Big Data Research 30 (2022) 100335

Contents lists available at ScienceDirect

Big Data Research

www.elsevier.com/locate/bdr

Detecting Seasonal Dependencies in Production Lines for Forecast Optimization

Gerold Hoelzl [a], Sebastian Soller [b,*], Matthias Kranz [a]

[a] University of Passau, Chair of Embedded Systems, Germany
[b] Almanara Research GmbH, Germany

ARTICLE INFO

Article history:
Received 23 September 2020
Received in revised form 7 June 2022
Accepted 6 July 2022
Available online 26 July 2022

Keywords:
Data mining
Real-time system
Maintenance prediction
Time series forecast
Seasonality analysis
Clustering

ABSTRACT

Huge amounts of data are produced inside an industrial production plant every minute. This data is getting more accessible by higher network and computing capabilities. This poses an opportunity to apply methods in real time to support the reliability of production machines. In theory every time series, that is currently monitored by for a breach of thresholds, can be extended with a forecast method. Classical approaches, such as ARIMA and Exponential Smoothing can be used for forecasting. To describe the signal and boost the forecast results we use a clustering method to group each unknown data stream in a seasonality class. This seasonality classes can be used for insight into intra and inter group behaviour between machines and add causality to factory wide correlations. We collected 10000 multiple day segments of multiple identical and different machines. We manually hand labelled the data segments for their seasonality pattern to compare and explain the clustering results. Classes, obtained through clustering, are used to adapt each single forecast model for every machine. For the forecast method we could show improved results by selecting the correct seasonality for each data stream.

1. Introduction

If we deal with big data, we often have a high degree of complexity and unknown information. In an industrial setup when analysing existing systems, the data patterns, characteristics and relationships of different sources are often unknown [1]. It is difficult to apply data analysis methods without these information.

In a previous study [2] we investigate identical lines of machines which are used for mass production. The setups are composed of identical machines by the same supplier but can show differences depending on location and product type produced. We observe the real-time sensor data of multiple production machines with unknown dependencies to either the previous production steps or environmental influences such as daytime or physical location. In this work we extend an outlier and trend detection system by adding a seasonality analysis to categorize the time series and make a prediction for the error probability. In our work [3] we compared the results of differently configured Holt-Winters setups depending on previous sorting of seasonality and trend. We aim to extend this method by a two step process of seasonality clustering 1.

We propose a system, that gathers data and adapts in an opportunistic [4,5] approach to use different techniques of data analysis. The approach allows the system to adapt without the necessity of manual actions by a controller. We implement a live system that is able to distinguish between a continuous erroneous change inside the data stream and isolated outliers which are caused by measurement errors for single work-pieces. We focus on continuous erroneous changes which allow the production worker to make corrections. A continuous erroneous change is defined, as a permanent shift of the observed variable towards the defined thresholds. We create a model, which allows to observe the machine park as a single entity with different seasonality patterns and their number of occurrences as characteristics. This information can be used by experts to directly counteract this behaviour, for results such as shift dependent changes, or weekly or bi-weekly changes due to cleaning plans. More complex measures, such as factory wide cooling systems can be planned accordingly due to the knowledge of the placement of machines with a daily seasonality.

We observe a production system where we get measurement values for every produced sample for multiple machines of the same type. Every sample is evaluated by a comparison to a positive and negative threshold. All produced parts need to be within these thresholds to be considered okay and not get sorted out. These tolerance intervals are manually set and can be shifted to

* Corresponding author.
E-mail address: sebastian.soller93@gmail.com (S. Soller).

https://doi.org/10.1016/j.bdr.2022.100335
2214-5796/

a small degree to adapt to small shifts of the measurements. If a sample exceeds the limits it is removed from the production process. It then gets manually checked for a measurement error and sorted out if the second test proofs the automatically detected error was correct. If multiple confirmed errors appear over a short distance the process parameters are evaluated by experts. We aim to start this evaluation process before an accumulation of errors appears. With this forecast the production time and more importantly faulty work pieces can be reduced.

The number of faulty parts can be reduced if a trend inside the data can be detected, and preemptive measures can be taken. We implement a system to dynamically select the best forecast method from different forecast models such as different ARIMA [6] configurations, exponential smoothing [7] and the Holt-Winters method [8]. The forecast can be compared to the positive and negative tolerance thresholds. To account for the data variance, we do not aim to predict the next sample of the time series. We make a forecast of the probability of a certain number of upcoming samples being outside the tolerance threshold.

We observe the differences between machines of identical construction type to select the forecast method. Depending on machine and production type behaviour, different forecast methods can perform differently. The selection is done by analyzing the seasonality [9] for each individual work-piece type and machine combination. For each machine, the optimal forecast method can result in a higher prediction accuracy when combined, in comparison to a single forecast method applied to all machines. Using this setup, a live adaptable solution for multiple production lines is created.

We conduct a pilot study to give experts insights into the machine behaviour and warnings of upcoming deviation. We implemented a system to detect trends and use a probability model to support the decision for the experts. The design goal was to keep the system simple and comprehensible in every step, which is why we do not use neural networks approach the problem. The decision is whether to intervene and for multiple erroneous trends on multiple machines a priority which machines need to be maintained first. For the pilot study we will not intervene on those errors but monitor if the predicted errors will occur.

We use seasonality information to boost the accuracy of the proposed prediction for the underlying system and use case. The further aim of the seasonality detection is to give insight into the characteristics of the observed production line or machine. The analysis of seasonality can give insight into unknown machine behaviour. A correlation between detected errors and error commonness for machines with seasonality and without seasonality can be examined.

2. Related work

To predict upcoming system deviations, we used a dynamic recognition system [2]. We collected time series data and analyzed the behaviour to detect a deviation from normal behaviour. The setup was used to detect critical errors which need to stop the process immediately to avoid damage to the production machines. These errors are sparse, but maintenance is done permanently on all production lines. In this work we aim to extend the system by adding a forecast for a threshold-based system. Instead of monitoring the time series for strong changes, this approach analyses the behaviour over a given time span. This analysis is used to estimate a probability of faulty parts for the next n samples. By combining both approaches we aim to alert the experts and workers on upcoming critical errors and support maintenance decisions on non-critical machines by ranking the faulty products probability.

The system we implemented fulfills the demands of a low cost, scalable, self-configuring architecture [10]. No additional manual input is needed to extend the system to additional production lines. The system is also capable of adapting to similar production system where single variable values are monitored by threshold systems. For the prediction of maintenance previous work uses machine parameters and product quality [11]. A study about the technical issues was conducted. The main issues for people inside the industry were gathered and the most valuable aspects were rated. The highest priority was the ability to continuously monitor the process parameters and if those range in between pre-defined thresholds.

