Zeitschrift Kunststofftechnik
Journal of Plastics Technology

www.kunststofftech.com · www.plasticseng.com

archivierte, peer-rezensierte Internetzeitschrift
archival, peer-reviewed online Journal of the Scientific Alliance of Polymer Technology

Scientific Alliance of the University Professors of Plastics Technology

eingereicht/handed in: 28.05.2018
angenommen/accepted: 21.09.2018

Prof. Dr.-Ing. Reinhard Schiffers[1], Prof. Dr. Katharina Morik[2],
M.Sc. Alexander Schulze Struchtrup[1], B.Sc. Philipp-Jan Honysz[2],
Prof. Dr.-Ing. Johannes Wortberg[1]
[1]Institute of Product Engineering (IPE), University of Duisburg-Essen
[2]Chair of Artificial Intelligence, TU Dortmund University

Anomaly detection in injection molding process data based on unsupervised learning

Plastic processing companies in high-wage countries are facing continuously increasing cost and quality pressures. In many applications, a 100 % quality control leads to unreasonable efforts. Hence, quality forecasting or control based on process data would be desirable. Neural Networks have been applied. However, their success depends on the appropriate labeling of the process data. Since during the process, it is usually unknown whether a good or bad part has been produced in one cycle, supervised machine learning is not applicable. Here, we present approaches to anomaly detection in injection molding process data by means of unsupervised machine learning.

Anomalie-Erkennung in Spritzgieß-Prozessdaten auf Basis von unüberwachtem Lernen

Kunststoffverarbeitende Unternehmen in Hochlohnländern sind mit einem kontinuierlich steigendem Kosten- und Qualitätsdruck konfrontiert. Eine 100 %-Qualitätskontrolle ist in vielen Anwendungsfällen jedoch mit einem nicht vertretbaren Aufwand verbunden. Qualitätsprognose bzw. -regelung auf Basis von Prozessdaten wäre wünschenswert. Neuronale Netzwerke wurden bereits angewandt. Ihr Erfolg beruht aber auf dem Vorhandensein von gemäß ok/not ok annotierten Daten. Da bei der Betrachtung von Prozessdaten meist nicht bekannt ist, ob in einem Zyklus ein Gut- oder Schlechtteil produziert wurde, sind die überwachten Lernverfahren nicht anwendbar. Wir stellen deshalb Ansätze zur Anomalie-Erkennung in Spritzgieß-Prozessdaten mittels unüberwachter maschineller Lernverfahren vor.

Anomaly detection in injection molding process data based on unsupervised learning

R. Schiffers, K. Morik, A. Schulze Struchtrup, P.-J. Honysz, J. Wortberg

1 INTRODUCTION

Plastic processing companies in high-wage countries are facing continuously increasing cost and quality pressures. Accordingly, manufacturers of injection molding machines are challenged to provide products and services that enable plastics processing companies to successfully deal with the trade-off between cost and quality to persist in global competition. Resilient manufacturing processes are characterized by the ability to recover from unexpected machine behavior. Accomplishing a resilient process implies solving two important tasks: First, anomalies in manufacturing data need to be detected at the earliest possible point in time during the process. Second, the machine operator needs to be informed about the found anomalies, so that he readjusts manufacturing parameters and, thus, prevents further anomalies. Ideally this approach shortens production intervals in which defective products are being manufactured, consequently leading to a smaller amount of manufactured bad parts, while cutting costs and saving resources.

Doing so, we are facing real-time constraints, as anomaly detection, operator information and process readjustment all together should not exceed the cycle duration.

Generally speaking, this leads to the following problem definition:

Given a set of data $Z = \{Z_1, \dots, Z_n\}$ of n observed production cycles or processes recorded at some robust operating point, indexed by a set $I = \{id_1, \dots, id_n\}$ (i.e. "cycle IDs"). Then, every $Z_i = \{\vec{x}_1, \dots, \vec{x}_m\} \subset \mathbb{R}^p$ in Z is a set, which is subject to a strict total order induced by time (time series property), consisting of m equidistant observations, a so called p-dimensional multivariate time series (MTS) of cycle id_i. Such a MTS inherently consists of d so called univariate time series (UTS), which are gathered simultaneously and contain the realizations of the features one usually observes, like the melt pressure or the clamping force in injection molding.

We express the anomaly status of a production cycle by a function $f(Z_i)$. This function is not known but must be estimated by a function $\hat{f}(Z_i)$, such that (a) $\hat{f}(Z_i) \approx f(Z_i)$, i.e. the actual anomaly status of an investigated cycle id_i mostly matches the predicted anomaly status, and (b) the delay d between the actual event of anomaly at cycle id_i, $f(Z_i) = \top$, and the prediction of the same anomaly at cycle id_j, $\hat{f}(Z_{j-d}) = \top$ where $j > i$ and $i = j - d$ will be minimized.

In this paper, we present a new method for anomaly detection that is based on unsupervised learning.

The paper is organized as follows. First, we give an overview of previous work for quality assurance and relevant machine learning methods. Second, our approach of unsupervised anomaly detection is described in general. It consists of sequencing, feature extraction, feature selection, and a complex modeling that optimizes possible clusterings and composes an ensemble of learned models. Third, we apply the novel model to data from an injection molding machine.

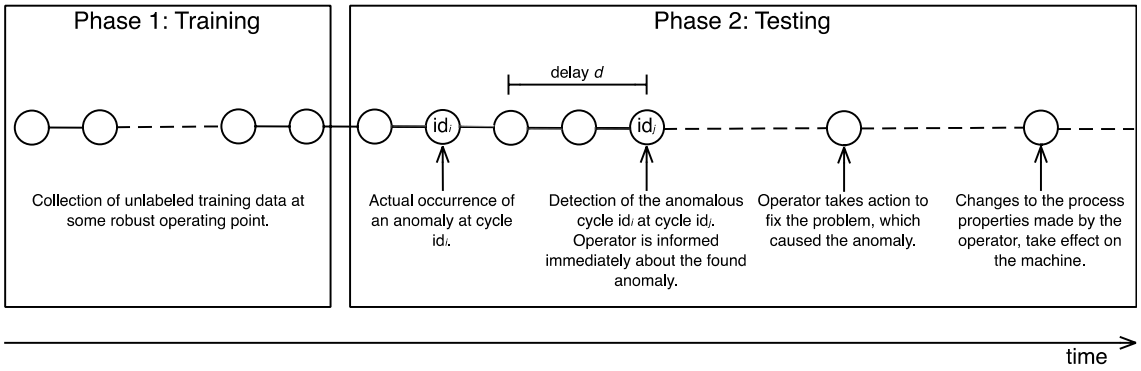

Figure 1: Concept of an anomaly detection system operating on data provided by a chain of production cycles. In the training phase a certain amount of process data is recorded to fit a machine learning model. In the testing phase every cycle is investigated to detect anomalies. Between the actual event of anomaly and the detection of that anomaly some delay d *may elapse, which is subject to minimization. Further unspecified and uncertain delays may occur, like between the anomaly detection and the subsequent intervention by the machine operator and between the intervention of the operator and the taking effect on the manufacturing system.*

1.1 State of the art and related work

State of the art injection molding machines allow highest reproducibility of the process setting parameters through precise axis movements. Nevertheless, internal and external interferences can result in process fluctuations leading to the production of reject parts. This holds true also at robust operating points. Obviously, the current situation demands for optimization of both process (plastic processing company) and machine (injection molding machine manufacturer). Regarding quality critical applications, optimization potentials on the process side are often exploited. Likewise, enhancements of (mechanical) machine technology are largely exhausted. Therefore, recent optimization

approaches in research and application are focusing on control concepts and data analysis.

In this regard, the following paragraphs deal with the state of the art in injection molding process monitoring and control, as well as in the field of data analysis and machine learning. This sets the stage for our combination of both fields resulting anomaly detection based on the clustering of injection molding process data.

Data acquisition

With regard to documentation requirements, injection molding machines offer the option of selecting a limited set of process parameters ("actual values") using a protocol function either manually via USB interface or automatically e.g. via Euromap 63 interface to a production host computer.

These actual values may be monitored with respect to fixed, predefined tolerance limits. In case a threshold is violated, an alarm or machine stop can be imposed. However, systematic process monitoring usually does not yet take place, because there is no suitable analysis infrastructure available that would allow adequate manual or even automated analysis on this basis.

To overcome the limitations of the selected number of actual values that are usually available, KraussMaffei offers the DataXplorer, a data logger that records up to 500 signals as continuous curves [4]. In this case the user can use the high resolution data to select individual features for further processing like process monitoring based on anomaly detection. Although the high number of signals as well as the high resolution establishes new possibilities for analysis, it should be mentioned that only those anomalies can be detected, that manifests in at least one process variable.

Predefined thresholds

A basic approach already used in injection molding implies a comparison between achieved process properties and planned process properties, e.g. alarming the machine operator if predefined thresholds are underrun or exceeded. However, these methods are commonly known for working under very special conditions only. Even if they are working as expected, they are yielding unreliable results, as also pointed out by Bauer *et al.* regarding the usage of such systems in intensive care medicine [18]. Furthermore, in usual injection molding applications, only a small fraction of the available of actual values is monitored, such as the cycle time or residual melt cushion, which does not provide a holistic process monitoring, while the quality of the set tolerance limits is largely dependent on the individual operator experience. Even if he decides to use threshold based on a multiple of the previous variables' standard deviation, he still has to decide which factor is appropriate for a certain variable. Finally, interactions between anomalies occurring in different variables or at different times are not combined. Even more, fixed tolerance limits may fail to

detect certain anomalies that can only be found when taking into account the combination of two or more variables at the same time.

Quality prognosis and control by ANN

In 1994 Häussler proposed a method for quality assurance in plastic manufacturing using so called artificial neural networks (ANN) [32]. ANNs are used by Häussler to realize an abstract quality prediction model, which maps different process parameters to the resulting quality characteristic of a product. Based on different kinds of training data, several experiments were carried out to assess the predictive performance of ANNs. For instance, he tried to predict different molding weight-related properties, since the molding weight is commonly recognized as a very important product property in many aspects [33]. Although Häussler achieved promising results, the described and performed experiments did require that prior knowledge about the actual quality properties to predict has been gained, which leads to a common supervised learning setting. However, it may be considered as too costly and not feasible to continuously monitor these quality properties as part of an inline quality assessment strategy, what makes it rather complicated to build up and maintain a training set of data. Nevertheless, Häussler discusses how to detect whether learned models respectively ANNs are still appropriate to predict quality properties of a product and describes how ANNs may be retrained to fit new process properties. Further approaches with regard to quality prognosis and control are proposed by [6].

Process-related quality management

In 2003 Haman among others dealt with the issue of quality inspection in injection molding processes [33]. He developed two kinds of regression models, taking different input features into account which are selected using different statistical approaches, like *F*- or *t*-testing as well as correlation analysis regarding the quality property to predict. These models are developed with the goal of online quality supervision: One model involved the parameters of the injection molding machine, while the other model used the current process state, consisting of current machining parameters and different measured process properties. Though this method seems to achieve acceptable results, it remains questionable whether learned models can be transferred to different situations, e.g. if significant process parameters did change. This is a pressing issue, as it may be very time-consuming and costly to collect new training data and retrain or adapt already learned models. Still, Haman’s work may be helpful to identify important features in the injection molding process data.

Since the standard concept focusing on the control and reproducibility of machine parameters has reached its performance limits and direct quality control could not yet prevail in practice, efforts have been made to find a compromise of both approaches. These recent control concepts are either based on control of process parameters (pvT-control) or are evaluating process parameters to adapt machine parameters (Adaptive Process Control). The

concepts are able to compensate external interferences, especially those coming from material property fluctuations, to some extent.

pvT-control

The pvT-control developed by [8], [9], [5], [3], is a control concept based on the conclusion that a direct control of cavity pressure (process variable) yields better results in terms of product quality than control of hydraulic pressure (machine parameter). The overall objectives of a constant part weight and geometry are tackled by a process phase dependent cavity pressure control. In the injection phase, constant mean shear rates at the melt front shall be provided which is realized by a constant cavity pressure gradient, i.e. a linear increase of the cavity pressure. In the holding phase a constant (cycle to cycle) volumetric shrinkage and constant part weight shall be reached which is achieved by constant specific volume when reaching ambient pressure. This requires a largely isochoric holding process which is in turn achieved by a process called pvT-optimization, i.e. an online cavity pressure control based on the calculated melt temperature in the cavity. Due to the fact that the control of the cavity pressure has to deal with a nonlinear controlled system with dead time, PID-controllers do not provide satisfactory results, therefore a model predictive control (MPC) with an artificial neural network (ANN) based process model is used. Further improvement of the control quality is achieved using so called norm optimal iterative learning controller (NOILC). In experiments with deliberately induced changes of cooling water supply temperature and barrel temperatures, pvT-control yields lower part weight fluctuations compared to standard process control. A simpler implementation that skips the step of pvT-optimization and instead aims at high reproducibility of the cavity pressure is the reference characteristic control by Arburg [54].

