Semi-Unsupervised Segmentation of Industrial Time Series Data

Dr. Albert Krohn, 21data.io
Adelina Barbur, Softing ROM s.r.l.
Dr. Christopher Anhalt, Softing Industrial Automation GmbH

In this article we want to share our experience on applying unsupervised clustering on a large industrial time series data set. The study was conducted during a project for the root cause analysis of system failures of a printing machine. Clustering was used to segment the time series data and classify segments as system states. We target this paper to an audience of engineers and data scientists working on industrial data and specifically focus on hands-on experience and practical obstacles. Throughout this paper we solely look at k-Means for clustering for its widespread use and ease of application and interpretation of the internals of the algorithm and results. We report on the experience and discoveries we found using k-Means for clustering and the influences that come from various preprocessing we apply on the time series data before and during clustering. We also motivate our work in the broader context of autonomous analytics.

1 Introduction

In the past few years, the awareness of the value of data has been spread over various markets. Whereas working with data in the finance area (stock market, fraud detection etc.) is a common discipline, the use of data in the industrial area, specifically manufacturing is still in its early days. In most cases it is limited to KPIs[1] (key performance indicators) like yield, quality, throughput and other metrics to be used by the supervising MES[2] or ERP[3] systems. Just recently the interest increased in using machine data on the production floor for optimizing processes, understanding influences and interactions of system variables or finding root causes of machine failures. This rising interest has been developed by world-wide trends named under Industry 4.0 or IIoT[4]. These movements proclaim the automatic and efficient use of data that is continuously produced by each step of a manufacturing process to feed knowledge to higher systems or back to the process with the aim to improve, automate and individualize or monitor the whole production chain and all involved machinery.
With that vision in mind, and the chance to enter into a new field of data mining that accounts for a large portion of data available on the planet, many companies and start-ups have been focusing on providing platforms and analytics tools to implement and deploy analytics on industrial data. Still, we feel that there is a gap between the acquisition of data and the tools and platforms for data storage and processing. This gap is normally filled by a professional data scientist working on the data and using tools to eventually extract the wanted knowledge from the data (if possible) and materialize the business value of IIoT. With this article we contribute to the field of automating machine learning algorithms for the use in industrial applications

2 Use Case and Motivation

While the majority of projects in the IIoT area still a represent rather manual and explorative data science job, we envision a system which closes the mentioned technology gap between the data lake and the data analytics tools by automating most of the manual steps a data scientist would go through. This research area has recently become known as *autonomous analytics* and is closely related to the field of *guided analytics*. Both disciplines try to minimize the interaction and necessity of a professional data scientist, whereas the latter automates the execution of data analytics based on commands of an expert user, while the former aims to completely automate a data analysis process from ingestion of data until the presentation of the result.

[1] KPI: key performance indicator
[2] MES: manufacturing execution system: a software to control and automate the whole production flow
[3] ERP: enterprise ressource planning system
[4] IIoT: Industrial Internet Of Things

2.1 Industrial use case

In this article we look at the industrial use case of gravure printing machines. We look at a sub type of rotational machines that is typically used for printing high circulation products like newspapers or advertising flyers. Those machines normally fill a whole production hall spanning 100m in length and several floors in height and run at a speed of up to 50km/h. Paper is fed into the machine from large paper rolls (typically 1.5m in diameter and up to 4m in width, several tons in weight) and runs at full speed through the machine where printing, cutting and folding takes place to produce a ready to use output product like a newspaper. To achieve maximum uptime and output of those machines, several strategies have been developed to be able to keep the printing and production process running all the time - even during the change of a paper roll. The basic idea is to glue a new paper roll at the end of a used-up paper roll and thus provide an endless flow of paper into the printing machine. Various strategies have been developed to change a paper roll during the run of such a machine like building up a repository of paper from the old paper roll in the machine, glue the start of a new paper roll on the end of the repository. Others hold both a new and an old paper roll at full speed and stick both paper webs together and cut the old paper roll at the same time during the production run.

