) Twitter_volume_CRM.csv



(f) Twitter_volume_CVS.csv



(g) Twitter_volume_KO.csv



(h) ambient_temperature_system_failure.csv

Figure 9: Point Anomalies. All illustrations by the author.

D Other Datasets

Dataset
Numenta Anomaly Benchmark
Tennessee Eastman Process Simulation Dataset
Turing Change Point Benchmark
Secure Water Treatment
M-Competitions
Yahoo! Webscope S5

Table 2: Univariate Datasets

Dataset
Data from: Machine Learning-based Anomaly Detection in Software Systems
3W Dataset
NASA Bearing Sensor Data
HOT SAX Datasets
Soil Moisture Active Passive (SMAP)
Mars Science Laboratory (MSL)
HIL-based Augmented ICS

Table 3: Multivariate-Datasets

E Algorithm Selection

Curated List
https://github.com/cuge1995/awesome-time-series
https://github.com/xephonhq/awesome-time-series-database
https://github.com/MaxBenChrist/awesome_time_series_in_python
https://github.com/cuge1995/awesome-time-series
https://github.com/yzhao062/anomaly-detection-resources
https://github.com/rob-med/awesome-TS-anomaly-detection
https://www.kaggle.com/general/185462

Table 4: Curated lists for open source anomaly detection libraries

Library	Models	Version	Latest Commit	Stars
AtsPy	Conglomeration of independent libraries: • Auto ARIMA • Prophet • N-Beats • Gluon-TS • and TBAT; Also: • Holt Winters	/	12th Nov 2020	320
GluonTS	Deep Learning: • Deep Factors for Forecasting [Wan+19] • DeepAR [FSG17] • Deep State Space [Ran+18] • Gaussian Processes • N-Beats [Ore+20] • Transformer • Wavenet Also Includes: • Prophet	0.6.4	7th Jan 2021	1.7k
pmdarima	SARIMAX	1.8	2nd December 2020	790
PyFlux	ARIMAX, Dynamic AR, Dynamic Linear Regression, EARCH, GAS, ARCH, GARCH...	/	16th December 2018	1.8k
sktime	ARIMA, Exponential Smoothing/Holt Winters, Polynomial Thresholding	0.5.1	6th Jan 2021	3.4k
ARCH	AR, Heterogeneous Autoregression, ARCH, GARCH, TARCH, EGARCH, ...	4.15	7th Jan 2021	630
Time Series Prediction TF2	LSTM, GRU, Wavenet, Transformer, N-Beats, GAN	/	25th December 2020	292
Transformers for Timer Series	Transformer	0.3	10th December 2020	231
Prophet	General Additive Model • trend is fitted via a logistic or a linear model • seasonality is fitted via Fourier series • also considers holidays	0.6	7th Jan 2021	12.1k
TBATS	Automated BATS/TBATS fitting	1.1	27th July 2020	84
N-Beats	Res- and fully connected layer	/	30th April 2020	326

Table 5: Forecasting-Focused Libraries, accessed last: 8th Jan 2021

Library	Models	Version	Latest Commit	Stars
Anomaly Detection Toolkit (ADTK)	Statistical Models: • AR • Seasonal Pattern/Decomposition • Generalized ESD-Test • Difference in Mean • Difference in QR/IQR • Difference in Volatility; ML: • Clustering	0.6.2	17th April 2020	582
Contextual Anomaly Detector	No Documentation. Winner of NAB 2016	/	10th August 2016	63
datastream.io	Focused on integration into ELK-stack; Gaussian-Distribution and Difference in Percentile	/	20th Feb 2018	804
DONUT	Variational Auto-Encoder [Xu+18]	/	6th March	298
hastic	Focused on Grafana. Thresholding on seasonality adjusted and exponentially smoothed data.	/	26th Dec 2020	274
luminol	Bitmap-Transformation [Wei+05], Single Exponential Smoothing;	0.4	9th Jan 2018	861
PyODDS / PyOD	Statistical Models: • Local Outlier Factor • Clustering Based Local Outlier Factor • Histogram-Based Outlier Score • Isolation Forest • kNN • One-Class SVM • PCA • Robust Covariance • Subspace Outlier Detection • Luminol Deep Learning: • LSTMAD[Mal+15] • DAGMM [Zon+18]	/	8th May 2020	132
rrcf	Robust Random Cut Forest [Guh+16]	0.4.3	10th June 2020	271
rnn ts anomaly detection	LSTM, GRU, SRU	/	21st Oct 2020	684
TODS: Time-series Outlier Detection System	DeepLog [Du+17], Telemanom Matrix Profile, PyOD algorithms	/	5th Jan 2021	258
Skyline	Ensemble of statistical tests like Kolmogorov-Smirnov, Grubbs, deviation from median...	2.0	22nd Dec 2020	284
Banpei	Hotelling's theory [Hot90] and Singular Spectrum Transformation	/	21st Sept 2020	222
NuPIC	Numenta HTM	/	23 Oct 2019	6.2k
MPF	Matrix Profile	/	26th Dec 2020	121

Table 6: Boundary Focused Detection Libraries, accessed last: 8th Jan 2021

Boundary-Libraries
PyODDS / PyOD
rrcf
Skyline
NuPIC
Forecasting-Libraries
GluonTS
AtsPy
PyFlux
sktime

Table 7: Chosen Libraries

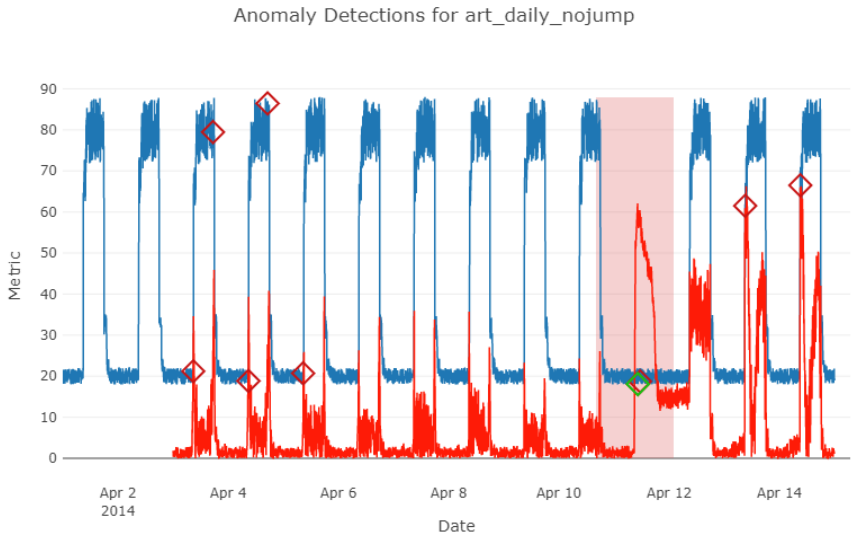
Forecasting Algorithm	Package/Implementation
Prophet [TL17]	AtsPy
Auto ARIMA [Smi+17]	pmdarima
ARIMA [Hyperparameters adopted from BW20]	pmdarima
Holt-Winters [Win60]	AtsPy
N-BEATS [Ore+20]	Implementation by the author
TBAT/S	AtsPy
DeepAnT [Mun+19]	Implementation by the author

Table 8: Chosen Forecasting Algorithms

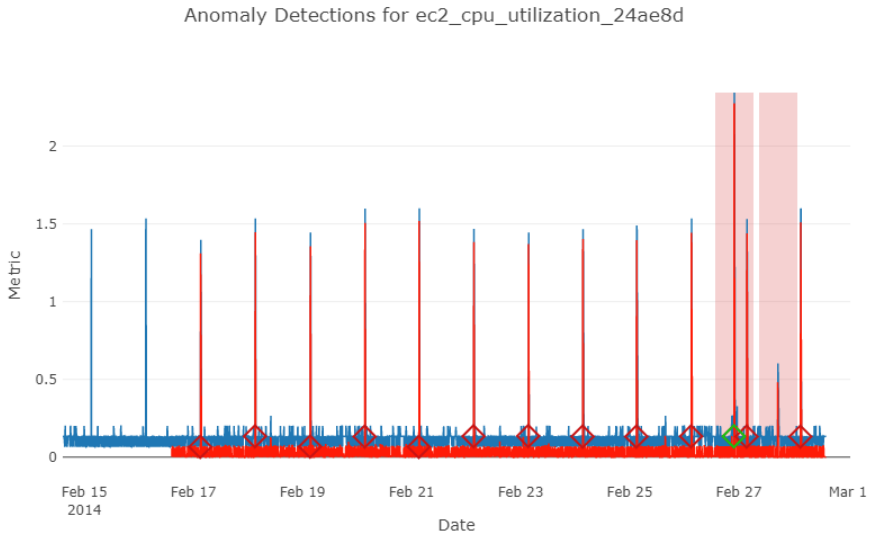
Boundary Algorithm	Package/Implementation
LOF [Bre+00]	PyODDS
AE [GBC16, pp. 499 sqq.]	PyODDS
CBLOF [HXD03]	PyODDS
DAGMM[Zon+18]	PyODDS
kNN [Mur12, pp. 16 sqq.]	PyODDS
Skyline	Skyline
OCSVM [Sch+99; TD04]	PyODDS
Robust Random Cut Forest [BMT19]	rrcf
LSTM-AD [Mal+15]	PyODDS
LSTM-ED [Mal+16]	PyODDS
Numenta Threshold Detector [Ahm+17]	Implementation by the author / NuPIC
Numenta HTM [Ahm+17]	Community-HTM

Table 9: Chosen Boundary Algorithms

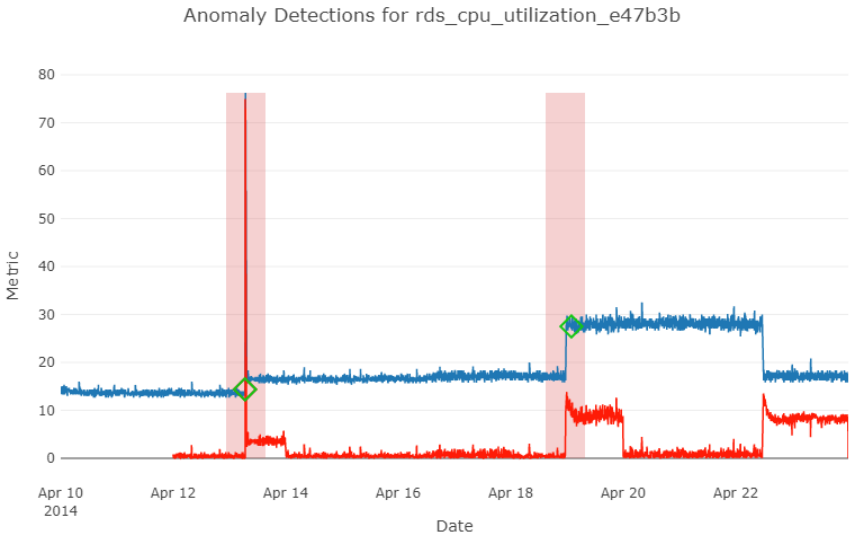
F Illustrations from Forecasting Algorithms



(a) Example detection of a cyclicity violation. True positives are displayed as green- and false positives as red diamonds. The red line shows error-terms from DeepAnT, the blue line shows the original time series.

