Stauder, Maximilian; Kühl, Niklas

AI for in-line vehicle sequence controlling: development and evaluation of an adaptive machine learning artifact to predict sequence deviations in a mixed-model production line

Flexible Services and Manufacturing Journal

Provided in Cooperation with:
Springer Nature

Suggested Citation: Stauder, Maximilian; Kühl, Niklas (2021) : AI for in-line vehicle sequence controlling: development and evaluation of an adaptive machine learning artifact to predict sequence deviations in a mixed-model production line, Flexible Services and Manufacturing Journal, ISSN 1936-6590, Springer US, New York, NY, Vol. 34, Iss. 3, pp. 709-747, https://doi.org/10.1007/s10696-021-09430-x

Flexible Services and Manufacturing Journal (2022) 34:709–747
https://doi.org/10.1007/s10696-021-09430-x

AI for in-line vehicle sequence controlling: development and evaluation of an adaptive machine learning artifact to predict sequence deviations in a mixed-model production line

Maximilian Stauder[1] · **Niklas Kühl**[1]

Accepted: 3 August 2021 / Published online: 15 August 2021

Abstract

Customers in the manufacturing sector, especially in the automotive industry, have a high demand for individualized products at price levels comparable to traditional mass-production. The contrary objectives of providing a variety of products and operating at minimum costs have introduced a high degree of production planning and control mechanisms based on a stable order sequence for mixed-model assembly lines. A major threat to this development is sequence scrambling, triggered by both operational and product-related root causes. Despite the introduction of Just-in-time and fixed production times, the problem of sequence scrambling remains partially unresolved in the automotive industry. Negative downstream effects range from disruptions in the Just-in-sequence supply chain, to a discontinuation of the production process. A precise prediction of sequence deviations at an early stage allows the introduction of counteractions to stabilize the sequence before disorder emerges. While procedural causes are widely addressed in research, the work at hand requires a different perspective involving a product-related view. Built on unique data from a real-world global automotive manufacturer, a supervised classification model is trained and evaluated. This includes all the necessary steps to design, implement, and assess an AI-artifact, as well as data gathering, preprocessing, algorithm selection, and evaluation. To ensure long-term prediction stability, we include a continuous learning module to counter data drifts. We show that up to 50% of the major deviations can be predicted in advance. However, we do not consider any process-related information, such as machine conditions and shift plans, but solely focus on the exploitation of product features like body type, power train, color, and special equipment.

Keywords In-line vehicle sequencing · Sequence scrambling · Supervised classification · Artificial intelligence · Concept drift

✉ Niklas Kühl
kuehl@kit.edu

Extended author information available on the last page of the article

Abbreviations

AI	Artificial Intelligence
ANN	Artificial Neural Network
AS/AR	Automated Storage and Retrieval
BTO	Build-to-Order
C	Trigger for the Relearning Cycle
D	Class of Delayed Instances
DoF	Degrees of Freedom
DSR	Design Science Research
FN	False Negative
FP	False Positive
I	Number of Instances for Pretraining the Artifact
ILVS	In-line Vehicle Sequencing
KPI	Key Performance Indicator
Λ	Parameter for the Page-Hinckley-Test
ML	Machine Learning
MMAL	Mixed-Model Assembly Line
OEM	Original Equipment Manufacturer
P	Class of punctual instances
PF	Product Family
PMS	Predictive Manufacturing System
PNR	Product Number
RG	Random Guess Classifier
SA	Sequence Adherence
SD	Sequence Deviation
$SD_{0.9-quan}$	90% Quantile
$SD_{0.95-quan}$	95% Quantile
SML	Supervised Machine Learning
SMOTE	Synthetic Minority Oversampling
STD	Standard Deviation
T	Binarization Threshold
TN	True Negative
TP	True Positive
XGBoost	Extreme Gradient Boosting

1 Introduction

Since Henry Ford made his famous statement, *"Any customer can have a car painted any color that he wants, so long as it is black"* in 1909, the principles of modern car manufacturing assembly lines have changed notably. A study by Stäblein et al. (2011) shows that the theoretical variety of products that can be configured and ordered from one *original equipment manufacturer* (OEM) in the automotive sector ranges between the factors of 10^9 to 10^{24}, depending on the car model and the market region. According to this study, a given product specification is assembled 2.26 times per year on average. For the US market, Fettermann and Freitas (2017) found

a variety of nearly 10^{34} across OEMs. However, for an average-sized automotive production plant with a yearly output of 250.000 cars, more than 100.000 unique configurations of cars are produced (EMEA 2019a, b). To cope with the resulting complexity, all major OEMs implement a *build-to-order* (BTO) strategy (Meissner 2010). In BTO, the production of goods is usually triggered by confirmed customer orders. To realize such a broad product portfolio without incurring exorbitant assembly and setup costs, the concept of *mixed-model assembly lines* (MMAL) has become the industrial standard for high-value, customizable goods such as passenger cars (Kern et al. 2017). The transformation to increase assembly sequence efficiency is facilitated by several mathematical models that consider numerous factors.

Dörmer et al. (2015) name five distinctive types of planning problems that are related to MMAL production operations. Besides the problems of line balancing, master production scheduling, and material flow, the authors identify both production sequencing as well as re-sequencing as core activities. In the remainder of this study, we will tackle both sequencing as well as re-sequencing while master production planning, line balancing, and material flow are out of scope. The downside of MMALs resides within their sensitivity to disruptions, leading to deviations in the process sequence (Lehmann and Kuhn 2019). The main factors that cause disruptions can be divided into *product-related* and *operations or process-related factors* (Meissner 2010). While product-related factors reside within the nature of the actual product, that is, the set of attributes which characterizes a product (color, type, size. etc.), process-related factors include process discipline, machine breakdowns, resource allocation, parts availability, and errors. Certain process-related factors, such as machine breakdowns or quality audits, are *dynamic* and not necessarily determined at the start of the production of a specific customer order. The majority of research focuses on those dynamic, process-related causes, leading to a continuous improvement of JIT production systems by giving those systems more flexibility to react to unforeseen situations (Rudolf et al. 2014; Lehmann and Kuhn 2019). However, those studies take a rather reactive perspective, i.e., the proposed strategies aim at redeeming sequence scrambling after the production process started. To adopt a novel perspective, the current work focuses on the influence of product-related factors on sequence deviations. This perspective is rather rare in current research and industry. Product-related factors are deemed *static*, which means that they do not change once the customer order is placed. As most of the factors become known once a customer order is submitted to the production plant, a prediction of the expected sequence deviation for that order can be made before the production starts. That allows us to take a prescriptive perspective, i.e., we aim to take counteraction against sequence scrambling before the production process starts. The objective is to identify and learn from hidden patterns in complex data. *Artificial intelligence* (AI) in general and *supervised machine learning* (SML) in particular provide the necessary techniques to achieve this goal by revealing hidden correlations; in this case, the patterns between product-related factors and sequence deviations (Kühl et al. 2019). The predictions allow the identification of potential disruptions before creating the assembly sequence, which in turn allows preventive counteraction. For our specific case, AI offers some advantages compared to

traditional mathematical models. Methods of SML allow us to utilize big data at comparably low cost. Furthermore, SML models can be tailored to react to changes in the underlying processes, i.e., a change in the distribution of products, production times, or scrambling rates.

To contribute to the area of *Predictive Manufacturing Systems* (PMS) (Peres et al. 2019) in general and sequence scrambling in particular, we address a current research gap by combining a singular product-related perspective with state-of-the-art AI tools. However, being aware that process-related features exert a significant influence on the stability of a sequence (Boysen et al. 2012; Lehmann and Kuhn 2019), we consciously isolate product-related factors in our analysis and use them as features of our prediction model.

In the remainder of this study, we review the current state of research and provide essential conceptual foundations. This is followed by a presentation of our proposed methodology that shows how an AI artifact, which integrates a SML classification model and an online learning module, can be implemented to predict sequence deviations. Through this methodological approach, we conduct a field study in collaboration with a leading global automotive manufacturer. Finally, we evaluate our results, provide a critical discussion, and conclude with findings, limitations, and prospects.

2 Literature review

At the outset, to ensure a common understanding, we provide a structured overview of the related work and present our approach in the context of existing knowledge (see Table 1).

Analysis and simulation of disturbances in ILVS have been the subject of a rich body of research which, in general, can be subdivided according to its focus and methodological approach. Table 1 frames the existing research and provides an overview of the most relevant concepts. We distinguish between a product-oriented perspective that targets inherent product characteristics, and a process-/operation-oriented view that deals with procedural, operational, and organizational factors. Methodologically, we distinguish between traditional concepts, namely simulation studies, heuristics or mathematical models, and perceptions based on the applied field of artificial intelligence.

First, the lower left quadrant of Table 1 includes literature based on traditional methods that has a process or operation-oriented focus. This perspective is adopted in various studies that evaluate the downstream effects of sequence scrambling: notably, the effect of individual workstations on sequence stability is quantified by Rudolf et al. (2014), whereas attainable stability levels and their preconditions are estimated by Lehmann and Kuhn (2019), using a simulation approach. The concept that denotes the reordering of products by performing a physical switch according to the desired sequence is *physical resequencing* (Boysen et al. 2012). A prerequisite of physical resequencing is the existence of at least one physical decoupling buffer within the line. An automated storage and retrieval buffer (AS/AR), which is located before the assembly line, enables the position switch of products (Inman 2003).

Table 1 Overview of related research approaches

		Method	
		Traditional	Artificial Intelligence
Focus	Product-oriented	*Postponement* (Fournier and Agard 2007) *Virtual Resequencing* (Inman and Schmeling 2003; Meißner et al. 2008; Boysen et al. 2009) *Optimal AS/AR Pre-filling* (Gunay and Kula 2016, 2018) *Intentional Sequence Scrambling* (Gusikhin et al. 2007; Meissner 2010)	Our approach
	Process-/operation-oriented	*Simulation of Downstream Effects* (Rudolf et al. 2014; Lehmann and Kuhn 2019) *AS/AR Buffer Sizing and Allocation* (Inman 2003; Urnauer et al. 2019) *Resequencing Strategies* (Ding and Sun 2004; Lahmar and Benjaafar 2007; Gujjula and Günther 2009; Boysen et al. 2011; Franz et al. 2014, 2015; Moetz et al. 2019)	*Multistage Quality Control* (Peres et al. 2019) *Outcome-Oriented Predictive Process Monitoring* (Teinemaa et al. 2017) *Remaining Time Prediction* (Tax et al. 2017; Polato et al. 2018)

Building on this study's results, authors subsequently developed alternative buffering systems to perform physical resequencing, such as *mix-banks* (Lahmar and Benjaafar 2007; Meissner 2010) and *pull-off tables* (Gujjula and Günther 2009). Besides the dimensioning of buffers and acknowledging joint capacity restrictions, Urnauer et al. (2019) provide a simulation-based optimization methodology for the location and allocation of buffer spaces between subsequent production steps. Present buffer operating strategies are based either on the utilization of downstream buffers (Boysen et al. 2011) or on specific shuffling areas that change the order of products. The harmful impact of inserting rescheduled products into an existing sequence in terms of worker utilization and work distribution can be reduced by employing a non-trivial strategy (Lahmar and Benjaafar 2007; Gujjula and Günther 2009; Franz et al. 2015). These *resequencing strategies*, specifically in the specific context of complex automotive supply chains and assembly procedures, are benchmarked in detail. It is possible, depending on the complexity of the strategy, to avoid at least 75% of the overload situations (Franz et al. 2014, 2015). Moetz et al. (2019) adopted a network perspective by developing a conceptual model to manage short-term disruptions, resulting in a hierarchical planning system. However, most authors neglect the practical implementation of such a system.

Second, along with the aforesaid process and operation-based strategies, the product-oriented approaches listed in the upper left quadrant of Table 1 utilize the similarities and disparities of product characteristics. Several studies confirm the

influence of substitution approaches, such as the flexibilization of the sequence, by *postponing* the assignment of products to customer orders (Fournier and Agard 2007) or by decoupling orders from physical products (Inman and Schmeling 2003). The latter, usually referred to as *virtual resequencing,* is extensively analyzed (Inman and Schmeling 2003; Fournier and Agard 2007; Meißner et al. 2008; Boysen et al. 2009). For postponement, *points of differentiation* are introduced. Products are uniformly produced up to the first point of differentiation. Fournier and Agard (2007) analyze the introduction of two points of differentiation in the automotive production system and indicate that a 39,70% reduction of missing vehicles in the assembly line is achievable. Since these authors accept a uniform distribution of feature frequencies, a real production system's results would be less profitable. However, the benefits of postponement emerge when differentiation is delayed, that is, when products are standardized as much as possible for as long as possible. Again, the authors ignore practical implementation. To enable efficient virtual resequencing, it is essential to pre-fill the buffers with spare products. Gunay and Kula (2016) evaluated the provisioning of an optimal buffer pre-filling level and mixture in their two-stage stochastic model, as did Jin et al. (2019) in a hybrid simulation model. Another product-oriented approach is the reduction of sequence scrambling by taking preventive action. Collectively, these approaches are known as *hedging strategies*. (Meissner 2010) distinguishes between two different types of hedging strategies. The first focuses on implementing a high degree of process discipline and lean thinking to counteract disruptions; the second involves generic, proactive measures to counteract sequence scrambling. A *least in-sequence probability heuristic* for mixed-volume production lines is introduced to intentionally manipulate the input sequence (Gusikhin et al. 2007). The proponents of this technique show that high-volume products have a higher probability of meeting their required sequence position than low-volume ones. Thus, as a precautionary measure, they propose moving low-volume products ahead in the production sequence to introduce an individual time buffer for those products. This strategy is known as *intentional sequence scrambling*. The authors prove that a reduction of deviations could be achieved by combining this strategy with virtual resequencing approaches, allowing a size reduction of the physical buffer space by up to 30% based on the assumption that only three distinct products are produced.

