Lehrstuhl Angewandte Informatik IV
Datenbanken und Informationssysteme
Prof. Dr.-Ing. Stefan Jablonski

Institut für Angewandte Informatik
Fakultät für Mathematik, Physik und Informatik
Universität Bayreuth

Master Thesis

Andreas Braun

28.03.2019
Version: Final

Universität Bayreuth

Fakultät Mathematik, Physik, Informatik
Institut für Informatik
Lehrstuhl für Angewandte Informatik IV

Master Thesis

Data Mining in Industrial Processes: Evaluation of different machine learning models for product quality prediction

Data Mining in industriellen Prozessen: Bewertung verschiedener Machine-Learning-Modelle zur Vorhersage der Produktqualität

Andreas Braun

1. Reviewer Prof. Dr.-Ing. Stefan Jablonski
Fakultät Mathematik, Physik, Informatik
Universität Bayreuth

2. Reviewer Dr. Stefan Schönig
Fakultät Mathematik, Physik, Informatik
Universität Bayreuth

Supervisors Prof. Dr.-Ing. Stefan Jablonski and Dr. Stefan Schönig

28.03.2019

Andreas Braun
Master Thesis
Data Mining in Industrial Processes: Evaluation of different machine learning models for product quality prediction, 28.03.2019
Reviewers: Prof. Dr.-Ing. Stefan Jablonski and Dr. Stefan Schönig
Supervisors: Prof. Dr.-Ing. Stefan Jablonski and Dr. Stefan Schönig

Universität Bayreuth
Lehrstuhl für Angewandte Informatik IV
Institut für Informatik
Fakultät Mathematik, Physik, Informatik
Universitätsstrasse 30
95447 Bayreuth
Germany

Abstract

For industrial companies it is increasingly important to extract knowledge from the available data about their manufacturing processes. An overview about the relevant field of time series data mining is given. The application goal is to further improve the quality control for a plastic profile extrusion process by using machine learning which assists in determining the product quality. The product quality is inherently connected to the production line and therefore the quality of an article is determined by classifying the state of the line into error time and production time. The executed data mining steps include the extraction of a raw dataset from the company's data lake, intensive data preprocessing and the selection of an appropriate algorithm. Five different machine learning algorithms with an optional initial dimensionality reduction step are explored and evaluated on precision and recall. The standalone random forest algorithm type works the most precise and has a reasonable recall for both considered data partitioning approaches, a state based and time series based one. Hence, it can be recommended as the best out of the five initially considered algorithm types. Even though the used strategy is specific to a representative article and the production line, it can analogously be adapted to other articles as well.

Zusammenfassung

Für Industrieunternehmen wird es immer wichtiger, Wissen aus den verfügbaren Daten über ihre Fertigungsprozesse zu extrahieren. Hierzu wird ein Überblick über das relevante Feld des Time-Series-Data-Mining gegeben. Das Ziel der Anwendung besteht darin, die Qualitätskontrolle für einen Kunststoffprofilextrusionsprozess durch maschinelles Lernen (Bestimmung der Produktqualität) weiter zu verbessern. Die Produktqualität ist inhärent mit der Produktionslinie verbunden, und daher wird die Qualität eines Artikels durch Klassifizierung des Zustandes der Linie in Fehlerzeit und Produktionszeit bestimmt. Zu den ausgeführten Data-Mining-Schritten gehören das Extrahieren eines Rohdatensatzes aus dem Data Lake des Unternehmens, eine intensive Datenvorverarbeitung und die Auswahl eines geeigneten Algorithmus. Fünf verschiedene Algorithmen zum maschinellen Lernen mit einem optionalen anfänglichen Dimensionsreduzierungsschritt werden auf Genauigkeit und Trefferquote untersucht und bewertet. Der Random Forest Algorithmus arbeitet am genauesten mit einer angemessenen Trefferquote, für die beiden betrachteten Datenpartitionierungsansätze, einem Zustand-basierten und einem Time-Series-basierten. Der Random Forest Algorithmus kann daher als der Beste von den fünf betrachteten Algorithmentypen empfohlen werden. Obwohl die verwendete Strategie spezifisch für einen ausgewählten Artikel und die Produktionslinie ist, kann diese analog auch auf andere Artikel adaptiert werden.