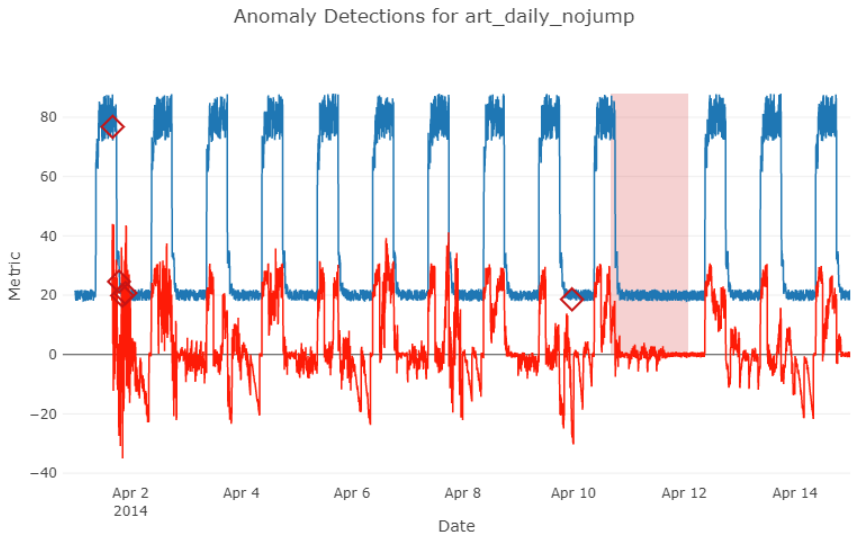


(b) A result from DeepAnT from a time series with regular spikes. The error-terms (red) closely resemble the original values (blue — mostly hidden underneath the red error-terms).

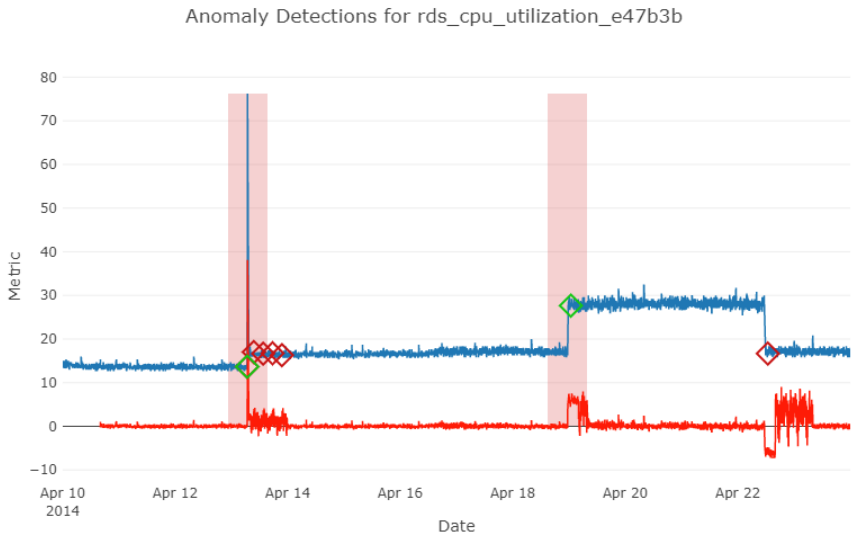


(c) DeepAnTs response to multiple changepoints. First, predictions occur mean shifted. However, DeepAnT quickly adapts to the new behavior.

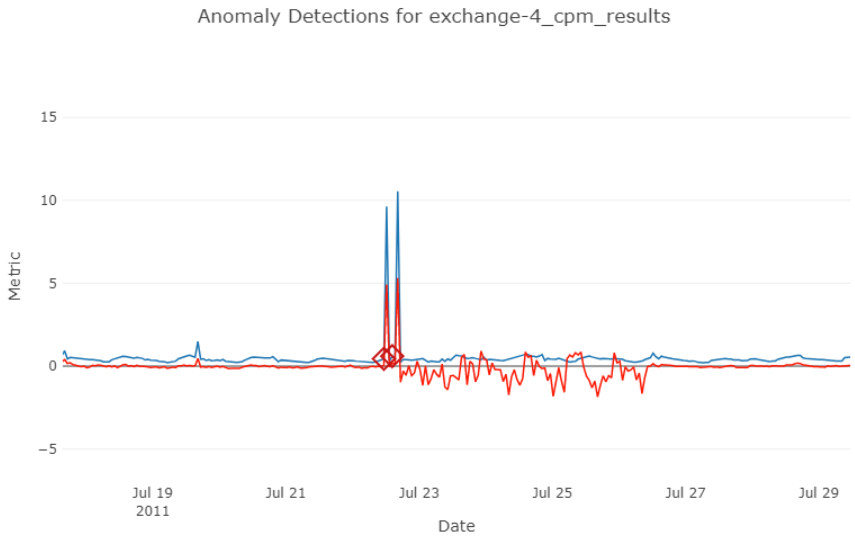
Figure 10: Examples from DeepAnT. Illustrations by the author.



(a) An example of N-BEATS' lack of sensitivity to long-term cyclicity

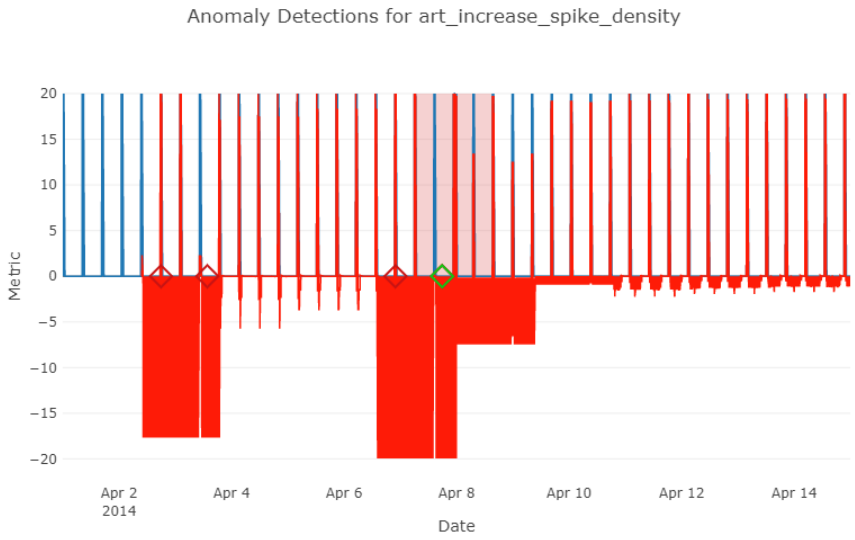


(b) Like DeepAnT, output from N-BEATS often resembles the original time series.

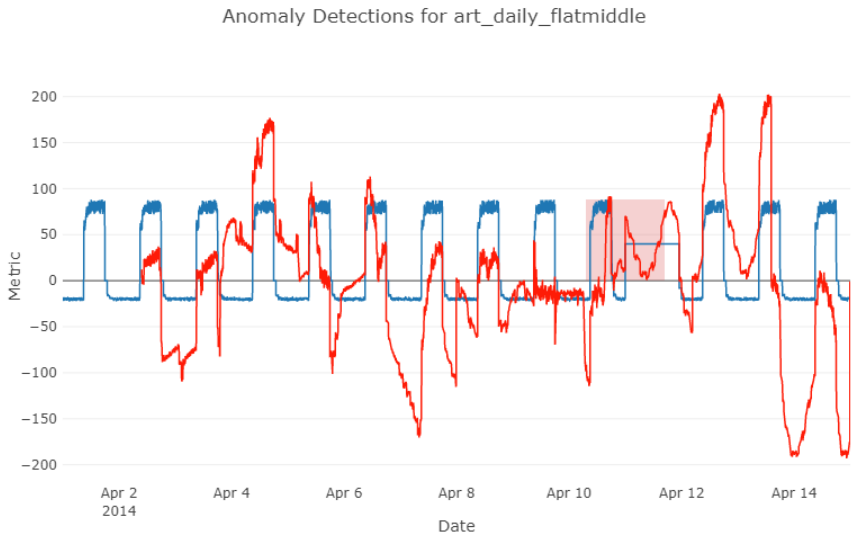


(c) Example of a value spike (Jul 23) that decreases prediction accuracy for almost four consecutive day.

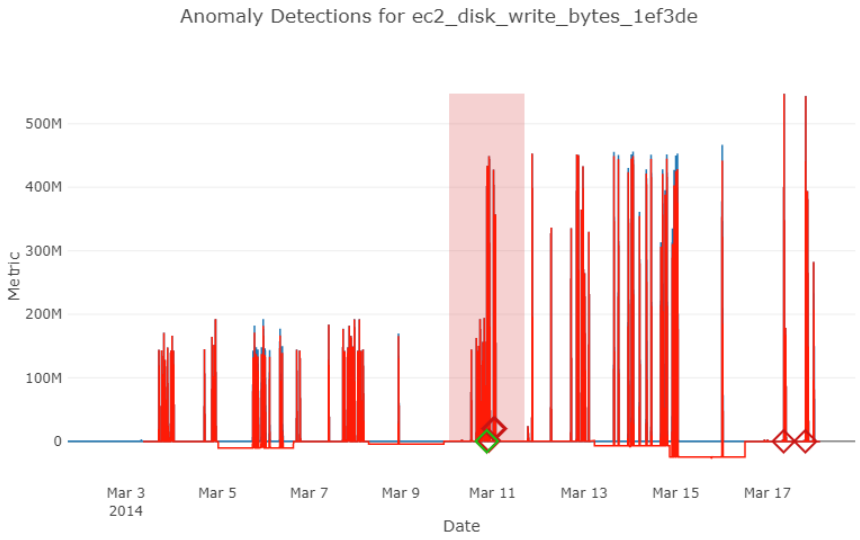
Figure 11: Examples from N-BEATS. Illustrations by the author.