Adaptive process control

Another approach aiming at a constant melt volume filled in the cavity is provided by [12]. The so called adaptive process control (APC) evaluates the melt pressure curve in the injection phase with regard to a reference curve and subsequently adapts the change-over point and holding pressure. This way, process fluctuation coming from material viscosity (induced by material batch, residual moisture content or temperature fluctuations) or the non-return valve's closing behavior can be compensated to some extent. The control concept bases on the fact that for velocity-controlled injection phases, the melt pressure is a process variable that serves as an indicator with regard to the named interferences in the way that each influence manifests in different curve shape deviations. Available implementations are KraussMaffei APC [1] and Engel iQ weight control [2], further background information can be found in [10, 11].

Both APC and pvT-control do not notify the machine operator about the actually reached part quality, anomalies or process fluctuations that cannot be compensated by the control. One workaround can be to manually set

boundaries with respect to the actuating variables, i.e. boundaries equal to the actuating variables' engagement limits.

Real-time quality prediction

Case studies in a hot rolling mill and for Basis Oxygen Furnace (BOF) showed that learning models directly from the process data allows to predict the quality of the process outcome in real-time [45]. While the preprocessing, feature extraction, feature selection demands time and interaction with the engineers, the learned model can be applied during the processing. While the case studies used the same general approach, the particular methods used differ. For the hot rolling mill, the process time series for each process step have been preprocessed independently. They were segmented and statistics have been calculated for each segment using the RapidMiner system [45]. This reduced the 60,000 raw time series values for a block a steel to about 2000 features. From these features, 218 were selected because they correlated with the quality of the process outcome. For the BOF end-time prediction in real-time, the data were synchronized and fused from the different IBAPDA (http://www.iba-ag.com) streams. Feature extraction and selection has been investigated on the basis of a large data collection from more than one year. Learned models predict the optimal end-time of the process, i.e. determine when the optimal steel quality has been achieved. During a BOF process features are extracted and learned models are applied in real-time on the streaming sensor data.

Feature extraction

The raw data are often not well suited for quality prediction. They might be redundant, irrelevant, or obtain expressiveness only after some transformation. This has been impressively shown for univariate time series by [18] introducing the phase space feature transformation. Approaches for symbolic representations of time series allow us to apply machine learning algorithms that demand symbols instead of real-valued time series. Eamonn Keogh's algorithm SAX creates a sequence of symbols from a large sample set of numerical time series [46]. One of the first algorithms automatically summarizing a data stream into intervals of increasing, stable, or decreasing values was used for robot movements [38]. A general framework for all the possible transformations of time series has been developed and implemented within RapidMiner [52]. Very often, the feature extraction is the clue to learning success. The approach of Deep Learning aims at representation learning. However, the tuning of the many hyperparameters is prohibitive for many applications.

Anomaly detection

The main idea of outlier or anomaly detection is to learn what is normal and then find the exception. One way of expressing normality is to learn a so-called *Minimum Enclosing Ball* (MEB), which encloses most of training data. The *Core Vector Machines* (CVMs) do this and consider all data within the data "normal",

those that fall outside the ball outliers [43]. In 2013 Stolpe *et al.* [42] proposed an approach to outlier and anomaly detection in vertically distributed input data, i.e. data that is recorded at different locations. Their *Vertically Distributed Core Vector Machine* (VDCVM) minimizes communication costs for the distributed learning of an overall normal state.

In our approach, we follow a similar intuition. Instead of optimizing one MEB, we cluster the data and consider examples in the largest cluster to be normal, those in small clusters anomalies. In this context, it is important to mention that this procedure is only valid for one operation point, since any operation point may include one normal and multiple anomalous clusters. Beside the central task of anomaly detection, the cluster affiliation may be used to check whether the correct operation point has been chosen.

Another view of normality comes from information theory: a model that compresses many observations that are normal needs only a few bits for the encoding, where a collection of many exceptions requires more bits. Hence, the minimal description length of models select the model of normal processing [47].

Concept drift

Concept drift or shift denotes change points in processes. In machine learning, it has been modeled for sequences of text classification tasks [48]. In statistics the level change in time series is investigated [53]. A survey of adapting to concept drifts is given by [51]. Albert Bifet has carefully modeled concept drift and adaptation on data streams [49,50]. He implements a two-step approach: the detection of a concept drift and then the selection or adaptation of a new model. This is exactly what we have shown in Figure 1.

2 PROCEDURE FOR ANOMALY DETECTION BASED ON UNSUPERVISED LEARNING

In this section, we propose a holistic method to detect anomalies in multivariate time series data which has been obtained from cyclic manufacturing processes. Where many papers just report on the application of a machine learning method for modeling, we present all the steps that are necessary, because preprocessing is decisive although often underestimated. In particular, preprocessing includes *sequencing* of time series to inspect data of different production stages distinctively, *feature extraction*- and *feature selection*-techniques. Subsequently, we will show in detail how the learning task of *Clustering* may be exploited to identify anomalous patterns in data, where no label for the process or product quality is given. We automatically optimize the parametrization of a Clustering algorithm and make the result robust through the application of an *Ensemble method*.

2.1 Sequencing

Industrial manufacturing processes, especially cyclic ones like the injection molding process [33], mostly consist of a well-defined sequence of production stages, which may run sequentially and (partially) concurrently. Rather than only investigating the data of a process as a whole, it can be more interesting to inspect the data of the various production stages separately. For instance, it may be more interesting for machine operators to consider the variance of the injection pressure during the injection and holding stage of the molding process rather than during the whole process. In the following, we want to describe formally how a given MTS of a cycle may be *sequenced* along different production stages.

In order to sequence time series data, it is required from the MTS that it contains so-called *trigger features*. Those features are associated with specific production stages and either yield $\top$ (i.e. true) or $\bot$ (i.e. false), when the respective stage is active or not. This allows to derive a sequenced MTS from the original MTS: Let Z_i be the MTS of an observed cycle id_i and t an index, by which the realizations of a trigger attribute T may be accessed. Then Z_i^T is the MTS of cycle id_i which is sequenced along the trigger attribute T and given by $Z_i^T = \{\vec{x}_t \in Z_i | \, \vec{x}_t[t] = \top\}$, where $\vec{x}_t[k]$ returns the k-th component of the vector $\vec{x}_t$.

2.2 Feature extraction

The set of features in a raw data set is often considered too uninformative for solving a specific learning task in an optimal way. Hence, the need for more expressive features is emerging. This problem obtains additional importance, as we are trying to explain found anomalous patterns in data to the machine operator in a meaningful way. Hence, from a given set of (raw) features new and more descriptive features are derived, that describe a process meaningfully by a *feature vector* rather than a MTS. In the following, we will discuss two approaches to solve this problem.

Domain-knowledge based feature extraction

A first imaginable solution could make use of already gained domain knowledge about the origin of anomalies in injection molding processes. From a technical point of view, this leads to the extraction of well-defined characteristics of different univariate time series which make up the MTS. An example of domain knowledge-based feature extraction in injection molding processes is given in Figure 2.

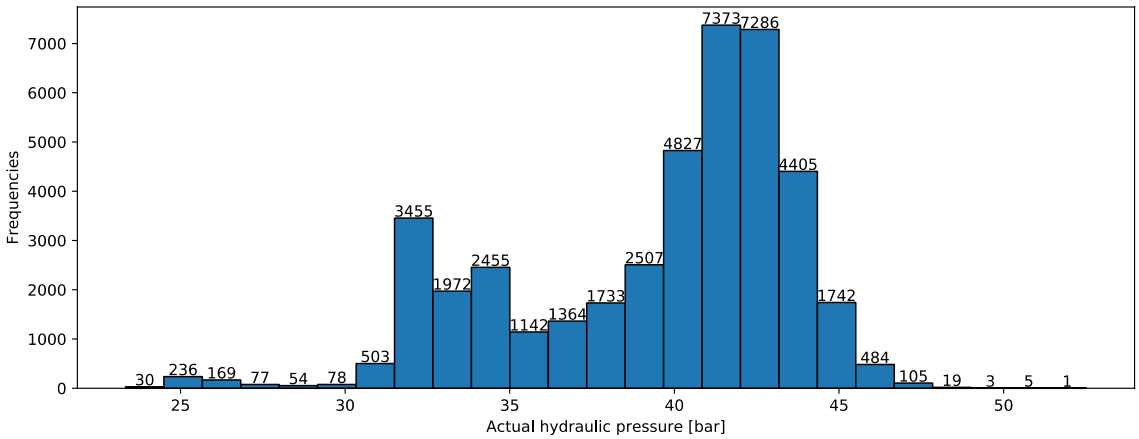

Figure 2: Histogram of 42025 recorded actual maximal hydraulic pressures during the process stage of plasticization with 25 bins. The histogram suggests that this feature could be modeled using one or more Gaussian distributions, which in turn could be exploited to classify anomalous observations.

The benefit of the domain knowledge-based feature extraction is that usually a "good" set of features is obtained which is appropriate to solve a given task, like the detection of anomalies. In the literature, there is a wide set of variables that have been developed to characterize different aspects of the injection molding process, cf. table 1. One should be aware that already in the step of feature extraction, there is a selection-influence since only the more frequently named features from literature are taken into account. Obviously, the task of domain-knowledge based feature extraction cannot be clearly separated from the task of feature selection.

Features available in technical literature
• Injection work [J] • Cooling water feed temperature [°C] • Barrel temperature [°C] • Maximum melt pressure in injection phase [bar] • Maximum cavity pressure [bar] • Injection speed [mm/s] • Cycle time [s] • Holding pressure [bar] • Holding time [s] • Clamp force [kN] • Cooling time [s] • Screw rotational speed in plasticizing phase [1/min] • Back pressure [bar] • Plasticizing torque [Nm] • Melt pressure at changeover point [bar] • Cavity pressure integral in holding phase [bar*s] • Plasticizing work [J] • Changeover point [mm] • Residual melt cushion [mm] • Plasticizing time [s] • Holding work [J] • Flow index [bar*s] • Metering stroke [mm] • Hydraulic pressure [bar] • Injection time [s] • Power consumption barrel heaters [W] • Power consumption plasticization [W]

Table 1: Domain-knowledge based features, cf. [6, 7, 10-17, 32, 33]

Automatic feature extraction

Another possible feature extraction method that could be employed to given data without prior domain knowledge, involves applying a set of predefined functions to all univariate time series (UTS). For instance minimal and maximal values for every UTS might provide valuable information. Given, that every function in that specific set maps from the space of UTS to real values, one could conflate the resulting values in a feature vector, which consequently describes the process. A training set consisting of these feature vectors may then be used in any subsequent unsupervised learning endeavors. We will establish this vague description now more formally:

First, we have to extend the MTS definition given in the introduction by a possibility to access an underlying UTS in a comfortable way. Let $Z_i = \{\vec{x}_1, \dots, \vec{x}_m\} \subset \mathbb{R}^d$ be the d-dimensional MTS of a cycle id_i, which consists of d univariate time series. Then, every k-th UTS with $k \in [1, d]$ of Z_i may be

accessed by computing the set $uni_k(Z_i) = \{\vec{x}_t[k] \mid \vec{x}_t \in Z_i\} \subset \mathbb{R}$. It can be seen, that the time series property holds true for every $uni_k(Z_i)$ given a MTS Z_i.

Now we will apply a set of functions to all underlying UTS of a given MTS. Hence, let $uni(Z_i) = \{uni_1(Z_i), \ldots, uni_d(Z_i)\}$ be the set of all UTS, which are included in a d-dimensional MTS Z_i. Furthermore, let $F = \{f_1, \ldots, f_l\}$ be a set of functions conducting a feature extraction, with every $f_i \in F$ mapping from $uni(Z_i)$ to $\mathbb{R}$. Now, all functions from F will be applied to a given UTS $uni_k \in uni(Z_i)$, thus constructing a sub feature vector $\vec{m}'_k = \left(f_1(uni_k), \ldots, f_l(uni_k)\right)^T$, which contains the mapped values. Computing all sub feature vectors for all UTS in $uni(Z_i)$ will yield a set of vectors $M'_k = \{\vec{m}'_1, \ldots, \vec{m}'_d\}$, which will be concatenated to a single feature vector $\vec{m}_i = (\vec{m}'_1, \ldots, \vec{m}'_d)^T$, that is specific to cycle id_i.

The choice of a suitable function for feature extraction is often the key to success. In Section 3.3 we show three features that are more sophisticated than the aggregate functions.

Applying this feature extraction methodology to all MTS from the original raw data set $Z = \{Z_1, \ldots, Z_n\}$ yields a set of feature vectors $M = \{\vec{m}_1, \ldots, \vec{m}_n\}$, which constitutes a possible training data set.