We aim to monitor the condition of the produced samples. Previous work has shown the necessity to give an alarm before reaching a critical threshold. A one-step-ahead prediction of the time series created by a regression tree showed the potential for the machine condition prognosis [12]. The observation of a machine resulted in a low prediction error which was measured with the root mean square error (RMSE). We aim to extend the approach and make it suitable for a multi machine setup and compare the result of different prediction methods. We aim to predict multiple steps ahead and apply an adaption beyond a single observed system.

For seasonality influenced forecasts exponential smoothing can be used [7]. Exponential smoothing methods were evaluated and the result was different methods for different application cases, depending on data properties. They also found weaknesses in practical setups due to default parameters being set. Compared to the Exponential smoothing we will compare to neural networks approaches, such as a Deep Neural Network (DNN). Neural networks have been previously used to forecast seasonal data [13]. Long short term memory (LSTM) neural networks have been used to forecast data with seasonal components [14].

Classification for trends was used in previous work [15]. They used a trend classification algorithm on stochastic time series data. They showed that the proposed methods can detect local and global trends inside the data and therefore can improve the prediction methods for the time series. The historical trends are used to forecast the trend using different classification techniques such as Naive Bayes, Decision Tree or SVM to predict a given day. We aim to use classification of trend and seasonality to predict the probability of an error.

The Mann Kendall approach has been previously used in different domains, that are influenced by seasonal trends, such as trends inside water quality [16], and has also been modified to fit regional trends [17]. The application of this method to test for trends and seasonality has been implemented previously [18]. We aim to implement a similar method, with a focus on detecting and defining the seasonality, instead of removing the seasonality for trend detection. We aim to apply a model which considers the uncertainty of the prediction [19]. We will use a probability distribution to account for the variability of the data and the prediction results.

The system needs to be capable to divide between an anomaly in the data that can cause an error and anomalies caused by a seasonality. Previous work used an algorithm to flag data influenced by seasonality to exclude anomalies [20]. A cumulative sum control chart algorithm is used to detect anomalies inside the data set. A uniform frequency of two minutes is assumed for the data. Their algorithm uses the additional information of an optimized threshold and seasonality to boost the accuracy on their data set with seasonal changes. For our proposed system we do not detect anomalies by using detection methods. We aim to implement a method to find that seasonality influenced data sets and change the behaviour of the forecast. For each time series or subproblem as called by Torres et al. [21] an individual pre-

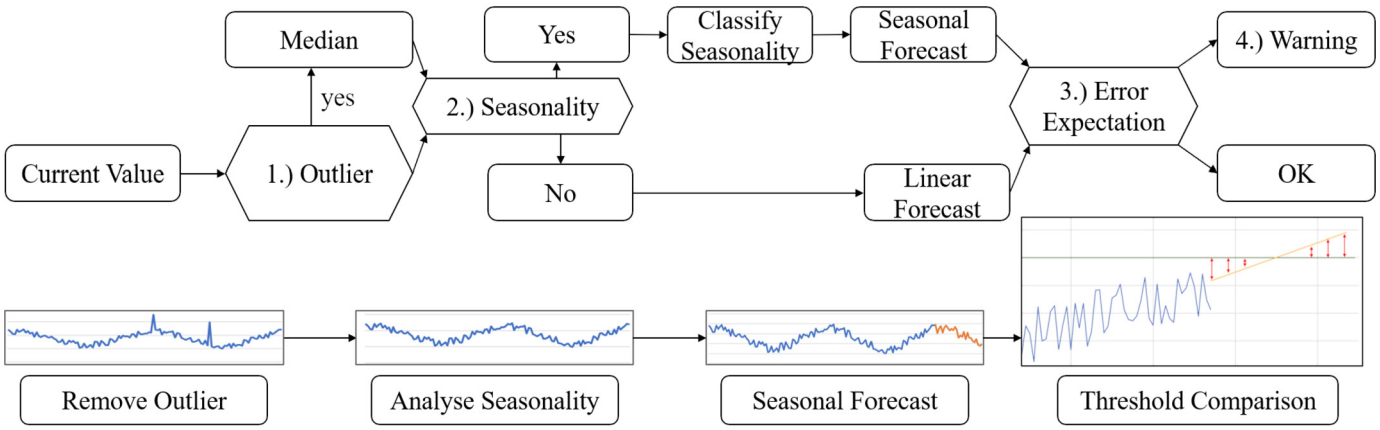

Fig. 1. For each time series a new incoming value is checked. 1.) Each detected outlier is removed by the median value of surrounding values. 2.) The seasonality of the time series is evaluated. Each time series is sorted into one of the three categories for seasonality, which are daily, yearly (long time), or neither. To predict the upcoming values of the time series a forecast method depending on the seasonality is used. 3.) The probability of system failures in the upcoming samples is calculated. 4.) If a high probability for a continuous error is detected and a warning is sent to a technician or foreman depending on the gravity of the error or the repeated occurrence.

diction method is selected. For their deep learning approach they carried out a grid search to optimize the hyper parameters. Single outliers inside the data need to be detected by our approach. Multiple approaches for anomaly detection have been proposed either using a statistical approach [22,23] or machine learning, such as neural networks [24]. These different approaches can be used interchangeable in the pipeline we aim to build.

In previous work the best forecast models were selected and compared to use the single most effective model for an underlying problem. We aim to extend these system setups by dynamically selecting the best forecast model for each underlying machine. We apply a clustering method to detect seasonalities in the data which can be a causality for relationships between different data [25].

3. System state and goal

We implemented the rule based seasonality detection system and used it for the feature selection in a live system as shown in Fig. 1. The adaptability of this system provided the advantage of a single configuration for multiple machines and machine groups. The only necessary information we need to configure by hand are either the endpoints of the Open Platform Communications Unified Architecture (OPC UA) or columns of a database. The information include the measurements, thresholds and machine identifier. From an initial setup of 48 machines, we use for the testing, we could extend the framework to 3 further groups of machines which include 93 machines with an implementation effort of less than one day. Furthermore, any machine with network capabilities and a necessity to monitor trends can be added into this setup. The system checks the probabilities of an error for each new measurement read or entered into the database. The main setup from a user perspective is to define the probability threshold, as it controls the number of warnings that would be sent out. For each work piece type we can train the model, as soon as we have a single session of the work piece recorded. For this first session, we can determine the seasonality, and boost the accuracy of the prediction model.