2.2 Root cause analysis

For the execution of root cause analysis, one would intuitively look for explanatory modeling, but a root cause analysis process will often comprise of a predictive modeling component as well. Despite a seemingly obvious distinction between the two, predictive and explanatory models usually get conflated.

Explanatory modeling focuses on testing causal hypotheses, whereas predictive modeling is used for predicting new or existing information. The difference between the two approaches is rooted in the philosophical yet statically relevant "correlation vs. causality" discussion. While correlation represents a relationship between factors, which can be positive or negative, it does not imply causation. Often, causality is difficult to prove and requires special experimentation.

If for predictive models the existence of correlation will suffice, explanatory models require an underlying causal hypothesis to be tested. The design of the analysis process starting from data preparation and ending with validation methods will vary depending on the scope. Dimensionality reduction is a common practice in data analysis; however, this is used differently in a predictive/ explanatory context. Used in predictive modeling, compression methods such as PCA[6] or SVD[7] can reduce the sampling variance and improve the algorithm's performance. In explanatory modeling, PCA[6] serves as a validation tool in the pre-testing phase.

In explanatory modeling, simple and transparent models are preferred due to their interpretability, hence the popularity of statistical based models, especially regressions. Complex "black box" algorithms such as neural networks are especially suited for predictive modeling, due to their high accuracy and robustness, but lack the insight to explanatory jobs.

When it comes to model validation and evaluation, two different performance criteria are being considered: explanatory power vs. predictive power. For explanatory models, assessing the strength of relationship depicted by the model is key. Evaluation and validation methods include specification tests (Hausman), goodness-to-fit tests, R type statistics and statistical significance tests. For predictive models, the focus is on good predictive accuracy and generalization ability, without overfitting the data. Therefore, validation and evaluation concentrates on examining overfitting, by comparing results from training data and hold out data sets. (Shmueli 2010; Shmueli and Koppius, n.d.)

Our use case is to find the root cause of production failures during the printing process, where the dominant problem is that of paper rips. A rip of paper needs a manual stop and re-setup of the whole machine thus causing downtime and affecting the machine's utilization, i.e. the OEE[5], which is an important performance indicator in manufacturing.

Several strategies in the data science for the root cause drill down of such a problem can be followed on:

- anomaly and outlier detection in time series processes
- condition monitoring techniques in the frequency domain (spectral analysis of vibrations etc.)
- variable influence on system events
- combinations of the above and many more

(Solé et al. 2017)

[5] OEE: overall equipment efficiency
[6] PCA: principal component analysis
[7] SVD: singular value decomposition

2.3 Data science challenge

In this article we want to focus on the third mentioned approach to root cause analysis and build a system to look at the variable influences on the event of a paper rip. The idea is to teach a predictor like regression or a decision tree to predict the paper rip event and then trace down the influence of variables or their interaction to find the root cause of the event in the system. There is a rich collection of papers targeting this approach, such as (He, Chen, and Yang 2014; Guo and Kang 2015; Demetgul 2012; Chen et al. 2004). This is a standard technique in root cause analysis of labeled data. But before we can work on the predictor, we have to segment the time series data of the machine, identify events and use the incidents to extract labeled training data and then train a predictor. Here again, we have several options to do that. The most intuitive way during an explorative data analytics session would be to use expert knowledge on the data and annotate manually the occurrence of such events. During that process we encountered several obstacles:

- even for an expert it was not always easy to find the events by just looking at plotted curves of measurements
- the pure amount of data we covered in our study would take days to be completely manually annotated
- the time-precision of manual annotation is limited

So we continued on this idea with the help of pattern mining. We simply extracted a pattern from the annotation and used that for mining repetitive occurrences. This worked surprisingly well but still needs an expert to find and precisely annotate multivariate time patterns to be used for event mining. It is similar to attach an additional sensor to the machine to actually inform about the situation with e.g. a light break on the running paper web.