2.3 Feature selection

As already described, not every feature is actually capable to discriminate between anomalous and normal cycles. The goal of the *feature selection*-preprocessing task is to reduce a given set of features by those features, which are irrelevant to a given task, like anomaly detection in our case. In this section, we will discuss two approaches that may be pursued, to gain a suitable subset of features, which solves subsequent learning tasks appropriately.

Domain knowledge-based feature selection

Similar to the approach presented in section 2.2, consulting domain experts for the selection of relevant features is the first choice. For our application, 11 features were selected, cf. Table 2.

Selected features
• Injection work [J] • Maximum melt pressure in injection phase [bar] • Melt pressure at changeover point [bar] • Holding work [J] • Average melt pressure in holding phase [bar] • Residual melt cushion [mm] • Average hydraulic pressure in plasticization phase [bar] • Plasticizing time [s] • Average duty cycle of each cylinder zone heater [%] • Average duty cycle of each hot runner [%] • Average temperature of each hot runner [°C]

Table 2: Selected features with respect to the initial feature set listed in table 1.

Algorithmic feature selection

Another possible approach to feature selection involves the usage of specialized algorithms. Researchers have investigated the topic of algorithmic feature selection deeply and developed different kinds of methods to accomplish this task. Algorithmic feature selection methods may be classified into three different groups, as pointed out by *Guyon* and *Elisseeff* and Saeys *et al.* [31,39]:

- **Filters** rank each provided feature independently using a scoring function and remove those which are yielding poor scores. Furthermore, filters may be characterized as either univariate, i.e. neglecting feature interactions, or multivariate, i.e. incorporating feature interactions [39]. These methods are known to be fast and are not tied to any learning algorithm.
- **Wrappers** assess different subsets of the original feature set by evaluating the resulting predictive performance using a learning algorithm, which is considered to be a "black box". Obviously, this introduces some problems, like an often-unacceptable runtime, which is dominated by the number of subsets to rate (e.g. the power set of the initial feature set, in the worst case) and the time to train the underlying learning model. Nevertheless, these methods are simple and take feature interactions into account.
- **Embeddings** are related to wrapper algorithms, as they also incorporate learning algorithms in their selection strategy. But unlike wrappers, embeddings do not consider the learning algorithm to be a "black box" and therefore inspect the learned model to choose features. This may, for instance, include the weight vector of a linear SVM that indicates,

which features are exposing a high discriminative power between two classes (e.g. anomalous or non-anomalous, if appropriate data is available). Therefore, embeddings are regarded as computationally more efficient than Wrappers.

Beside the runtime considerations which are especially important regarding the use of wrappers and embeddings, the final choice on a specific selection algorithm should also be made by reflecting the additionally introduced hyper parameters which may need to be optimized or chosen appropriately and therefore increase the complexity of the anomaly detection method. Within this article, we will use the Minimum Redundancy Maximum Relevance (MRMR) algorithmic feature selection method, described in [27] and [40], which is a fast multivariate filter that cares about feature interactions and strives for a more general selection of features.

2.4 Modeling

Anomaly detection often suffers from the difficulty that no label information, like the actual anomaly status, is available to learn a classification model from given training data, which directly makes a binary decision regarding anomalies given testing data. This especially applies to industrial data which have been gained from cyclic processes that only run in a short time interval and therefore do not expose the possibility to economically assess the quality in an in-line manner. Consequently, there is no practicable way to make use of the widespread and heavily investigated techniques of supervised learning which rely on label information. Instead, other methods, like those from unsupervised learning, have to be considered. In this section we will discuss the unsupervised learning task of *Clustering* and describe in detail how and when it may be exploited to infer anomaly information. Additionally, we will introduce the concepts of *Ensemble learning* and link it to the Clustering learning task.

2.4.1 Clustering

A prominent learning task within the group of unsupervised learning involves the identification of so-called clusters, which, informally, are mutually disjunctive subsets of a training data set. Those clusters contain vectors that somehow are "close" or "similar" to each other (expressed by a distance or similarity function). In this section we will briefly introduce the accompanying learning task of Clustering in a more formal way, describe how a found partitioning of feature vectors may be exploited to infer anomaly information and which requirements have to be met to accomplish this successfully.

Clustering is a learning task, which may be described as follows [44]: Let $M = \{\vec{m}_1, \ldots, \vec{m}_n\} \subset X$ be a set of feature vectors in some space X (i.e. the training data) and $dist: X \times X \rightarrow \mathbb{R}^+$ a distance function, like the Euclidean distance. Furthermore, let $q: \mathcal{P}(\mathcal{P}(X)) \rightarrow \mathbb{R}$ be a quality function. Now, find a clustering, so that $\mathcal{C} = \{C_1, \ldots, C_k\}$ is a set of clusters which maximizes $q(\mathcal{C})$, with $\forall\, i \in$

$\{1,\ldots,k\}: C_i \subset M$ and $\forall\, i \neq j \in \{1,\ldots,k\}: C_i \cap C_j = \emptyset$. So-called clustering algorithms are able to find a partition of the training data set, which satisfy the clustering constraints that are mentioned here.

Now, the goal is to infer anomaly information from a found clustering. To achieve this, we will need to make two important assumptions about the properties of the training data: First, we assume, that the provided features are actually capable to discriminate between anomalous and normal cycles. In context of clustering that means that the "anomalous" feature vectors (i.e. feature vectors constructed from anomalous cycles) are sufficiently dissimilar from "normal" feature vectors (i.e. feature vectors constructed from normal cycles). Given that this assumption holds true, appropriate cluster algorithms will ideally group normal feature vectors into one "normal" or "non-anomalous" cluster and anomalous feature vectors into one or more other "anomalous" clusters. However, it is not known which cluster actually contains the normal feature vectors. Consequently, this assumption is not enough to infer any anomaly information from the clustering and thus additional statements about the training data set are necessary.

Hence, in conjunction with the first assumption we make the second assumption which states that although the true empirical distribution of anomalous and non-anomalous cycles is *not* known, the training data set describes more normal than anomalous cycles. This assumption should hold true for most data sets which have been obtained from real-world industrial plants and gives us the possibility, to identify the cluster of normal feature vectors, as it is the cluster with the highest cardinality.

As described in Figure 3, a found clustering can then be exploited, to classify each cycle, regarding its anomaly status.

We can establish this procedure more formally, by defining a decision map which maps the cycle ID under investigation either to $\top$ (i.e. anomalous) or $\bot$ (i.e. normal). This map can be defined as follows: Let $M = \{\vec{m}_1,\ldots,\vec{m}_n\}$ be a set of feature vectors, which is derived from cycle data identified by an index set $I = \{id_1,\ldots,id_n\}$ and obeys the above-mentioned assumptions. Furthermore, let $\mathcal{C} = \{C_1,\ldots,C_k\}$ be a clustering of M. Additionally, let $C_{max} \in \mathcal{C}$ be the cluster with the highest cardinality, i.e. $\nexists C_i \in \mathcal{C}: C_{max} \neq C_i \wedge |C_i| > |C_{max}|$. The decision map $r_{\mathcal{C}}: I \rightarrow \{\top, \bot\}$ is then given by:

$$r_{\mathcal{C}}(id_i) = \begin{cases} \top, & \vec{m}_i \notin C_{max} \\ \bot, & \text{else} \end{cases}$$

At this point, we already provide means to rate every cycle in a training data set individually with respect to its anomaly. However, it is currently not possible to test new cycles for membership in the largest cluster. Hence, we need to inspect the cluster model in order to turn the clustering into a classification or rating of anomalies.

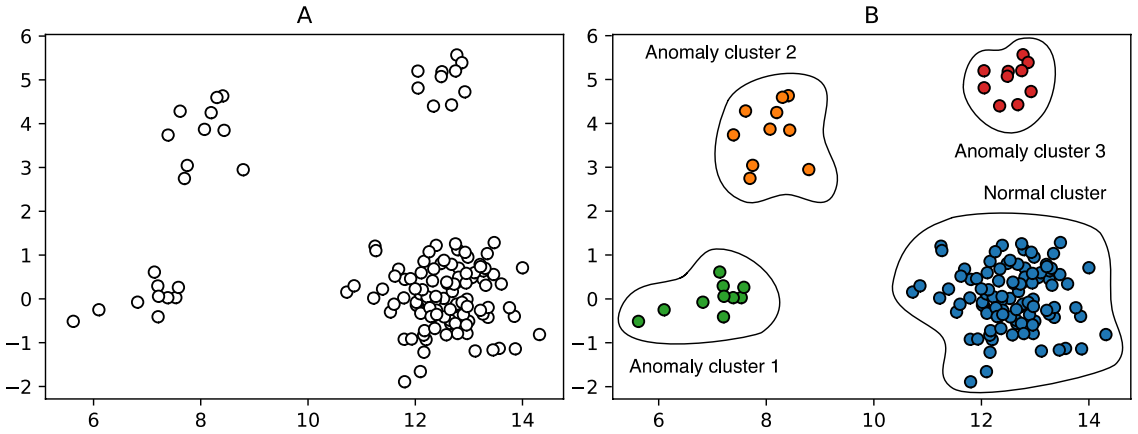

Figure 3: ***A:*** *A scatter plot of a two-dimensional unlabeled toy dataset which clearly exhibits three low- and one high-density cluster.* ***B:*** *A clustering algorithm identified the clusters and tagged the points accordingly. If the above-mentioned assumptions hold true, the "blue" cluster contains all normal vectors as it is the cluster with the highest cardinality, while all other clusters contain anomalous vectors. Hence, it is possible to rate all feature vectors and thus all cycles individually. However, there is no "natural clustering", which is inherently included in a data set and therefore found clusterings differ from algorithm to algorithm and are additionally influenced by the chosen clustering parameters (cf. section 2.4.2).*

2.4.2 Optimizing unsupervised learning

How data is partitioned into different clusters heavily depends on the choice of the used algorithm and its parametrization. In this section we will assume a fixed clustering algorithm that is given by a map $Cl: \mathcal{P}(X) \times \mathbb{R}^p \to \mathcal{P}\big(\mathcal{P}(X)\big)$. It takes a data set from some space X and a real parameter vector $\vec{\theta} \in S$ in order to partition the data set according to quality criteria. S is the search space for parameter vectors. Additionally, we demand from Cl to be fully deterministic, i.e. always yielding the same clustering for a given set of data and parameter vector. Which particular clustering algorithm should be chosen, will not be discussed here but in section 3.3. Here, we specify the properties an optimal clustering for anomaly detection should fulfill.

Within distance of clusters

The first property, one usually demands from a clustering is, that all included clusters are rather compact with respect to the defined distance or similarity measure:

Let $M = \{\vec{m}_1, \ldots, \vec{m}_n\} \subset X$ be a set of feature vectors, $dist: X \times X \to \mathbb{R}^+$ a distance function and $C \in \mathcal{P}(M)$ an investigated cluster. The dispersion of that cluster $\sigma_s^2: \mathcal{P}(M) \to \mathbb{R}$ may be derived from the well-known statistical sample variance and is thus given by:

$$\sigma_s^2(C) = \frac{1}{|C|}\sum_{\vec{m}\in C} dist(\vec{x},\vec{c})^2$$

in which, $\vec{c} \in X$ is the centroid, i.e. the "middle point", of the cluster, formally defined by:

$$\vec{c} = \frac{1}{|C|}\sum_{\vec{m}\in C} \vec{m}$$

As we are trying to assess the quality of the whole clustering, rather than a single cluster, we will extend this definition to provide a possibility to determine the overall compactness of the clustering and hence define a new map $\sigma_f^2: \mathcal{P}(\mathcal{P}(M)) \to \mathbb{R}^+$. This map will effectively average all dispersions from all clusters in a clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$ and is thus defined as follows:

$$\sigma_f^2(\mathcal{C}) = \frac{1}{|\mathcal{C}|}\sum_{C\in\mathcal{C}} \sigma_s^2(C)$$

We want to minimize the within distance.

Between distance of clusters

Another important quality criterion in clustering refers to the distance between the different clusters which is subject to maximization. To quantify this criterion, we refer to Fahad *et al.* and introduce a map $\Delta: \mathcal{P}(\mathcal{P}(M)) \to \mathbb{R}^+$, which measures the mutual distance between all clusters for a clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$, their respective centroids $\bar{\mathcal{C}} = \{\overrightarrow{c_1}, \ldots, \overrightarrow{c_k}\}$ and the Euclidean norm $\|\cdot\|$, as follows [29]:

$$\Delta(\mathcal{C}) = \frac{2}{k^2 - k}\sum_{i=1}^{k}\sum_{j=i+1}^{k} \|\overrightarrow{c_i} - \overrightarrow{c_j}\|$$

$\Delta(\mathcal{C})$ will approach zero, as the cluster centroids become closer.

We want to minimize the inverse between distance $-\Delta(\mathcal{C})$.