(a) An example of computational instability from Holt-Winters.

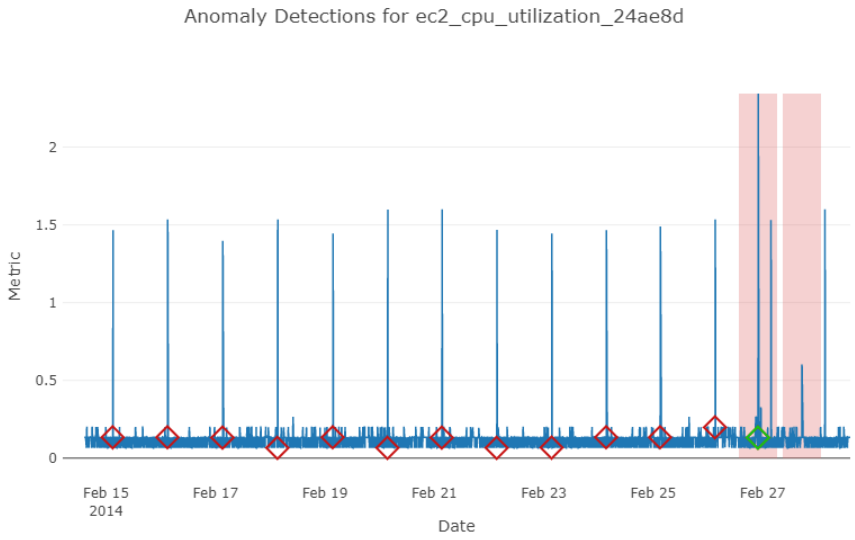


(b) An example of inferred trends from Holt-Winters that are not found within the data.

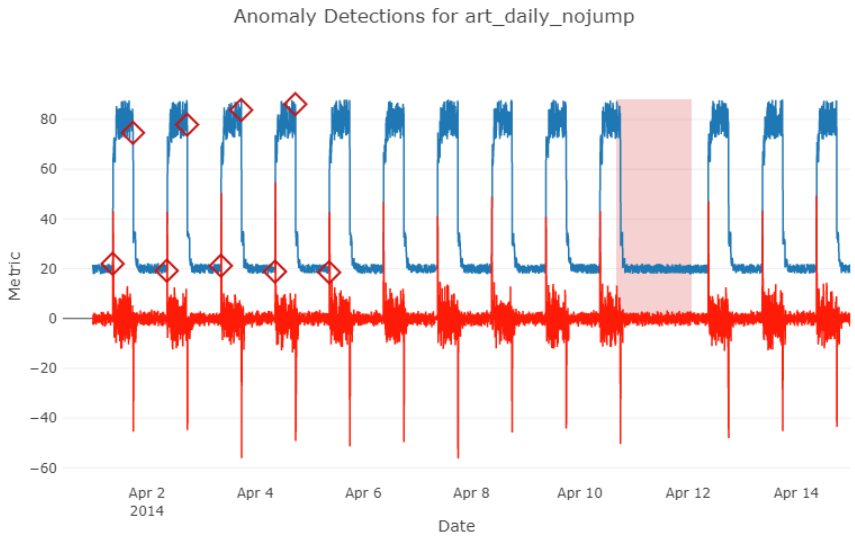


(c) A more ordinary example from Holt-Winters.

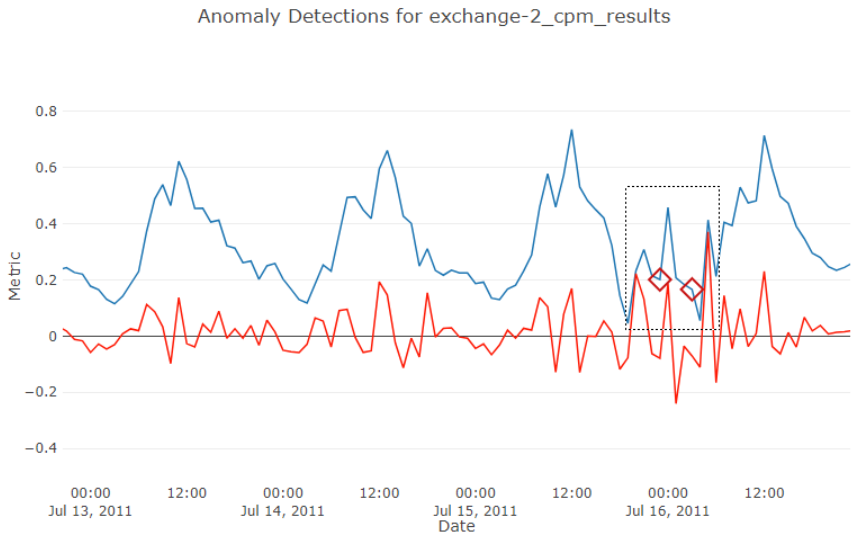
Figure 12: Examples from Holt-Winters. Illustrations by the author.



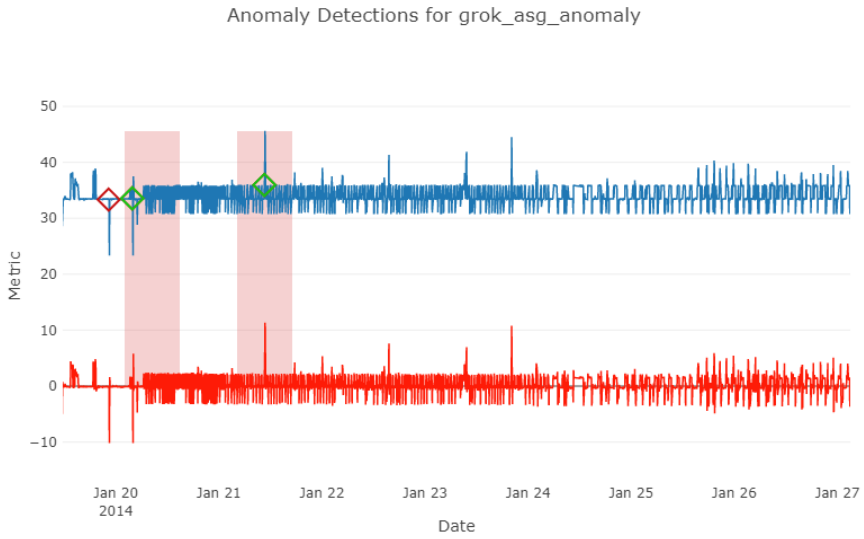
(a) High false positives rate on time series with many spikes.



(b) ARIMA is unable to adopt long-term cyclicity.

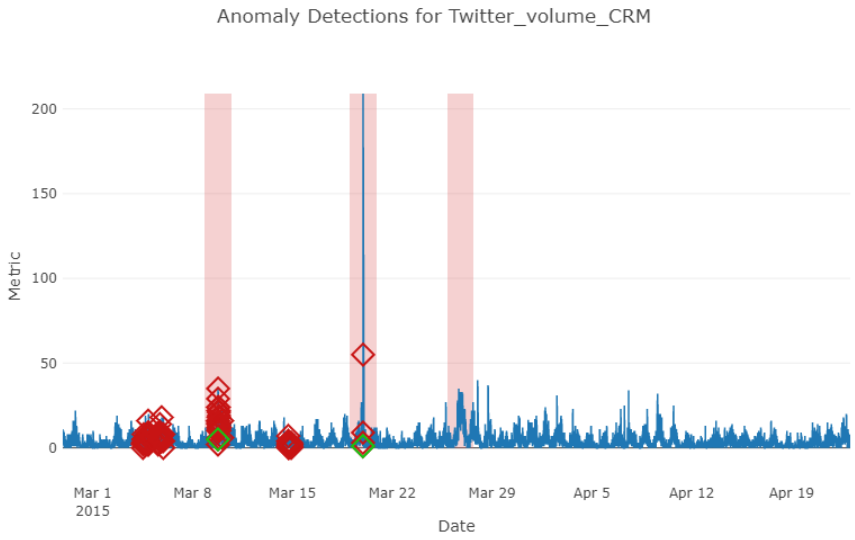


(c) Time series values are colored blue, while error-terms from ARIMA are red. The pattern (dotted box) from the time series deviates from previous time steps.

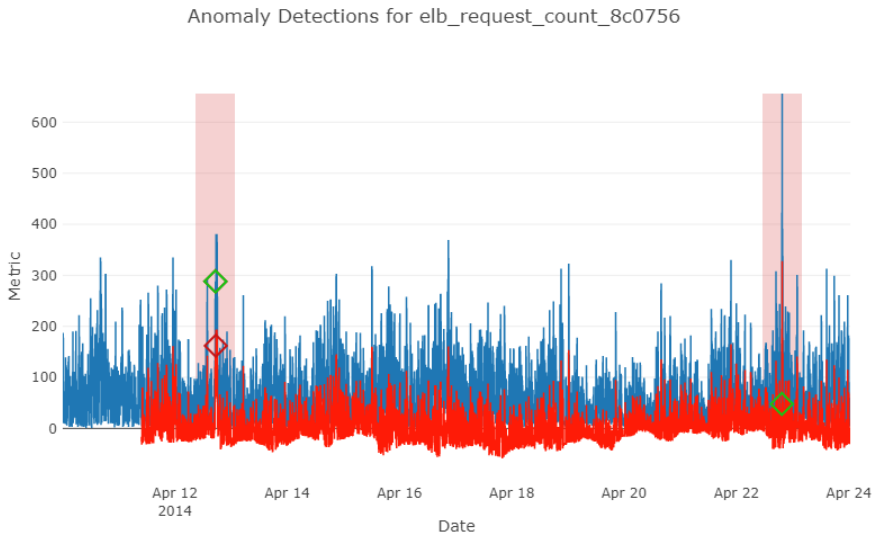


(d) On more chaotic time series, output from ARIMA closely resembles the original time series.