From the perspective of autonomous analytics this is not a satisfying situation: we still need an expert (annotation) or even invest money and engineering effort (additional sensor) to support our very first step of the data science process. And there is more to be considered: Using pattern mining, we could not distinguish between problems that occurred during maintenance of the machines, trials and other states of the machine.
So the analysis of this brings up the question: can we - instead of using more and more expert knowledge and effort - by the means of machine learning determine the *system state* of the machine?

If we were able to do that, we would still need another step to select the system states of interest, especially those where certain system states occur in sequence. But the *automatic classification of the system state* of the machine will help reduce the interaction between experts and algorithms to a minimum, and still provide us with equally valuable information and a precise annotation in time.

In this paper we will show how k-Means clustering can indeed be used to identify system states in our use case and what properties of k-Means need attention when applied to such a data challenge.
The idea of automating the root cause analysis and taking clustering for the segmentation of data brings us now to the following chain of algorithms we designed for the root cause analysis:

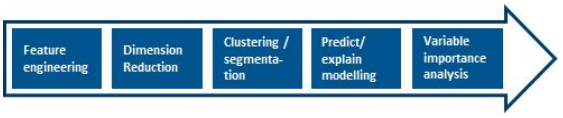

Figure 1 - Autonomous analytics for root cause analysis

2.4 Using clustering for system state identification

To support the intuition of our approach, we first want to give a visual example of what clustering of system states is, how it looks on our data, and discuss why we expect that it generally works.

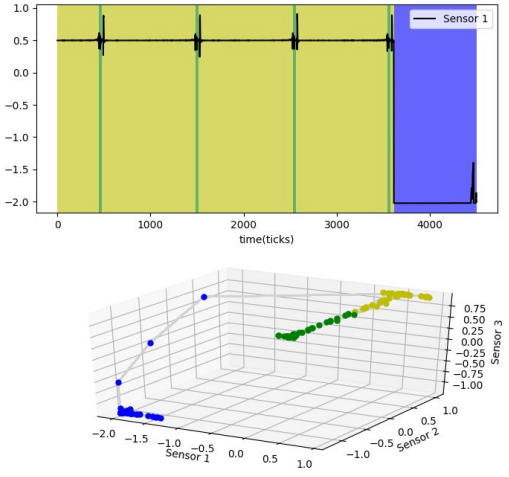

Figure 2 - Time series clustering and trajectory

Figure 2 shows a sensor plot over time, and the result of clustering into 3 states as a background color. On the upper image, we see how the clustering separated the system states we were

looking for. For this example, we used several manually selected variables. On the lower image we see the same time area covered but the data plotted as 3d trajectory over the first 3 sensors involved in the clustering. We also find Sensor 1 again on the x-Axis and see that the blue area of the upper picture can be found on x-Axis at approx. "-2" in the lower left corner. The clustering can be found on the second image as coloring of the measurements on the trajectory over time. It is easily visible that the different states the machine passes over time can be separated in the 3D-space spanned by the first 3 sensors. In the following chapter we will extend this concept and look at the machine trajectory in an n-dimensional space spanned by features we derive from pre-processing the data. These features (dimensions) can be

- the raw (normalized) data itself like in this simple example
- features derived from the raw data through preprocessing like combination or sliding window processes
- abstract coordinates resulting from a transformation like PCA[6] on the pre-processed data

When we interpret our time series data as a trajectory in an n-dimensional space, we can - like Figure 2 intuitively shows - separate the system states by finding clusters of points in the space. It is obvious that the number and selection of variables, the post processing like sliding window aggregation as well as a possible projection of data (like PCA) will affect the capability of a cluster algorithm to separate the system states just from the point clouds in the space. We will discuss exactly those influences in the following chapter and then conclude how it affects an autonomous way of processing and propose a pragmatic solution.