We recognize the error types (I) continuous errors and (II) outlier errors as it can be seen in Fig. 2. The system is capable to distinguish between continuous errors and outliers. A continuous error is a continuous change in the time series, which leads to a certain amount of erroneous work pieces. It is either a change in average, slope or variance, which leads to exceeding pre-defined thresholds. An outlier is a single observation or measurement

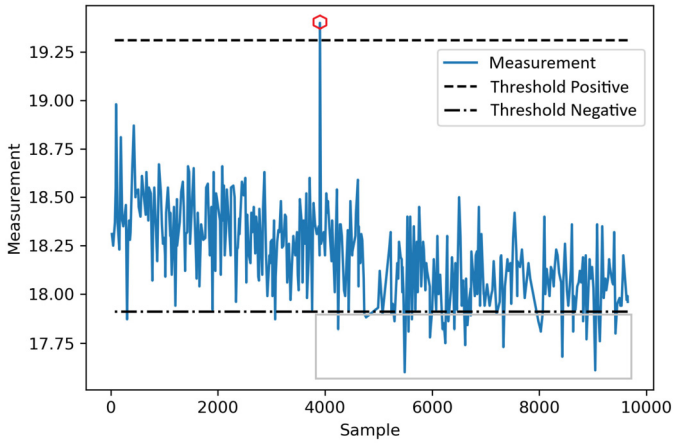

Fig. 2. The different error types that can occur in the system. The area in the box is a continuous error. For a continuous error, not every sample needs to be outside the threshold, but a certain amount of work pieces. For the forecast, the probability of a given number of work pieces being outside the threshold is used. The continuous error needs to be dealt with by making corrections. The highlighted sample (marked with a circle) is an example for an outlier. These outliers can either be caused by measurement or process errors, as for example human influences.

which differs from the median by an absolute deviation [22]. These outliers can be removed by a median filter. It is important for the system to be able to differ between a continuous error and an outlier error. This is important for the observers workflow. A continuous error needs to be resolved manually or adjusted by the system if only a small change appears. For the observed production setup an average of 23% of faulty outliers do not trigger any kind of process, where a manual correction needs to be done by a machine controller. Meanwhile 77% of faults, caused by a continuous error, need to be corrected in several steps on the production site.

We forecast the continuous errors for linear trends towards the tolerance values. The continuous errors can appear after a maintenance or through a trend towards the tolerance limits during the production process due to wear. Trends inside the data need to be observed and the probability of a continuous error triggers an alarm for the machine controller. We implemented a decision support system to decide which errors need to be reacted to on high priority. For each detected continuous error, we rate the probability of failure and rank it for the machine controller.

Each result gets sent to a database with the machine name and probability of failure in the next n steps, for multiple n. A business intelligence software is used to visualize the order of maintenance importance. For critical cases an automatic email is sent, which includes the machine identifier, probability of error and an image of the threshold comparison 2.

Beyond the information of the forecast, we had to question the origin of the seasonality. To get a deeper understanding of connections between this behaviour for multiple machines, we implement a system to recognize and sort seasonality more precisely. We can group together machines, which are assigned to identical seasonality categories, and investigate their common modalities. The fluctuations of data can not only be used to boost a forecast method, but are analyzed for their influence on product quality and if measures can be taken to counteract this fluctuation.

4. Method

4.1. Adaptive model

For the model adaptability we use a classification of the underlying machine and its behaviour. The classification is used to sort the data streams for each machine to the correct prediction method. The classes are created using the rate of the seasonality for each machine, stemming from wear of long time usage and surroundings as shown in Fig. 1.

Before detecting the seasonality of the time series, we preprocess the data. For each value of the time series we calculate the mean absolute deviation (MAD) [26] and the median $\overline{x}$ of the data for n values.

$$MAD = 1/n \sum |x_i - \overline{x}| \tag{1}$$

A value is defined as a strong outlier if it differs 3 MAD from the median, as it was shown to be a robust method in related work [22]. The outlier is replaced by the median of the time series. We decided to use this statistical method instead of a machine learning approach, as it allows us to clean the data on premise using a micro controller. For further applications on a centralized system with more computing capabilities we can look to use an auto encoder to detect outliers of the data. For the resulting time series with removed outliers we calculate the autocorrelation coefficient. The value with lag k is calculated by the autocovariance function divided by the variance function for the sample Y_i [27]:

$$r_k = \frac{\sum_{i=1}^{N-k}(Y_i - \overline{Y})(Y_{i+k} - \overline{Y})}{\sum_{i=1}^{N}(Y_i - \overline{Y})^2} \tag{2}$$

We calculate an autocorrelation on each data stream. The local maxima of each autocorrelation are calculated. We compare the local maxima to the number of daily samples to determine to seasonality of the underlying time series. The data we observe does not provide a uniform sampling frequency. The time each work piece needs for productions differs for every type and is influenced by necessary stops for refilling and reparation.

The expected seasonality is calculated. We determine the number of average daily and yearly measurement points and the correct location of the peaks in the autocorrelation function when a seasonality exists. We can visualize the results by comparing the actual numbers of samples for a day and the n-th peaks of the autocorrelation function in Fig. 3. For the autocorrelation coefficient in Fig. 3, we had an expected seasonality of 62. The daily seasonality was obtained by calculating the average of the daily work-piece count. The autocorrelation of the function had its local maxima at 0, 61, 123, 186 and 244. The value at the position 0 can be ignored, since it is the correlation of the time series to itself without a shift and is a maximum in any autocorrelation. For each further value we can divide the peak by its number. For the time series, we get 61, 61.5, 62 and 61. We can compare the expected seasonality and the position of the peaks and use the values as features for a classification. The seasonality classes are:

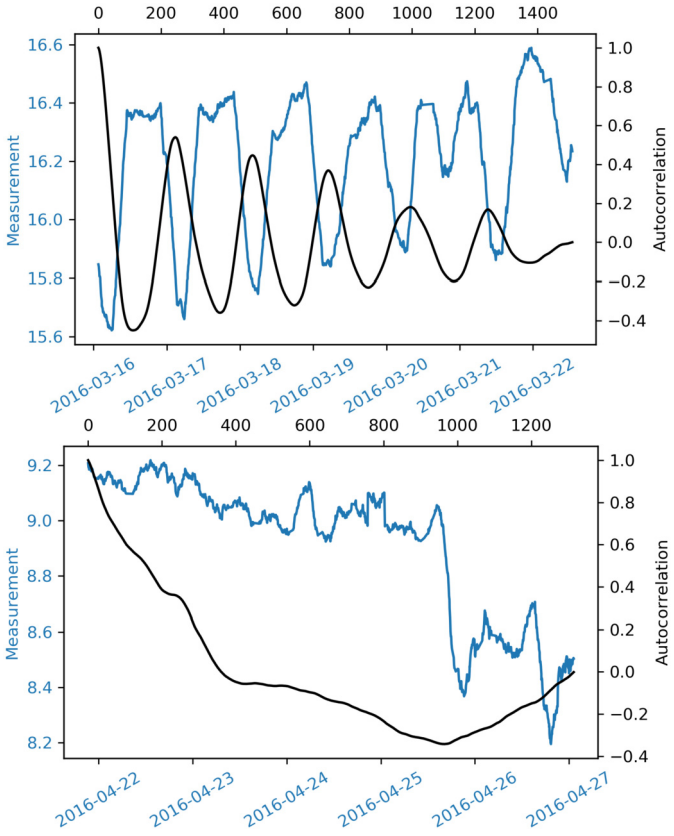

Fig. 3. The top Graphic shows a time series which can be identified as daily seasonal by visual inspection. Its corresponding autocorrelation function shows peaks at clear intervals. The bottom graphic is not influenced by seasonality. Its corresponding autocorrelation function does not show evenly distributed peaks.

- No seasonality: No seasonality or seasonality not matching the defined classes, such as weekly or bi-weekly is detected. These changes can mostly attribute to a trend instead of a seasonality over the production time of a single product type.
- Seasonality: We can detect a data fluctuation over different time frames. The peaks for the maxima and minima are at the same time for every day. Daily influences, such as temperature, humidity or the workers shift can influence the production process.

Using the results, we can use the features of the measured seasonality and the expected seasonality to classify the seasonality type of the time series. In this work we focus on the detection of the daily seasonality for the forecast. For a yearly seasonality detection, long time data over multiple production intervals for an identical product type are observed. The average values for the weekly production intervals over multiple months and years are used with an identical method, as is shown for the daily seasonality in this work.

For multiple time series we get the following matrix for a classification of the daily seasonality for 5 randomly selected time series:

$$\begin{matrix} 1st & 2nd & 3rd & Expected \\ \begin{pmatrix} 61 & 123 & 186 & 62 \\ 77 & 153 & 236 & 79 \\ 75 & 174 & 216 & 72 \\ 38 & 91 & 192 & 74 \\ 138 & 243 & 368 & 122 \end{pmatrix} \end{matrix} \tag{3}$$

Each row represents the 1st maximum, 2nd maximum, 3rd maximum and expected seasonality for a single time series. The expected seasonality is the average number of daily produced workpieces. The first 3 maxima were selected as most product types get produced for a minimum of 3 days. For comparability of each time series and classification we divide the maxima by the expected seasonality to get the matrix:

$$\begin{matrix} 1st/Expected & 2nd/Expected & 3rd/Expected \\ \begin{pmatrix} 0.98 & 1.98 & 3 \\ 0.97 & 1.94 & 2.99 \\ 1.04 & 2.42 & 3 \\ 0.51 & 1.23 & 2.59 \\ 1.13 & 1.99 & 3.02 \end{pmatrix} \end{matrix} \tag{4}$$

For the prototype study, labels are applied for a train data set to train a selected classification method. For the train data set we can manually label the matrix as seasonal or non-seasonal by optical inspection. For the presented matrix, the labels can be assigned as follows:

$$\begin{matrix} 1st/Expected & 2nd/Expected & 3rd/Expected & Seasonality \\ \left(\begin{array}{ccc|c} 0.98 & 1.98 & 3 & daily \\ 0.97 & 1.94 & 2.99 & daily \\ 1.04 & 2.42 & 3 & daily \\ 0.51 & 1.23 & 2.59 & no \\ 1.13 & 1.99 & 3.02 & daily \end{array}\right) \end{matrix} \tag{5}$$

If the time series is classified as a daily seasonality, a model with a seasonal component will be selected. A partial auto-correlation function (PACF) and an auto-correlation function (ACF) are used to determine the order of the components of the prediction model with a seasonal component.

4.2. Seasonality detection and classification

As an initial approach we apply different clustering algorithms to create seasonality classes. This method is based on the idea, that the data provided by the autocorrelation should be separable into different classes. Fluctuations over different time periods should be sorted into different classes. From the clustering we get the number of classes and an initial labelling for the classification process. Classification methods are compared using the classes defined by the clustering to classify new segments. We extend our simplified approach by using a row-wise normalisation of the Array (3) and using all maxima detected divided by the number of appearance. The resulting matrix is given as:

$$\begin{matrix} Expected & 1st & 2nd & 3rd & \dots & nth \\ \begin{pmatrix} 1.01 & 0.99 & 0.99 & 1.01 & \dots & x.xx \\ 1.02 & 0.99 & 0.98 & 1.01 & \dots & x.xx \\ 0.94 & 0.98 & 1.14 & 0.94 & \dots & x.xx \\ 1.34 & 0.68 & 0.82 & 1.16 & \dots & x.xx \\ 0.97 & 1.09 & 0.96 & 0.97 & \dots & x.xx \end{pmatrix} \end{matrix} \tag{6}$$

We evaluate the usage of two different feature sets for the classification methods. The first feature set includes all columns of Array (6) as features. The second feature set is calculated from the values of Array (6) to get a feature vector with a fixed length of three. The features of this vector are the expected seasonality, the calculated mean and variance, of all rows from the 1st to the nth maxima. For both datasets we apply identical clustering methods. For the clustering we compare the k-means, Agglomerative Clustering and DBSCAN method of the scikit-learn python library [28]. We apply the k-means method on the assumption of the seasonality classes: daily, yearly and neither seasonality which results in a three cluster setup. We also apply 4, 5 and 6 clusters and inspect the results for each cluster and their attributes. The Agglomerative Clustering and DBSCAN have the advantage of supporting an unknown number of classes. We optimize the DBSCAN function to have the least amount of outliers, while obtaining a reasonable amount of classes by tuning the distance parameter. Non-seasonal behaviour should be detected as outliers, since the noisy data will result in random maxima inside the autocorrelation.

Commonly used approaches were selected, to show the possibility of an automatic machine learning approach. We used a support vector machine [29] and a decision tree [30] for the classification of the clusters after the decision of the number of classes is made by the clustering. We use the classifier to classify each new segment in the live production process. Especially the decision tree was selected, as a system for this basic approach is similar to manually programmed if statements. This allowed us to inspect the results of data segments, which were predicted worse, than the average segment. The goal was to find an explanation why the forecast for specific product types and machine combinations was incorrect. If the prediction was worse due to a wrong classification of the seasonality, we could inspect the decision tree, which features, and values led to a wrong classification. This aspect of the process was done due to wishes of management, to communicate to engineering, which product types would not follow one of the described patterns.