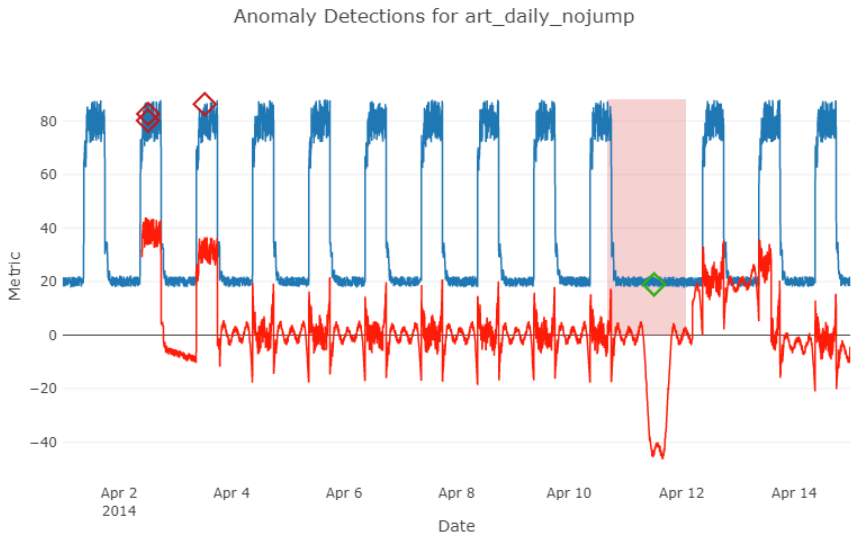
Figure 13: Examples from Auto-ARIMA. Illustrations by the author.



(a) 57 false positives on a single time series.



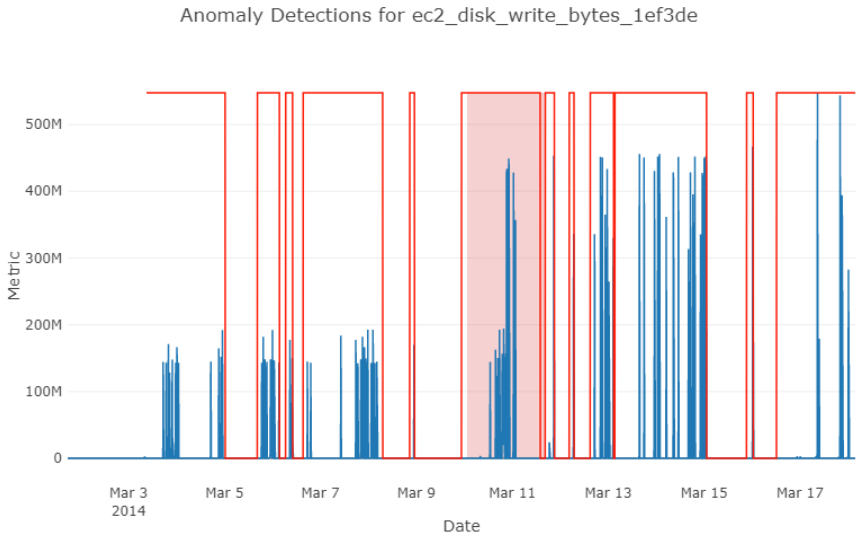
(b) Prophet often assumed seasonality within the data, producing a curvy output.



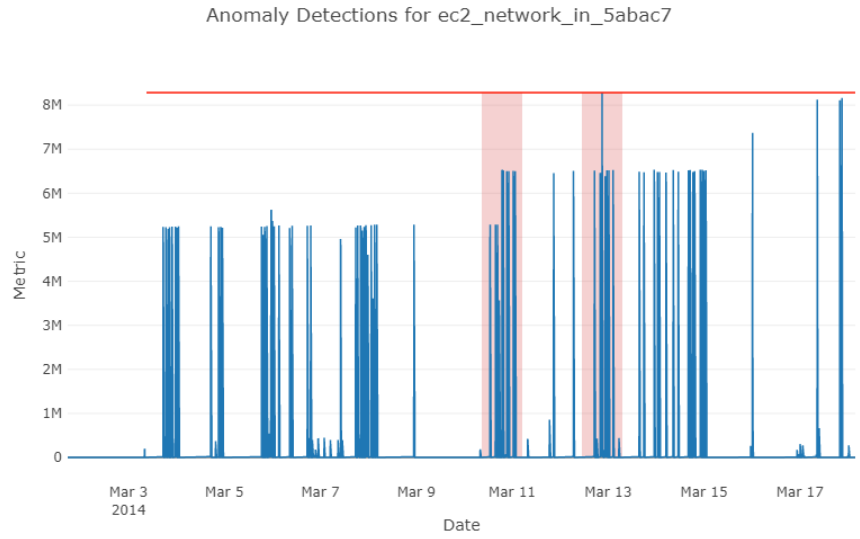
(c) Successful adaptation to the cyclicity of a simple time series.

Figure 14: Examples from Prophet. Illustrations by the author.

G Illustrations from Boundary Algorithms

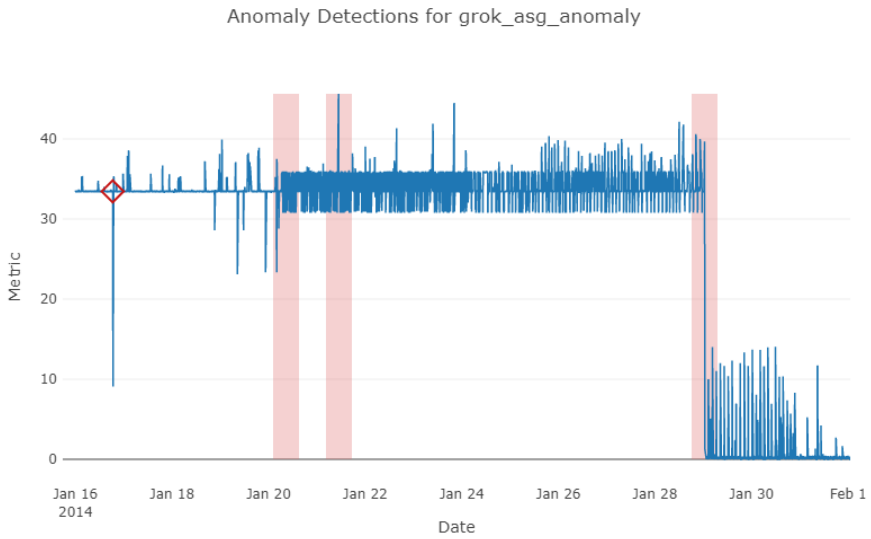


(a) Output from OCSVM seems to lay *boxes* around observations with value > 0. Useless output for the task of anomaly detection.

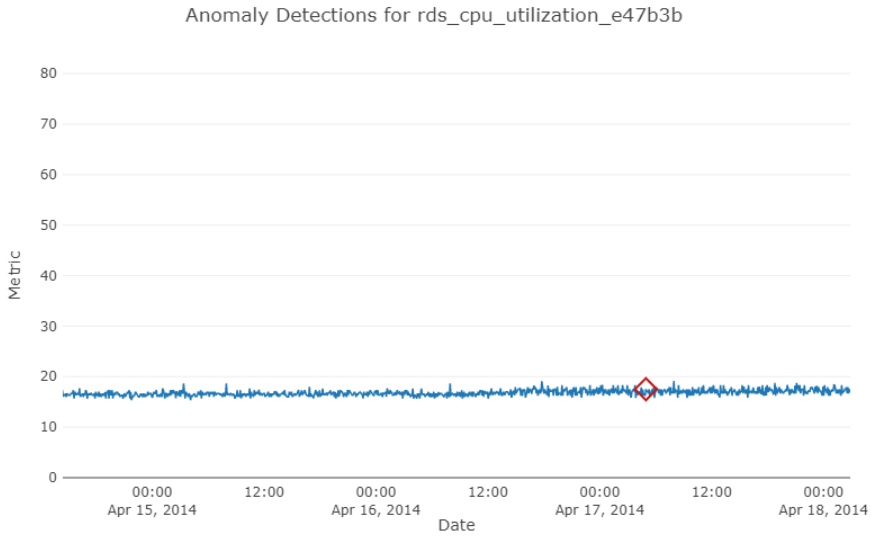


(b) In this example, OCSVM claims every that every observation is anomalous.

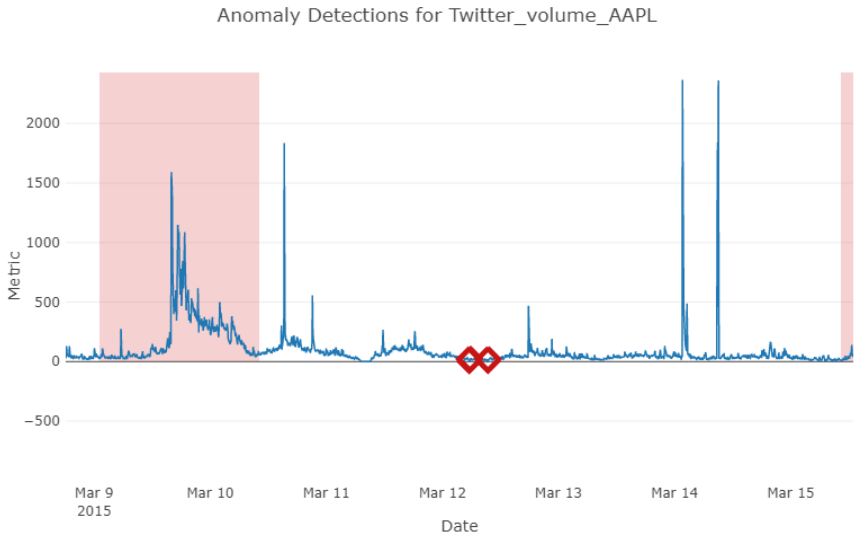
Figure 15: Unusable outputs from OCSVM. Blue lines represent observations from the time series. Red lines represent the output from OCSVM. Illustration by the author.



(a) Output from LOF, where no detections are produced on a seemingly simple example. Anomalous time windows are marked red.

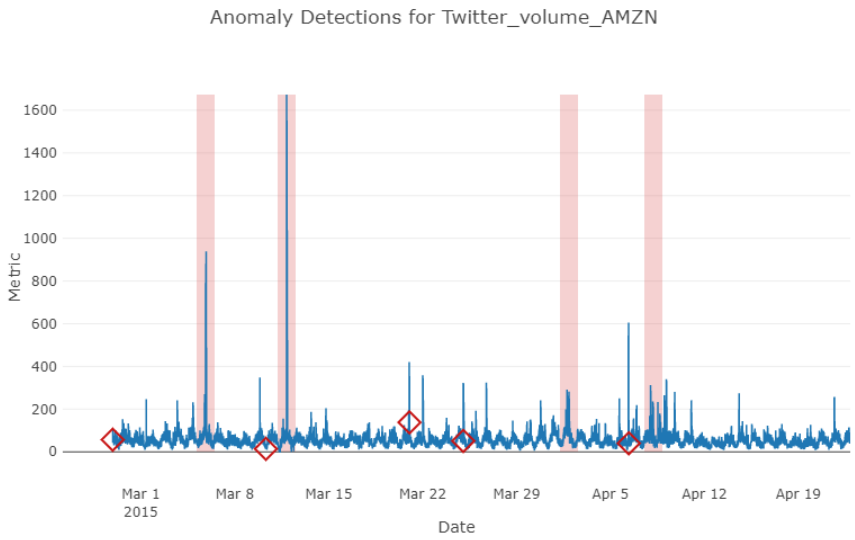


(b) Example of an unreasonable detection by LOF.