3 Clustering of Time Series Data

Using clustering algorithms to identify areas of interest in any sort of data is not new. There is numerous related work, especially for the application of clustering on time series data. For completeness, we would like to direct the interested reader to survey papers like (Aghabozorgi, Shirkhorshidi, and Wah 2015) or (Rani and Sikka 2012) as a good starting point. We would also like to reference on (Lin, Keogh, and Truppel 2003) and point out that we are not using sliding window techniques to find patterns in the time-domain with the samples of windows being the dimensions, but will instead use the features-space (sensors, PCA, …) as our dimensions always. So we will be looking for clusters in the space spanned by variables like in Figure 2. When using clustering algorithms, there is also the problem that distance measurements lose their meaning when approaching high-dimensional spaces. This phenomenon is discussed in detail in (Aggarwal, Hinneburg, and Keim 2001). So we have to keep an eye of the number of dimensions during preprocessing to avoid that this effect negatively influences the clustering algorithm.

3.1 The test data

The test data is an excerpt of approx. 8 hours of machine data with approx. 100 sensors sampled parallel at 2 seconds rate (~14k data vectors).

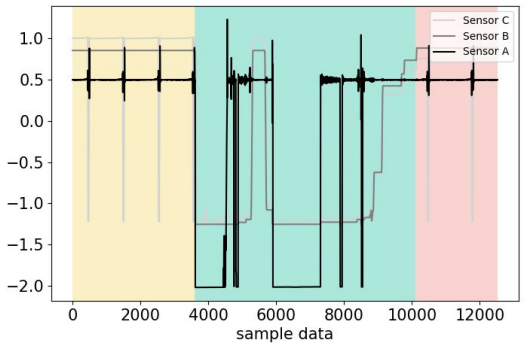

Figure 3 - the test data

Our test data is basically divided into three areas (i.e. three system states): a normal operation (yellow), a machine failure (green), and normal operation again (red). During the normal operation, we also see some periodic behavior (repetitive spikes). This is another system state we eventually want to distinguish. It is the changing of paper rolls during a printing process.

3.2 Normalizing data

As a first step of preprocessing data, it is often very helpful to normalize or standardize data. For k-Means clustering it is an essential and mandatory step as we will see in short. The intuition of it is easy to understand: K-Means uses the Euclidian distance as a measure for proximity and equally weights all dimensions. If one dimension uses a totally different scale (e.g. much higher values), it will influence the distance measure a lot more than the others and bias the clustering.

In Figure 3 we see the results of clustering (k=3) with raw and normalized data. We can see, that on the raw data, the normal operation on the left and right side is not put into the same cluster, which in

[6] PCA: principal component analysis, here used for dimensionality reduction

our understanding will be an unusable clustering, plus on the right, there is no separation of the periodic system state anymore. Obviously, one or two variables dominated the distance measure.