4.3. Prediction model

If we aim to include seasonal trends when found, we can use the Holt-Winters method. We focus on the ARIMA, Exponential Smoothing and Holt-Winters in detail in our previous work [3]. The model allows observing seasonal trends inside the data for the forecast. For data without a seasonality we will use double exponential smoothing and for data only defined by a level we can apply simple exponential smoothing. The best forecast method for the underlying data stream is used as a result in values for the next upcoming pieces. After applying the best forecast method to get a forecast, we need to apply the uncertainty on the forecast values. The variance of the data is considered for our forecast. We calculate the standard deviation of the train data as a benchmark. For comparison to this benchmark we calculate the distance between the positive and negative thresholds of the system and the forecast obtained by the Exponential Smoothing model. We calculate an array that includes the distance for each sample of the forecast to each desired threshold. The array for the distance to the negative threshold is calculated by subtracting the prediction from the negative threshold value. For the positive distance we subtract the positive threshold value by the prediction. We visualise the calculation for the difference as an example in Fig. 4. Let *forecast(i,n)* denote the expected number of erroneous work pieces between times *i* and *i+n*. Let *probability(i)* denote the probability that the work piece at time i is erroneous. Assuming independence of errors, we get the equation:

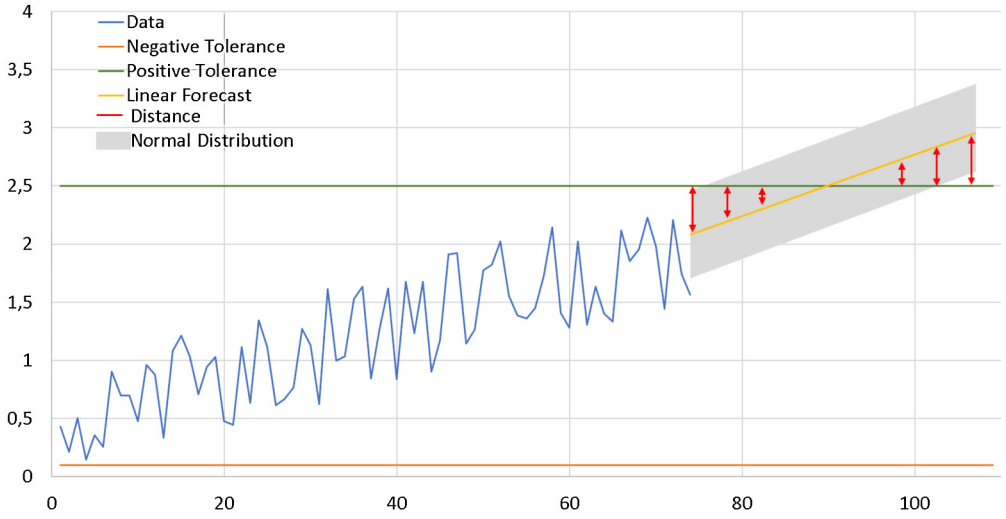

Fig. 4. A generated time series and its linear forecast. For each sample the value of the positive tolerance limit is subtracted from the linear forecast. For this example with the parameter n set to 5 to get the probability of the 6 upcoming work pieces i to i+5 we would obtain an array containing the values -0.4, -0.3, -0.1, 0.1, 0.3, 0.4. For the distance to the negative threshold we get the values -2, -2.1, ..., -2.7, -2.8. For every value we can use the distance to the tolerance line to calculate the probability of lying inside the tolerance lines. The calculation for the probability of being outside the positive tolerance is shown in Table 1. We did not include the negative tolerance in the presented table, as the probability of the forecast being below the negative tolerance value within a normal distribution is zero for all points within the forecast.

Table 1
We can get the distance for the next n samples of the forecast in Fig. 4. This distance is divided by the standard deviation of 2.7 to get the distance in standard deviations of for each point from the threshold. For the presented example we show the distance to the positive threshold, as the probability calculation for crossing the negative threshold was 0 for all values.

Distance	-0.4	-0.3	-0.1	0.1	0.3	0.4
Distance/Standard Deviation	-1.5	-1.1	-0.4	0.4	1.1	1.5
Probability	7%	14%	34%	66%	86%	93%

$$forecast(i,n) = probability(i) + probability(i+1) + \ldots + probability(i+n) \tag{7}$$

For each distance value we can calculate the amount of n standard deviations below the critical threshold. We calculate the standard deviation of the n preceding values of the forecast. Before calculating the standard deviation, we make the data stationary by differentiation. For the visualized data of Fig. 4 we get a standard deviation of 0.27. We divide the calculated distance of the positive and negative array by the standard deviation. We can derive the probability of each sample lying inside or outside the threshold by using the normal distribution. To assume normal distribution, we used a Shapiro Wilk test to proof this assumption [31].

For each sample of the prediction we can get the probability of error by using the normal distribution of the series and the position of the sample within the distribution. We get the table of probability of each point in the prediction of Fig. 4 by dividing by the standard deviation and looking up in a normal distribution table as can be seen in Table 1. We can define the amount of values that the system needs to consider in the forecast. We can select the forecast window manually for the prediction of a single value or extend to multiple values. For the decision of the time frame a fixed value of work pieces or the number of work pieces matching a fixed time frame can be used. The probabilities for all values of the probability row of the Table 1 are summed up. For the probabilities of Table 1 the expected number of faulty work pieces is calculated by adding the probabilities $0.07 + 0.14 + 0.34 + 0.66 + 0.86 + 0.93$. For this segment of 6 upcoming work pieces, 3 are expected to be faulty. We can rank each production line by its number of upcoming faulty work pieces. This ranking supports short time maintenance decisions, as the more critical lines get attention first.

For the evaluation of the system, we will make a comparison between the predicted amount of errors and the actual detected errors in the system. We need to exclude single errors since the system is designed to detect continuous errors and there is no way to reliably predict the measurement or process errors. The measurement and process errors can be caused by already slightly broken parts being processed by the machine.