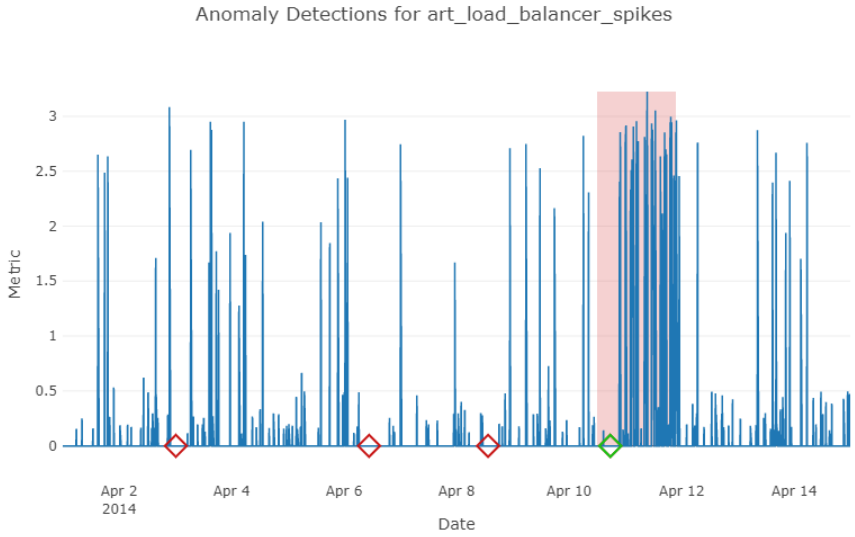


(c) Another example of an unreasonable detection by LOF.

Figure 16: (Seemingly) unreasonable detections from LOF. Detections are depicted as colored diamond symbols. Illustration by the author.

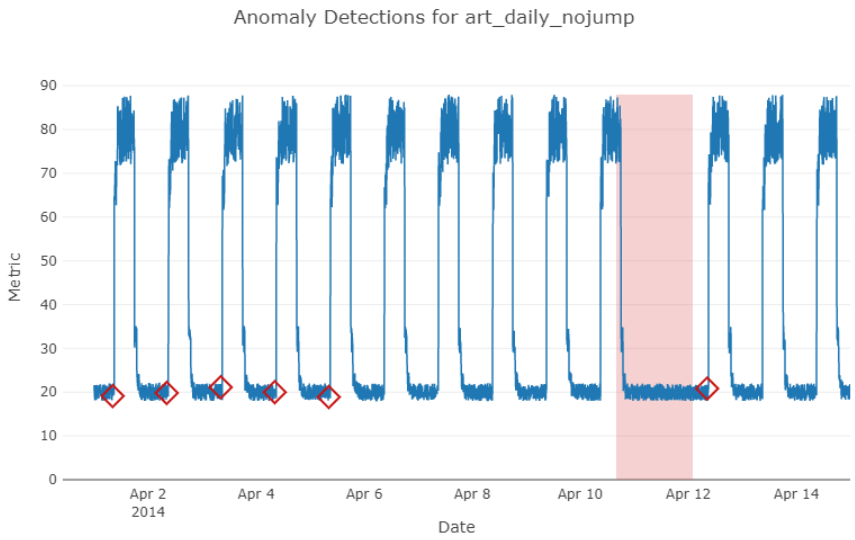


(a) False positives from DAGMM

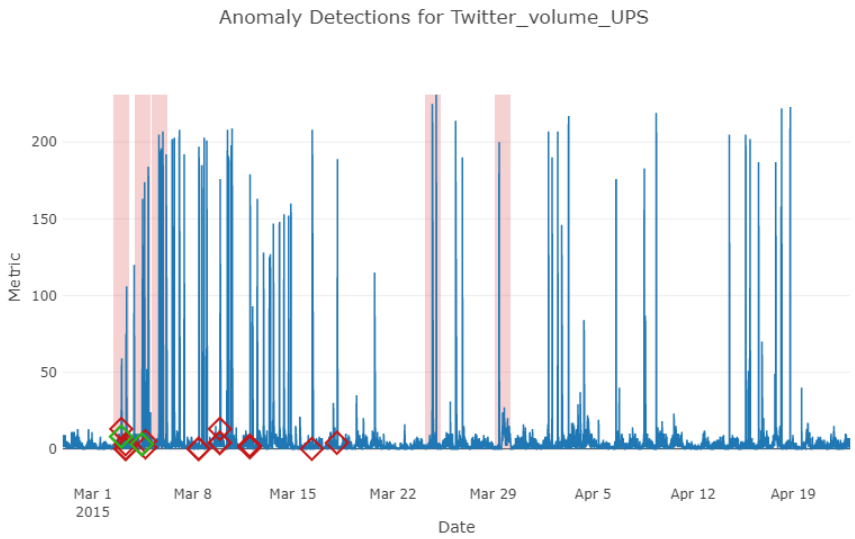


(b) A representative true positive from DAGMM. The reason for the anomaly is the high value spike (inside the red area). DAGMM (unreasonably) attributes the anomaly to a normal value approximately two days before.

Figure 17: Examples from DAGMM. Illustrations by the author.

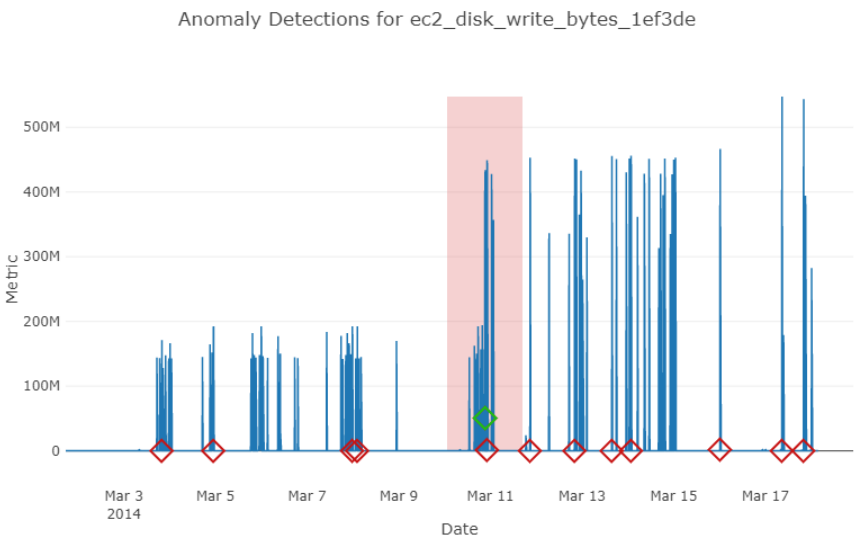


(a) Example detection of a cyclicity violation. Note that the anomaly was only detected as values began rising again for the normal pattern.

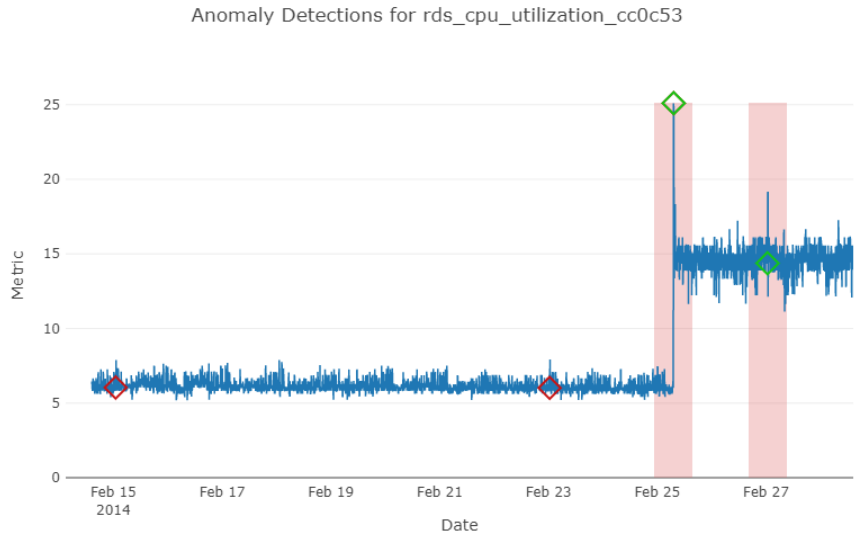


(b) A representative result from RRCF, where a complex time series confuses the algorithm.

Figure 18: Examples from RRCF. Illustrations by the author.

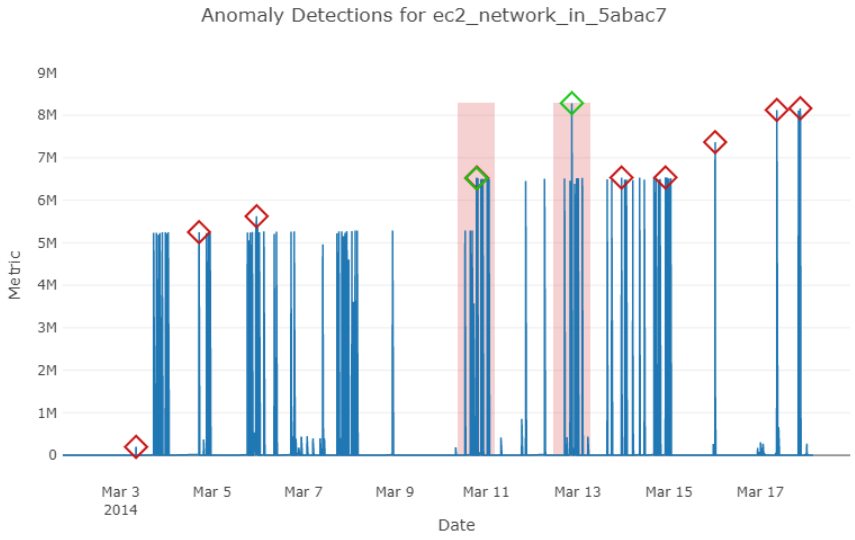


(a) High false positives rate on time series with many spikes.