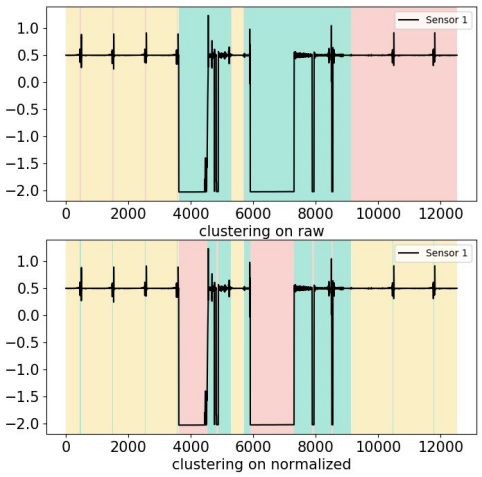

Figure 3: Clustering raw and normalized

3.3 Feature engineering

Feature engineering is one of the most important and most challenging tasks in data science. It is a process of data preparation before we put that data into an algorithm like clustering. The idea is that - instead of using the data as is - we derive new variables from the data and use those instead of the raw data. The choice of transformation is totally free. Typically, approaches like linear combination of inputs or polynomial functions on the input variables are used. The example in this chapter uses another often used technique: we produce statistical properties of the signal with a sliding window: We apply a sliding window on each sensor separately and derive statistical measures like mean, std, kurtosis, skewness that characterize the behavior of that signal inside the current sliding window. Those statistical measures are then taken instead of the raw data. Doing so, we multiply the number of variables by the number of statistical measures we take: the resulting feature-space is much larger than our variable space. Whether or not this is useful and which transformation is the best for the underlying data is - as mentioned before - the process of feature engineering. There is no one-fits-all recipe. In Figure 4 we give an example of the outcome for a certain selection of sensors. Our goal was to make a good separation of the system state. The second image is a sub-selection of variables of the system where the clustering is not able to find the clusters we are looking for anymore. Then when we apply the sliding window process on exactly that data selection and with the derived features we improve the discriminative power for the system states: the spikes on the first and last third of the time are separated again.

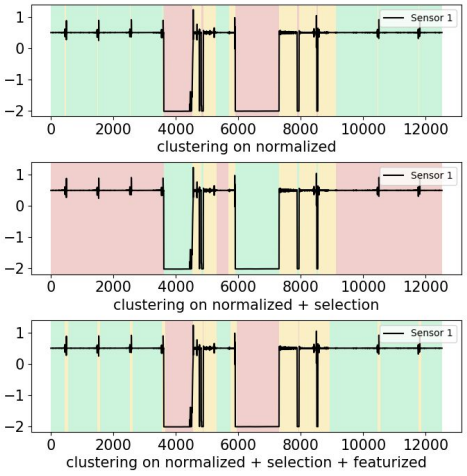

Figure 4: Clustering normalized and featurized

3.4 Dimensionality reduction of data

When using a high dimensional space as the basis for a cluster algorithm, we encounter the problem, that the power of discrimination decreases with increasing dimensions due to the influence on the distance measure in a high dimensional space (like discussed in the introduction of this chapter). It is not easy to say which number of dimensions should be considered high, but as the effect is gradual, it is always worth considering dimensionality reduction to improve the discrimination ability of a clustering algorithm which uses a distance measure like the Euclidian distance. In our case, we use PCA for this, which basically orders the variables by their variance and then puts them together in a linear combination, thus creating a new feature space of uncorrelated orthogonal dimensions. PCA makes sure that the newly created features show high variability over time, so these features are a perfect starting point for a clustering algorithm. Figure 5 shows the result. In our case, we set the PCA reduction to 90% explained variance, resulting in a dimension reduction from 372 down to 20 on the example of Figure 5. The clustering results are exactly the same, but we can now profit from the strong dimensionality reduction and lower the k-means calculation time from 9.0s down to 200ms, which is another important aspect of dimensionality reduction.