5. Pilot study and data set

We collected a data set of 48 production machines with threshold monitored time series to apply the forecast. In a time frame of five years we collected 10137 segments which can be seen in Fig. 2 and 3. To test the proposed setup we implemented a data selection and prediction setup to compare the different forecast methods to each other. The data selected are the 100 most common product types. For each product type 20 segments were selected, each inheriting 3 to 7 days of continuous production. For the implementation of the ARIMA, Holt and Holt-Winters exponential smoothing approach we use the python library statsmodels [32]. We iterate over each segment to forecast the next segment of 20 work pieces depending on all earlier values of the segment. For each segment we calculate the mean squared error to evaluate the forecast. For each forecast model we get 100 times 20 mean squared error measurements to compare to each other for the best possible method.

To select workpieces which show a daily seasonality, where the method using the season and no trend produced best results, we use the expected seasonality and the n first maxima of the median filtered autocorrelation. These values are used for the clustering and classification. We separated the work-pieces fragments into a test data set and a training data set. The training data set consists of 500 random fragments and the test data set consists of the other 1500 fragments. The 500 segments of the training set were labelled according to their best performing method and a classifier was trained with the features expected seasonality, maxima one, maxima two, maxima three and the label method.

For the live production forecast we use the forecast and apply a distribution to every sample measured, as shown in Fig. 4. If the probability for errors surpasses a certain threshold, as for example 25% errors in the upcoming 20 workpieces, an alarm is sent out. This threshold can be changed freely by the recipients of the alarm and was selected individually, as technical staff wanted

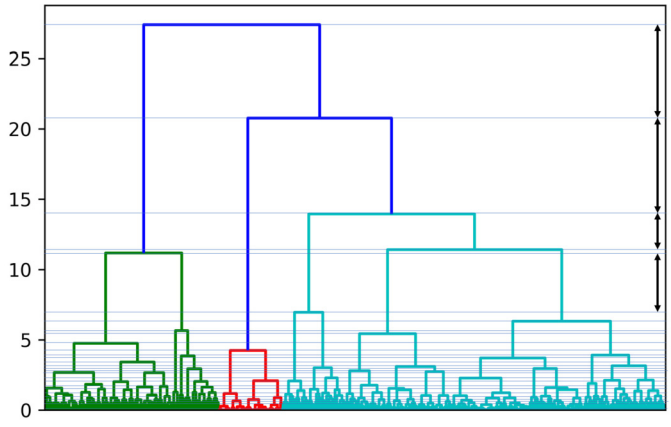

Fig. 5. Number of classes detected by Agglomerative Clustering depending on distance threshold and corresponding dendogram.

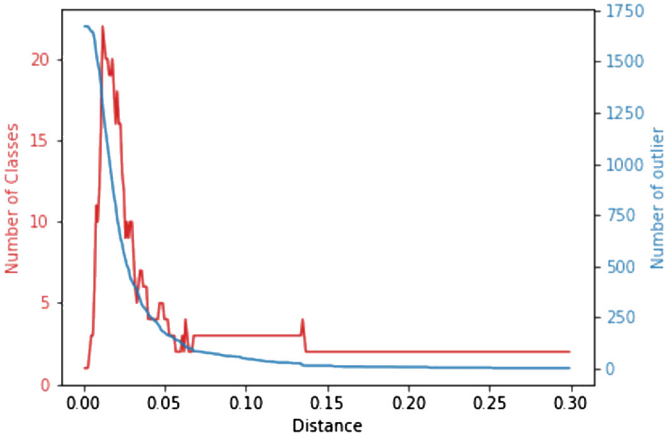

Fig. 6. Number of classes detected by DBSCAN depending on distance threshold.

Table 2
Accuracy of Decision Tree and Support Vector Machine.

Number of classes	2	3	6
Support Vector Machine (Maxima)	0.97	0.94	0.90
Decision Tree (Maxima)	0.99	0.96	0.94
Support Vector Machine (Statistical)	0.93	0.91	0.86
Decision Tree (Statistical)	0.95	0.93	0.91

to get informed on more nuanced changes, such as people on the shop floor. The implemented system is live as today and used by production to counteract erroneous machine behaviour and give a better understanding of difficult product types.

6. Results

6.1. Clustering and classification

For the agglomerative clustering we evaluated the distance threshold for the merging of clusters. Depending on the threshold we get a different amount of classes. Using a low threshold results in multiple classes up to the number of samples in classes for using a threshold close to zero. The dendogram in Fig. 5 was created using the ward linkage. In the dendogramm the biggest range is between three clusters and two clusters with a range of 7 and between two clusters and one cluster with a range of 6. The next consistent number of clusters are six clusters with a range of 5 and four clusters with a range of 2. For the DBSCAN method we compared the change of the distance metric to the number of outliers the system detects in Fig. 6. The number of classes is unstable with a high amount of outliers until we get 3 clusters and less than 10 percent of outliers. The method then stays stable on 2 clusters.

We applied the plausible cluster numbers 2, 3 and 6 obtained by the agglomerative clustering and DBSCAN using the kmeans clustering. The kmeans clustering resulted in similar clusters as the Agglomerative clustering. For the classification we used the cluster sizes 2, 3 and 6, with the classes assigned by the Agglomerative clustering in Table 2. The Decision Tree classification outperformed the support vector machine for every cluster number. For the support vector machine we get an accuracy of 0.97, 0.94 and 0.90. The Decision tree showed an accuracy of 0.99, 0.96 and 0.94. This was done using each maxima and the expected seasonality as feature. Using the statistical features mean and variance we achieve lower results.

6.2. Forecast

To evaluate the results of the prediction we use the RMSE, as we filter out outliers earlier which would strongly influence RMSE results. For all 100 work-piece types 96 showed the lowest RMSE for above 18 segments for a single selected method. Using only a seasonal parameter performed best for 42 types, using neither a seasonal nor a trend parameter for 27 types, using only a trend parameter for 16 types and a seasonal and a trend parameter for 11 types. The remaining workpieces showed a mixture of models as the best performing model for different segments. The overall performance for each work piece can be maximized by selecting the correct model before the forecast. For each single method the average error for all workpieces can be compared in Table 3. The DNN performs similar to the best exponential smoothing configurations with an RMSE of 0.049. The best single method for the underlying data set is the LSTM with an RMSE of 0.047. The mixture model performs 4.2% better than each single model with an RMSE of 0.045.
We can use the seasonality gained by the autocorrelation to select the correct method for every work-piece type and machine combination to achieve the best possible RMSE as can be seen in Table 3. By always selecting the best method during the online process we get the best possible performance for the overall system.

7. Discussion

An important idea of the clustering of the seasonality is to provide expert information in the first step. By recognizing big range cluster sizes, these cluster sizes can be analysed. The direct decision without manual assistance or analysis by a human is to take the cluster number with the biggest decision threshold range by the agglomerative clustering for further analysis. Further, the analysis of the cluster numbers can also give additional insight into the data. The biggest difference between the dominant cluster sizes 3 and 6 for the manually labelled dataset showed in Fig. 7, that the

Table 3
RMSE comparison of each method.