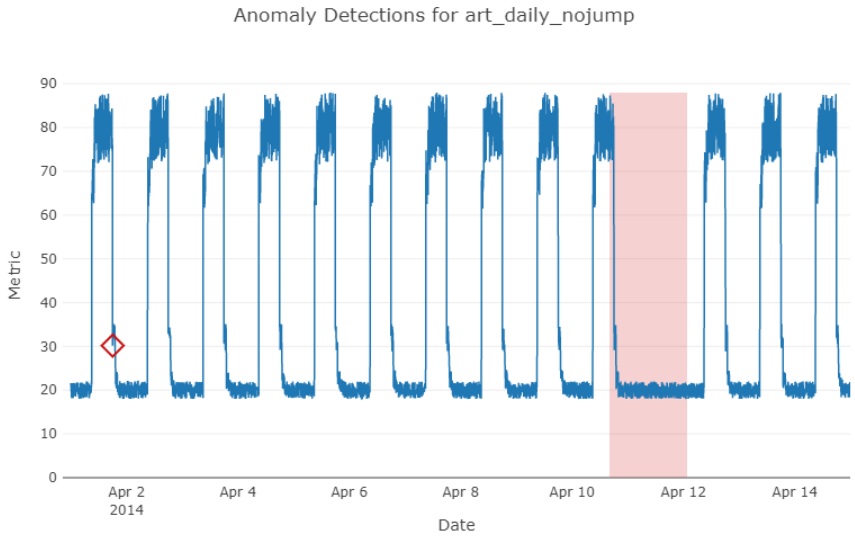


(b) Successful detection of two spikes.

Figure 19: Examples from NDT. Illustrations by the author.

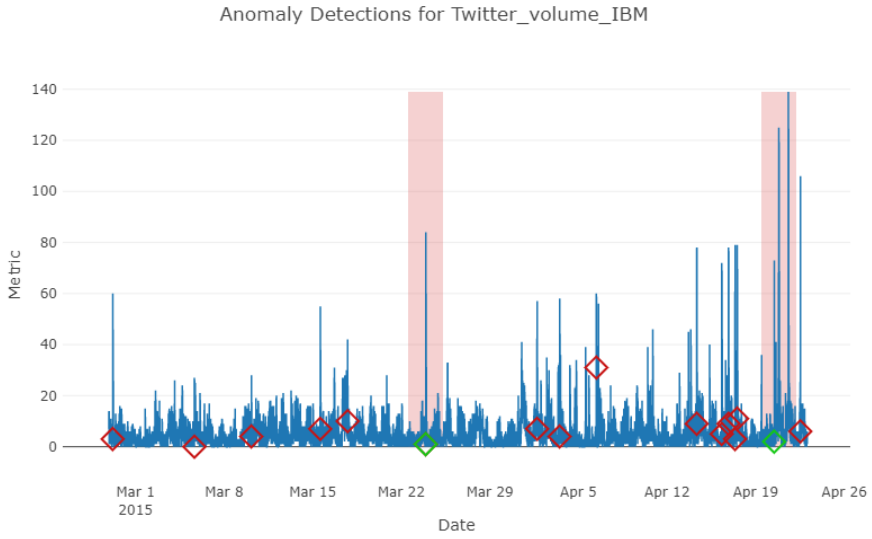


(a) High false positives rate on time series with many spikes.

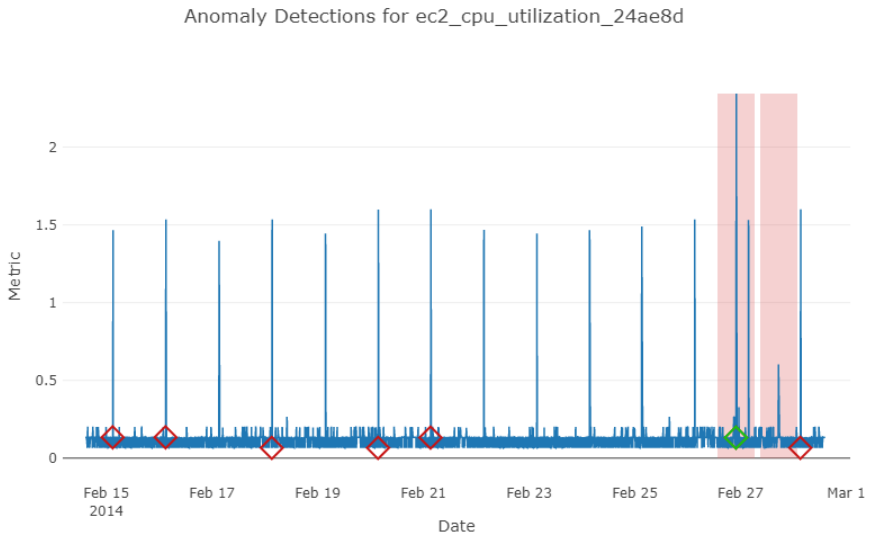


(b) Cyclicity violations remain undetected by LSTM-AD.

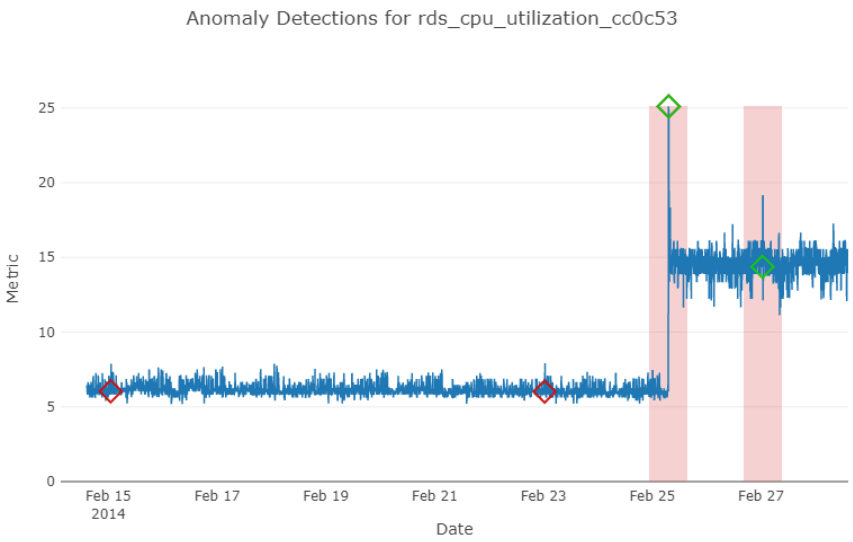
Figure 20: Examples from LSTM-AD. Illustrations by the author.



(a) High false positives rate on (some) time series with irregular spikes.

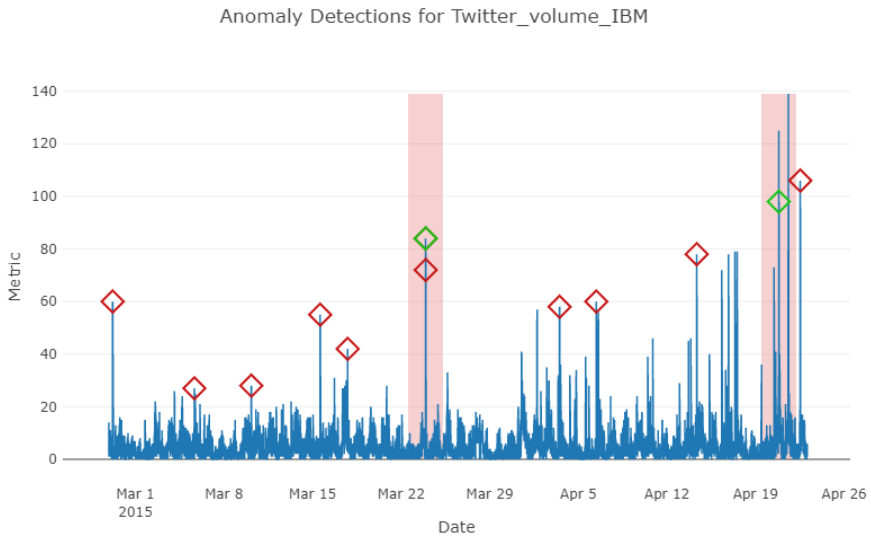


(b) Adaptation to more regularly spiking time series.

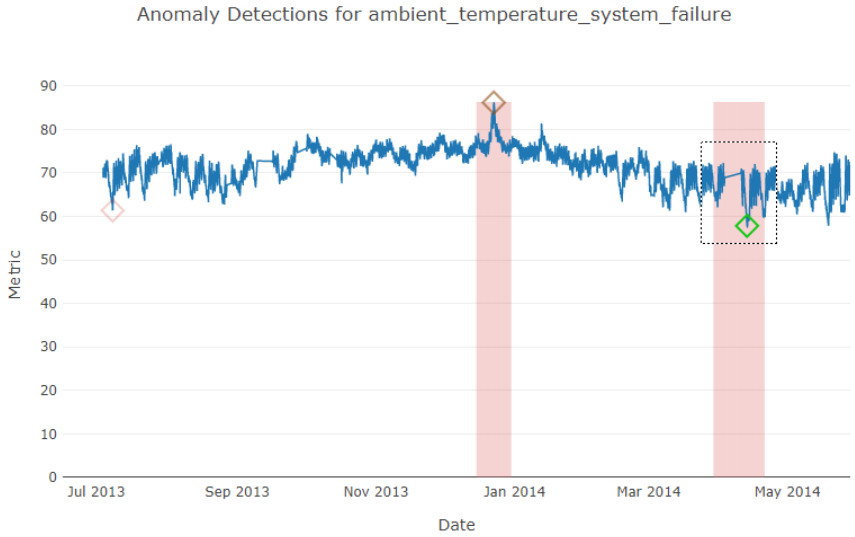


(c) Example of a good result from CBLOF.