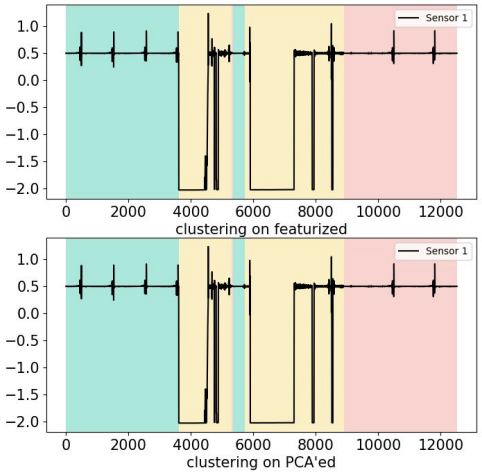

Figure 5: Clustering normalized and PCA'ed

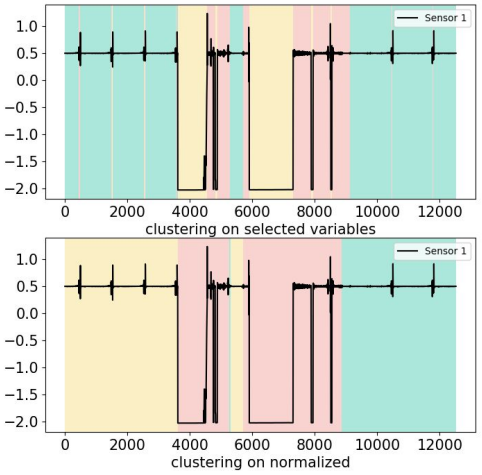

Figure 6: Clustering with variable selection

3.5 Feature engineering on variables

The most obvious way to improve the capabilities of a clustering algorithm is to manually select "features" of the existing data set which explain the system states the best. For the data set we look at, there is some expert knowledge needed to understand the variables and their meaning in the system. The overall machine is divided in several parts, each of which is responsible for different production steps like

- feed in the paper from a paper roll
- hold constant tension and speed on the paper feed
- change the paper roll during a running process
- print different color layers on the paper
- cut the paper into stripes
- cut the stripes into parts
- fold the paper to form the final product (newspaper, catalogue…)

Now, if we talk about a whole system, we must be very precise which part of the system is meant. Let's say we consider the process of changing a paper roll during the ongoing process as a *system state*, would that also be a system state for the printing area of the same machine? Such questions can only be answered by experts operating the machine - this is where *domain knowledge* and the *data model* come into play.

So, if we are interested in the system states "paper roll change", we should rather look at the variables that are directly involved in that process to get a useful separation by clustering. Let's consider the basic data model, where variables are now grouped along their mechanical position inside the machine. We compare the clustering result of variable selection in Figure 6.

We can clearly see that the clustering using all variables does not distinguish the spike areas on the first and last third of the data anymore, whereas the clustering on the selected variables does that indeed. So with the selection of variables that belong to the machine parts we want to gain insight, we can in fact gain better and more insight than just using all variables

3.6 Interactive hierarchical clustering

In the previous chapter we have illustrated probably the most important aspect of unsupervised learning on such datasets: Including expert's knowledge into the process of data mining seams to make a lot of sense[7]. So if we take this idea further (especially for k-Means), we will end up in an interactive process, where we iteratively cluster the data and evaluate the result with experts. We basically fit the parameters of our process (feature engineering, parameter k) to the expected result, which is not given as a verification set but given as the expert sitting in front of a visualization. In this subchapter we want to improve the clustering iteratively and hierarchically cluster the states by just varying

- the number of clusters k
- areas to merge or re-cluster

We do no specific selection of variables (take all) nor any preprocessing besides normalization. In Figure 7 we can see the result of the interactive clustering as images top down:

[7] Note that this is a fundamental change moving from totally unsupervised to a semi-unsupervised approach as we now include domain knowledge to *evaluate* the results of our machine learning process

1) cluster all data using k = 2
2) re-cluster[8] the first cluster using k=3
3) merge the re-clustered classes blue and violet to just blue
4) re-cluster all the data with a k=6

We can clearly see that the hierarchical clustering and merging which is done interactively with the use of expert knowledge leads to the requested result (sub-image "merging") in just 3 steps, whereas the pure clustering (the last image), even with more clusters, does not produce the required results.