	LSTM	DNN	Season:No Trend:No	Season:Yes Trend:No	Season:No Trend:Yes	Season:Yes Trend:Yes	Mixture Model
RMSE	0.047	0.049	0.053	0.049	0.049	0.051	0.045

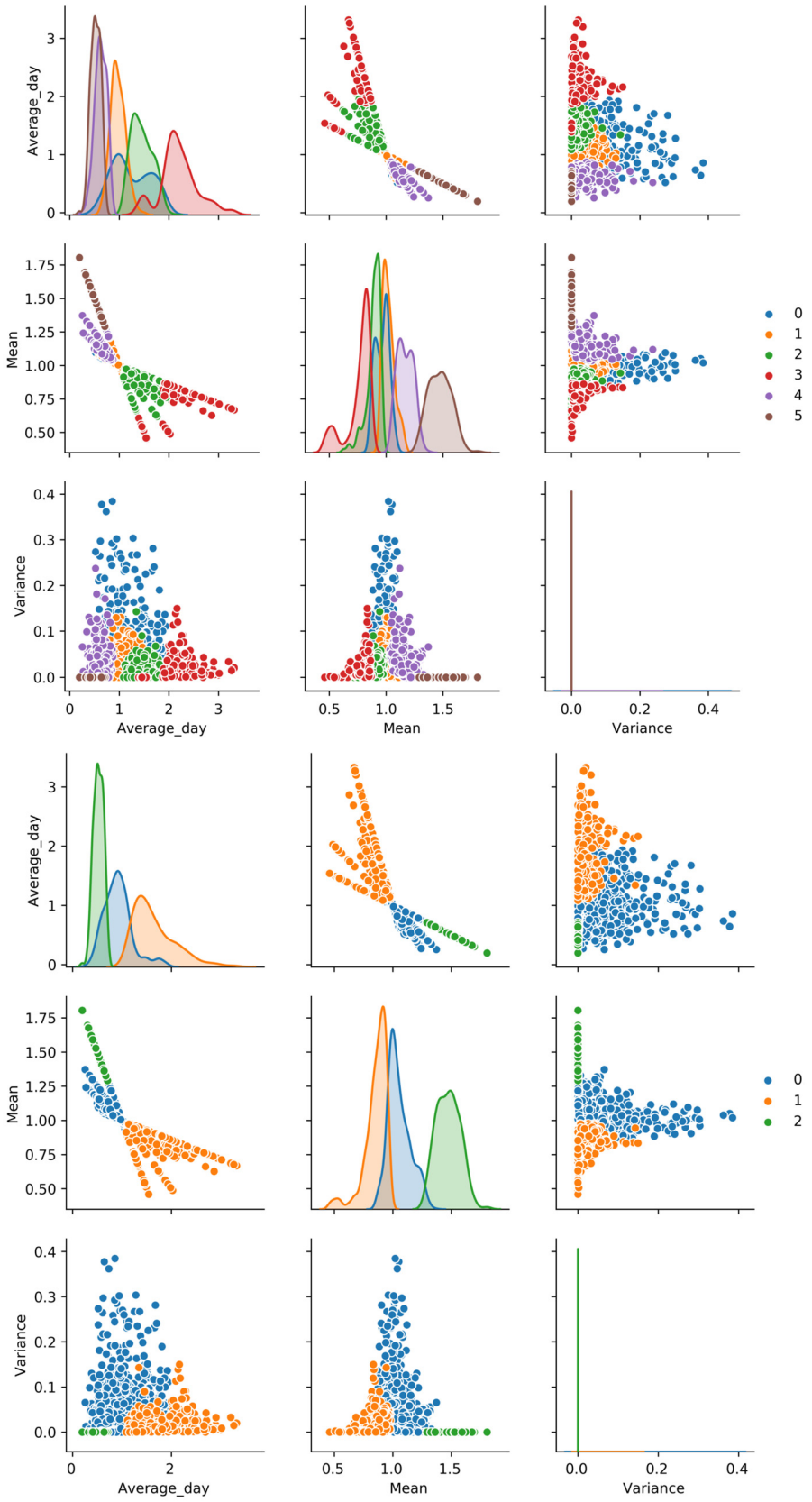

Fig. 7. Separability of classes depending on the selection. The Figure in the top shows the separation of the clusters using six and on the bottom using three, as pre-defined number of clusters. The classes two and three merge into the cluster one in the bottom. We selected the mean, average day and variance for an easier visualisation of 3 features instead of the average day length and 1st to nth maxima.

outcome of the three clusters could be labelled as daily seasonality, intra daily seasonality and no seasonality.

For the further forecast this detection would be feasible, as it does not matter for methods such as the Holt-Winters if the given frequency is daily or intra daily, as long as the correct frequency is selected. For an expert this information can provide valuable information. In a factory setting the knowledge about intra day seasonality can give additional information. The behaviour of different working shifts or underlying production processes, such as refill or maintenance cycles can be analysed. The applied cluster number for the further analysis and forecast can be decided by the department or factory wide by experts or automatically selected by a method finding the biggest distance in the dendogram in Fig. 5 between cluster sizes.

A challenge of this method is the evaluation of the clustering. The initial assessment and hand labelling of the data segments assumed only the daily seasonality and sorted the data into daily seasonality, no pattern and trend. Therefore we can only compare to this ground truth. The problem with re labelling the data for the 2 and 6 cluster setup is, that the expert would be biased, since we would create a ground truth specific for a result.

8. Conclusion and future work

Identical setups of production lines behave differently depending on the surroundings of every line. A forecast was used to estimate the amount of upcoming erroneous workpieces. The adaptive selection of the correct components and models for the forecast method has boosted the overall accuracy of the prediction and outperforms each single selected best model. The models only applying the trend or the season also outperformed the setup using both at the same time in our production setup. The dynamic model selection provides the ideal model for the observed online system and can be introduced to new types or production lines and attentively select the correct forecast model. We extend the setup we use on to different use cases, as we can access an array of data inside our network, where this kind of forecast is potentially useful. The strength of the application is the capability to extend to data on different application fields, such as machine temperature, water pressure and energy consumption. Using the detected seasonality each production cycle for a workpiece can be classified to a fitting forecast model. The information about the seasonality can be used by experts to analyze why the different production lines behave as they do. First inspections lead to influence factors, such as a location close to gateways or exposure to sunlight. The clustering of classes is extended by a classification model for the known classes from the ground truth. The classification, clustering, outlier detection and forecast methods can be adapted and exchanged and tested in future work, as we used basic statistical approaches for this work. The accuracy of the model classification is tested for different machine learning methods and changed features, such as the usage of more maxima or including the minima of the autocorrelation. The adaptive selection of prediction methods was implemented on different applications and use cases with or without a threshold-based system. We see our system as a step forward towards an autonomic intelligent system that uses dynamically configured recognition processes based on trend and seasonality analysis to raise the overall recognition performance.