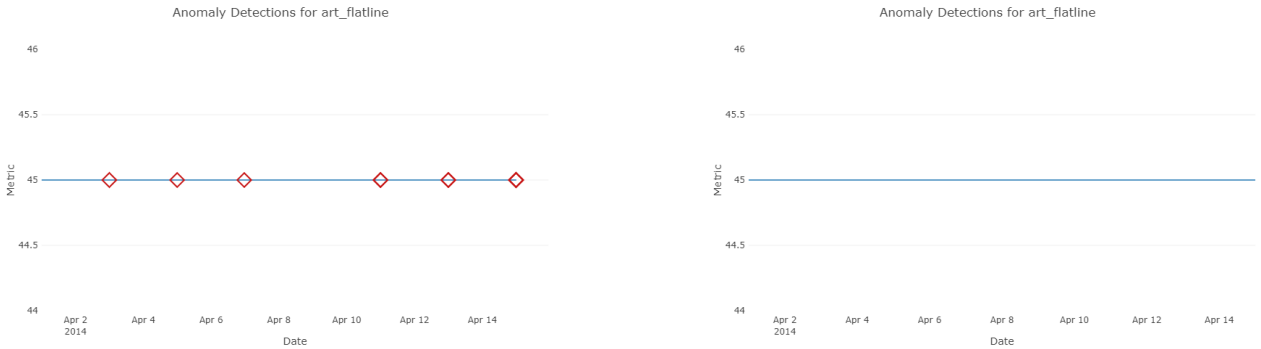
Figure 21: Examples from CBLOF. Illustrations by the author.



(a) High false positives rate on (some) time series with irregular spikes.

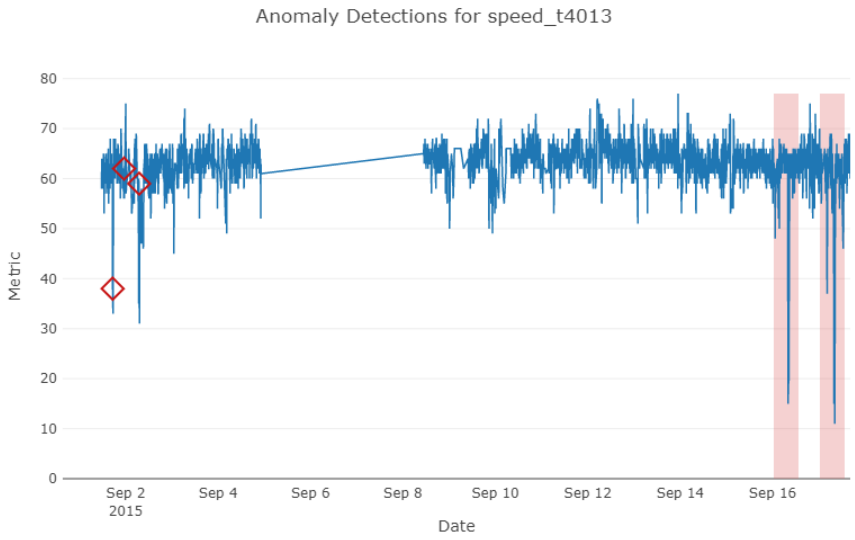


(b) A harder anomaly (dotted box) detected by LSTM-ED.

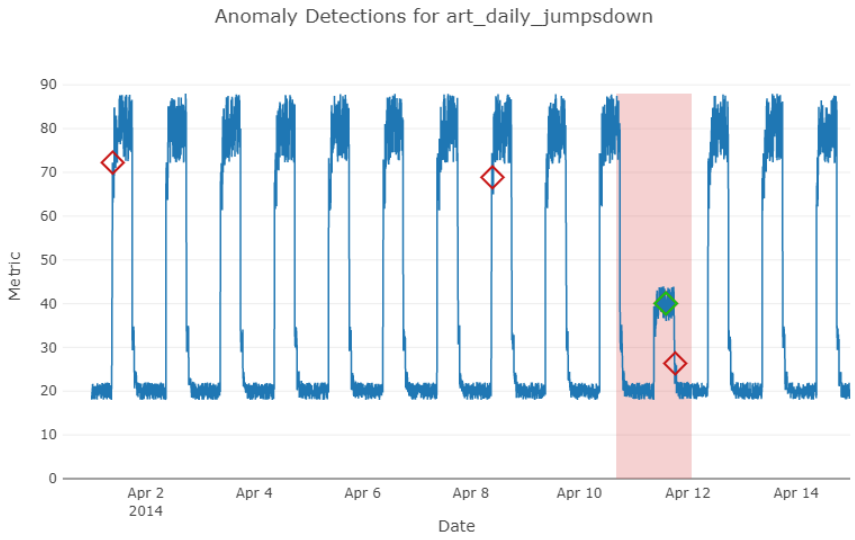


(c) Left: Results from LSTM-ED on one of the artificial time series (straight line). Right: Results from CBLOF on the same artificial time series.

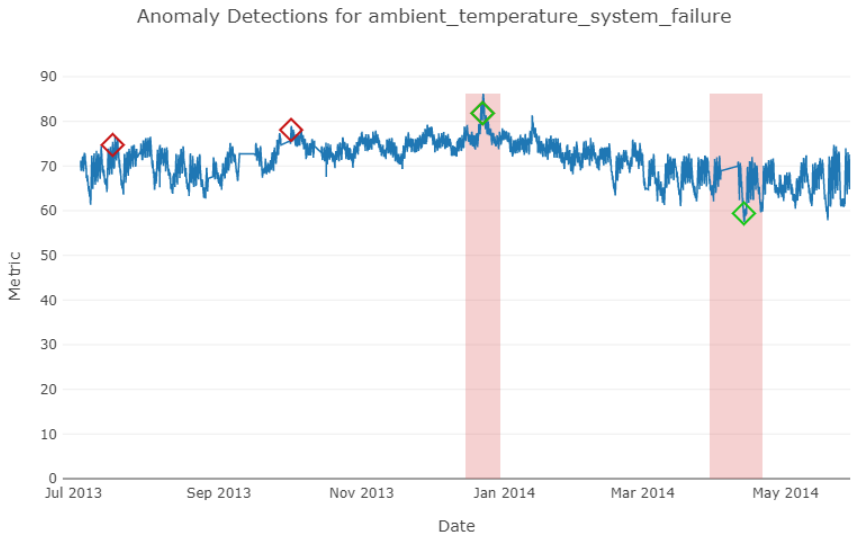
Figure 22: Examples from LSTM-ED. Illustrations by the author.



(a) Two simple spike anomalies missed by kNN.



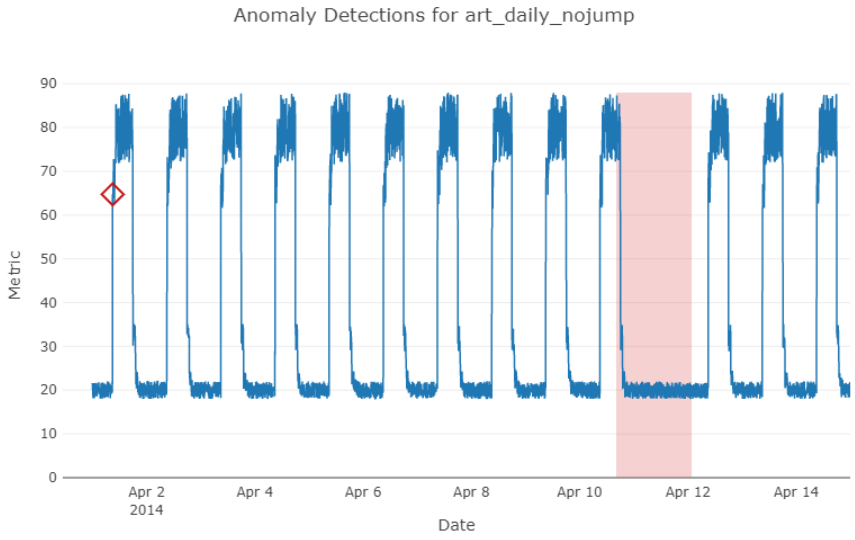
(b) A simple cyclicity violation that was detected by kNN, presumably due to its unusual value range.



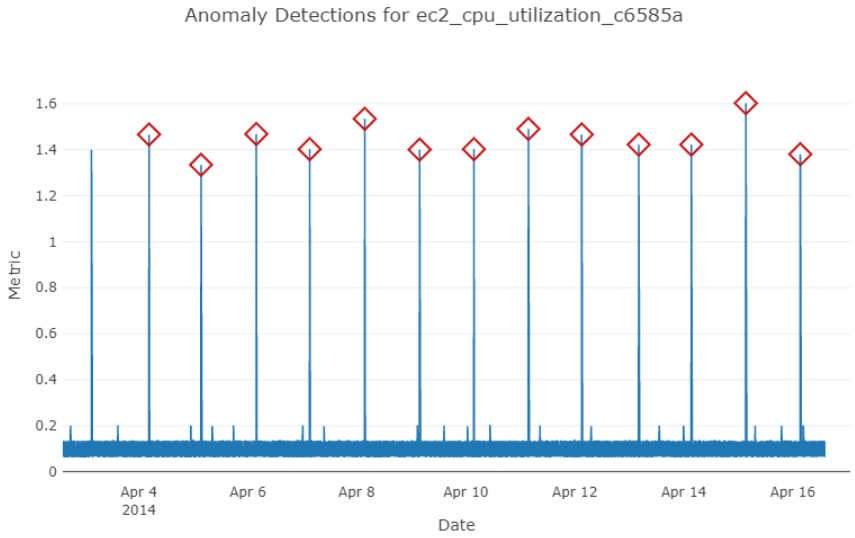
(c) A difficult anomaly detected by kNN (rightmost anomaly).

Figure 23: Examples from kNN. Illustrations by the author.

Figure 24: Two representative examples from the Numenta Threshold Detector. Illustrations by the author.

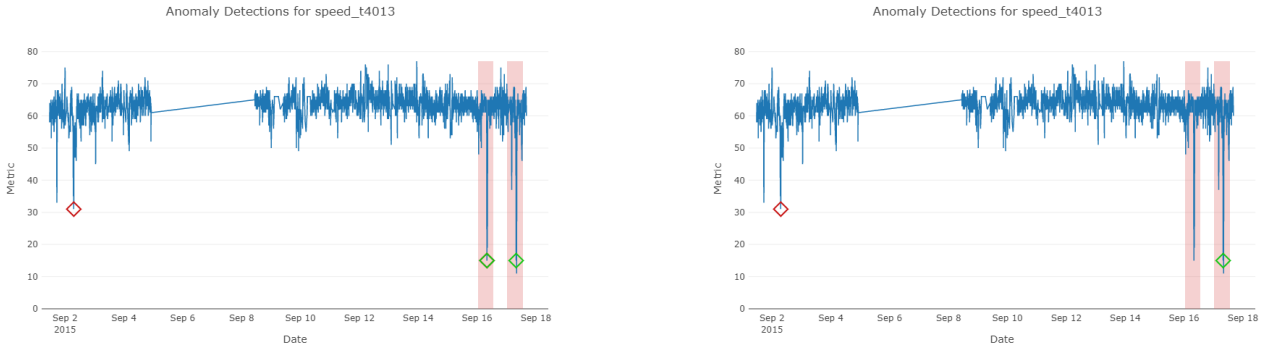


(a) Skyline was unable to detect cyclicity violations.