Third, beyond these traditional methods, it is necessary to discuss existing work utilizing predictive models. These models that adopt a process-oriented view are summarized in the lower right quadrant of Table 1. While most of the traditional methods focus on reactive actions to restore the sequence, predictive models have the ability to guide proactive processes. The ability to predict the behavior of a running process is regarded as a key capacity of modern businesses (Houy et al. 2010). In a promising study, Peres et al. (2019) developed an automated defect detection tool based on SML. The authors showed how the early detection of defects, based on quality measures that are taken from inspection machines within the production line, leads to a reduction of scrap and deviation in the assembly line sequence. An automotive industry case study provided accuracy scores of 93% for the detection of defects in the automotive body shop. More recently, approaches also emerged in the area of predictive process monitoring and process mining. Evermann et al. (2017),

based on a process backlog of past events, proposed the application of a deep learning concept to predict the next step in a consecutive business process. A slightly different approach, proposed by Teinemaa et al. (2017), is the prediction of a process outcome. While business processes gained a lot of attention regarding process mining, manufacturing processes have been neglected (Son et al. 2014). Some outcome-oriented concepts regarding the exploitation of manufacturing process data focus on the prediction of the remaining time of a running process (Tax et al. 2017; Polato et al. 2018). All these studies use AI-based methodologies. Next, we indicate the foundations of the In-line vehicle sequencing concept.

3 Foundations

The concept of ILVS was first introduced by Morrison (1991). He points out that ILVS is a precondition for Just-in-Sequence (JIS). Sometimes, both terms are used interchangeably. ILVS, in turn, is an essential part of intermixed production on MMAL. Because of parallel developments, ILVS is known as the *pearl necklace concept* or *pearl-chain concept*, especially in the automotive industry (Weyer and Spath 2009; Klug 2017; Lehmann and Kuhn 2019). The process of creating the assembly line sequence is referred to as *initial sequencing*. The outcome of initial sequencing is a technically feasible and economically favorable chronological ordering of customer-specific purchase orders and is known as *assembly line sequence* or *frozen sequence* (Lehmann and Kuhn 2019). An assembly line sequence is considered favorable if "*vehicles with high-content options are spaced apart from each other to balance the work content*" (Morrison 1991, p. 547).

Currently, there are more complex, multi-criteria mathematical models that foster the initial sequencing of customer orders along a fixed planning horizon, for example, a day or a week (Drexl and Kimms 2001; Boysen et al. 2009; Giard and Jeunet 2010; Mayrhofer et al. 2011; Zhang and Gen 2011; Saif et al. 2019). In all the presented models, economic factors are the key to evaluate the quality of an assembly line sequence. These factors include the efficient utilization of machinery and workers, the setup costs, keeping to the delivery date promised to the customer, and the restriction of technical feasibility as well as minimizing searching costs for JIS parts on the assembly line. At the time of initial sequencing, JIS delivery schedules are submitted to the suppliers. It is important that the information lead time, that is, the time span between the initial sequencing and the entry into the assembly line is longer than the suppliers' parts delivery lead time. However, both the supplier and the producer benefit from advanced knowledge of the sequence. Parts are only delivered to the assembly line if needed, thus saving costs on inventory, buffer space, handling, and potential waste (Rudolf et al. 2014; Lehmann and Kuhn 2019).

Several required pre-assembly steps are placed between the commencement of production at the *sequencing point* (Swaminathan and Nitsch 2007), where the initial sequencing takes place and the start of the assembly line (Fig. 1). Each step adds to the risk of disordering the sequence due to products falling behind or taking over. While Meissner (2010) distinguishes between *product* and *process-related* factors, Grinninger (2012) distinguishes between *product*, *process,*

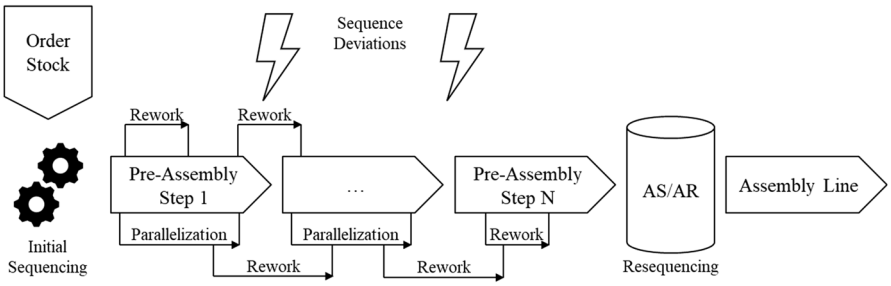

Fig. 1 Mixed-model production line

and *infrastructural-related* factors and further subdivides process-related factors into *steering*, *material* and *quality*. Product-related factors include differences in processing times, unsynchronized processes, the stage of the product lifecycle, and extra work. Process-related factors include missing parts, locally deviating conformity with objectives, steering errors, missing transparency, and the non-availability of information, as well as quality audits, machine breakdowns, and worker shifts. Infrastructural factors relate to split-and-merge-architectures, resequencing restrictions, and inappropriate conveyor technology. The negative effect of sequence deviations is called *unintended sequence scrambling*.

Several different *Key Performance Indicators* (KPIs) support the quality assessment of an ILVS production line. Meissner (2010) proposes different measures to evaluate sequence stability. The *sequence deviation* (SD) denotes the delta between the position of an order in the output and the input sequence:

$$SD_i = pos_i^t - pos_i^{t-1}$$

While pos_i^t represents the position number in the achieved sequence, pos_i^{t-1} represents the sequence position in the planned sequence that was inserted at the start of production. Therefore, SD denotes the degree of deviation for each order from its designated position in the input sequence. Relatively speaking, early orders have an $SD_i < 0$ and late orders an $SD_i > 0$. The *sequence adherence* (SA) is a measure to assess the stability level of the entire assembly sequence. The number of violations v of the assembly sequence is summed and divided by the total number of orders n in the respective sequence:

$$SA = 1 - \frac{1}{n} \sum_{i=1}^{n} v_i [\%]$$

Values between 0 and 100% can be obtained. Usually, the definition of a sequence violation is restricted to orders with sequence deviations of $SD_i > 0$, because the intended delaying of early orders is always possible at relatively low effort (Günther 2017). However, sequence deviations are interdependent. Each early product forces later products to be late.

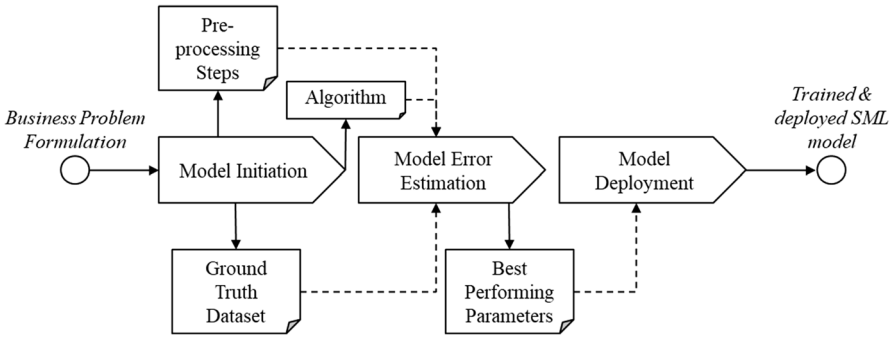

Fig. 2 Process model for supervised machine learning model based on Hirt et al. (2017)

The costs incurred by sequence scrambling emerge in multiple divisions, for example, logistics costs of the buffer space to enable physical resequencing, labor costs of the manual resequencing work, and capital costs of growing stocks. The sequence's quality primarily determines the efficiency of the assembly line's utilization. Imbalanced work across products and workstations, leading to situations of work overload or wasting resources, constitutes a main cost block in the automotive production systems (Franz et al. 2014). If deviations become too severe, the logistics department cannot guarantee the timely provision of the required material to the assembly line. As a result, empty cycles in the assembly sequence or termination of the assembly process may occur.

Machine learning (ML) is one of several method sets applied within the area of artificial intelligence. Kühl et al. (2019) provide a conceptual framework to specify the contribution of machine learning to AI and to enhance terminological clarity. The work at hand uses ML and AI interchangeably.

4 Methodology

Knowledge discovery is an evolving and dynamic area of research and, as a result, includes many different process models (Brodley and Smyth 1995; Kurgan and Musilek 2006; Choudhary et al. 2009). Hirt et al. (2017) provide a well-defined and specifically designed process model for SML tasks. In the next section, we follow the general steps of *model initiation*, *model error estimation*, and *model deployment* (Fig. 2). After describing the development of our SML classification model, we emphasize the need to consider continuous online learning. Hence, we embed the developed SML classification model in an AI artifact that features an additional online learning module and a continuous data input stream.

4.1 Business problem formulation

Before developing the SML classification model, it is necessary to clearly formulate the prevailing business problem. As discussed previously, disruptions in the

assembly line sequence involve a loss in the production efficiency of ILVS production systems. The objective of production managers is a sequence adherence of 100%, implying that the achieved assembly line sequence is identical to the input sequence. Losses arise as soon as a product is delayed. Sequence deviations that are successfully reversed before entering the assembly line do not harm sequence adherence. Sequence deviations in pre-assembly steps can be reversed by applying physical resequencing in the AS/AR buffer. The resequencing capability of the AS/AR buffer is an important parameter to assess the degree of acceptable sequence deviations in pre-assembly steps. Therefore, production control mechanisms aim to keep the sequence deviations in pre-assembly steps below the capacity of the AS/AR buffer.

To maximize sequence adherence, the business objective of AI artifact development is to identify products that are likely to produce a sequence deviation that exceeds the capacity of the AS/AR buffer. Considering the systems' resequencing capability, a threshold T in terms of a critical sequence deviation measure is derived from differentiating between two distinct product classes: punctual instances $p \in P, whereSD_p \leq T$ and delayed instances $d \in D, whereSD_d > T$. Accordingly, only instances $d \in D$ with sequence deviations above T are relevant to the business problem under consideration. Therefore, we formulate the problem as a classification task and use an externally provided threshold to binarize the target variable *SD* into two classes. Future research could formalize the business problem as a regression or a multi-class classification task.

4.2 Model initiation

Having formulated the business problem, we initiate the development of the classification model. SML models perform very well on large, high-dimensional datasets (Kotsiantis 2007). To obtain convincing results from a SML classification model, it is necessary to provide sufficient learning data (Wuest et al. 2014). The initiation steps comprise *problem exploration* and *data gathering*, *performance measure selection*, *preprocessing,* and *algorithm selection*. They require a preliminary understanding of both the business problem and the application domain. Whereas the business problem serves as the main input for model initiation, the outputs include a *preprocessing pipeline*, the *ground truth dataset,* and a suitable classification algorithm.

4.3 Problem exploration and data gathering

The exploration of problem-specific features is fundamental to successful SML classification models. First, all available product-related features from past instances must be identified, collected, and merged. The result is a set of categorical or numerical variables describing historical instances. Typically, product-related features are organized in *feature families*. Each family contains at least two distinct product feature values; for example, the feature family *color* could have the values *black*, *red*, or *white*. Second, the target variable must be defined. The sequence deviation is either

Table 2 Exemplary dataset of historic samples with assigned feature values and binary label

ID	Fam 1	Fam 2	Fam 3	…	Fam N-1	Fam N	Target Variable
1	Value A	Value A	Value D		Value B	Value A	D
2	Value B	Value D	Value G		Value A	Value E	P
…							
M	Value A	Value C	Value C		Value A	Value J	P

directly retrieved from information storage systems or derived from related data. The target variable in the proposed SML classification model is the class that an instance belongs to. Based on the threshold T, the binary target variable is derived from the sequence deviation measure by performing a *binarization*. The result of the data gathering step is a list of M historic instances, described by a set of N product-related features and the target variable (Table 2).