Cluster Count

As we are trying to exploit the clustering technique for anomaly detection, we do not expect to detect an unfeasible high quantity of clusters, as this could make it difficult to discriminate sharply between "normal" and "anomalous" clusters. Hence, we aspire to keep the number of found clusters as low as possible. To measure this, we define a simple map $c: \mathcal{P}(\mathcal{P}(X)) \to \mathbb{N}$, as follows:

$$c(\mathcal{C}) = |\mathcal{C}|$$

Noise Count

Some cluster algorithms expose the possibility to identify noisy vectors, which cannot be unambiguously assigned to a single cluster. For anomaly detection

purposes, it is undesirable to identify a high number of noisy vectors because those points cannot be classified with respect to cluster membership. More formally, let $M = \{\vec{m}_1, \ldots, \vec{m}_n\} \subset X$ be a set of feature vectors and $\mathcal{C} = \{C_1, \ldots, C_k\}$ the accompanying clustering. Furthermore, let $n(M, \mathcal{C}) = \{\vec{m} \in M | \nexists C \in \mathcal{C} : \vec{m} \in C\}$ be the function, which computes the set of all unassigned feature vectors. The number of noise vectors is then given by the simple map c_N:

$$c_N(M, \mathcal{C}) = |n(M, \mathcal{C})|$$

Optimization problem

Ultimately, we compose a multi-objective- or Pareto optimization problem out of the given four quality criteria. The parameter vector $\vec{x}$, also called the *solution*, determines the found clustering and is subject to the optimization. Usually, $\vec{x}$ is an element of some p-dimensional space S, the *search space,* and describes the possible values that $\vec{\theta}$ may accept, whereby $p \in \mathbb{N}$ is the number of parameters the algorithm requires. Now, we can establish the optimization problem, given a set of feature vectors M and a clusterer $Cl(\cdot,\cdot)$ as follows:

$$min \qquad \sigma_f^2(Cl(M, \vec{x})), c(Cl(M, \vec{x})), c_N(Cl(M, \vec{x})), -\Delta(Cl(M, \vec{x}))$$

$$s.t. \qquad \vec{\theta} \in S$$

From the fact that we are facing a Pareto optimization problem, two implications arise: First, there is no *single* optimum which has to be found. Rather, there exists a set of *Pareto optimal solutions*, *Pareto frontier*. Second, it is not possible to solve the Pareto optimization problem "directly", i.e. by using numerical methods which deliver exact solutions to some extent. Hence, alternative approaches to that optimization problem are necessary. Evolutionary algorithms are one of them and will be briefly sketched in the following.

Evolutionary optimization

Evolutionary algorithms are characterized by the fact that they preserve more than one solution, known as the *population* which is usually of fixed, predefined size (usually formalized as $\mu \in \mathbb{N}$), whereby the single solution is commonly called the *individual*. Technically, a population is a subset of the search space, which in turn mostly is restricted by some lower and upper bounds for each dimension respectively parameter. During the process of evolutionary optimization, a population is generated initially and constitutes the *current population*. In every optimization step, known as the *generation*, an offspring population is generated from the current population, by using specialized operations with biological origins, like the *crossing-over* of two individuals and the *mutation* of a single individual. Afterwards, every individual from the union set of offspring and current population is *evaluated*, which means that the individuals' *fitness vector* is computed. This vector contains the resulting values from the maps included in the optimization problem and is a member of the so-

called *objective space*. Subsequently, a selection of individuals from the union set of offspring and current population is performed. Selection algorithms like the *NSGA-II* [26] identify the so-called *Non-dominated sets* respectively the *Non-dominated frontiers*, which are approximations to the (unknown) Pareto frontier that preserve the same properties a (true) Pareto frontier demands but limited to the given set of individuals. Subsequently, the NSGA-II successively transfers the best individuals to a new population (so-called *elitism*) until the previously fixed size of the population is reached. Elitism is an important concept as it ensures that the best individuals so far are part of the new population for the next generation, which is essential to optimization. Afterwards, the new population constitutes the new *current population* and a new generation may begin. Evolutionary optimizers terminate when specific termination criteria are met, which usually means that a predefined number of generations (usually formalized by $G \in \mathbb{N}$) has been computed. Lastly, the optimization routine returns the final population.

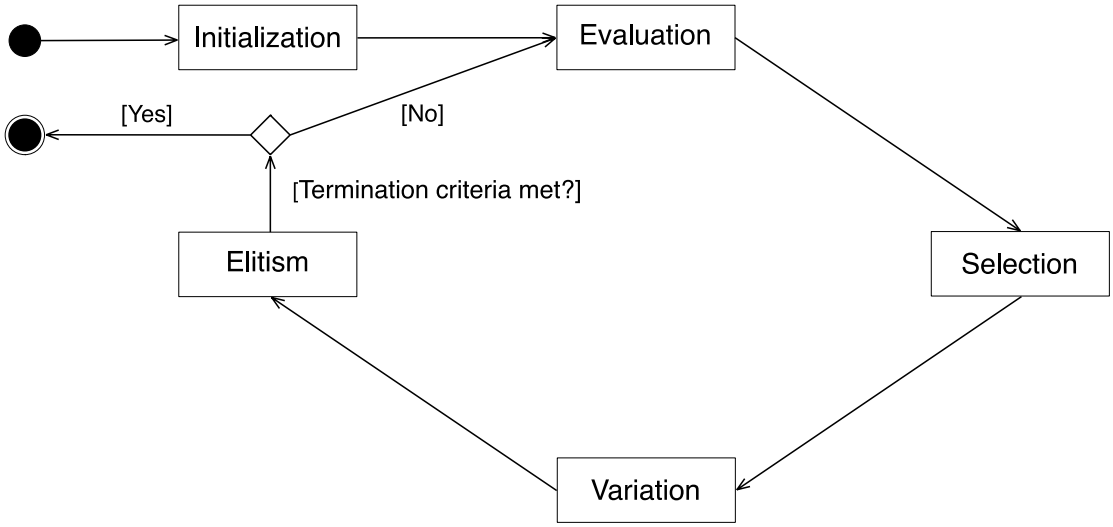

Figure 4: Steps which are performed during an evolutionary optimization, as described in [23].

Which solution should be selected?

As already stated, there is no single solution that solves a Pareto optimization problem optimally, but rather a set of optimal solutions exist. Hence, from a set of solutions, a single solution has to be picked to solve the original task. Within the scope of this article, we will pick a single solution from a set of evaluated solutions according to the following strategy:

1. It has been mentioned before that we aspire to minimize the number of clusters. Still, we need to find at least two clusters in order to distinguish a normal and an anomalous cluster. Hence, we specify a tolerable interval of cluster number as $[2,5]$. Any solution, which provoked a number of clusters that does not lie within this interval will be removed.
2. If applicable, we will keep only those solutions which provoked the lowest number of noisy points, while removing all other solutions from the population. We choose this approach, as noisy points cannot be assigned to any cluster, which in turn makes it impossible to rate the respective cycles.

It is possible that there exists more than one solution which suffices the specified strategy. In this case, an arbitrary solution from the set of feasible solutions will be picked. It is also possible that the set of feasible solutions is empty. In this particular case, we consider the optimization procedure to have failed and the system that initiated the optimization should fallback to a predefined error routine.

2.4.3 Making the analysis robust: Ensembles

We made two assumptions: first, there are many more observations of normal processing than there are for anomalies; second, the provided feature set is capable of discriminating between anomalous and normal feature vectors. Although this sounds trivial, it may be hard and time-consuming to identify such a set of features, especially in complex and highly integrated industrial systems.

Therefore, it may be beneficial to use multiple sets of features within the anomaly detection system, if the suitability of a single chosen feature set is doubtful. Choosing multiple feature sets instead of one single feature set consequently leads to multiple training sets, which all need to be evaluated using the Clustering technique as described in section 2.4.1 and merged together appropriately. The idea of combining several "base learners" in order to acquire a more robust prediction model [34] is subsumed under the term *Ensemble methods*. These methods assume that every learned model is subject to errors (e.g. by providing an inappropriate selection of features), which may become manageable when different models are put together [20]. In this section, we will present such an Ensemble approach, which exploits different clustering decision maps to calculate an anomaly rating for every cycle.

Given a set of training data sets that are each derived from the same set of raw data provided by the production system, each is clustered delivering a

clustering decision map. The set of all decision maps is the *committee*. A cycle is rated *anomalous* if the majority of the committee considers the cycle to be anomalous. Otherwise, the cycle is considered to be non-anomalous. The ratio of the decision maps classifying the cycle as anomalous to those doing not is more informative than a simple classification. It allows the operator to assess the certainty of the rating. We will establish this method now more formally.

Let $I = \{id_1, \ldots, id_n\}$ be the index set of cycles to rate and $M^* = \{M_1, \ldots, M_k\}$ be a set of training data sets, with every data set being of size n and containing feature vectors derived from cycles given in I, i.e. $\forall\, M \in M^* \,\forall\, id_i \in I \,\exists\, \vec{m}_j \in M : i = j$. Furthermore, let $F^* = \{F_1, \ldots, F_k\}$ be the set of feature sets, which respectively are used in M^*. Finally, let $R = \{r_{\mathcal{C}_1}, \ldots, r_{\mathcal{C}_k}\}$ be a set of clustering decision maps (cf. section 2.4.1), that have been obtained by clustering the data sets in M^*.

In order to perform the majority voting and therefore calculate the anomaly decision for a cycle $id_i \in I$, we need to capture the number of "positive" votes and "negative" votes. To do this, we define $V_i^{\top} = \left\{F_j \in F^* \,\middle|\, r_{\mathcal{C}_j}(id_i) = \top\right\}$, the set of feature sets, which were responsible to classify the cycle id_i as *anomalous* and $V_i^{\perp} = \left\{F_j \in F^* \,\middle|\, r_{\mathcal{C}_j}(id_i) = \perp\right\}$, the set of feature sets, which were responsible to classify the cycle id_i as *normal*. Now, we introduce a new decision map $r_{\mathrm{Ens}} : I \to \{\top, \perp, \dashv\}$, which accomplishes the majority voting on behalf of the Ensemble:

$$r_{\mathrm{Ens}}(id_i) = \begin{cases} \top, & |V_i^{\top}| > |V_i^{\perp}| \\ \perp, & |V_i^{\perp}| > |V_i^{\top}| \\ \dashv, & \text{else} \end{cases}$$

r_{Ens} will yield $\top$, when the majority of clustering decision function decided, that the cycle id_i should be considered anomalous, respectively $\perp$, when the majority considered the cycle to be normal. Finally, $\dashv$ is returned, when no majority exists for both choices (indifference). It is possible to avoid this situation by providing an odd number of clustering decision maps.

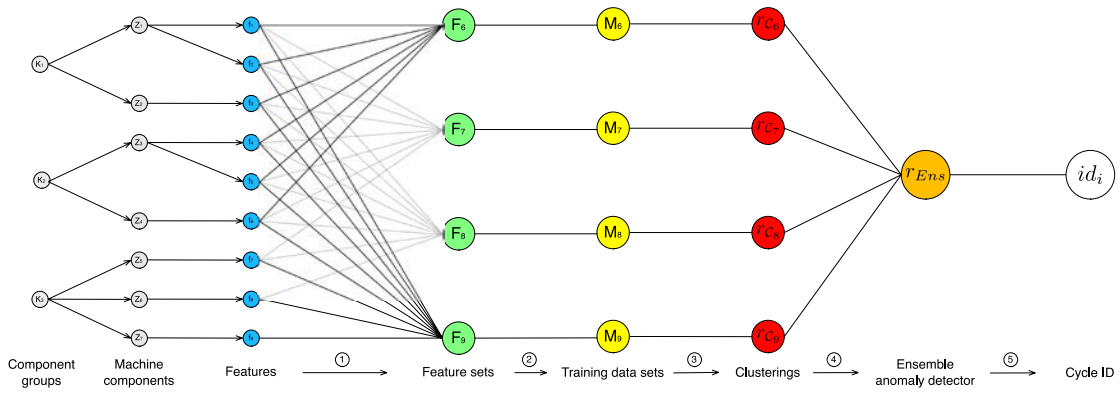

Figure 5: Scheme of an Ensemble anomaly detection system which operates on different training data sets in an Industrial context. ***1.*** *Extracted features which are derived from raw features provided by different machine components are bundled in different feature sets. Machine components in turn may be members of a component group (e.g. the different heating zones attached to the barrel, which may be grouped together). The edges between the different feature set nodes and feature nodes indicate that a forward selection-algorithm for feature selection is used.* ***2.*** *Feature sets define which (extracted) features are used to describe the cycle within training data sets.* ***3.*** *The training data sets are used afterwards to compute clusterings and the accompanying clustering decision function.* ***4.*** *The Ensemble decision function r_{Ens} conflates all clusterings using a majority voting approach.* ***5.*** *The Ensemble decision function is used to rate a cycle, which is uniquely identified by its cycle ID.*

3 APPLICATION

This chapter shows the data processing pipeline as applied to industrial injection molding data. Although the presented case involves quality issues of the molded parts, we want to point out that the presented approach for anomaly detection is not limited to product quality issues, but is of general nature that can deal with any kind of problem that manifests in the available process data.