Future applications can combine our previous approach [2] to detect impending errors which need to be fixed immediately and the approach of this work to support maintenance decisions depending on the necessity for each machine.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] X. Jin, B.W. Wah, X. Cheng, Y. Wang, Significance and challenges of big data research, Big Data Res. 2 (2) (2015) 59–64.
[2] S. Soller, G. Hölzl, M. Kranz, Predicting machine errors based on adaptive sensor data drifts in a real world industrial setup, in: 2020 IEEE International Conference on Pervasive Computing and Communications (PerCom), IEEE, 2020, pp. 1–9.
[3] S. Soller, M. Kranz, G. Hoelzl, Adaptive error prediction for production lines with unknown dependencies, in: Proceedings of the 10th International Conference on Web Intelligence, Mining and Semantics, 2020, pp. 227–234.
[4] G. Hoelzl, M. Kurz, A. Ferscha, Goal oriented recognition of composed activities for reliable and adaptable intelligence systems, J. Ambient Intell. Humaniz. Comput. 5 (3) (2013) 357–368.
[5] P. Halbmayer, G. Hoelzl, A. Ferscha, A dynamic service module oriented framework for real-world situation representation, in: The 6th International Conference on Adaptive and Self-Adaptive Systems and Applications, ADAPTIVE 2014, May 25–29, Venice, Italy, 2014, pp. 79–84.
[6] S.C. Hillmer, G.C. Tiao, An arima-model-based approach to seasonal adjustment, J. Am. Stat. Assoc. 77 (377) (1982) 63–70.
[7] E.S. Gardner Jr, Exponential smoothing: the state of the art—part ii, Int. J. Forecast. 22 (4) (2006) 637–666.
[8] C. Chatfield, The Holt-Winters forecasting procedure, J. R. Stat. Soc., Ser. C, Appl. Stat. 27 (3) (1978) 264–279.
[9] S. Hylleberg, Modelling Seasonality, Oxford University Press, 1992.
[10] V.C. Gungor, G.P. Hancke, Industrial wireless sensor networks: challenges, design principles, and technical approaches, IEEE Trans. Ind. Electron. 56 (10) (2009) 4258–4265.
[11] J. Lindström, H. Larsson, M. Jonsson, E. Lejon, Towards intelligent and sustainable production: combining and integrating online predictive maintenance and continuous quality control, Proc. CIRP 63 (2017) 443–448.
[12] B.-S. Yang, M.-S. Oh, A.C.C. Tan, et al., Machine condition prognosis based on regression trees and one-step-ahead prediction, Mech. Syst. Signal Process. 22 (5) (2008) 1179–1193.
[13] G.P. Zhang, M. Qi, Neural network forecasting for seasonal and trend time series, Eur. J. Oper. Res. 160 (2) (2005) 501–514.
[14] X.-H. Le, H.V. Ho, G. Lee, S. Jung, Application of long short-term memory (lstm) neural network for flood forecasting, Water 11 (7) (2019) 1387.
[15] L. Anghinoni, L. Zhao, D. Ji, H. Pan, Time series trend detection and forecasting using complex network topology analysis, Neural Netw. 117 (2019) 295–306.
[16] R.M. Hirsch, J.R. Slack, R.A. Smith, Techniques of trend analysis for monthly water quality data, Water Resour. Res. 18 (1) (1982) 107–121.
[17] D.R. Helsel, L.M. Frans, Regional Kendall test for trend, Environ. Sci. Technol. 40 (13) (2006) 4066–4073.
[18] M.M. Hussain, I. Mahmud, pymannkendall: a python package for non parametric Mann Kendall family of trend tests, J. Open Sour. Softw. 4 (39) (2019) 1556.
[19] M. Compare, E. Zio, Predictive maintenance by risk sensitive particle filtering, IEEE Trans. Reliab. 63 (1) (2014) 134–143.
[20] A. Murugan, Analytical algorithm to detect anomalies with seasonality patterns using machine learning techniques, Int. J. Pure Appl. Math. 119 (7) (2018) 149–154.
[21] J.F. Torres, A. Galicia, A. Troncoso, F. Martínez-Álvarez, A scalable approach based on deep learning for big data time series forecasting, Integr. Comput.-Aided Eng. 25 (4) (2018) 335–348.
[22] C. Leys, C. Ley, O. Klein, P. Bernard, L. Licata, Detecting outliers: do not use standard deviation around the mean, use absolute deviation around the median, J. Exp. Soc. Psychol. 49 (4) (2013) 764–766.
[23] J. Hochenbaum, O.S. Vallis, A. Kejariwal, Automatic anomaly detection in the cloud via statistical learning, arXiv preprint, arXiv:1704.07706, 2017.
[24] R. Corizzo, M. Ceci, N. Japkowicz, Anomaly detection and repair for accurate predictions in geo-distributed big data, Big Data Res. 16 (2019) 18–35.
[25] H. Hassani, X. Huang, M. Ghodsi, Big data and causality, Ann. Data Sci. 5 (2) (2018) 133–156.
[26] T. Cleff, Deskriptive Statistik und Explorative Datenanalyse, Springer, 2015.
[27] R. Adhikari, R.K. Agrawal, An introductory study on time series modeling and forecasting, arXiv preprint, arXiv:1302.6613, 2013.
[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[29] A.J. Smola, B. Schölkopf, A tutorial on support vector regression, Stat. Comput. 14 (3) (2004) 199–222.

[30] S.R. Safavian, D. Landgrebe, A survey of decision tree classifier methodology, IEEE Trans. Syst. Man Cybern. 21 (3) (1991) 660–674.
[31] P. Mishra, C.M. Pandey, U. Singh, A. Gupta, C. Sahu, A. Keshri, Descriptive statistics and normality tests for statistical data, Ann. Card. Anaesth. 22 (1) (2019) 67.
[32] S. Seabold, J. Perktold, Statsmodels: econometric and statistical modeling with python, in: Proceedings of the 9th Python in Science Conference, Austin, TX, vol. 57, 2010, p. 61.