4.4 Performance measure

We consider various prediction performance measures for the binary classification task. The aim is to attain either the highest possible precision by minimizing the frequency of false predictions of punctual products as delayed—or the highest possible recall by maximizing the rate of detected delayed products. Focusing only on a single measure leads to mutual losses. Depending on the inferred costs, a suitable trade-off ensures that the overall error costs are minimized. Therefore, our key performance measure is the f_β*-score as recommended by* (Powers 2011)) to counter prediction biases:

$$f_\beta\text{-}score = \left(1 + \beta^2\right) * \frac{precision * recall}{\beta^2 * precision + recall}$$

The parameter β can be adjusted to overweigh one of the two metrics, depending on the individual necessity of the model.

4.5 Preprocessing pipeline

The quality and representation of data instances used to train the classification model is a critical success factor of knowledge discovery in SML (Kotsiantis et al. 2006). To attain a high level of training data quality, the collected set of raw data is subject to a set of preprocessing techniques. All preprocessing steps are incorporated into the *preprocessing pipeline*. Unfortunately, there is no unique, undisputable set of preprocessing steps that performs best across datasets. The selection of favorable preprocessing techniques depends on the data characteristics, as well as on the classification model's architecture. Accordingly, we present an extract of preprocessing techniques deemed suitable for the task at hand (Table 3). However, the ultimate implementation of the preprocessing techniques is limited to the specific

Table 3 Overview of proposed preprocessing techniques

Problem	Preprocessing technique	Description
Illegal/Irregular values	Filter rules (Kotsiantis et al. 2006)	Dismisses invalid, undesirable or duplicated instances based on filter rules
	Outlier Removal	Dismisses outliers that are not representative
Machine interpretability	One-hot-encoding	Transforms categorical columns into multiple binary columns
	Label encoding	Transforms categorical columns into a numerical column
	Standardization	Standardizes continuous feature columns
Feature dimensionality	Manual selection	Considers only manually selected features for the prediction model
	Frequency removal	Removes data columns with little change to the feature values
	Automated grouping	Combines columns with very similar trends, i.e. collinearity
Class imbalance	Imbalanced training	Trains on an imbalanced dataset; alternatively, adjust the class loss functions
	Undersampling (Bach et al. 2017)	Creates a subset of the majority class by removing samples
	Oversampling (Bach et al. 2017)	Creates a superset of the minority class by replicating
	Synthetic sampling (Blagus and Lusa 2013)	Synthetically creates new samples to enlarge the minority class
Missing relevant features	Neighborhood features	Considers features of preceding products
	Batch features	Considers statistical features of the current production batch

application case. Therefore, a final selection of preprocessing techniques depends on an evaluation using the application data.

Some of the preprocessing steps are *mandatory* to enable implementation, that is, the removal of illegal, misspelled, and mislabeled variables or the conversion of values into a machine-readable format. An *instance selection* algorithm is employed to remove invalid instances from the dataset based on domain-specific *filter rules*. The use of filter rules requires knowledge of applicable filter criteria. If this knowledge is not feasible, an *outlier removal* algorithm serves as an alternative to detect faulty instances not representative of the dataset. The outlier removal algorithm dismisses instances that exceed a permission range in terms of absolute or statistical values, for example, variance (Kotsiantis et al. 2006).

A classification model can only interpret numerical data. We apply different *feature encoding* techniques to convert categorical values into machine-readable, numeric values. A *one-hot-encoding* transforms categorical data into binary columns. The number of binary columns per feature column depends on the cardinality of the feature's values. This ensures that the individual categorical values remain independent. Alternatively, *label-encoding* is suitable if the categorical variables can be brought into an ordinal dependency, for example, sizes or distances. *Standardization* manipulates the distribution of continuous values without changing the information content. Standardized values are easier to interpret by a machine learning model (García et al. 2016).

Although the dataset can be used to train an ML model, certain additional preprocessing steps would increase its performance. *Optional* preprocessing steps aim to increase the model's performance. Hence, we propose different techniques to reduce the dimensionality of the dataset. The first option is a *manual feature selection*, that is, the reduction of the potential space of factors based on expert knowledge. To achieve satisfactory results from manual selection, it is necessary to retain features with a high impact while removing those of lesser or no importance. However, this requires some knowledge of the influence of individual features. A data-centered technique to reduce the dimensionality of the feature space is *frequency removal*, that is, removal of feature columns with only one predominant feature value. Lastly, the technique of *automated grouping* identifies linear-dependent features. Since these linear-dependent features do not add value to our dataset, they can be removed. A combination of multiple dimensionality reduction algorithms is also conceivable.

Many SML applications face the problem of imbalanced datasets. The imbalance leads to the biased training of the model. A *class balancing* technique is applied to balance the training data while the test data remain untouched. A balanced training dataset reduces the danger of prediction biases favoring the majority class (More 2016). Different sampling techniques are available to cope with class imbalance. We present trivial techniques that include imbalanced training with and without *weighted losses*. The weighted losses regulate the classifier during training by overrating errors on the minority class according to the inverse of its relative frequency. Besides *imbalanced training*, we also identify *under*, *over,* and *synthetic sampling* techniques. The science behind sampling is linked to an extensive field of research (Chawla et al. 2002; Batista et al. 2004; He and Garcia 2009; Bach et al. 2017;

Boardman et al. 2018). The literature recommends *synthetic methods*, although they infer an additional overlap between the classes, which disadvantages the prediction performance (Chawla et al. 2002).

Lastly, an extension of the feature space could be beneficial. For the SML task in an ILVS scenario and to account for effects that may occur in a pattern of subsequent instances, relevant *feature engineering* techniques cover the inclusion of *neighborhood features*. *Batch features* aggregate the product features of preceding instances over a longer time period to elucidate macro-effects resulting from the distribution of product-related features across a larger production batch. Both features account for interdependency effects in the assembly line sequence.

4.6 Classification model

The selection of a suitable classification algorithm is important. Depending on the specificity of the task, a suitable algorithm is chosen on the basis of various technical and environmental factors (Reda et al. 2019). Generally, an algorithm's applicability, prediction performance, and resource efficiency are important.[1] Its degree of transparency is an additional factor. For the SML task, we consider four predictive classification algorithms: A *Logistic Regression,* an *Artificial Neural Network* (ANN), a *Random Forest*, and an *Extreme Gradient Boosting* (XGBoost), which we will shortly introduce. A Logistic Regression models the probability of a certain class by assigning a probability between 0 and 1, based on a logistic function of the data (Tolles and Meurer 2016). An ANN is a collection of connected nodes (so-called neurons), which are inspired by the biological neurons from the brain. Each connection can transmit a signal to other neurons. Signals travel from the input layer to the output layer, typically after traversing other layers multiple times (Hassoun 1995). A Random Forest is a so-called ensemble learning model (Opitz and Maclin 1999) for which operates by building multiple decision trees during training and outputting the class that is the mode of the individual trees (Breiman 2001). Similar to Random Forests, XGBoost is a technique producing a prediction model in the form of an ensemble of so-called "weak learner" prediction models, more precisely, boosted trees (Chen and Guestrin 2016). For more information on the inner workings of the utilized classification models, we refer to related literature.

Depending on the classification algorithm, several parameters that cannot be learned from the data by training, known as *hyperparameters*, must be set. For the ANN, the relevant hyperparameters include the number of layers, the number of neurons per layer, and the activation functions. For an XGBoost, they include the booster technique and the maximum depth of the tree. For a Random Forest, the minimum number of samples that are required to split and the number of trees in the forest are important. The process of finding a suitable parameter setting is referred to as *hyperparameter tuning*.

[1] In addition, *fairness* is also deemed important in some current applications (Barocas et al. 2018).

4.7 Model error estimation

The selection of both suitable preprocessing techniques and the classification algorithm affects the prediction model's performance. Due to combinatorial cardinality, computing resource restrictions often preclude an exhaustive evaluation of all possible combinations of preprocessing techniques and classification algorithms. To cope with limited resources, we propose the independent testing of preprocessing steps and algorithms. Accordingly, we evaluate optional preprocessing techniques using a default hyper-parametrized XGBoost classification algorithm. To perform hyperparameter tuning, we use a set of default preprocessing techniques. A *search space* is defined for the relevant hyperparameters to perform a *grid search*, that is, to evaluate each permutation of the specified parameter values (Bergstra et al. 2011).

Within each step of preprocessing and algorithm tuning, we can either perform an *exhaustive grid search*, that is, an evaluation of each possible combination, or we can use a heuristic method based on a *greedy strategy* to evaluate the different alternatives. An exhaustive grid search requires extensive computing resources, whereas a greedy strategy is computationally more efficient because alternative techniques are evaluated in sequence. We propose the incorporation of *nested k-fold cross-validation* (Varma and Simon 2006; Cawley and Talbot 2010) for both the exhaustive grid search and the heuristic method. This involves stepwise nested cross-validation, thus performing the hyperparameter tuning and model-error estimation independently. The stepwise nested k-fold cross-validation is superior to a simple *train-test-split* as it prevents the previously discussed possibility of *overfitting* (Cawley and Talbot 2010).

After independently obtaining suitable preprocessing steps and hyperparameters for the classifiers, a cross-algorithm comparison identifies the most suitable classification algorithm. To enable this, each model is provided with identical data and again evaluated on their tuned hyperparameters using a k-fold cross-validation. The algorithms are compared based on the achieved mean f_β*-score*, and stability, that is, variance across the cross-validation runs. The training of the final model is based on the best-performing algorithm and its hyperparameter settings. In the final training, all available data samples serve as training instances. However, as we incorporate online learning, we do not deploy our SML classification model at this point. In the next chapter, we describe how to embed the SML classification model in an AI artifact comprising a trained and evaluated SML classification model and an additional online learning module, based on the inputs of a continuous data stream.

4.8 AI-artifact and deployment simulation

At this point, the SML classification model should be ready for deployment. However, within an industrial context, the data source is usually not stationary, especially if the data is collected over a long period of time (Baena-Garcia et al. 2006). Therefore, continual evaluation under real-world conditions is necessary. According to Peres et al. (2018), the joint exploitation of real-time and historical

data is a prerequisite to ensure the long-term prediction stability and validity of data-driven prediction models. Thus, continual evaluation is a key factor in the successful deployment of our AI artifact.

Once the SML classification model is trained and evaluated, it is constantly fed with a *data stream*. The stream provides the model with features from new customer orders, as well as with new measures of sequence deviations obtained from completed production runs, that is, new values of the target variable. The submission of a new order triggers prediction before the commencement of the production of the corresponding product. The provision of sequence deviations measures allows continual performance monitoring by comparing the predictions with the actual sequence deviation measures. After deployment, the SML classification model is subject to continuous change. Changes in the underlying data may involve drifts in the distribution of product feature frequencies, entirely new features, or learning and scaling effects of existing features due to maturity in a product lifecycle. These sudden or incremental changes are widely referred to as *concept drifts* (Schlimmer and Granger 1986; Widmer and Kubat 1996). To cope with the changes in the underlying data and to avoid manual adjustments, a concept drift or *online learning module* is incorporated. Four different operating strategies are proposed for the online learning module (Table 4).

In the *offline scenario*, the model keeps on producing predictions on the incoming feature stream, but it is not updated. Whenever new instances with target variables are available, the SML classification model could be updated. Existing work on online learning strategies describes two trivial strategies. First, *incremental learning* describes a scenario where the model is updated, based on new, unseen data, but where it retains its full knowledge of old training instances (Baena-Garcia et al. 2006). The advantage of incremental learning is that it maximizes the size of the available training set. As a downside, obsolete instances are never dismissed from the training set. Second, and in contrast, the *batch retraining* strategy dismisses past training instances and completely discards the old model (Baier et al. 2019). A new prediction model is exclusively trained on the most recent data samples provided by the data stream. This strategy ensures that the updating of the classification model is always based on the most recent data. However, the number of training samples is reduced significantly, and long-term patterns may no longer be captured.

Based on *drift detection techniques*, it is possible to fuse the advantages of both strategies. We propose combined *active decision learning* that adapts its relearning strategy to its estimation of the underlying data. Whenever a new batch of data samples is provided, the model performs a statistical test to determine the occurrence of concept drift. A batch retraining is triggered when the data distribution or pattern changes. Otherwise, the new data samples are incrementally learned, thereby extending the knowledge of the classification model. A suitable detection algorithm is the *Page-Hinckley-Test* which detects concept drifts based on the observation of performance means. A drift is assumed if a mean exceeds a threshold value λ (Mouss et al. 2004). Whenever a concept drift is detected, the batch relearning strategy is implemented. When no concept drift is detected, incremental learning is performed.