3.1 Course of event

In the present use case, a KraussMaffei 80-100 CX hydraulic injection molding machine is used to produce molded parts from a PET / PBT blend with 15% glass fiber content, which are subject to stringent quality requirements. In addition to the standard equipment, the machine is equipped with a KraussMaffei DataXplorer for continuous data acquisition. After setting up the process, the production runs on schedule for several weeks.

This shifts with a change in the material batch used. Although the molded parts are subject to quality control, the bad part production (violation of a dimensional tolerance) resulting from material property changes is not noticed until after a delay for two reasons: First, there is a general time delay between production and the sample based quality assessment due to logistic reasons. Second, in this very case, two factors of influence foster the usual time delay: On the one hand, the time of the scrutinized part's removal is so that the newly supplied material batch arrived shortly after the previous parts removal from the machine and on the other hand, the subsequent quality assessment is scheduled only after a weekend break.

After the unacceptable dimensional deviation is detected, it is first attempted to realize a good part production by adapting the machine settings, but without success. To narrow down the cause of the fault - which is still unknown at this time - the affected tool is equipped on another injection molding machine. Also in this case, it comes to a bad part production. A material-related cause is confirmed later in laboratory tests which show corresponding differences in viscosity and pvT-behavior.

For the purpose of this paper we shall ignore the described steps carried out for root cause analysis and rather investigate how unsupervised learning can contribute to a quick recognition of critical process states through anomaly detection.

3.2 Dataset

In the above mentioned use case, a KraussMaffei DataXplorer has been used for continuous data acquisition. Based on the present injection molding machine configuration, an MTS with 33 signals and 14 trigger signals that can be used for sequencing, has been recorded with a sample rate of 200 Hz, cf. table 3. During the production order, which took 14 days of production, a total of 42035 cycles have been recorded.

This data set incorporates 5191 cycles which are considered to be *anomalous*, and 36844 cycles,which are considered to be *normal*. The decision on which cycles should be considered to be anomalous or normal has been made by quality spot checking the manufactured parts.

Info	Signals	Triggers
• Cycle start time • Cycle number • Cycle ID • Screw-diameter • Screw L/D-ratio • Machine number • Order number	• Inj1PrsAct [bar] • Inj1VelAct [mm/s] • Inj1PosAct [mm] • Inj1RpmAct [1/min] • ClpVelAct [mm/s] • ClpPosAct [mm] • ClpFceAct [kN] • Eje1VelAct [mm/s] • Eje1PosAct [mm] • Inju1PosAct [mm] • Inju1VelAct [mm/s] • Pmp1PrsAct [bar] • Inj1HopTmpAct [°C] • Inj1HtgTmp1Act [°C] • Inj1HtgTmp2Act [°C] • Inj1HtgTmp3Act [°C] • Inj1HtgTmp11Act [°C] • Inj1HopEdAct [%] • Inj1HtgEd1Act [%] • Inj1HtgEd2Act [%] • Inj1HtgEd3Act [%] • Inj1HtgEd11Act [%] • MldHtg1Tmp1Act [°C] • MldHtg1Tmp2Act [°C] • MldHtg1Tmp3Act [°C] • MldHtg1Tmp4Act [°C] • MldHtg1Tmp5Act [°C] • MldHtg1Ed1Act [%] • MldHtg1Ed 2Act [%] • MldHtg1Ed 3Act [%] • MldHtg1Ed 4Act [%] • MldHtg1Ed 5Act [%] • OilTmp1Act [°C]	• TrigClpCls (T_1) • TrigClpOpn (T_2) • TrigInj1 (T_3) • TrigHld1 (T_4) • TrigPlst1(T_5) • TrigInj1Dcmp2 (T_6) • TrigCool (T_7) • TrigClpCfb (T_8) • TrigClpCfr (T_9) • TrigEje1Fwd (T_{10}) • TrigEje1Rew (T_{11}) • TrigInju1Fwd (T_{12}) • TrigInju1Rew (T_{13}) • TrigInj1Dcmp1 (T_{14})

Table 3: Available Process Data Structure

3.3 Experiments

In order to apply our Ensemble anomaly detection method, we have to discuss first, which features should be provided to the algorithm. As proposed in section 2, this includes the question, along which trigger features the given MTS will be sequenced, which features will be extracted and which features will be selected. Subsequently, we have to deal with the question which clustering algorithm should be chosen to cluster the feature vectors, how the evolutionary optimizer should be configured and how many cluster decision maps we want to take into account within the Ensemble.

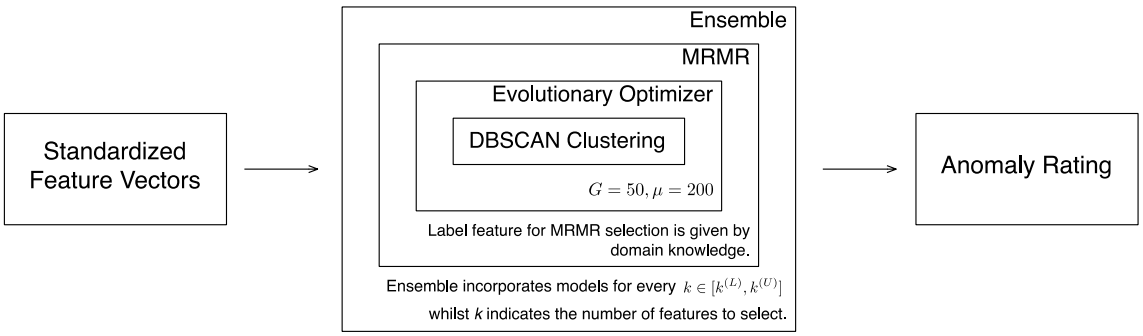

Figure 6: Scheme of Anomaly Detection experiments we will conduct within this section. All feature vectors which have been gathered previously will be standardized to eliminate biases towards different features. Afterwards, we will conduct a 10-fold stratified cross validation which partitions the training data set of standardized feature vectors that way that every split in training and testing set is characterized by an equal distribution of labels. Now, we will train the anomaly detection model using the training data sets and assess the predictive performance of the learned models using the testing data sets, as every model yields a binary anomaly rating for given testing data.

Standardization

As features may be scaled differently and therefore introduce an unwanted bias towards other features when distance or similarity functions are used, we will standardize all features first, before incorporating them in any learning endeavor or model. The process of standardization is described in [21].

Sequencing

First, every MTS from the data set will be sequenced along all Trigger features which are described in Table 1. For subsequent preprocessing steps it is important that all cyclic processes share a set of performed process stage, which is as large as possible. Neglecting this demand can lead to feature vectors of different dimensionality which consequently incorporate a different set of features. This is especially the case when an automatic feature extraction

should be performed. Hence, exceptions are made for *TrigInj1Dcmp1* (decompression 1) and *TrigInju1Rew (injection unit rewind)*, as the process stages that are associated with these triggers are not mandatory within the injection molding process and are performed on request of the machine operator. In our dataset, 41825 cycles share the same set of performed process stages which respectively are associated with the set of Trigger features $T_{Seq} = \{T_1, \ldots, T_{12}\}$. In addition to sequencing a given MTS, we will also provide the unsequenced MTS of every cycle to the next preprocessing step.

Feature extraction

We used the following set of extraction functions:

- $max(Z)$: Returns the maximal value appearing in an UTS Z.
- $min(Z)$: Returns the minimal value appearing in an UTS Z.
- $avg(Z)$: Returns the arithmetic average from all values included in an UTS Z.
- $variance(Z)$: Returns the sample variance from all values included in an UTS Z.

In addition, three more sophisticated features are extracted.

- **Injection and holding work:** As we are dealing with empirical values rather than analytical functions, we have to integrate the respective base features, Inj1PrsAct during the injection cycle stage respectively during the holding stage in our case, numerically. Within this article, we will use the trapezoidal rule for numerical integration purposes which approximates the definite integral by fitting trapezoids under the curve that is given by the values (cf. [55]).
- **Melt pressure at changeover point:** In simple terms, the melt pressure at the changeover point is the first observation of the melt pressure during the holding cycle stage. Formally, let $uni_{InjPrs}(Z_i^{T_4})$ be the UTS of the injection pressure at the holding stage of some cycle id_i. Then, $uni_{InjPrs}(Z_i^{T_4})$ is an ordered set, where the first observation x_0 is clearly defined and represents the melt pressure at the changeover point.
- **Residual melt cushion:** Analogous to the melt pressure at the changeover point, we will proceed with the residual melt cushion which represents the last observation of the screw position during the holding cycle stage. Again, let $uni_{InjPos}(Z_i^{T_4})$ be the UTS of the screw position at the holding stage of some cycle id_i. The last observation x_n is then clearly defined and represents the residual melt cushion.

Feature partitioning for groups of components

According to logical groups or component groups of the injection molding machine, we partition the original feature set (cf. table 3), in order to possibly identify anomalies in those groups and make anomaly reports more descriptive.

In Table 5 we give a partition of the original feature set into the two important component groups, namely the *clamping unit and hot runners* and the *Plasticization unit without cylinder heaters*.

Plasticization unit without cylinder heaters	**Clamping unit**	**Hot runners**	**Cylinder heaters**
Inju1PosAct [mm]	ClpVelAct [mm/s]	MldHtg1Tmp1Act [°C]	Inj1HopTmpAct [°C]
Inju1VelAct [mm/s]	ClpPosAct [mm]	MldHtg1Tmp2Act [°C]	Inj1HtgTmp1Act [°C]
Inj1PrsAct [bar]	ClpFceAct [kN]	MldHtg1Tmp3Act [°C]	Inj1HtgTmp2Act [°C]
Inj1VelAct [mm/s]	Eje1VelAct [mm/s]	MldHtg1Tmp4Act [°C]	Inj1HtgTmp3Act [°C]
Inj1PosAct [mm]	Eje1PosAct [mm]	MldHtg1Tmp5Act [°C]	Inj1HtgTmp11Act [°C]
Inj1RpmAct [1/min]		MldHtg1Ed1Act [%]	Inj1HopEdAct [%]
		MldHtg1Ed2Act [%]	Inj1HtgEd1Act [%]
		MldHtg1Ed3Act [%]	Inj1HtgEd2Act [%]
		MldHtg1Ed4Act [%]	Inj1HtgEd3Act [%]
		MldHtg1Ed5Act [%]	Inj1HtgEd11Act [%]
Label			
max(Inj1PrsAct)	max(ClpFceAct)	avg(MldHtg1Ed1Act)	max(Inj1HtgTmp1Act)

Table 5: Partitioning of the Signals given in Table 2 according to different component groups

Nevertheless, we will also consider the unpartitioned set of features in a separate experiment, in order to compare the "generic" model with the more specialized ones.

Clustering

To calculate a clustering decision map for the given set of feature vectors, we will need to make a choice on a clustering algorithm. Within this article, we chose the DBSCAN algorithm by Ester *et al.* [28]. In contrast to other clustering algorithms, DBSCAN is capable of identifying noisy points and, most importantly, finding clusters of non-convex shape, as we are not able to make statements about the shape of the clusters we expect to find. To accomplish this, DBSCAN takes the density structure in the respective space into account and, starting from an arbitrarily chosen point, recursively expands clusters, if a certain amount of points lies within a defined radius measured by a distance function which will be the Euclidean distance. Consequently, two hyper parameters are introduced which have to be optimized: First, the amount of points that have to lie within the radius, usually formalized by $minPts \in \mathbb{N}$, and the radius itself, formalized by $\epsilon \in \mathbb{R}^+$. The set of points that lie within the ϵ radius of a point is also called the "ϵ-neighborhood" of that specific point. Sticking to the framework we defined in section 2.4.2, DBSCAN takes a parameter vector $\vec{p} \in \mathbb{N} \times \mathbb{R}^+$ to calculate the clustering.

Ensemble

As described in section 2.4.3, we want to use an ensemble anomaly detector to rate cycles. In order to do this, we will reduce the dimensionality of the feature vectors in our training data set M to $k \in [k^{(L)}, k^{(U)}]$ using the MRMR approach, hence gathering a set of training data sets $M^* = \{M_{k^{(L)}}, \ldots, M_{k^{(U)}}\}$ which is used in the ensemble. Within this article, we will set $k^{(L)} = 2$ and $k^{(U)} = 10$. Since the label feature which is used by MRMR is always included in the selected feature subset, the selection of k features yields a feature subset of $k + 1$ size. As we are conducting five experiments, we will need to specify five meaningful label features for MRMR within the Ensemble, which are as follows:

- **Domain knowledge:** Maximum Injection Pressure during the Injection Stage
- **Unpartitioned feature set:** Overall Maximum Injection Pressure
- **Clamping unit:** Maximum clamping force
- **Hot runners:** Average duty cycle of the first hot runner heater
- **Cylinder heaters:** Maximum temperature of the barrel at the first zone.