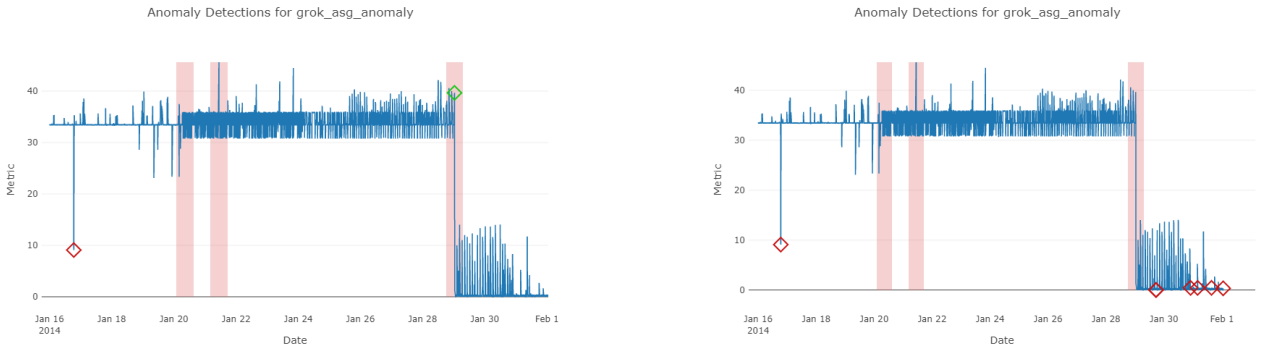


(b) An example of spikes that are outside the moving 3-Sigma rule and thereby lead to many false positives.

Figure 25: Examples from Skyline. Illustrations by the author.

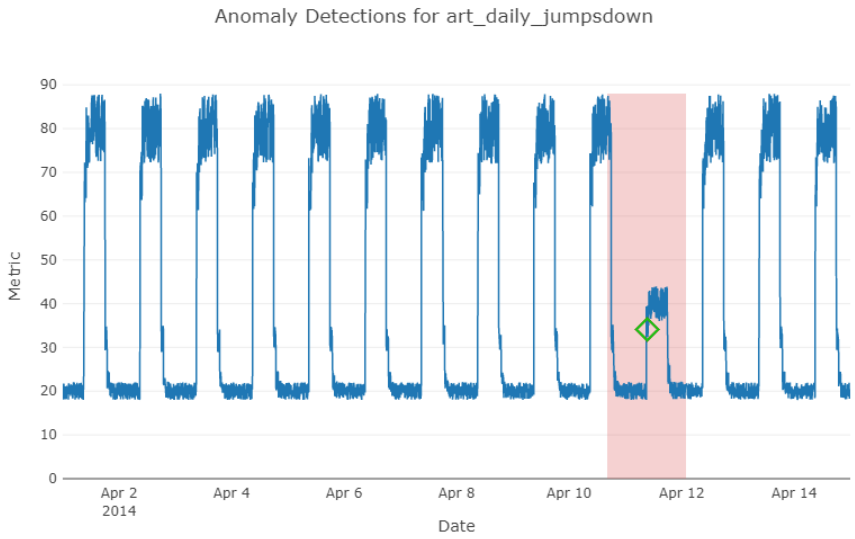


(a) AE (left) is able to produce one more true positive than LSTM-ED (right) on this example

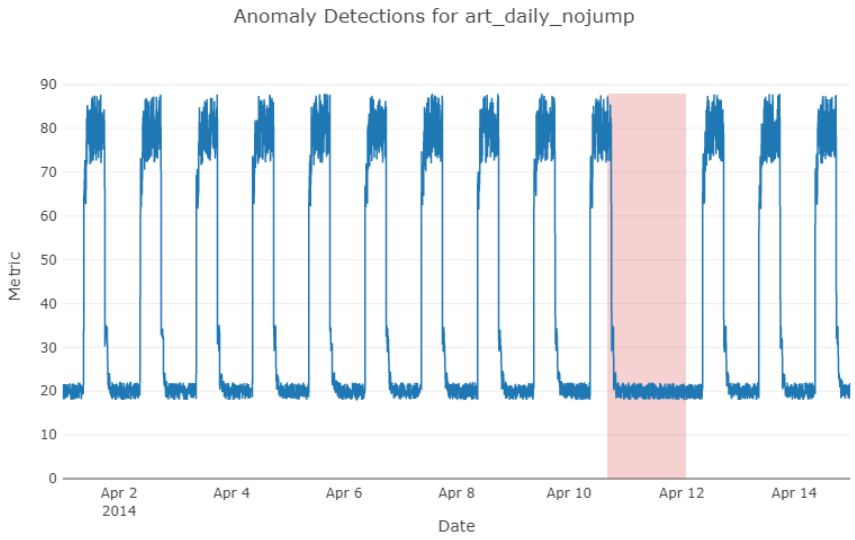


(b) AE (left) produces one more true positive and five less false positives than LSTM-ED (right) on this example.

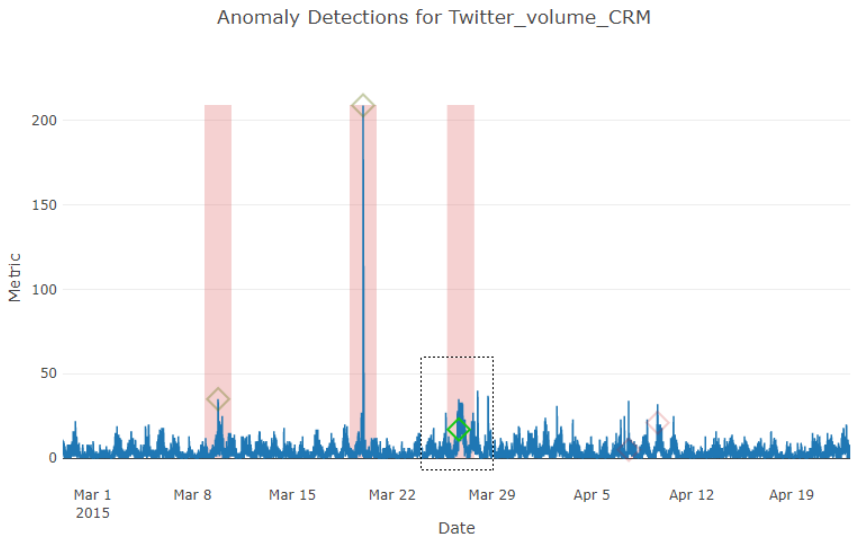
Figure 26: Comparison of AE and LSTM-ED on two time series. Illustrations by the author.



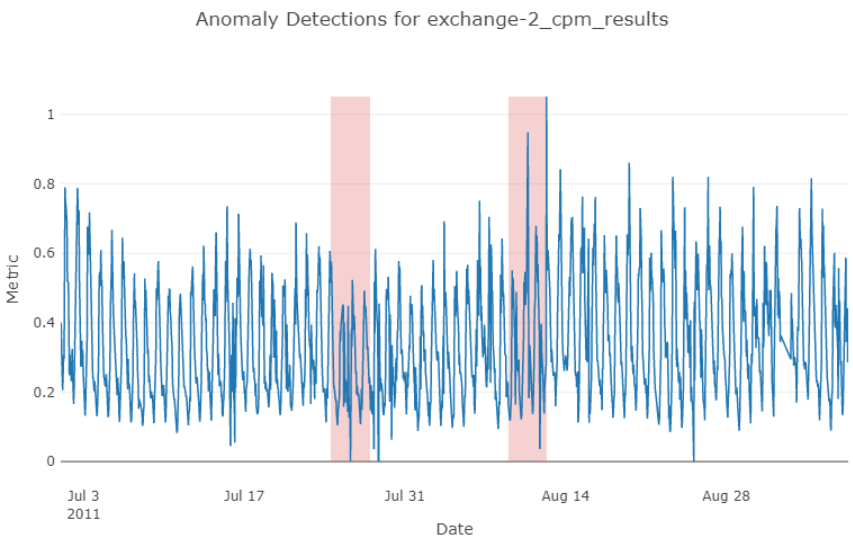
(a) HTM was able to detect simple cyclicity violations in the artificial time (and thereby simple) series.



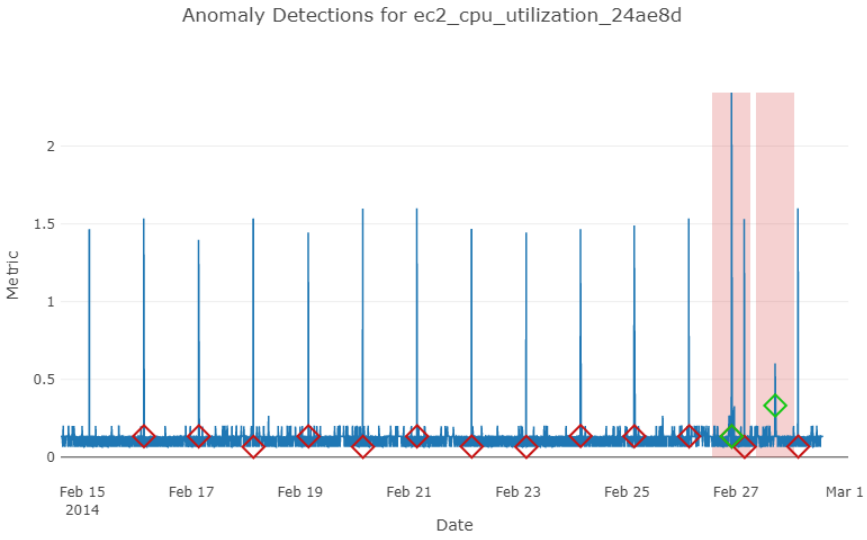
(b) More difficult cyclicity violations were not detected.



(c) HTM was more sensitive for some subtle anomalies.



(d) ...while other subtle anomalies were overlooked.



(e) HTM did not adapt to regular yet infrequent value spikes.

Figure 27: Examples from HTM. Illustrations by the author.