Table 4 Proposed online learning strategies for the concept drift module

Strategy	Description	Advantages	Disadvantages
No online learning	Does not update the SML classification model	Low effort	Loses validity due to concept drifts
Incremental learning (Baena-Garcia et al. 2006)	Updates the classification model by expanding the training set for each new data sample	Growing number of training instances	Never dismisses obsolete data samples
Batch retraining (Baier et al. 2019)	Discards the classification model and dismisses training data. Retrain solely on the new data batch	Always trained on up-to-date data samples	Unable to capture patterns over long periods
Active Decision Learning	Recognizes concept drifts and applies a reasonable relearning strategy	Combines strengths by applying the more beneficial strategy	Need to implement a decision function; computationally more expensive

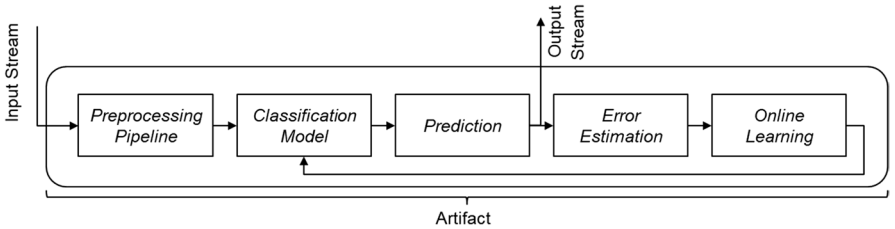

Fig. 3 AI artifact comprising the preprocessing pipeline, a trained SML classification model and an online learning module

The online learning module is integrated with our AI artifact and maintains the recency of the SML classification model. The architecture of the artifact is indicated in Fig. 3. The center of the artifact is the SML classification model, which is initially trained on historical data samples drawn from a database. During runtime, new instances are provided via the input stream. Once the new instances pass the preprocessing pipeline, the SML classification model generates a prediction. The prediction is propagated back via the output stream. Additionally, an error estimation is performed to monitor the SML classification model by comparing predicted and actual sequence deviations. The error is fed into the online learning module. According to the implemented online learning strategy, the online learning module triggers either no change to the model, or an incremental update, or dismission and entire retraining. Based on the proposed methodology, we continue by accessing the industrial application case.

5 Case study

We conducted a thorough field study to implement and evaluate the methodology within a real-world industrial production environment. We used data from a MMAL provided by a leading global automotive OEM. The daily output of the production line varies between 565 to 612 vehicles within the two-year scope of this industrial dataset. The dataset comprises more than 200,000 vehicles that were assembled between January 2018 and March 2020. The dataset consists of several different subsets, which feature the order details (body type, color, etc.), information regarding the production process (delays, position in the sequence), and other information such as the week of production. Each vehicle in the dataset is identified by a unique vehicle identifier. The subsequent discussion of our application case involves three aspects. First, the provision of further insights into the automotive production system. Second, the implementation of our AI artifact based on the specificities of the presented case. Finally, the presentation of our results and an evaluation of the SML classification model and the AI artifact.

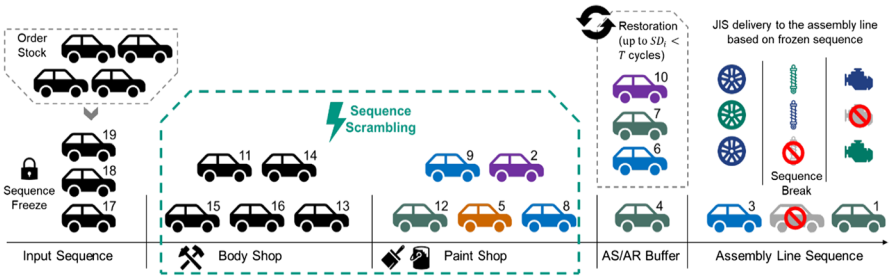

Fig. 4 Sequencing along the production line in the automotive industry based on Lehmann and Kuhn (2019)

5.1 Foundations of the automotive production system

Typically, the sequence-based value chain of automotive OEMs comprises three consecutive production segments: *body shop*, *paint shop,* and *assembly line* (Fig. 4). Based on the *order stock,* an input sequence is formed according to the requirements and restrictions of the assembly line. In the automotive industry, the initial sequencing takes place at least three to four days before the start of production (Boysen et al. 2012). The production starts with the assembly of metal sheets to form vehicle bodies in the *body shop*. Once the assembly in the body shop commences, a physical entity for each customer order exists, and each work-in-progress item is assigned to a unique customer order. The metal bodies are painted in the *paint shop* according to the purchase order. The painted bodies are stored in an AS/AR buffer. Subsequently, the painted bodies are released from the AS/AR buffer and transferred to the start of the final assembly line. At the exit of the AS/AR buffer, the quality of the achieved assembly sequence is estimated. Once released from the buffer, missing bodies can no longer be inserted into the sequence.

5.2 Business problem formulation

As unintended sequence scrambling occurs in the body and paint shop, the study focuses on these two segments. The AS/AR buffer supports physical resequencing and is regarded as a deterministic system. However, the latter is beyond the scope of this study and is a topic for future research. The assembly line itself is not a major source of deviations. The assembly work is based on an independent MMAL with minimum or no interference by other production lines. The system's resequencing capability determines the choice of our SML classification model's threshold. The AS/AR has a total capacity of 653 slots for models of the considered MMAL. The buffer is designed as an automated storage and retrieval system with random access, and it has both a decoupling and a resequencing effect. Almost 50% of the buffer space is specifically reserved for physical resequencing. The remaining capacity is designed to decouple the assembly line from pre-assembly steps. However, in general, the production system does not allow any kind of virtual resequencing

Table 5 Anonymized sample from the industrial dataset

Vehicle ID	Model code	Production family					
		~1AAB	~1ABC	~1ACD	…	~1ZZY	SD
A-22–4321	8AB3DE	~2E00	~21TA	~21RN	…	~28HH	38
A-22–5492	7G5CFG	~2E01	~21TR	~21RN	…	~28HG	-11
A-22–5826	8AB3DE	~2E01	~21TA	~21RP	…	~28HL	533

or postponement. In accordance with experts from the field of study, $T = 400 cycles$ is chosen as the threshold to binarize this study's target variable. The main KPI to measure the quality of the overall sequence is the sequence adherence SA. The objective is to reach $SA > 95\%$. The SD is determined separately for each vehicle.

The leading root causes of sequence deviations in the body shop are due to a varying effort for rework. Some models include specific features such as rear spoilers and scarce body types that require manual polishing of the body. Those bodies are released from the sequence and inserted again at a later stage for the manual polishing step. Another reason is that some special classes of customer orders inhere a manual quality check, e.g., models for public exhibitions, automotive fairs, or orders that require armored standards. In the paint shop, the sources of sequence scrambling are manifold. Certain types of paint need to be applied twice to the body. The same is required for some convertibles or two-toned bodies. Another root cause of scrambling in the automotive paint shop is the formation of blocks of bodies applied with the same color to reduce the effort of manually changing the spray cans. However, if specific colors are ordered very rarely, the block formation leads to high deviations for those orders.

5.3 Model initiation

The goal of this study is the development of an AI artifact that exploits product features to predict the deviation behavior of each individual entity in a mixed-model sequence. The prediction model is designed as a SML model in accordance with the presented methodology.

5.4 Problem exploration and data gathering

To distinguish the technical specifications of orders, a system of three-digit, alphanumeric *production-numbers* (PRN) is used. The system is organized in *production-families* (PF). Each family features a set of PRNs that is unique across PFs. A *PRN String* comprises all PRNs related to an order and provides a finite bill of materials. The PRN String is submitted to the production plant by the sales department. Apart from PRNs, the color-codes, model type, and the planned starting and desired finishing dates of production are utilized to chronologically sort the orders. Table 5 shows three exemplary instances, namely a selection of PFs from the industrial dataset, their respective PRNs, and the sequence deviation measure for these instances.

Table 6 Distribution of sequence deviations across production segments

Production segment	SD_{MIN}	SD_{MAX}	SD_{MEAN}	$SD_{0.9-quan}$	$SD_{0.95-quan}$	SD_{STD}	$\|D\|/\|P\|$
Body shop	−212	5524	0	59	116	129.47	1.161%
Paint shop	−365	4759	0	105	169	125.17	0.881%
Total	−434	5346	0	112	210	169.91	2.282%

Notes: $SD_{0.9-quan}$ *= 90% quantile,* $SD_{0.95-quan}$ *= 95% quantile, STD = standard deviation*

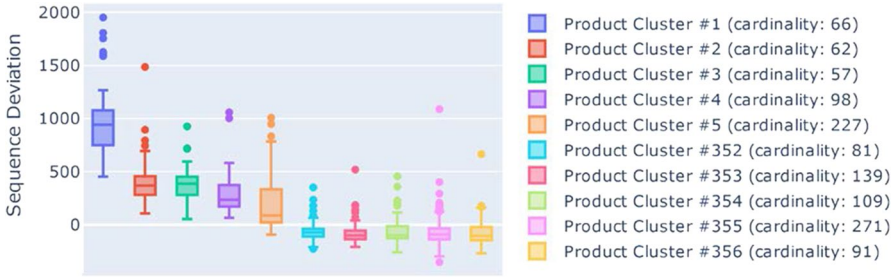

Fig. 5 Sequence deviation distribution of vehicle clusters

The available industrial dataset consists of 211,584 vehicles of 98 different models assembled between January 2018 and December 2019. Each vehicle is described by 198 PFs, whereas between two and 91 different PRNs were captured per PF. Binary PFs either describe optional features, for example, an optional Head-up Display, or exclusionary features, such as the fuel system (diesel or gasoline). More diverse PFs, for example, include an optional rear spoiler (five options) or the sales region of a vehicle (91 values). On average, 6.3 distinct values existed within each PF.

A proper analysis of the ground truth dataset shows underlying effects and particularities of the industrial dataset (Table 6). The sequence deviation measures SD_{MIN} and SD_{MAX} are similar across the individual production segments. For the paint shop, the quantiles are slightly higher. Overall, the magnitude of sequence deviations within the body and paint shop are comparable. For classification purposes, we are specifically interested in the share of vehicles that belong to the class of delayed vehicles $d \in D, where SD_d > 400 cycles$. The measures imply that 2,28% of the total vehicles have a critical deviation of $SD_i > 400 cycles$. However, the impact of the body shop is slightly more severe than that of the paint shop. The distribution of vehicles across classes P and D shows that the industrial dataset is imbalanced by a factor of approximately 44. This means that the punctual class P includes 44 times more vehicles than the delayed class D.

According to our product-oriented view, we analyze the behavior of identically configured vehicles in the industrial dataset. Two vehicles are deemed to be identical if all feature values are equivalent. We build clusters of identical vehicles with a cardinality of at least 50 instances. Overall, 356 clusters are found. We present the five vehicle clusters with the highest and the lowest SD_{MEAN} in Fig. 5.

If product-related features are the only root cause of sequence deviations, vehicles in the same cluster should behave uniformly regarding their sequence deviation. However, within clusters of equivalent vehicles, we observe non-uniform behavior. Hence, we propose two further explanations for the observed patterns. First, sequence deviations are interdependent because preceding vehicles affect the deviation of subsequent vehicles and vice versa. The dependencies arise when product-related features of neighboring vehicles have effects on other vehicles. Although all vehicles within a group ultimately have an identical set of features, the neighborhood environment in terms of preceding vehicles in the sequence may be very different. Second, the influence of process-related factors is not incorporated in the model. Nevertheless, the observed SD_{MEAN} allows a 'fuzzy' differentiation of the vehicle groups.

5.5 Performance measure

To evaluate the prediction performance, we prefer the f_β*-score* as the performance metric. The overall goal is to minimize prediction errors. However, based on the business application, we must set β to determine the trade-off between recall and precision. Based on the domain knowledge of experts from the application case, the cost of missing a delayed vehicle in the prediction is higher than the cost of a falsely predicted delayed vehicle. Consequently, we regard recall to be more important than precision. Therefore, we choose $\beta = 2$ to overweigh recall against precision.

5.6 Preprocessing pipeline

To implement the model, we use the *Python* programming language (version 3.7). Moreover, we utilize several well-recognized machine learning packages, such as *Scikit-Learn* (version 0.22) (Pedregosa et al. 2011), *TensorFlow* (version 2.0.1), and *Keras* (version 2.3.1) (Géron 2019). As discussed earlier, conducting an exhaustive grid search is computationally expensive. Hence, we opted to implement a greedy heuristic to find favorable preprocessing techniques. An overview of this is presented in Table 7. The default techniques (^{d}) and the mandatory techniques are marked separately (^{m}). The set of default techniques is used to evaluate preceding preprocessing steps. Once a favorable technique is identified, it is kept for the evaluation of the next preprocessing steps. The favorable techniques are marked separately ($^{+}$). Wherever possible, the option not to perform a preprocessing step is included. Since no algorithm selection and hyperparameter tuning have taken place at this point, we perform the preprocessing based on an XGBoost classifier with default hyperparameters. To prevent the model from overfitting, we implement sixfold cross-validation.