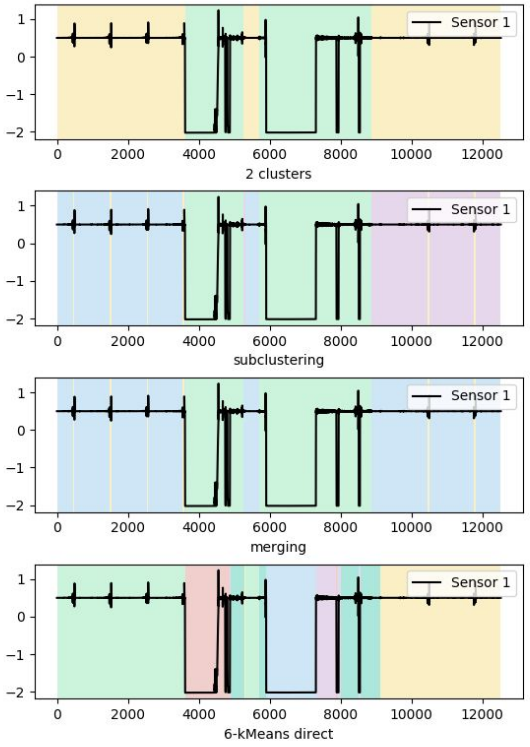

Figure 7: Hierarchical clustering

Instead, most of the variation is found in the middle of the time area and without any further guidance, more and more clusters will be found there.

4 Conclusion

In the previous chapter we have shown practical results on different approaches on the time series segmentation of our example data set using the well known k-Means clustering algorithm. Different preprocessing was applied to the data such as

- sub-selection of variables involved
- transformation with PCA
- hierarchical clustering
- interactive / guided analytics
- featurizing data with sliding window

The target of the clustering approach was to mathematically support the autonomous analytics for the root cause analytics presented in chapter 2. The results clearly show that there is a danger to produce mostly unuseful results when there is no expert supervision involved. We can see that e.g. in the case of the 6-kMeans clustering in Figure 7: we get clusters, but not the ones we were looking for.

This is an important finding in the context of autonomous analytics. Based on the experience on this data set and others we have worked on, it seems unfeasible to use a completely unsupervised approach for the first step of our autonomous analytics chain. In other words: we can't do an automatic system state detection without involving domain knowledge in this step.

Still, with minimum involvement of domain knowledge like the guidance of the hierarchical clustering or the selection of variables, we see satisfying results.

4.1 Pragmatic best practice for clustering time series with k-Means

As there is no clear winner in the list of approaches we selected, and there are many more preprocessing possible which are not discussed here, we will now combine the best approaches into a pragmatic engineering solution. We propose a combined approach that should

- minimize user interaction
- minimize inclusion of domain knowledge
- run fast and efficient
- be able to be generalized for other data sets

From the experiments results, we would now

1) select variables using the domain knowledge (was shown to be superior to just taking all data)
2) generate simple statistical features (e.g. mean, std, min, max) with a sliding window (was shown to increase discriminative power in some cases)
3) do PCA to avoid the problem with distance measures in high-dimensionality spaces (Aggarwal et al. 2001) (it was also shown to speed up the process significantly)
4) do an interactive hierarchical 2 step clustering of the maximal complexity of
    a) clustering
    b) merging clusters
    c) sub clustering

[8] the re-clustering with k=2 would have been the intuitive selection, but unfortunately, using k=2 didn't separate the spikes from the flat areas, only k=3,4,5 did the separation.

d) merging clusters again

In Figure 8 we see the results of exactly these process steps. The iterative clustering needed just k=2 and re-cluster with k=2 compared to the version in chapter 3.6 where k=2 was followed by a k=3 plus merging the results of the sub-clustering. The used variable selection and PCA pre-processing speeds up the k-Means in our example to 8 ms compared to 9 seconds (for the case of using all variables and featurize them). The results look as expected and involve only minimal user interaction (select variables and select the area to sub-cluster or merge)