Acknowledgement

First of all, I would like to thank Stefan Jablonski who made it possible to write this thesis with an external partner and for all the good advice throughout the meetings accompanying this thesis. Secondly I would like to thank the people at Rehau AG I worked together with, Michael Spangenberg, Sandra Romeis and Markus Thaeter, who advised me in scientific, subject-specific and data understanding related questions.

Furthermore, I thank my family for their support and especially my parents for helping me become the person I am today.

Contents

Introduction 1

In the last several years large amounts of data have been stored in the process industry. At first they have mostly been used for technical checks or process log requirements [Ge+17]. Nowadays it is increasingly important to extract knowledge of the available data of a company to gain a competitive advantage. In industrial companies the manufacturing processes play a crucial role, as they directly add value to the products and determine the product quality [Wue15].

The quality of the final product is essential as customers expect products free from error and defect. Manufacturers therefore apply in-house quality control mechanisms to ensure that only a minimal amount of faulty products leave the factory. To further minimize shipment of low quality products, an ongoing development of quality control takes place. One approach in this continuous improvement process is to utilise the techniques found in data mining. In manufacturing, data mining allows quality improvements of large and complex processes [Wue+16] that are challenging to analyse by conventional methods. Thereby large amounts of data with high dimensionality (>1000) can be analysed effectively [Wue+16].

In modern production lines it is common that various machine parameters are monitored during production. The measured parameters along the production process up until the final product allow to draw inferences about the overall quality. Nevertheless it is challenging to find the best performing algorithm as each problem is different in its own way and "the best fitting algorithm has to be found in testing various ones in a realistic environment"[Wue+16].

The core question of the internal quality control is whether the quality of the final product is good enough and thus if the product can be sold. The goal is to further improve the quality control in a plastic profile extrusion process by using data mining methods to assist in determining the product quality. The new approach must work precisely to keep the amount of falsely detected bad quality products at a minimum.

The existing implemented quality control mechanisms provide quality information for the analysed manufacturing process. Labels for poor and good quality products are available and so classification algorithms are applicable. The best fitting algorithm for the data at hand is found by exploring five chosen algorithms using two different settings. One makes use of state based partitioning and the other uses the time series based one. The best performing algorithms are inferred and lastly two algorithms are combined to see if further improvement can be achieved.

Objective and constraints in the thesis

The first goal of this work is to provide an overview on how to handle and process manufacturing data in the context of time series data mining. The second goal is to find a suitable algorithm to automatically determine the resulting article quality during production, while its purpose is to complement the existing quality control.

The data mining process during this thesis begins with the initial documentation and extraction of semi-structured data from a data lake and therefore the time consuming steps extract, transform and load (ETL) are executed. Furthermore, this thesis does not go into depth for each analysis step as an detailed investigation involves an extensive comparison of the available methods, which is not the focus here. Furthermore, quality determination parameters are specific to the article and to the production line it is produced on and so one representative article is chosen to do a meaningful comparison of algorithms.

Overview about the thesis

The first chapter introduces the problem and describes the motivation to solve this problem. The second chapter provides a theoretical overview about the possible approaches of time series data mining and quality related data mining in manufacturing. The third chapter describes the choice and application of specific methods for the problem at hand. In the next chapter a depiction and description of the results for the described approaches is given. In the final chapter the findings are discussed and conclusions are made.

Theoretical Background 2

At first a short overview about data mining in general is given. Then follows a section about data mining and quality in manufacturing. Next is a general overview about the disciplines of time series data mining, with special focus on representation and distance of time series. After that, possible partitioning approaches for the dataset of a manufacturing process are introduced. This results in an overview showing how to handle and process manufacturing data in the context of time series data mining.

2.1 Data Mining

In this section at first the CRISP-DM and the data processing pipeline are compared and next are the common data types with their portability into each other.

2.1.1 CRISP-DM and the Data Processing Pipeline

One of the most popular standards used in data mining is the cross-industry standard process for data mining (CRISP-DM) [Cha+00]. Therefore, it is used as reference for the project management during the implementation of this thesis. An overview is shown in Fig. 2.1, that displays its generic tasks.