The mandatory steps of *instance selection* and *feature transformation* are performed first. We use filter rules to remove pre-series and scrapped vehicles from the industrial dataset. We perform a one-hot-encoding to convert the categorical features to a machine-readable format. One-hot-encoding is preferred because it does not imply any relationship between feature values within the same family. After filtering, 199,277 datapoints remained in the industrial dataset. Due

Table 7 Comparison of f_2-*scores* for the proposed preprocessing techniques. Mandatory (m), optional default (d), and favorable techniques (+) are marked separately

Preprocessing step	Strategy	f_2-*score*	Precision	Recall
Instance cleaning	Filter Rules [m]	–	–	–
Feature transformation	One-Hot-Encoding [m]	–	–	–
Feature selection	Use All Features [d]	0.358	0.661	0.321
	Manual Selection [+]	**0.388**	0.568	0.360
Frequency cut-off	No Cut-Offs [d]	0.388	0.568	0.360
	0.5% / 0%	0.393	0.574	0.364
	0% / 0.5%	0.388	0.576	0.359
	0.5% / 0.5% [+]	**0.395**	0.576	0.365
	…			
	2% / 2%	0.379	0.585	0.348
Feature selection	**No Grouping [d/+]**	**0.395**	0.576	0.365
	Automated Grouping	0.395	0.543	0.370
Sampling	None (Equal Losses)	0.305	0.860	0.263
	None (Weighted Losses)	0.320	0.118	0.557
	Random Undersampling	0.292	0.097	0.585
	Replicative Oversampling	0.317	0.117	0.555
	SMOTE Sampling [d]	0.395	0.576	0.365
	SMOTE + ENN Sampling [+]	**0.403**	0.516	0.382
	SMOTE + Tomek Sampling	0.389	0.571	0.360
	ADASYN Sampling	0.373	0.595	0.341
Feature engineering	**No Additional Features [d/+]**	**0.395**	0.576	0.365
	Neighborhood Features	0.293	0.821	0.252
	Batch Features	0.358	0.765	0.316
Instance selection	No Filter Rules [d]	0.395	0.576	0.365
	Filter Rules [+]	**0.462**	0.526	0.448

Important results are given in bold

to feature encoding, the number of columns increased to 1,249 binary feature columns. We continue with the evaluation of feature selection techniques. The *manual feature selection* is based on the distinction between those features that impact the processing in the body and paint shop and those that do not have a direct influence. We test if a subset of the features sufficiently represents the influencing factors. This subset features PRNs which describe basic measures of vehicles, such as their body shape, color, propulsion system, and quality classification, but which neglect other features that are irrelevant to the processes in the body and paint shop. A subset of 21 out of 198 PFs is identified, and its corresponding, one-hot-encoded columns are kept. We generalize the problem space by reducing the number of features. The results show that an 8.4% increase in the f_2-*score* is achieved on average. Due to the generalization of the feature space, the precision decreases while the recall increases. In proceeding, we keep to the

strategy of manual selection. *Frequency cut-offs* remove features based on frequency thresholds. A lower threshold determines how often a feature, as a minimum, must occur in the industrial dataset, whereas an upper threshold determines how often a feature, at most, could occur. We tested different cut-off levels in 0.5% intervals. A cut-off tuple of (0.5%/0.5%) for the upper and lower threshold increases the f_2-*score* by 1.8%. Both precision and recall increase. An *automated grouping* condenses linear-dependent features into new, representative parental features. Although the grouping boosts the recall, there is no positive effect on the f_2-*score*. Therefore, we exclude automated grouping from further evaluation.

To cope with the imbalance, several different *class balancing* techniques are evaluated for the training step. To balance the industrial dataset, *SMOTE + ENN* sampling turns out to yield the highest f_2-*score* with an increase of 2.0%. This corresponds with studies that confirm the supremacy of this technique (More 2016). In general, synthetic sampling algorithms outperform static sampling techniques. Interestingly, the synthetic sampling algorithms, especially if combined with *ENN* or *Tomek*, increase recall and decrease precision. According to Boardman et al. (2018), this is due to the expansion and generalization of the minority class boundaries and the shift of the classification bias toward the minority class. This is achieved by removing instances that are misclassified by their three nearest neighbors. The disadvantage of SMOTE + ENN is that this sampling requires significant computational resources. For our industrial dataset, each training iteration takes about five additional hours, which is an increase of 900%. Due to limited resources, we exclude SMOTE + ENN from the further evaluation of preprocessing steps. We continue the preprocessing evaluation, as well as the algorithm hyperparameter tuning by implementing SMOTE. However, we use SMOTE + ENN for the final evaluation of the model.

Moreover, we evaluate if *feature engineering* increases prediction performance and test if the performance increases due to the inclusion of the neighborhood features of the five preceding vehicles. Additionally, we test batch features, that is, rolling means of features for a short-term horizon of 100 cycles and a long-term horizon of 600 cycles. The shorter horizon accounts for a timespan of approximately half a shift, whereas the longer horizon captures the effects of an entire day. The horizons are set based on the knowledge of domain experts. The expansion of the feature space by these additional features does not notably improve prediction performance. Lastly, we perform another *instance selection* strategy based on filter rules. Here, the selection is optional as it aims to dismiss those vehicles from the training set that are mainly affected by process-related factors, that is, vehicles that were subject to quality checks or product audits. According to our product-oriented view, the influence of process-related factors is neglected. The vehicles that are subject to product audits and quality checks are randomly selected from the sequence and cannot be identified a priori. In total, by removing 1,903 affected vehicles from the training set, an increase of 14.6% is observed in the f_2-*score*.

Based on the evaluation results, we derive the preprocessing pipeline as the successive application of filter rules, one-hot-encoding, upper and lower frequency

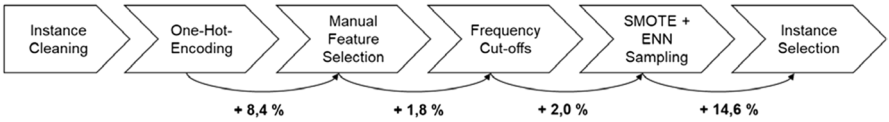

Fig. 6 Preprocessing pipeline with respective increases of the average f_2-*score*

cut-offs at 0.5%, SMOTE + ENN, and the removal of audited and quality-checked vehicles (Fig. 6).

5.7 Classification model

We pre-selected three algorithms for further evaluation: An Artificial Neural Network, an Extreme Gradient Boosting Classifier, and a Random Forest Classifier. Each algorithm is separately hyperparameter tuned. To reduce a preprocessing bias, we apply the default preprocessing steps (^{d}) for the hyperparameter search.

5.8 Model error estimation

We test and evaluate different preprocessing techniques based on a non-hyperparameter-tuned classification algorithm. We implement a grid search to determine hyperparameters for the classification algorithms. The hyperparameter grid search is performed on an Intel Xeon CPU E5-2667 v4 with 3.20 GHz, 4 processors with 8 physical kernels each, and 64 GB of RAM. A detailed summary of the hyperparameter grid search, as well as the final hyperparameter selection, is presented in Table 12 (see the Appendix). The hyperparameters are used later to evaluate and benchmark the algorithms.

Again, we emphasize the evaluation under real-world conditions. To simulate the effects of an online learning model, the industrial dataset is chronologically ordered and fed into the model instance-by-instance. This enables the monitoring of the models' performance over time and a simulation of real-time behavior. We use *Scikit-Multiflow* framework to stream data (Montiel et al. 2018). The first relevant parameter to be selected is the size of the *initial training batch I*. The initial batch is used to pretrain the SML classification model before its deployment in the AI artifact. We use one quarter of the available data to pretrain, that is, 50,000 data points. Once the stream of feature values commences, the model makes batchwise predictions. An online learning module runs in parallel to trigger relearning. The relearning cycle can be either *continual*, that is, in fixed batch sizes, or *continuous*, that is, separately for each new instance. We decided to evaluate continual relearning with a *fixed trigger* that sets off relearning after receiving C new instances of the minority class. At least one new instance per class is required to improve the model on new, unseen data. We evaluate different values for C to find a suitable parameter setting. If the minimum cycle $C = 1$ is chosen, that is, the maximum relearning frequency, *quasi-continuous* learning is applied. However, $C > 1$ is recommended to prevent

Table 8 Evaluation of different online learning strategies and relearning cycles based on average f_2-*scores*

$I = 50{,}000$	$C = 10$	$C = 100$	$C = 250$
No online learning	0.1602	0.1210	0.0910
Incremental learning	0.4177	0.3819	0.3568
Batch retraining	0.3822	0.3471	0.2726
Active Decision Learning	**0.4183**	0.3910	0.3885

Important results are given in bold

overfitting. The relearning batches differ in size whenever datapoints of the minority class are not uniformly distributed along the time horizon. If delayed vehicles occur more frequently, the relearning cycle is shortened, and vice versa. Four different strategies are implemented and evaluated to cope with concept drifts: *No Online Learning*, *Incremental Learning*, *Batch Retraining*, and *Active Decision Learning* with different values of λ for the Page-Hinckley-Test.

First, we build the SML classification model according to our prior findings on favorable preprocessing steps and the selected algorithm hyperparameter set. We train the model on the first 50,000 data points and simulate a constant stream of new vehicle features to trigger predictions according to parameter C. Afterwards, we compare the predictions with the true values and monitor the performance of the model over time. A detailed summary of the parameter search for the Page-Hinckley-Test is presented in Table 13 (see the Appendix). The best strategy is selected based on the average f_2-*scores* that we achieve (Table 8).

Generally, we observe a higher average f_2-*scores* for shorter relearning cycles. A possible explanation is the advantages of shorter reaction times on the local pattern, which outweigh the disadvantages of smaller training batches. For every evaluated relearning cycle, the active decision learning algorithm outperforms alternative strategies. Therefore, we recommend the implementation of an active decision learning strategy that selects a suitable strategy for every batch from either batch retraining or incremental learning.

6 Results and evaluation

The evaluation is divided into two parts: first, the evaluation of the offline SML classification model; second, the evaluation of the AI artifact incorporating online learning. For the evaluation, we merge our findings—separately derived for each classification algorithm—on favorable preprocessing steps and hyperparameters.

6.1 Offline SML classification model

We evaluate three distinct classification algorithms based on their mean f_2-*scores* (Table 9). The evaluation of the algorithms' performance is based on 12-fold cross-validation. Each iteration of the cross-validation provides an f_2-*score* of the model on unseen data. The minimum and maximum scores denote the performance boundaries of the classifier. Besides the mean performance, the variance is also of interest. The

Table 9 Evaluation of classification algorithms

Classification Algorithm	f_2^{MAX}	f_2^{MIN}	f_2^{MEAN}	$f_2^{Std.Dev.}$	$precision^{MEAN}$	$recall^{MEAN}$	Improve-ment
RG ($p=0.0228$)	–	–	0.0230	–	0.0228	0.0228	1892%
RG ($p=0.5$)	–	–	0.0960	–	0.0228	0.5	377%
RG ($p=1$)	–	–	0.105	–	0.0228	1	336%
Log. Regression	0.4450	0.1880	0,3414	0.1230	0.2030	0.4115	34%
ANN	0.4975	0.2629	0.3809	0.0771	0.2605	0.5244	20%
XGBoost	0.4990	0.4189	**0.4582**	0.0225	0.5058	0.4478	–
Random Forest	0.4921	0.4087	0.4554	0.0218	0.5397	0.4385	0.6%

Important results are given in bold

standard deviation of the performance measure over the cross-validation runs provides an estimation of the expected error on future data. The lower the standard deviation of a classifier, the higher the estimation of its prediction confidence. For purposes of illustration and comparability, we also provide the means of precision and recall.

Because of the novelty of the presented methodology, no benchmark model exists that predicts sequence deviations in a comparable manner. In the absence of a feasible metric or 'gold standard', also considering that humans do not perform this task, the machine learning algorithms are benchmarked against different *random guess strategies*. We assess three different *random guess classifiers* (RG): an informed RG that guesses by using the underlying class distribution ($p=0.0228$), a uniformed RG that guesses with a probability of $p=0.5$, and an RG that assigns all instances to the minority class ($p=1$). The latter performs best in maximizing the f_2-*score*. Moreover, we apply a multi-variate logistic regression as an additional baseline. To compare the different performances, we conduct an ANOVA with Tukey HSD-Post-hoc Tests. The detailed results of both tests are depicted in Tables 14 and 15 in the Appendix. The ANOVA is highly significant (p-value < 0.00001), which means there are significant differences between the classification performances of the individual models. The Tukey HSD Post-hoc test reveals significant differences between all direct comparisons, except for the comparison between XGBoost and Random Forest, which, in conclusion, can therefore be seen as equal. For simplification, we will exemplarily discuss our results in the upcoming chapters based on the performance of the XGBoost algorithm.

Looking exemplarily at the XGBoost, the improvement compared to random guessing is above 300%, and compared to the logistic regression, it is above 34%. The Random Forest Classifier performs similarly and, relatively, the ANN performs worst. The XGBoost reaches a precision of 0.5058 and a recall of 0.4478; that is, almost every second delayed vehicle is identified in advance. Moreover, the relatively low variation across cross-validation runs indicates that the SML classification model is not subject to overfitting.