Evolutionary optimization

We optimize the parameters of DBSCAN by an evolutionary algorithm using Polynomial mutation described in [25] for the mutation of single individuals. The crossing-over of two individuals uses the simulated binary-crossover described in [24]. The final selection of the whole set of individuals in every step uses the NSGA-II approach [26]. Furthermore, every Evolutionary optimization routine is run with $G = 50$ and $\mu = 200$.

We also have to deal with the question how the search space should be restricted. The goal one usually pursues by restricting the search space is to keep the space of possible solutions as small as possible. This is beneficiary to runtime, as a tinier space has to be investigated to find optimal solutions. However, it is very important to choose an appropriate restriction to make sure that optimal solutions are not excluded from the search space.

As the $minPts$ parameter is a natural number which still somehow is interpretable in higher-dimensional feature vector spaces, it is conceivable to determine the lower and upper boundary of this parameter manually. This is not the case for the ϵ parameter, as we are usually dealing with high-dimensional data spaces that often do not expose the possibility to visualize data and set ϵ parameter boundaries that way.

Therefore, we will estimate the lower and upper bound of the ϵ parameter, $\epsilon^{(L)}$ and $\epsilon^{(U)}$, on the basis of the user-defined boundaries of the $minPts$ parameter, $minPts^{(L)} \in \mathbb{N}$ and $minPts^{(U)} \in \mathbb{N}$. To do so, we respectively will average the $minPts^{(L)}$- and $minPts^{(U)}$-nearest points over all feature vectors in M, given a distance function $dist: X \times X \to \mathbb{R}^+$, like the Euclidean distance.

More formally, let $D \in \mathbb{R}^{|M|\times|M|}$ be a distance matrix, incorporating the distances between all elements of M, i.e. $D_{i,j} = dist(\vec{m}_i, \vec{m}_j)$. Furthermore, let D^{Sorted} be a version of D, which is row-wise sorted in ascending order. Now, we define the set of distances between all points and its $minPts^{(L)}$-nearest point, $D_{Low} = \left\{ D^{\text{Sorted}}_{i,minPts^{(L)}} \middle| \vec{m}_i \in M \right\}$, and the set set of distances between all points and its $minPts^{(U)}$-nearest point, $D_{Up} = \left\{ D^{\text{Sorted}}_{i,minPts^{(U)}} \middle| \vec{m}_i \in M \right\}$. The ϵ boundaries are then given by averaging over all values in their respective set:

$$\epsilon^{(L)} = \frac{\sum_{x \in D_{Low}} x}{|D_{Low}|}, \epsilon^{(U)} = \frac{\sum_{x \in D_{Up}} x}{|D_{Up}|}$$

It should be kept in mind that this estimation method has a runtime complexity of $\mathcal{O}(n^2)$ given n vectors to calculate the distance matrix which has a non-negligible impact on the overall runtime.

For the given two parameters of DBSCAN to optimize, namely ϵ and $minPts$, the search space is now restricted by $minPts^{(L)}$ and $minPts^{(U)}$, which is provided by the user, and $\epsilon^{(L)}$ respectively $\epsilon^{(U)}$, which is computed as mentioned above. Within this article, we will set $minPts^{(L)}$ to 10, $minPts^{(U)}$ to 100 and estimate the ϵ-boundaries as described.

The complexity of the optimization step may become unfeasible. Especially when it comes to evolutionary optimization, the runtime of the Ensemble is dominated by the number of individuals (μ) to consider per generation, the number of generations itself (G) and the complexity of the Clustering algorithm. Hence, it is very important to parameterize the Ensemble appropriately, either by using low-cost clustering algorithms or by limiting the μ and G parameter of the evolutionary optimizer, in order to keep the overall runtime in a reasonable range.

Turning a DBSCAN cluster model into a decision map

Now, we exploit a DBSCAN clustering of a training set for assessing the anomaly of a previously unseen data set. This requires detailed clustering information about the points in the largest cluster, in particular, which points in the largest cluster are "core points" (i.e. points, which have more than or exactly $minPts$ in their ϵ-neighborhood) and which points are "border points" (i.e. points, which are members in a ϵ-neighborhood of a core point but are not core points itself). According to Ester *et al.* we can safely assume that a (new) testing point is supposed to be at least a border point of the largest cluster and thus being a member of it when the point lies in the ϵ-neighborhood of a core point [28]. This allows us to specialize the more general clustering decision map we gave in section 2.4.1, tie it to the DBSCAN algorithm and, hence, provide means to test new data for membership in the largest cluster. Normally, this can be written as follows.

Again, let $M_{train} = \{\vec{m}_1, \ldots, \vec{m}_n\} \subset \mathbb{R}^d$ be a set of qualified feature vectors which is derived from cycle data identified by an index set $I_{train} = \{id_1, \ldots, id_n\}$ and

constitutes the training set. Now, let $\mathcal{C}_{train} = \{C_1, \ldots, C_k\}$ be a DBSCAN clustering of M, whereby $C_{max} \in \mathcal{C}_{train}$ is the cluster with the highest cardinality. DBSCAN will now partition every cluster in a set of core and border points during the process of clustering. As our anomaly detection method only relies on the largest cluster C_{max}, we need to inspect its core points, given by the set K_{max}, in order to assess the membership of an arbitrary testing point in the largest cluster. As described, we need to check if the testing point is a member of the ϵ-neighborhood of any core point, which should be formalized in the following. Let $\vec{m} \in \mathbb{R}^d$ be a vector and $dist: \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^+$ a distance function, then the ϵ-neighborhood of $\vec{m}$ is defined by $\epsilon_{\vec{m}} = \{\vec{v} \in \mathbb{R}^d | \ dist(\vec{m}, \vec{v}) \leq \epsilon\}$. Now, the specialized clustering decision map $r_{\mathcal{C}}: I_{test} \to \{\top, \bot\}$ for a set of testing points $M_{test} = \{\vec{m}_1, \ldots, \vec{m}_n\} \subset \mathbb{R}^d$ and the accompanying index set I_{test} is given by:

$$r_{\mathcal{C}}(id_i) = \begin{cases} \top, & \nexists \vec{k} \in K_{max}: \vec{m}_i \in \epsilon_{\vec{k}} \\ \bot, & \text{else} \end{cases}$$

Predictive performance

The experiments are to assess the predictive performance of our anomaly detection method. Having turned the clusterings into classification, we can apply the confusion matrices [41] and *Precision* and the *Recall*, which are commonly used for determining the predictive performance of a binary classifier:

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$Recall = \frac{tp}{tp + fn}$$

Additionally, we will consider the Accuracy of our predictions, which measures the effectiveness of our method in a more general way.

$$Accuracy = \frac{tp + tn}{tp + fn + fp + tn}$$

Furthermore, we need to introduce the method of cross-validation in short, which is needed to estimate the *true* error of the model. Learning a model on behalf of the whole data set could lead to so-called *overfitting*, which generally expresses the issue that learned models may incorporate more parameters that can be justified by the data. Although these models seem to expose a good predictive performance at first sight, they may fail to predict new and unseen data. Hence, we are interested in estimating the *true* error of our model. One method to achieve this is to *cross-validate* our model. Within the scope of this paper, we will partition the original data set using the so-called 10-fold stratified sampling approach, which ensures, that all sets within the partition approximately share the same label proportions [35], i.e. the anomaly status in our case, and has been found to be better compared to the regular cross-validation [35].

Explaining anomalies

One important goal of our article is to provide means that allow machine operators to get a decent insight into the systems evaluation of individual anomalies. That means that the machine operators should not just know that the system predicted an anomaly for an individual cycle but also, why the system made that specific decision.

A first step towards this goal has already been made by partitioning the raw features into different "component groups" which reflect rough logical groups of the considered molding machine. This approach exhibits the possibility to learn anomaly detection models which are specific to the group of components and hence, ideally, makes it easier for machine operators to identify problems at specific points of the manufacturing system.

Another important aspect involves the inspection of the learned meta model, i.e. the collection of Clusterings within the Ensemble. The first thing we want to come up with, is to trivially quantify the number of times a feature is involved in the meta model. The idea is, that a feature, which is largely involved in the model, also exposes a high relevance towards the anomaly detection process. The machine operator then may consider the most important features to identify the root cause of a possible anomaly. Hence, we continue with the notation we developed in section 2.4.3 and again consider $V_i^{\top}$, the set of feature sets, which were responsible to classify a cycle id_i as *positive*. Now, we weight the importance of a specific feature f, given $V_i^{\top}$ as follows:

$$W_i^f = \frac{\left|\{F \in V_i^{\top} | f \in F\}\right|}{|V_i^{\top}|}$$

Another possible anomaly metric involves the computation of the number of models which rated a cycle as positive. If more Clusterings decided the cycle to be anomalous, it should be more likely that the cycle actually is anomalous. Analogous, to the feature importance, we want to establish this idea more formally. In order to do this, we additionally consider $V_i^{\perp}$, the set of feature sets which were responsible to classify a cycle id_i as *negative*, as described in section 2.4.3. We then may establish the metric W_i^{+} as follows:

$$W_i^{\top} = \frac{\left|V_i^{\top}\right|}{\left|V_i^{\top}\right| + \left|V_i^{\perp}\right|}$$

$W_i^{\top}$ will yield a value between zero and one, while one indicates, that all decision maps rated the cycle id_i as anomalous. The metric will yield zero, when all decision maps predicted, that the specific cycle is normal.

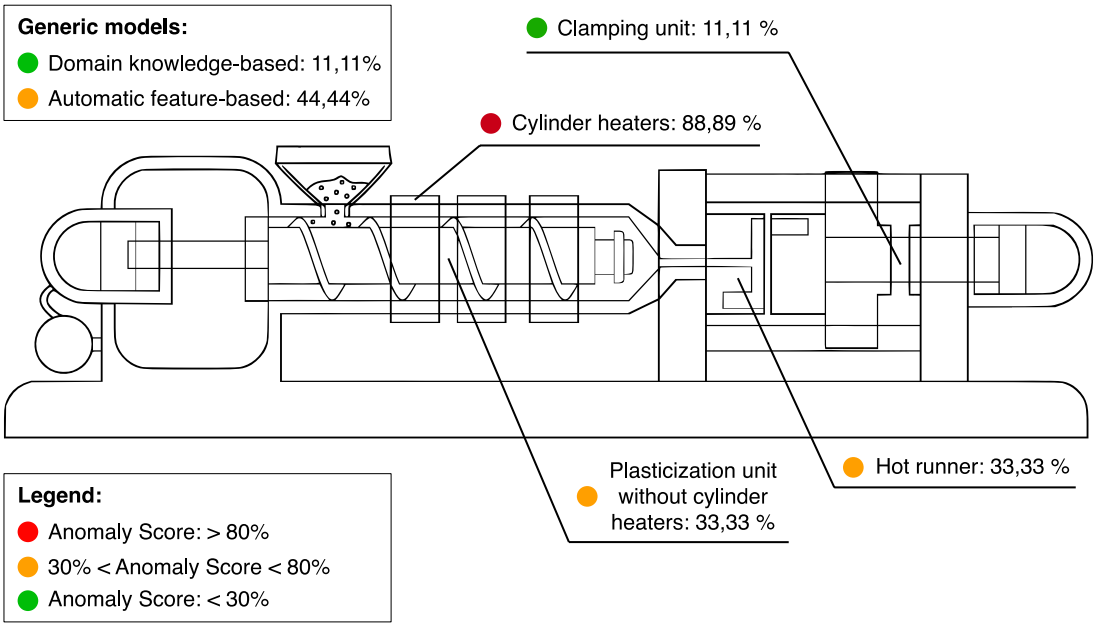

Figure 7: Anomaly Report of an arbitrarily chosen cycle from the given data set, which conflates the $W_i^{\top}$-metric from four different learned models. Two of the learned models target specific parts of the machine, while the other two models infer anomaly information in a more general way. The Injection molding machine sketch is taken from [37].

Transferability

One important challenge that has to be dealt with is how the validity of a learned model is affected when dealing with new data that has been produced under changed process conditions, e.g. a changed operation point. Minor changes to the machine settings are quite frequently taken by machine operators to maintain product quality. Due to the Ensemble approach, it is expected that small changes will have no significant impact on the model validity.

In contrast, significant changes may downgrade predictive performance. This demands to learn anew. Since our holistic method includes the feature selection and the optimization of the clustering, the method can easily be applied to data from cycles that no longer are covered by the previously learned decision map. Performing the overall analysis process can also become necessary because of changes of mold, material or injection molding machine. It is expected that in these cases most parts of a learned (cluster) model lose their validity, while some general process variable interrelations hold true despite significant system configuration shifts.

3.4 Results and Conclusions

We have deployed our methods and investigated particularly the influence of the various feature sets on the quality of learning.