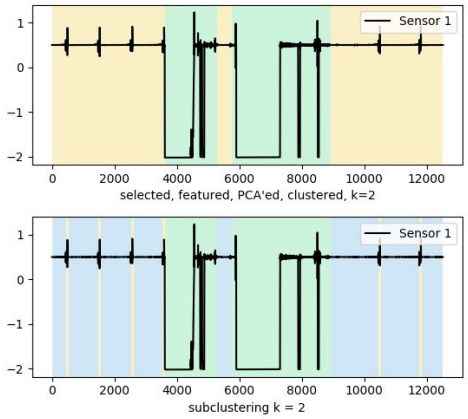

Figure 8: Proposed solution for state clustering

5 Outlook

We motivated our work in the context of autonomous analytics on time series data. Before applying any other anomaly detection, predictive or explanatory modeling, it is in most cases necessary to segment and classify the time series data. Doing so, we can e.g. avoid confusing online-learning algorithms for anomalies with the data produced during the setup of the system or maintenance of machine parts. Equally - explanatory models can only work on labeled time areas to find interesting changes of a state to another. We have shown that following some simple and pragmatic guidelines we can achieve a robust and useful state separation for the machine as a data preparation step. One of these guidelines is to change the pure unsupervised approach of k-Means clustering into a semi-supervised approach of guided hierarchical clustering.

Although not detailed in this paper, k-Means is also a feasible algorithm for outlier isolation. The classical "*learn the normal behavior to detect any deviations from it*" approach can be followed. However, due to the nature of time series data, and especially data sourced from continuous sensor readings where we expect a transition between system states (figure 2, lower image) the behavior of k-Means in the context of anomaly detection might be different. If isolation is done based on cluster distribution, these transition points, which probably sit at the outer border, could be incorrectly labelled as outliers. Hence, adding extra clusters to the desired number of clusters could be required to account for these "in between clusters" data points, in order to get the desired separation and avoid detecting too many false anomalies.

Besides clustering, there are other powerful methods to segment time series like decision trees, pattern miners. In our ongoing work we will also evaluate those on an experimental way and try to extract best practices with autonomous analytics in mind.

References

Aggarwal, Charu C., Alexander Hinneburg, and Daniel A. Keim. 2001. "On the Surprising Behavior of Distance Metrics in High Dimensional Space." In *Lecture Notes in Computer Science*, 420–34.

Aghabozorgi, Saeed, Ali Seyed Shirkhorshidi, and Teh Ying Wah. 2015. "Time-Series Clustering – A Decade Review." *Information Systems* 53: 16–38.

Chen, M., A. X. Zheng, J. Lloyd, M. I. Jordan, and E. Brewer. 2004. "Failure Diagnosis Using Decision Trees." In *International Conference on Autonomic Computing, 2004. Proceedings.*, 36–43.

Demetgul, M. 2012. "Fault Diagnosis on Production Systems with Support Vector Machine and Decision Trees Algorithms." *International Journal of Advanced Manufacturing Technology* 67 (9-12): 2183–94.

Guo, Lijie, and Jianxin Kang. 2015. "A Hybrid Process Monitoring and Fault Diagnosis Approach for Chemical Plants." *International Journal of Chemical Engineering* 2015: 1–9.

He, Bo, Tao Chen, and Xianhui Yang. 2014. "Root Cause Analysis in Multivariate Statistical Process Monitoring: Integrating Reconstruction-Based Multivariate Contribution Analysis with Fuzzy-Signed Directed Graphs." *Computers & Chemical Engineering* 64: 167–77.

Lin, Jessica, Eamonn Keogh, and Wagner Truppel. 2003. "Clustering of Streaming Time Series Is Meaningless." In *Proceedings of the 8th ACM*

SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery - DMKD '03. doi:10.1145/882095.882096.

Rani, Sangeeta, and Geeta Sikka. 2012. "Recent Techniques of Clustering of Time Series Data: A Survey." *International Journal of Computer Applications in Technology* 52 (15): 1–9.

Shmueli, Galit. 2010. "To Explain or to Predict?" *Statistical Science: A Review Journal of the Institute of Mathematical Statistics* 25 (3): 289–310.

Shmueli, Galit, and O. Koppius. n.d. "The Challenge of Prediction in Information Systems Research." *SSRN Electronic Journal*. doi:10.2139/ssrn.1112893.

Solé, Marc, Victor Muntés-Mulero, Annie Ibrahim Rana, and Giovani Estrada. 2017. "Survey on Models and Techniques for Root-Cause Analysis." *arXiv [cs.AI]*. arXiv. http://arxiv.org/abs/1701.08546.