Since we are interested in the sensitivity of the SML classification model, a threshold T was selected based on the systems' resequencing capability to binarize the target value for our binary classification task. We evaluate how a change

Table 10 Evaluation of online learning strategies

Online Learning Strategy	f_2^{MAX}	f_2^{MIN}	f_2^{MEAN}	$f_2^{Std.Dev.}$	Improvement
No Online Learning	0.7692	0.0	0.1602	0.1424	161%
Incremental Learning	1.0	0.0	0.4177	0.3933	0.1%
Batch Retraining	1.0	0.0	0.2654	0.3380	57%
Active decision learning	1.0	0.0	**0.4183**	0.3915	–

Important results are given in bold

of T influences the prediction performance of the current model. For $T = 300$ the XGBoost delivers an f_2-*score* of 0.4202, and for $T = 200$ an f_2-*score* of 0.3543. However, as the underlying class distribution is affected by a change of T, another set of preprocessing steps could yield better results.

6.2 Online AI-artifact

After identifying the required preprocessing steps and tuning hyperparameters, and selecting the best-performing algorithm, the deployment of our artifact was simulated to access the performance in a real-world scenario from our field study. A quarter of the available data instances is used for initial training. The remaining instances are used to simulate real-time streaming data. This evaluation of the artifact under real-world conditions is important to determine its long-term predictive ability and to estimate its future performance on unseen data. By using a portion of the available data for the simulation of a run-time environment, it is possible to estimate the artifacts' performance after deployment. Moreover, by providing boundaries for subsequent data batches, the evaluation facilitates an understanding of performance variations over time. We evaluate different strategies to cope with underlying data drifts. The relearning strategies are benchmarked against the offline SML classification model that does not incorporate any online learning. We optimize the parameter λ for a Page-Hinckley-Test and evaluate different relearning cycles. The best results are produced by a relearning cycle of $C = 10$. The evaluation of relearning strategies based on $C = 10$ is presented in Table 10. For the Page-Hinckley-Test, $\lambda = 5$ is chosen.

Each of the evaluated relearning strategies outperforms the model that is deployed without an online learning strategy. Compared to the setting without online learning, the improvement realized by the active decision learning strategy is significant. This underlines the need to continuously update a deployed machine learning model. An active learning strategy that combines the advantages of incremental learning and batch retraining performs best on the industrial dataset and outperforms offline learning by 161%. However, we observe that the model's performance highly depends on the specificity of the relearning batches, as indicated by a high standard deviation and a wide range of f_2-*scores* between zero and one. This confirms the presence of local patterns in our industrial dataset that cannot be captured by our AI artifact.

7 Conclusion

Our research focuses on making classification predictions about sequence scrambling effects on a mixed-model assembly line for in-line vehicle sequencing. At the outset, we identified a research gap concerning the leveraging of AI tools, especially in the analysis of product-related features. Whereas previous research focused on operational or process-related features, our practice-based results demonstrate that product features contribute significantly to the prediction of sequence deviations. The influence of other aspects, such as process-specific reasons, were not in the scope of the study. The subsequent results have considerable benefits for our industry partner, as revealed by several interviews. As domain experts from our case company confirm, the predictive power of the model goes beyond human capabilities, especially if applied to the high number of cars that are assembled. The industry partner considered it very useful that the model has an integrated drift-detection and re-learning pipeline, as different car types shift over time. In addition to the predictions, the development of the AI artifact revealed dependencies between product features and process weaknesses, as well as previously unknown sequence deviations. For example, a major reason for sequence scrambling in the body shop appears due to manual rework issues. The AI artifact helped us to identify a certain car type that is typically subject to rework based on insufficient tooling at the shop floor.

Our study bridges a relevant research gap by adopting a product-oriented view to construct a prediction model. While prior research mainly focused on the exploitation of process-oriented factors, we isolate product features for our analysis. Instead of making an ex-post analysis, we develop an a priori prediction model. We contribute a generalizable methodology that describes the development of an AI artifact by combining our product-oriented view and machine learning techniques with state-of-the-art online learning strategies. Our methodology provides a development framework for practitioners to design and implement a supervised machine learning classification model on a mixed-model assembly line. We also provide an overview of suitable preprocessing techniques and evaluate different classification algorithms. Our study shows that product-oriented factors play a compelling role in the prediction of sequence scrambling based on an extensive assessment of preprocessing techniques, classification algorithms, and online learning strategies.

The study provides new insights into the relationship between product variability and sequence stability. Compared to the analysis of operational or process-oriented factors as provided by (Rudolf et al. 2014; Lehmann and Kuhn 2019; Urnauer et al. 2019; Moetz et al. 2019), the predictions are comparably cheap to obtain because no additional data is needed and it is not necessary to interfere with the running processes. Moreover, no manual data processing is required. The employment of an AI artifact is possible at a relatively low cost compared to human-based analyses. The benefit of obtaining these predictions are manifold. First, the predictions can be used when considering improvements to the initial sequencing. While building the sequence, knowledge of expected delays could facilitate preventive counteraction. The input sequence can be manipulated intentionally to account for expected delays, i.e., the position of a vehicle in the input

sequence can be adjusted to reverse the negative scrambling effect predicted by the AI artifact. This could ultimately lead to an increase in sequence adherence, thus a reduction of costs. Second, to identify process weaknesses, it is possible to increase the transparency of the prediction results. The XGBoost algorithm allows the analysis of feature impacts, which can guide future process improvement projects.

By utilizing an XGBoost classification algorithm that is trained on product features, the supervised machine learning model can identify 44% of the delayed vehicles with a precision of 50%. Based on our industrial dataset, we prove that the prediction outperforms random guessing by a factor of more than 300%. The simulated deployment within an AI artifact produced an f_2-*score* of 41% in the online learning scenario under real-world conditions by using a quarter of the industrial dataset for pretraining and the remainder for continuous streaming. We show how local effects influence the performance variation of the artifact over time and estimate that a relatively short relearning cycle of $C = 10$ instances of the minority class yields the best results. Compared to an offline scenario, an active decision learning strategy, which either performs incremental relearning or batch retraining based on a Page-Hinckley-Test, ensures time-stable prediction performance and long-term validity. The best online learning strategy, according to our assessment, yields an improvement of 161% compared to the deployment of the offline artifact.

Our research helps to solve the problem of sequence deviations that is one of the most detrimental downsides of modern production systems in the area of mass-customization on mixed-model production lines. Based on existing data, production line managers can reduce the number of missing products in the assembly line by almost 50% at relatively low costs. The artifact facilitates a sustainable increase in resource efficiency and thus a reduction of costs, as the early detection of delayed vehicles helps to reduce manual resequencing work of JIS parts and facilitates a high level and smooth utilization of assembly line resources. The methodological design allows the adaptation and transference of the concept to different domains and application industries. The AI artifact, including its preprocessing pipeline, SML classification model, and online learning module, can be tailored to different production settings. However, the results are subject to review, especially to evaluate whether the performance of the AI artifact justifies its integration into active processes. A higher level of prediction performance and stability could facilitate its implementation in the industry.

The artifact is limited by the fact that, due to computing resource restrictions, we did not perform an integrated preprocessing and hyperparameter optimization. Potentially, the simultaneous adjustment of preprocessing techniques and hyperparameters yields better results. Moreover, we selected preprocessing techniques based on a greedy heuristic. An exhaustive search could reveal that another preprocessing pipeline may yield better results. Depending on some of its underlying assumptions, the model's performance could change significantly. If the system's resequencing capability is decreased, for example, due to the gains of using the predictions to improve the sequence, different thresholds T must be set for the classification algorithm. Another limitation is the restrictions on product-oriented features. The integration of a broader range of factors and (virtual) sensors (Martin et al. 2021) thus presenting a more comprehensive view,

would enhance prediction performance. The online learning shows that the continuous updating of the model significantly enhances its long-term mean validity. However, we noted a high standard deviation across relearning batches. This may be the result of the particularities of the industrial dataset. Although we evaluated different online learning strategies, the development of more advanced strategies could increase prediction performance over time and support long-term validity. Deriving specific actions from the predictions is beyond the scope of this study. We discussed the possibility of incorporating the predictions during the initial sequencing to counter expected delays. Moreover, the predictions can be analyzed to identify underlying weaknesses of the production process for vehicles with a specific set of features—possibly by introducing the transparency of the model (Vössing et al. 2019). Once the predictions are used to intentionally scramble the input sequence, the online learning module needs to take these effects into account. The validity of the AI artifact assumes that no actions are taken.

Further research is needed to integrate the proposed methodology and existing operational or process-oriented studies. An integrated approach that simultaneously acknowledges all relevant factors is required to improve the results of this study. Further steps to improve the model could include continuous predictions, that is, a process-mining methodology that updates the predictions based on the concurrent state of the production line, buffer filling level, and machine conditions. This methodology can adjust predictions when issues arise during the runtime of the production line and when factors that were not known at the start of the production process are internalized. Apart from a binary classification, the problem can also be modeled as a regression. By discriminating vehicles according to their expected degree of delay, the results of a regression model would allow even more appropriate counteractions.

In accordance with the work of Peres et al. (2019) on multistage quality control using machine learning, we underline the potential improvements of artificial intelligence models to the stability of automotive production processes. However, currently, there is no comparable study that investigates the influence of product-related features on sequence scrambling using SML models. Our results indicate that, pending future research, improvements could be achieved by combining product, operational, and process-oriented features within a single AI artifact. Moreover, the assessment of advanced online learning strategies is an important area for further research. However, the generalizability of the model is limited to mixed-model production lines with a stabilized assembly line sequence. Additional studies are required to derive specific actions from the predictions. Further studies should also consider the interdependencies that arise when actions are taken to intentionally scramble the sequence while the model is updated and making predictions. The question needs to be answered how counteractions affect the stability of the prediction model. Nonetheless, our AI artifact's current performance facilitates the introduction of substantive improvements to complex production processes on mixed-model assembly lines and generates long-term value by utilizing artificial intelligence tools.

Appendix

See Tables 11, 12, 13, 14 and 15.

Table 11 Supervised Machine Learning Report Card according to Kühl et al. (2020)

Model initiation		
Problem statement	Predict sequence scrambling in automotive production based on product-specific features	
Data gathering	Data regarding the orders from the SAP-based order-execution-system as well as production-related data from the SAP-based manufacturing-execution-system	
Data distribution	Two classes (delayed, punctual) based on a threshold of 400 sequence cycles selected for the business problem. Class distribution of 97,7% punctual and 2,3% delayed	
Sampling	No sampling	
Data quality	1,6% incomplete instances that were removed from the dataset	
Data preprocessing methods	See Table 7	
Feature engineering and vectorizing	See Table 7	
Performance estimation		
Parameter optimization	Yes	
	Search space	See Table 12
	Search algorithm	Grid search
Data split	Nested cross-validation, 6 outer folds, 12 inner folds	
Algorithm	XGBoost, Random Forest Classifier, ANN	
Sampling	Synthetic Minority Oversampling Technique with Edited Nearest Neighbors (SMOTE + ENN)	
Performance metric	F_2-score as a compromise between precision and recall with a focus on recall. Based on the domain knowledge of experts from the application case, the cost of missing a delayed vehicle in the prediction is higher than the cost of a falsely predicted delayed vehicle	
Performance evaluation	Average F_2-score performance on outer folds: 0.4582 (336–1892% better than baseline)	
Code Availability	https://github.com/AI-in-Sequence-Scrambling/Sequence-Scrambling-Project	

Table 12 Classification algorithms hyperparameter grid search

Algorithm	Parameter	Search grid	Selected value
ANN	Hidden layer	{4, 8}	8
	Layer types	{'Dense' + 'Dropout'}	'Dense' + 'Dropout'
	Input activation	{'relu'}	'relu'
	Input neurons	{50, 250, 500}	50
	Dropout rate	{0.3, 0.5}	0.5
	Hidden neurons	{50, 250, 500}	50
	Hidden activation	{'relu', 'sigmoid'}	'sigmoid'
	Output activation	{'sigmoid'}	'sigmoid'
	Output neurons	{1}	1
	Optimizer	{'adam'}	'adam'
	Epochs	{4, 8}	4
	Batch size	{50, 250, 500}	50
XGBoost	Booster	{'gbtree', 'gblinear', 'dart'}	'dart'
	Eta	{0.1, 0.2, 0.3, 0.4, 0.5}	0.2
	Gamma	{0, 0.1, 0.2, 1}	0.1
	Max. depth	{3, 6, 12}	6
	Lambda	{0.,5 1, 2, 6, 12}	2
	Alpha	{0, 0.5, 1, 4, 8}	1
	Tree method	{'auto'}	'auto'
	Parallel trees	{1, 3, 6, 12}	6
Random forest	N estimators	{100, 250, 500, 1000, 5000}	1000
	Criterion	{'gini', 'entropy'}	'entropy'
	Min sample split	{2, 5, 10, 25, 100}	10