For measuring the predictive performance using accuracy, precision and recall, we have to know the ground truth label for each cycle. This can be achieved e.g. by applying the quality of the molded parts as a criterion for the anomaly status. However, in this case there are two drawbacks: First, there has to be a 100% quality assessment and second, anomalies that have no direct impact on the part quality are not considered. In our case of the presented industrial application we can make a good guess of the actual anomaly status since we know the point in time after which the material batch change resulted into scrap production. Assuming that all cycles are normal beforehand and anomalous after, the assessment of the predictive performance tends to yield slightly poorer results since there might be some anomalous cycles before the shift due to other reasons. Still this is a resilient assumption since no major quality or other process issues have been observed during this time.

Additionally, we measured precision and recall in both classes, anomalous and normal, to gain more differentiated statements about the performance of the classifiers.

Experiment	**Precision (normal)**	**Precision (anomalous)**	**Recall (normal)**	**Recall (anomalous)**
Plasticization unit without cylinder heaters	91,50%	16,37%	48,04%	67,97%
Clamping unit	88,55%	31,70%	91,87%	15,35%
Hot runners	95,41%	50,48%	90,25%	68,53%
Cylinder heaters	89,23%	14,20%	40,21%	62,44%
Domain Knowledge Features	90,22%	23,95%	84,03%	29,32%
Unpartitioned set of features	91,00%	19,11%	65,87%	48,31%

Table 7: Predictive performance within the six experiments that have been conducted. All metrics are given as averages over the ten cross-validation instances that have been computed.

Concerning the precision, every model yielded an acceptable precision of at least almost 90%, i.e. 90% of all predictions, which resulted in *normal*, were actually correct. A closer look on the models' performance does not expose a high variance in precision: The model based on domain knowledge-features yields comparable results with respect to the models that are based on algorithmic feature selection.

Regarding the recall metric, the domain knowledge-model again exposes an average performance for the normal cycles compared to the models which are based on automatically extracted features. The domain knowledge-model

predicts 84% of all normal cycles correctly. The model of unpartitioned, automatic features yielded a recall of 65%, which may be considered to be acceptable while the models for cylinder heaters and plasticization unit without cylinder heaters respectively yield poorer result. In contrast to that, the models for clamping unit and hot runners perform excellent with recall metrics of approximately 92% and 90% respectively. The model of the cylinder heaters exposed the worst recall with approximately 40%.

Focusing on the anomalous cases, all models yielded a low precision, i.e. that the amount of true-anomalous cycles compared to all "anomalous" predictions is small. That is, there are many messages pointing at anomalies, where there is none. From a comparative perspective, we can state again that the domain knowledge-model exposes a mean precision (about 24%), while the hot runner model is the only model with an acceptable performance, yielding a precision of about 50%.

However, for the anomaly detection, a more decisive criterion is the recall of the anomalous cycles. If we consider the anomaly detection an information for the operator, it is most important to not miss an anomaly. In contrast to the metrics we considered until now, here, the specialized models of plasticization unit without cylinder heaters, hot runners respectively the cylinder heaters are exposing a high predictive performance (approx. 68%, 69% and 62%), and we observe remarkable low recall values with respect to the domain knowledge-model (approx. 29%) and the model based on the clamping unit feature set (approx. 15%). Since it is most important to recognize as many of the true anomalies, most specialized models outperform the unpartitioned model.

Although there is a general objective to achieve high values, i.e. values close to 100%, for accuracy, precision and recall, one should take a closer look at criterions' meanings with regard to manufacturing objectives though. While accuracy gives rather general information on a model's predictive performance, precision and recall have a more specific meaning that can be taken into account when inferring process adaptations. In other words, the reasons why the values of the different criterions should be as high as possible differs.

As the precision provides information on the percentage of cycles with a specific predicted anomaly status that actually have this status, the precision of "normal predictions" is a quality issue, while the precision of "anomalous predictions" is an efficiency issue. Let us for the following examples assume that the anomaly rating is linked to the parts' quality: In the first case, one aspires towards high values, since each false prediction implies one molded part that does potentially not meet the quality criteria while the cycle is rated normal. In the second case, one aspires towards high values, since each false prediction implies one molded part that does most probably meet the quality criteria, while it is disposed due the anomalous rating.

In contrast to that, the recall provides information on the percentage of cycles with a certain predicted anomaly status compared to those cycles that are actually of that status. Again assessed with respect to the manufacturing, this

means that the recall of “normal predictions” is an efficiency issue, while the recall of “anomalous predictions” is a quality issue.

Consequently, in case of molded parts where quality is the major criterion for product success, in case of a tradeoff, one would prefer high values for precision of normal predictions and recall of the anomalous predictions to those for precision of anomalous predictions and recall of normal predictions. In the cases where the product price predominates, it is vice versa. In the presented industrial application that is known to be rather quality critical, one would prefer the algorithmic feature subsets to the unpartitioned feature set and the domain knowledge-based feature set.

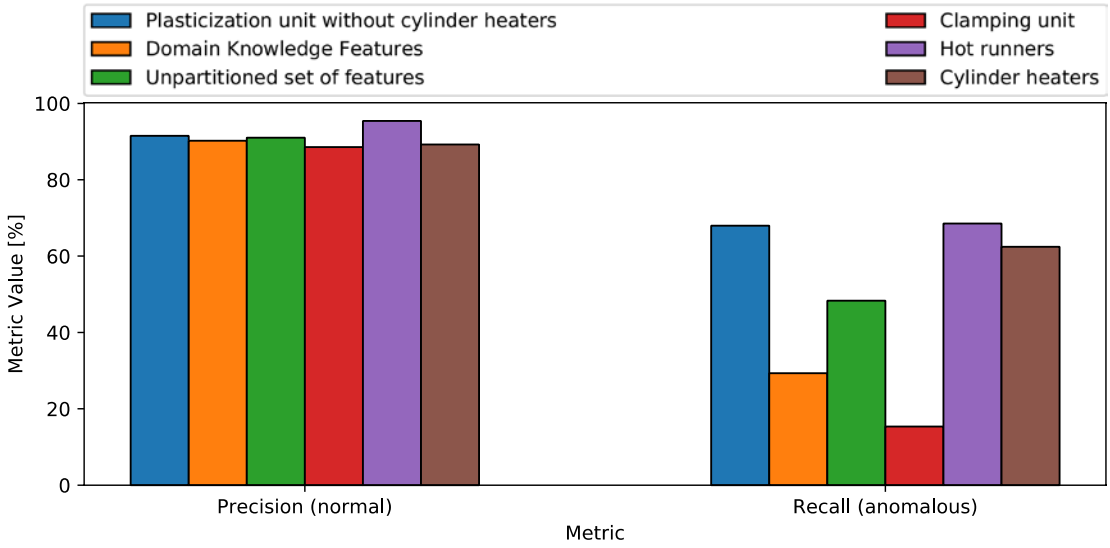

Figure 8: Bar chart representing the achieved precision (predicting normal cycles) and recall (predicting anomalous cycles) across our experiments. These metrics are indicative for the predictive performance of our models, given a quality-critical production process.

4 SUMMARY AND OUTLOOK

We presented the overall pipeline of Anomaly Detection in Injection Molding. Due to lacking detailed quality information, the common application of supervised learning techniques is not possible. Hence, we developed an unsupervised approach which makes use of clustering to infer anomaly information from data. The clustering model is turned into a classification by a decision map based on cluster cardinality – the larger a cluster, the more normal are its members. We leveraged the clustering learning technique using Ensembles in order maximize prediction stability. Our pipeline is applicable

under varying circumstances because the Ensemble approach decreases the variance of clustering. Moreover, the pipeline is also easily transferred to new data sets, since an evolutionary optimization adapts the clustering to the data.

Since the quality of features most often determine the success of predictions, we discussed extensively how features may be extracted from multivariate time series and how to select features that are meaningful and hence are beneficial to our anomaly detection learning task.

The goal of the overall analytical process is to inform the operator of a machine timely and in understandable terms about possible anomalies. Hence, we developed a display showing the component and the anomaly rating which expresses the certainty of the prediction.

Finally, we applied our method on a real industrial data set. For one of the four component groups, namely the hot runners, the precision of normal cycles was excellent with more than 95%. Also, all other feature sets yielded very good results regarding the precision of at least 88% of correct predicted normal cycles. Taking into account the recall metrics for anomalous cycles, the models for plasticization unit without cylinder heaters, hot runners and cylinder heaters yield good results still predicting approximately two thirds of the actually anomalous cycles correctly.

Since the results are promising, future work on our technique may include engineering towards more sophisticated features. There is a plethora of methods that extract features form time series and more approaches wait to be tested. Another interesting topic is to scale up the learning and its optimization by parallel processing, using more special hardware, or streaming algorithms.

Furthermore, it is desirable to provide the machine operator with more detailed information about the reason of occurring anomalies, going beyond the machine component that is concerned. This root cause analysis process includes also the challenge of dealing with potentially unreliable sensor data. Once the underlying interference is known, recommendations for action or even automatic adaptations to the machine setting may be provided, ultimately achieving a closed-loop control for anomaly detection and problem solution.

Abbreviations

Abbreviation	Description
Inj1PrsAct	Actual melt pressure
Inj1VelAct	Actual screw advance speed
Inj1PosAct	Actual screw position
Inj1RpmAct	Actual screw rotational speed
ClpVelAct	Actual clamp unit velocity
ClpPosAct	Actual clamping unit position
ClpFceAct	Actual clamping force
Eje1VelAct	Actual ejector velocity
Eje1PosAct	Actual ejector position
Inju1PosAct	Actual injection unit position
Inju1VelAct	Actual injection unit velocity
Pmp1PrsAct	Actual hydraulic pressure
Inj1HopTmpAct	Actual barrel temperature at hopper position
Inj1HtgTmp1Act	Actual barrel temperature zone 1
Inj1HopEdAct	Actual duty cycle flange cooling
Inj1HtgEd1Act	Actual duty cycle barrel heater 1
MldHtg1Tmp1Act	Actual temperature hot runner 1
MldHtg1Ed1Act	Actual duty cycle hot runner heater 1
OilTmp1Act	Actual hydraulic oil temperature
TrigClpCls	Trigger close clamping unit
TrigClpOpn	Trigger open clamping unit
TrigInj1	Trigger injection
TrigHld1	Trigger holding
TrigPlst1	Trigger plasticizing
TrigInj1Dcmp1	Trigger decompression 1
TrigInj1Dcmp2	Trigger decompression 2
TrigCool	Trigger cooling
TrigClpCfb	Trigger clamping force build-up
TrigClpCfr	Trigger clamping force release

TrigEje1Fwd	Trigger ejector forward
TrigEje1Rew	Trigger ejector rewind
TrigInju1Fwd	Trigger injection unit forward
TrigInju1Rew	Trigger injection unit rewind

ANN	Artificial neural network
APC	Adaptive process control
CVM	Core vector machine
fn	False negative
fp	False positive
MEB	Minimum enclosing ball
MPC	Model predictive control
MRMR	Minimum redundancy maximum relevance
NSGA	Nondominated sorting genetic algorithm
UTS	Univariate time series
MTS	Multivariate time series
PBT	Polybutylene terephthalate
PET	Polyethylene terephthalate
tn	True negative
tp	True positive
VDCVM	Vertically distributed core vector machine

References

[1]	Holzinger, G.; Schiffers, R.; Moser, S.; et al.	An Adaptive Filling to Packing Switchover Method for Injection Molding SPE-ANTEC Tech. Papers, Las Vegas (USA), (2014).
[2]	Caliendo, H.	Injection Molding: Engel launches new version of iQ weight control software URL: https://www.ptonline.com/products/injection-molding-engel-launches-new-verision-of-iq-weight-control-software, accessed November 27, (2017).
[3]	Hopmann, C.; Abel, D.; Heinisch, J; et al.	Self-optimizing injection molding based on iterative learning cavity pressure control Production Engineering – Research and Development. Springer Verlag, (2017). DOI: 10.1007/s11740-017-0719-6
[4]	Kruppa, S.; Schiffers, R.; Busl, M.; et al.	Advanced Data Acquisition and Analysis for Injection Molders SPE-ANTEC Tech. Papers, Anaheim (USA), (2017).
[5]	Michaeli, W.; Schreiber, A.; Lettowsky, C.	Optimierte Prozessführung beim Spritzgießen von Thermoplasten auf Basis von Prozessgrößen Zeitschrift Kunststofftechnik 4 (2008) 1
[6]	Vaculik, R.	Regelung der Formteilqualität beim Spritzgießen auf der Basis statistischer Prozessmodelle Dissertation, RWTH Aachen (1996)
[7]	Girth, M.	Methoden und Hilfsmittel zur prozessnahen Qualitätssicherung beim Spritzgießen von Thermoplasten Dissertation, RWTH Aachen (1992)
[8]	Michaeli, W.; Gruber, J.	Prozessführung beim Spritzgießen – Direkte Regelung des Werkzeuginnendrucks steigert die Reproduzierbarkeit Zeitschrift Kunststofftechnik 1 (2005) 1
[9]	Gruber, J.	Prozessführung beim Thermoplastspritzgießen auf Basis des Werkzeuginnendrucks Dissertation RWTH Aachen (2005)