Table 13 Online learning parameter search

$I = 50{,}000$	$C = 10$	$C = 100$	$C = 500$
No online Learning	0.0619	0.1210	0.0910
Incremental Learning	0.4137	0.3819	0.3568
Batch Retraining	0.2654	0.3471	0.2726
Active Decision Learning ($\lambda = 0.05$)	0.4105	0.3711	0.3848
Active Decision Learning ($\lambda = 0.1$)	0.4190	0.3638	0.3885
Active Decision Learning ($\lambda = 0.25$)	0.4084	0.3828	0.3565
Active Decision Learning ($\lambda = 1$)	0.4095	0.3861	0.3571
Active Decision Learning ($\lambda = 2$)	0.4152	0.3891	0.3784
Active Decision Learning ($\lambda = 5$)	0.4183	0.3892	0.3404
Active Decision Learning ($\lambda = 10$)	0.4142	0.3910	0.3854

Table 14 ANOVA between comparison of model performances

Sum of Squares	DoF	Variance	F-metric	*P*-value
0.4966	3	0.165549	30.025	5.12e−16

Table 15 Tukey–Kramer HSD Post-Hoc Tests of model performances

Groups	Means	Difference	Lower	Upper	*P*-value	Signif
ANN vs LogisticRegression	[0.3809, 0.3414]	−0.0395	−0.078	−0.001	0.04178	*
ANN vs RandomForest	[0.3809, 0.4554]	0.0745	0.036	0.113	0.00001	*****
ANN vs XGBoost	[0.3809, 0.4582]	0.0773	0.0388	0.1158	0.00000	*****
LogisticRegression vs RandomForest	[0.3414, 0.4554]	0.114	0.0755	0.1525	0.00000	*****
LogisticRegression vs XGBoost	[0.3414, 0.4582]	0.1168	0.0783	0.1553	0.00000	*****
RandomForest vs XGBoost	[0.4554, 0.4582]	0.0028	−0.0357	0.0413	0.99762	NS

*$p < 0.01$, ***** $p < 0.0001$, n.s. = not significant

Acknowledgements This paper and the research behind would not have been possible without the support of our industry partners. We would especially like to thank Vanessa Markert, Michael Minkwitz and Gerhard Rudolf for their openness and support. The opportunity to work with their input was fundamental to the success of this article. We thank them very much for their thoughtfulness and their patient answers to our questions. We also thank our colleagues at the KSRI for sharing their expertise with us. Their knowledge helped to improve the work at hand and saved us from many errors. Special thanks go to Gerhard Satzger, who, with his many years of industry and research experience, has always stood by our side with help and advice and firmly believed in the success of the work. Moreover, we thank our peer-reviewers and editors for their insightful comments that allowed us to improve our work significantly. We highly appreciate their efforts and time to raise the level of our work.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

Bach M, Werner A, Żywiec J, Pluskiewicz W (2017) The study of under- and over-sampling methods' utility in analysis of highly imbalanced data on osteoporosis. Inf Sci 384:174–190. https://doi.org/10.1016/j.ins.2016.09.038

Baena-Garcia M, del Campo Ávila J, Bifet A, Gavald R, Morales-Bueno R (2006) Early Drift Detection Method. In: Joao Gama, Ralf Klinkenberg, Jesus S. Aguilar-Ruiz (eds) The Fourth International Workshop on knowledge discovery from data streams. 17th European conference on machine learning and the 10th European Conference on principles and practice of knowledge discovery in databases, Berlin, 18–2 September 2006.

Baier Lucas, Kühl Niklas, Satzger Gerhard (2019) How to cope with change? Preserving Validity of Predictive Services over Time. In: Tung Bui (ed) Proceedings of the 52nd Hawaii International Conference on System Sciences. Hawaii International Conference on System Sciences, 8–11 January 2019.

Barocas Solon, Hardt Moritz, Narayanan Arvind Hari (2018) Fairness and machine learning: limitations and opportunities. Available online at https://www.semanticscholar.org/paper/Fairness-and-Machine-Learning-Limitations-and-Barocas-Hardt/bae7f0b3448a3eac77886f2a683c0cf9256bb8bf.

Batista GEAPA, Prati RC, Monard MC (2004) A study of the behavior of several methods for balancing machine learning training data. Special Interest Group on Knowledge Discovery Data Exploration Newsletter 6(1):20–29. https://doi.org/10.1145/1007730.1007735

Bergstra James S, Bardenet Rémi, Bengio Yoshua, Kégl Balázs (2011) Algorithms for Hyper-Parameter Optimization. In: Shawe-Taylor J, Zemel RS, Bartlett PL, Pereira F, Weinberger KQ (eds) Advances in Neural Information Processing Systems 24 (NIPS 2011). Advances in Neural Information Processing Systems 24. Granada, pp 2546–2554. Available online at http://papers.nips.cc/paper/4443-algorithms-for-hyper-parameter-optimization.pdf.

Blagus R, Lusa L (2013) SMOTE for high-dimensional class-imbalanced data. BMC Bioinformatics 14:106. https://doi.org/10.1186/1471-2105-14-106

Boardman Jonathan, Biron Kyle, Rimbey Ryan (2018) Mitigating the effects of class imbalance using SMOTE and Tomek link undersampling in SAS. Kennesaw State University. Available online at https://pdfs.semanticscholar.org/bf3e/68c3e9cfe50b75897d6e6296c45f5bd30f82.pdf.

Boysen N, Golle U, Rothlauf F (2011) The car resequencing problem with pull-off tables. Bus Res 4(2):276–292. https://doi.org/10.1007/BF03342757

Boysen N, Scholl A, Wopperer N (2012) Resequencing of mixed-model assembly lines: survey and research agenda. Eur J Oper Res 216(3):594–604. https://doi.org/10.1016/j.ejor.2011.08.009

Boysen Nils, Fliedner Malte, Scholl Armin (2009) Sequencing mixed-model assembly lines: Survey, classification and model critique. Survey, classification and model critique. Euro J Oper Res192(2):349–373. https://doi.org/10.1016/j.ejor.2007.09.013.

Breiman L (2001) Random forests. Mach Learn 45(1):5–32. https://doi.org/10.1023/A:1010933404324

Brodley Carla, Smyth Padhraic (1995) The process of applying machine learning algorithms. In: Armand Prieditis, Stuart J Russell (eds) Proceedings of the Twelfth International Conference on Machine Learning. Tahoe City, USA, 9–12 July 1995.

Cawley G, Talbot N (2010) On over-fitting in model selection and subsequent selection bias in performance evaluation. J Mach Learn Res 11:2079–2107

Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) SMOTE: synthetic minority over-sampling technique. J Artificial Intell Res 16:321–357. https://doi.org/10.1613/jair.953

Chen Tianqi, Guestrin Carlos (2016) XGBoost: A scalable tree boosting system. In: Balaji Krishnapuram, Mohak Shah, Alex Smola, Charu Aggarwal, Dou Shen, Rajeev Rastogi (eds) KDD 2016. Proceedings of the 22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 22nd ACM SIGKDD International Conference. San Francisco, California, USA, 13–17 August 2016. Association for Computing Machinery, pp 785–794.

Choudhary AK, Harding JA, Tiwari MK (2009) Data mining in manufacturing: A review based on the kind of knowledge. J Intell Manuf 20(5):501–521. https://doi.org/10.1007/s10845-008-0145-x

Ding F-Y, Sun H (2004) Sequence alteration and restoration related to sequenced parts delivery on an automobile mixed-model assembly line with multiple departments. Int J Prod Res 42(8):1525–1543. https://doi.org/10.1080/00207540310001645156

Dörmer J, Günther H-O, Gujjula R (2015) Master production scheduling and sequencing at mixed-model assembly lines in the automotive industry. Flex Serv Manuf J 27(1):1–29. https://doi.org/10.1007/s10696-013-9173-8

Drexl A, Kimms A (2001) Sequencing JIT mixed-model assembly lines under station-load and part-usage constraints. Manage Sci 47(3):480–491. https://doi.org/10.1287/mnsc.47.3.480.9777

EMEA (2019a) Automobile assembly and engine production plants in Europe | ACEA - European Automobile Manufacturers' Association. Available online at https://www.acea.be/statistics/article/automobile-assembly-engine-production-plants-in-europe, updated on 3/18/2020, checked on 3/18/2020.

EMEA (2019b) EU passenger car production | ACEA - European Automobile Manufacturers' Association. Available online at https://www.acea.be/statistics/article/eu-passenger-car-production, updated on 3/18/2020, checked on 3/18/2020.

Evermann J, Rehse J-R, Fettke P (2017) Predicting process behaviour using deep learning. Decis Support Syst 100:129–140. https://doi.org/10.1016/j.dss.2017.04.003

Fettermann DC, Freitas FC (2017) Automobile variety in emerging countries: a comparative study between Brazil and USA. Eng J 21(4):325–338. https://doi.org/10.4186/ej.2017.21.4.325

Fournier X, Agard B (2007) Improvement of earliness and lateness by postponement on an automotive production line. Int J Flex Manuf Syst 19(2):107–121. https://doi.org/10.1007/s10696-007-9022-8

Franz C, Hällgren E, Caap Koberstein Achim (2014) Resequencing orders on mixed-model assembly lines: Heuristic approaches to minimise the number of overload situations. Int J Prod Res 52(19):5823–5840. https://doi.org/10.1080/00207543.2014.918293

Franz C, Koberstein A, Suhl L (2015) Dynamic resequencing at mixed-model assembly lines. Int J Prod Res 53(11):3433–3447. https://doi.org/10.1080/00207543.2014.993046

García S, Ramírez-Gallego S, Luengo J, Benítez J Manuel, Herrera Francisco (2016) Big data preprocessing: methods and prospects. Big Data Analytics 1(1):1–22. https://doi.org/10.1186/s41044-016-0014-0

Géron Aurélien (2019) Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow. Concepts, tools, and techniques to build intelligent systems, 2nd ednn.

Giard V, Jeunet J (2010) Optimal sequencing of mixed models with sequence-dependent setups and utility workers on an assembly line. Int J Prod Econ 123(2):290–300. https://doi.org/10.1016/j.ijpe.2009.09.001

Grinninger Jürgen (2012):Schlanke Produktionssteuerung zur Stabilisierung von Auftragsfolgen in der Automobilproduktion. Dissertation. Technischen Universität München, Munich. Lehrstuhl für Fördertechnik Materialfluss Logistik.

Gujjula Rico, Günther Hans-Otto (2009) Rescheduling blocked workpieces at mixed-model assembly lines with Just-In-Sequence supply. Department of Production Management, TU Berlin, Germany. Available online at https://www.academia.edu/29265570/Rescheduling_blocked_Workpieces_at_Mixed-Model_Assembly_Lines_with_Just-In-Sequence_supply.

Gunay E Elcin, Kula Ufuk (2016) A stochastic programming model for resequencing buffer content optimisation in mixed-model assembly lines. Int J Prod Res 55(10):2897–2912. https://doi.org/10.1080/00207543.2016.1227101

Gunay E Elcin, Kula Ufuk (2018) A two-stage stochastic rule-based model to determine pre-assembly buffer content. J Industrial Eng Int 14(4):655–663. https://doi.org/10.1007/s40092-017-0252-4

Günther Manuel Till (2017) Produktionssteuerung nach dem Perlenketten-Prinzip am Beispiel der Automobilfertigung. Dissertation. Friedrich-Schiller-Universität Jena. Wirtschaftswissenschaftliche Fakultät.

Gusikhin O, Caprihan R, Stecke KE (2007) Least in-sequence probability heuristic for mixed-volume production lines. Int J Prod Res 46(3):647–673. https://doi.org/10.1080/00207540600824300

He H, Garcia EA (2009) Learning from imbalanced data. IEEE Trans Knowl Data Eng 21(9):1263–1284. https://doi.org/10.1109/TKDE.2008.239

Hirt Robin, Kühl Niklas, Satzger Gerhard (2017) An end-to-end process model for supervised machine learning classification: From problem to deployment in information systems. Karlsruhe Service Research Institute (KSRI), Karlsruhe Institute of Technology (KIT). Karlsruhe, Germany.

Houy Constantin, Fettke Peter, Loos Peter, van der Aalst, Wil MP, Krogsti, John (2010) BPM-in-the-large – Towards a higher level of abstraction in business process management. In: Marijn Janssen, Michael Rosemann (eds) E-Government, E-Services and Global Processes. 21st Joint International Federation for Information Processing TC 6 and TC 8 International Conferences. Brisbane, 20 - 23 September 2010. International Federation for Information Processing (IFIP Advances in Information and Communication Technology, 334), pp 233–244.

Inman RR (2003) ASRS sizing for recreating automotive assembly sequences. Int J Prod Res 41(5):847–863. https://doi.org/10.1080/0020754031000069599

Inman RR, Schmeling DM (2003) Algorithm for agile assembling-to-order in the automotive industry. Int J Prod Res 41(16):3831–3848. https://doi.org/10.1080/00207540310001595792

Jin Chun, Vogl Alexander, Wortmann Dirk, Leng Jinling (2019) A hybrid simulation model for optimal color-batching resequencing in paint shop. In: Umut Durak (ed) SummerSim '19: Proceedings of the 2019 Summer Simulation Conference. 2019 Summer Simulation Conference. Berlin, 22–24 July 2019. Society for Computer Simulation International.