[10]	Schiffers, R.	Verbesserung der Prozessfähigkeit beim Spritzgießen durch Nutzung von Prozessdaten und eine neuartige Schneckenhubführung Dissertation Universität Duisburg-Essen (2009)
[11]	Heinzler, F.	Modellgestützte Qualitätsregelung durch eine adaptive, druckgeregelte Prozessführung beim Spritzgießen Dissertation, Universität Duisburg-Essen (2014)
[12]	Kruppa, S.	Adaptive Prozessführung und alternative Einspritzkonzepte beim Spritzgießen von Thermoplasten Dissertation, Universität Duisburg-Essen (2015)
[13]	Johannaber, F.; Michaeli, W.	Handbuch Spritzgießen 2. Auflage, Hanser Verlag (2004) DOI: 10.3139/9783446440982
[14]	Bourdon, R.	Zur Optimierung der Prozessrobustheit beim Spritzgießen Dissertation, Universität Erlangen-Nürnberg (1994)
[15]	Eben, J.	Identifikation und Reduzierung realer Schwankungen durch praxistaugliche Prozessführungsmethoden beim Spritzgießen Dissertation, Technische Universität Chemnitz (2014)
[16]	Al-Haj Mustafa, M.	Modellbasierte Ansätze zur Qualitätsregelung beim Kunststoffspritzgießen Dissertation Universität-Gesamthochschule Essen (2000)
[17]	Schreiber, A.	Regelung des Spritzgießprozesses auf Basis von Prozessgrößen und im Werkzeug ermittelter Materialdaten Dissertation, RWTH Aachen (2011)
[18]	Bauer, M.; Gather, U.; Imhoff, M.	The Identification of Multiple Outliers in Online Monitoring Data Technical Report, Sonderforschungsbereich 475, Universität Dortmund (1999)

[19] Bhaduri, K.; Matthews, B. L.; Giannella, C.
Algorithms for speeding up distance-based outlier detection
Proceedings of KDD'11, pp. 859-867, (2011)
DOI: 10.1145/2020408.2020554

[20] Brown, G.
Ensemble Learning
In C. Sammut, & G. I. Webb (ed.), Encyclopedia of Machine Learning, pp. 312-320. Boston, MA: Springer US. (2010)

[21] Burkschat, M.; Cramer, E.; Kamps, U.
Beschreibende Statistik: Grundlegende Methoden der Datenanalyse
Springer Spektrum. (2012)
DOI: 10.1007/978-3-642-30013-4

[22] Das, K.; Bhaduri, K.; Votava, P.
Distributed anomaly detection using 1-class SVM for vertically partitioned data
Stat. Anal. Data Min., 4, pp. 393-406. (2011)
DOI: 10.1002/sam.10125

[23] Deb, K.
Introduction to Evolutionary Multiobjective Optimization
In J. Branke, K. Deb, K. Miettinen, & R. Słowiński (ed.), Multiobjective Optimization: Interactive and Evolutionary Approaches, pp. 59-96. Berlin, Heidelberg: Springer Berlin Heidelberg. (2008)
DOI: 10.1007/978-3-540-88908-3

[24] Deb, K.; Beyer, H.-G.
Self-Adaptive Genetic Algorithms with Simulated Binary Crossover
Evol. Comput., 9, pp. 197-221. (2001)
DOI: 10.1162/106365601750190406

[25] Deb, K.; Deb, D.
Analysing Mutation Schemes for Real-parameter Genetic Algorithms
Int. J. Artif. Intell. Soft Comput., 4, pp. 1-28. (2014)
DOI: 10.1504/IJAISC.2014.059280

[26] Deb, K.; Pratap, A.; Agarwal, S., et al.
A fast and elitist multiobjective genetic algorithm: NSGA-II
IEEE Transactions on Evolutionary Computation, 6, pp. 182-197. (2002)
DOI: 10.1109/4235.996017

[27] Ding, C. H.; Peng, H.
Minimum Redundancy Feature Selection from Microarray Gene Expression Data
2nd IEEE Computer Society Bioinformatics Conference (CSB 2003), 11-14 August 2003, Stanford, CA, USA. 0, pp. 523-529. Los Alamitos, CA, USA: IEEE Computer Society. (2003)
DOI: 10.1109/CSB.2003.1227396

[28] Ester, M.; Kriegel, H.-P.; Sander, J., et al.
A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise
Proceedings of the 2nd Internation Conference on Knowledge Discovery and Data Mining (KDD) pp. 226-231. AAAI Press. (1996)

[29] Fahad, A.; Alshatri, N.; Tari, Z., et al.
A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis
IEEE Transactions on Emerging Topics in Computing, 2, pp. 267-279. (2014)
DOI: 10.1109/TETC.2014.2330519

[30] Gan, J.; Tao, Y.
DBSCAN Revisited: Mis-Claim, Un-Fixability, and Approximation
Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, pp. 519-530. New York, NY, USA: ACM. (2015)
DOI: 10.1145/2723372.2737792

[31] Guyon, I.; Elisseeff, A.
An Introduction to Variable and Feature Selection
Journal of Machine Learning Research (JMLR), Special Issue on Variable and Feature Selection, 3, 1157-1182. (2003)

[32] Häussler, J.
Eine Qualitätssicherungsstrategie für die Kunststoffverarbeitung auf der Basis künstlicher Neuronaler Netzwerke
Dissertation, Universität Gesamthochschule Essen - Fachbereich 12 - Maschinentechnik. (1994)

[33] Haman, S.
Prozessnahes Qualitätsmanagement beim Spritzgießen
Dissertation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität Chemnitz. (2003)

[34] Hastie, T.; Tibshirani, R.; Friedman, J.
The Elements of Statistical Learning: Data Mining, Inference, and Prediction
(2nd ed.). Springer. (2009)
DOI: 10.1007/978-0-387-84858-7

[35] Kohavi, R.
A Study of Cross-validation and Bootstrap for Accuracy Estimation and Model Selection
Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2, pp. 1137-1143. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. (1995)

[36] Kriegel, H.-P.; Kröger, P.; Zimek, A.
Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering
ACM Transactions on Knowledge Discovery from Data, 3, 1:1--1:58. (2009)
DOI: 10.1145/1497577.1497578

[37] Rockney, B.
Injection Molding Machine. How to work injection moulding
University of Alberta Industrial Design: URL: https://commons.wikimedia.org/wiki/File:Injection_moulding.png, accessed May 10, (2018). Picture licensed under CC BY 3.0 (https://creativecommons.org/licenses/by/3.0/deed.en, accessed May 10 2018). All labeling has been removed from the scheme.

[38] Morik, K.; Wessel, S.
Incremental Signal to Symbol Processing
In K. Morik, V. Klingspor, & M. Kaiser (ed.), Making Robots Smarter -- Combining Sensing and Action through Robot Learning, pp. 185-198. Kluwer Academic Publ. (1999)
DOI: 10.1007/978-1-4615-5239-0_11

[39] Saeys, Y.; Inza, I.; Larrañaga, P.
A Review of Feature Selection Techniques in Bioinformatics
Bioinformatics, 23, pp. 2507-2517. (2007)
DOI: 10.1093/bioinformatics/btm344

[40] Schowe, B.; Morik, K. — Fast-Ensembles of Minimum Redundancy Feature Selection. In O. Okun, G. Valentini, & M. Re (ed.), Workshop at ECML PKDD on Supervised and Unsupervised Ensemble Methods and their Applications - SUEMA 2010. (2010) DOI: 10.1007/978-3-642-22910-7_5

[41] Sokolova, M., Lapalme, G. — A systematic analysis of performance measures for classification tasks. Information Processing & Management, 45, pp. 427-437. (2009) DOI: 10.1016/j.ipm.2009.03.002

[42] Stolpe, M.; Bhaduri, K.; Das, K., et al. — Anomaly Detection in Vertically Partitioned Data by Distributed Core Vector Machines. Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III, pp. 321-336. Springer. (2013) DOI: 10.1007/978-3-642-40994-3_21

[43] Tsang, I. W.; Kwok, J. T.; Cheung, P.-M. — Core Vector Machines: Fast SVM Training on Very Large Data Sets. J. Mach. Learn. Res., 6, pp. 363-392. (2005)

[44] Wrobel, S.; Joachims, T.; Morik, K. — Maschinelles Lernen und Data Mining. In Handbuch der Künstlichen Intelligenz, pp. 405-472. de Gruyter. (2013) DOI: 10.1524/9783486719796

[45] Stolpe, M.; Blom, H.; Morik, K. — Sustainable industrial processes by embedded real-time quality prediction. In Computational Sustainability, pp. 201–243. Springer. (2016) DOI: 10.1007/978-3-319-31858-5_10

[46] Lin, J.; Keogh, E.; Wei, L., et al. — Experiencing SAX: a novel symbolic representation of time series. In Data Mining and Knowledge Discovery, 15(2): pp. 107–144, (2007). DOI: 10.1007/s10618-007-0064-z

[47] Siebes A.; Vreeken J.; van Leeuwen M.
Item sets that compress
In Joydeep Ghosh, Diane Lambert, David B. Skillicorn, and Jaideep Srivastava, (ed.), Proceedings of the 6th SIAM International Conference on Data Mining, pp. 395–418, (2006).
DOI: 10.1137/1.9781611972764.35

[48] Klingenberg, R.; Joachims, T.
Detecting concept drift with support vector machines
In Pat Langley (ed.): Proceedings of the Seventeenth International Conference on Machine Learning (ICML), pages 487–494, San Francisco, CA, USA, Morgan Kaufmann. (2000)

[49] Bifet, A.
Classifier Concept Drift Detection and the Illusion of Progress
In: Rutkowski L., Korytkowski M., Scherer R., Tadeusiewicz R., Zadeh L., Zurada J. (eds) Artificial Intelligence and Soft Computing. ICAISC 2017. Lecture Notes in Computer Science, vol 10246. Springer, Cham (2017)
DOI: 10.1007/978-3-319-59060-8_64

[50] Gomes, H.; Bifet, A.; Read, J., et al.
Adaptive random forests for evolving data stream classification
In: Machine Learning, 106(9): pp. 1469–1495, (2017).
DOI: 10.1007/s10994-017-5642-8

[51] Gama, J.; Žliobaitė, I.; Bifet A., et al.
A survey on concept drift adaptation
ACM Comput. Surv., 46(4):44:1–44:37, (2014).
DOI: 10.1145/2523813

[52] Mierswa, I.; Morik, K.
Automatic feature extraction for classifying audio data
Machine Learning Journal, 58:127–149, (2005).
DOI: 10.1007/s10994-005-5824-7

[53] Lanius, V.; Gather, U.
Robust online signal extraction from multivariate time series
Computational Statistics Data Analysis, 54(4):966 – 975, (2010).
DOI: 10.1016/j.csda.2009.10.009

[54] Schäfer, O. Controlled part quality
In Today – the Arburg Magazine, 35, p. 19 (2007)

[55] Givens, G. H.; Hoeting, J. A. Numerical Integration
In Computational Statistics (eds G. H. Givens and J. A. Hoeting). (2013)
DOI: 10.1002/9781118555552

Bibliography

DOI 10.3139/O999.02052018
Zeitschrift Kunststofftechnik / Journal of Plastics Technology 14 (2018) 5; page 301–347

ISSN 1864 – 2217

Keywords:
Injection molding, process monitoring, anomaly detection, unsupervised learning, cluster analysis

Stichworte:
Spritzgießen, Prozessüberwachung, Anomalie-Erkennung, unüberwachtes Lernen, Cluster-Analyse

Autor / author:
Prof. Dr.-Ing. Reinhard Schiffers
Prof. Dr.-Ing Johannes Wortberg
M.Sc. Alexander Schulze Struchtrup
Institute of Product Engineering
University of Duisburg-Essen
Lotharstr. 1
D-47057 Duisburg

E-Mail: reinhard.schiffers@uni-due.de
Webseite: www.uni-due.de/kkm
Phone: +49 (0) 203/379-2500
Fax: +49 (0) 203/379-4379

Prof. Dr. Katharina Morik
B.Sc. Philipp-Jan Honysz
Chair of Artificial Intelligence
TU Dortmund University
Otto-Hahn-Str. 12
D-44227 Dortmund

E-Mail: katharina.morik@tu-dortmund.de
Webseite: www-ai.cs.tu-dortmund.de
Phone: +49 (0) 231/755-5100
Fax: +49 (0) 231/755-5105

Verlag / Publisher:
Carl-Hanser-Verlag GmbH & Co. KG
Wolfgang Beisler
Geschäftsführer
Kolbergerstraße 22
D-81679 München
Phone: +49 (0)89/99830-0
Fax: +49 (0)89/98480-9
E-Mail: info@hanser.de

Redaktion / Editorial Office:
Dr.-Ing. Eva Bittmann
Jannik Werner, M.Sc.
E-Mail: redaktion@kunststofftech.com