Kern W, Lämmermann H, Bauernhansl T (2017) An integrated logistics concept for a modular assembly system. Proc Manufact 11:957–964. https://doi.org/10.1016/j.promfg.2017.07.200

Klug Florian (2017) Das Perlenkettenprinzip der stabilen Auftragsfolge in der Automobillogistik. In: Ingrid Göpfert, David Braun, Matthias Schulz (eds) Automobillogistik. Stand und Zukunftstrends. Wiesbaden: Springer Fachmedien Wiesbaden, pp. 137–160.

Kotsiantis SB, Kanellopoulos D, Pintelas PE (2006) Data preprocessing for supervised leaning. Int J Comput Sci 1(1):1306–4428

Kotsiantis SB (2007) Supervised machine learning: A review of classification techniques. In: Ilias G. Maglogiannis (ed):Emerging artificial intelligence applications in computer engineering. Real word AI systems with applications in eHealth, HCI, information retrieval and pervasive technologies. Amsterdam, Washington, DC: IOS Press (Frontiers in artificial intelligence and applications, v. 160).

Kühl Niklas, Goutier Marc, Hirt Robin, Satzger Gerhard (2019) Machine learning in artificial intelligence: towards a common understanding. In: Tung Bui (ed) Proceedings of the 52nd Hawaii International Conference on System Sciences. Hawaii International Conference on System Sciences: Hawaii International Conference on System Sciences (Proceedings of the Annual Hawaii International Conference on System Sciences).

Kühl Niklas, Hirt Robin, Baier Lucas, Schmitz Björn, Satzger Gerhard (2020) How to conduct rigorous supervised machine learning in information systems research: The Supervised Machine Learning Reportcard (in press). https://doi.org/10.5445/IR/1000124438.

Kurgan LA, Musilek P (2006) A survey of knowledge discovery and data mining process models. Knowledge Eng Rev 21(1):1–24. https://doi.org/10.1017/S0269888906000737

Lahmar M, Benjaafar S (2007) Sequencing with limited flexibility. IIE Trans 39(10):937–955. https://doi.org/10.1080/07408170701416665

Lehmann M, Kuhn H (2019) Modeling and analyzing sequence stability in flexible automotive production systems. Flex Serv Manuf J 192:349. https://doi.org/10.1007/s10696-019-09334-x

Martin D, Kühl N, Satzger G (2021) Virtual Sensors. Bus Inf Syst Eng 63(3):315–323

Mayrhofer W, März L, Sihn W (2011) Planning assistance for pearl chain forecasts and personnel assignment planning of sequenced assembly lines. Manufacturing Technol 60(1):481–484. https://doi.org/10.1016/j.cirp.2011.03.044

Meißner S, Grinninger J, Kammermeier F (2008) Stabilisierung des Auftragsabwicklungsprozesses durch flexible Auftragszuordnung. Zeitschrift Für Wirtschaftlichen Fabrikbetrieb 103(12):893–897. https://doi.org/10.3139/104.101379

Meissner S (2010) Controlling Just-in-Sequence Flow-Production in *Logistics Research* 2(1):45–53. https://doi.org/10.1007/s12159-010-0026-5

Moetz A, Stylos-Duesmann P, Otto B (2019) Schedule instability in automotive production networks: the development of a network-oriented resequencing method. IFAC-PapersOnLine 52(13):2810–2815. https://doi.org/10.1016/j.ifacol.2019.11.634

Montiel Jacob, Read Jesse, Bifet Albert, Abdessalem Talel (2018) Scikit-Multiflow: A multi-output streaming framework. J Mach Learn Res 19 (72):1–5. Available online at http://jmlr.org/papers/volume19/18-251/18-251.pdf.

More Ajinkya (2016) Survey of resampling techniques for improving classification performance in unbalanced datasets. Available online at https://arxiv.org/pdf/1608.06048.pdf.

Morrison KR (1991) Animation—a new dimension in computer simulation of automotive assembly processes. Comput Ind Eng 21(1–4):547–551. https://doi.org/10.1016/0360-8352(91)90150-5

Mouss Hayet, Mouss Djamel M, Mouss Nadia K, Sefouhi Linda (2004) Test of Page-Hinckley, an approach for fault detection in an agro-alimentary production system. In: Proceedings of the 5th Asian Control Conference. Melbourne, Australia, 20–23 July 2004. 3 volumes. Available online at https://www.semanticscholar.org/paper/4ca7fa5e1cdae6557397a934f8c7521769b81427.

Opitz D, Maclin R (1999) Popular ensemble methods: an empirical study. J Artif Int Res 11(1):169–198

Pedregosa Fabian, Varoquaux Gaël, Gramfort Alexandre, Michel Vincent, Thirion Bertrand, Grisel Olivier et al (2011) Scikit-learn: Machine learning in Python. J Mach Learn Res 12:2825–2830. Available online at http://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdfUR - http://www.jmlr.org/papers/v12/pedregosa11a.

Peres R Silva, Barata Jose, Leitao Paulo, Garcia Gisela (2019) Multistage quality control using machine learning in the automotive industry. IEEE Access 7:79908–79916. https://doi.org/10.1109/ACCESS.2019.2923405

Peres R Silva, Dionisio Rocha Andre, Leitao Paulo, Barata Jose (2018) IDARTS—towards intelligent data analysis and real-time supervision for industry 4.0. Comput Ind 101:138–146. https://doi.org/10.1016/j.compind.2018.07.004

Polato M, Sperduti A, Burattin A, de Leoni M (2018) Time and activity sequence prediction of business process instances. Computing 100(9):1005–1031. https://doi.org/10.1007/s00607-018-0593-x

Powers D (2011) Evaluation: from precision, recall and F-measure to ROC, informedness, markedness & correlation. J Mach Learn Technol 2:37–63

Reda Mariam Moustafa, Nassef Mohammad, Salah Akram (2019) Categorization of factors affecting classification algorithms selection. Int J Data Mining Knowledge Management Process 9(4). Available online at https://pdfs.semanticscholar.org/5f22/5d6594e21d7feeeb5973ecb3f6a0b7588024.pdf.

Rudolf G, Noyan N, Giard V (2014) Modeling sequence scrambling and related phenomena in mixed-model production lines. Eur J Oper Res 237(1):177–195. https://doi.org/10.1016/j.ejor.2014.02.041

Saif U, Guan Z, Zhang Li, Zhang F, Wang B, Mirza J (2019) Multi-objective artificial bee colony algorithm for order oriented simultaneous sequencing and balancing of multi-mixed model assembly line. J Intell Manuf 30(3):1195–1220. https://doi.org/10.1007/s10845-017-1316-4

Schlimmer JC, Granger RH (1986) Incremental learning from noisy data. Mach Learn 1(3):317–354. https://doi.org/10.1007/BF00116895

Son Sook Young, Yahya Bernardo Nurgroho, Song Minseok, Choi Sangsu, Sung Nakyun (2014) Process mining for manufacturing process analysis: a case study. In: Chun Ouyang, Jae-Yoon Jung (eds) Proceedings of the Second Asia Pacific Conference on Business Process Management. Asia Pacific Conference on Business Process Management. Brisbane, 3–4 July 2014. Available online at https://www.researchgate.net/profile/Sangsu_Choi/publication/271910986_Process_Mining_for_Manufacturing_Process_Analysis_A_case_Study/links/565dfdc408ae4988a7bd2915/Process-Mining-for-Manufacturing-Process-Analysis-A-case-Study.pdf.

Stäblein T, Holweg M, Miemczyk J (2011) Theoretical versus actual product variety: how much customisation do customers really demand? Int J Oper Prod Manag 31(3):350–370. https://doi.org/10.1108/01443571111111955

Swaminathan, Jayashankar M, Nitsch Thomas R (2007) Managing product variety in automobile assembly: the importance of the sequencing point. Interfaces 37(4):324–333. Available online at https://pdfs.semanticscholar.org/7642/4cdb85d821f017608bc72d6b9f2cb295c5e1.pdf.

Tax Niek, Verenich Ilya, La Rosa Marcello, Dumas Marlon (2017) Predictive business process monitoring with LSTM neural networks. In: Eric Dubois, Klaus Pohl (eds) Advanced information systems engineering. Proceedings of the 29th International Conference, vol. 10253. 29th International Conference on Advanced Information Systems Engineering. Essen, Germany, 12–16 June 2017. International Conference on Advanced Information Systems Engineering; CAiSE: Springer (10253), pp 477–492. Available online at http://arxiv.org/pdf/1612.02130v2.

Teinemaa Irene, Dumas Marlon, La Rosa Marcello, Maggi Fabrizio Maria (2017) Outcome-oriented predictive process monitoring: Review and benchmark. ACM Transactions on Knowledge Discovery from Data (1).

Tolles J, Meurer WJ (2016) Logistic regression: relating patient characteristics to outcomes. JAMA 316(5):533–534. https://doi.org/10.1001/jama.2016.7653

Urnauer Christian, Bosch Eva, Metternich Joachim (2019) Simulation-based optimization of sequencing buffer allocation in automated storage and retrieval systems for automobile production. In: Proceedings of the 2019 Winter Simulation Conference. 2019 Winter Simulation Conference (WSC). National Harbor, MD, USA, 8–11 December 2019. IEEE, pp 1602–1611.

Varma S, Simon R (2006) Bias in error estimation when using cross-validation for model selection. BMC Bioinformatics 7:91. https://doi.org/10.1186/1471-2105-7-91

Vössing Michael, Potthoff Felix, Kühl Niklas, Satzger Gerhard (2019) Designing useful transparency to improve process performance—evidence from an automated production line. In: ECIS 2019 proceedings of 27th European Conference on Information Systems (ECIS), Stockholm & Uppsala, Sweden, June 8–14, 2019. Research Papers: Association for Information Systems (AIS).

Weyer M, Spath D (2009) Das Produktionssteuerungskonzept „Perlenkette“. Zeitschrift Für Wirtschaftlichen Fabrikbetrieb 104(12):1126–1130. https://doi.org/10.3139/104.110224

Widmer G, Kubat M (1996) Learning in the presence of concept drift and hidden contexts. Mach Learn 23(1):69–101. https://doi.org/10.1023/A:1018046501280

Wuest T, Irgens C, Thoben K-D (2014) An approach to monitoring quality in manufacturing using supervised machine learning on product state data. J Intell Manuf 25(5):1167–1180. https://doi.org/10.1007/s10845-013-0761-y

Zhang W, Gen M (2011) An efficient multiobjective genetic algorithm for mixed-model assembly line balancing problem considering demand ratio-based cycle time. J Intell Manuf 22(3):367–378. https://doi.org/10.1007/s10845-009-0295-5

Maximilian Stauder holds a degree in Industrial Engineering and Management from the Karlsruhe Institute of Technology (KIT). During his studies, he extensively worked on diverse types of analytical models, including mixed-integer-programming problems and network flow models. His focus areas are online planning of learning models and the evaluation of their performance. After he successfully defended his thesis at Karlsruhe Service Research Institute (KSRI) in 2020, he graduated (with distinction) as Master of Science. Since 2020 he is working as a Data Scientist and Artificial Intelligence Consultant at Schwarz Group. Maximilian focuses on applications of machine learning (ML) in production and industry. Mainly he is working with IoT Data for process automation and predictive maintenance. In addition, Maximilian is creating Computer Vision (CV) solutions based on machine vision systems that allow automatic quality control checkpoints in the food and beverage industry.

Niklas Kühl is head of the Applied AI in Services Lab at the Karlsruhe Service Research Institute (KSRI) and Institute of Information Systems and Marketing (IISM) at the Karlsruhe Institute of Technology (KIT). He is also working as a Managing Consultant Data Science at IBM. Niklas has been working on machine learning (ML) and artificial intelligence (AI) in different domains since 2014. Niklas' general research focuses on expanding theory by design knowledge for Artificial Intelligence (AI) solutions that enable the responsible and effective use of AI in productive industrial systems. In particular, he engages in AI solutions that turn IoT data into smart services, in enabling AI in the collaboration of different parties within value chains, and in creating AI solutions that are robust across their lifecycle. Niklas is internationally collaborating with the HEC Paris, the Wharton School of UPenn, and the MIT-IBM Watson AI Lab. He has published his work on the utilization of machine learning in journals like Business & Information Systems Engineering, Electronic Markets, Journal of Service Management, Journal of Cleaner Productions, as well as conferences like ICIS, ECIS, HICSS, IEEE CBI, ECML-PKDD, and ACM COMPASS.

Authors and Affiliations

Maximilian Stauder[1] · **Niklas Kühl**[1]

[1] Karlsruhe Service Research Institute (KSRI), Karlsruhe Institute of Technology (KIT), Kollegiengebäude am Kronenplatz (Geb. 05.20), Kaiserstraße 89, 76133 Karlsruhe, Germany