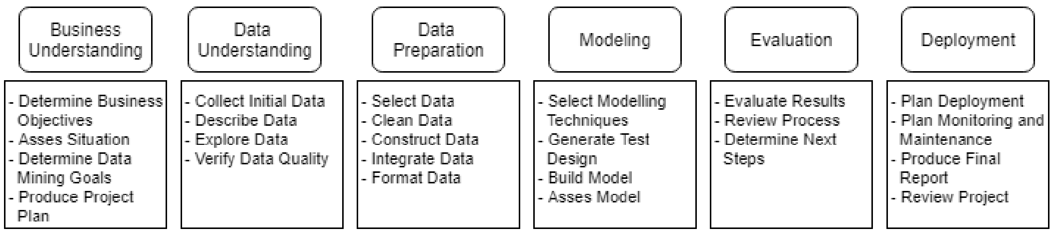


Fig. 2.1: CRISP-DM: Generic tasks based on [Cha+00].

Another point of view onto data mining is given by Fig. 2.2, in which more relevance is given to the aspect of data processing. Data mining is described as a data processing pipeline that is quite similar to the CRISP-DM, but more emphasis is on the need to implement pipelines during data mining.

As seen in Fig. 2.1 the data collection step of the pipeline is the same as "Collect Initial Data" in the second column within the data understanding of CRISP. Feature

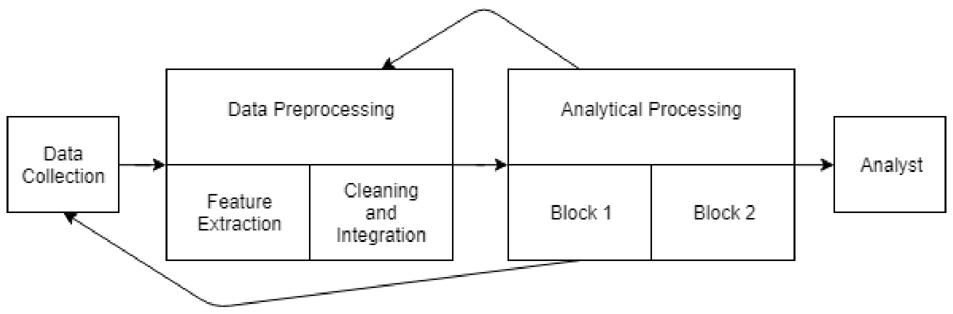


Fig. 2.2: Data processing pipeline based on [Agg15].

extraction in the pipeline framework is connected to "Construct Data" in the CRISP-DM solution. Cleaning and integration are analogous again. The next step is the analytical processing or modelling step that generates output that needs further evaluation by an analyst.

2.1.2 Data Types and Type Portability

In this chapter a overview is given about the different data types that are common in the field of data mining. Furthermore a directed graph points out possible data type transformations.

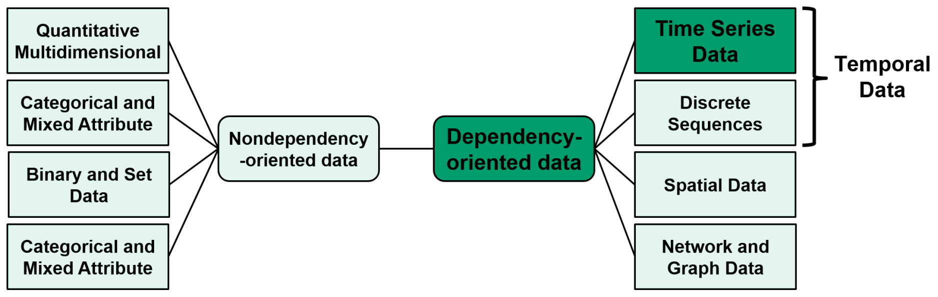


Fig. 2.3: Data types [Agg15].

As shown in Fig. 2.3 data is either of the dependency-oriented or nondependency-oriented type. Data of the nondependency-oriented type is more common and more simple to work with [Agg15] as no dependencies are specified between the data item or the attributes. While working with dependency-oriented data there exists an implicitly or explicitly defined relationship between the data items [Agg15]. In the case of time series data the relationship is implicit, as two successive values are related to another in the time dimension [Agg15].

In this work, especially the time series data is important because the analysed dataset is mostly measured by sensors. The goal is to mine the data regarding the time dimension to incorporate the time dependent relation in the analysis.

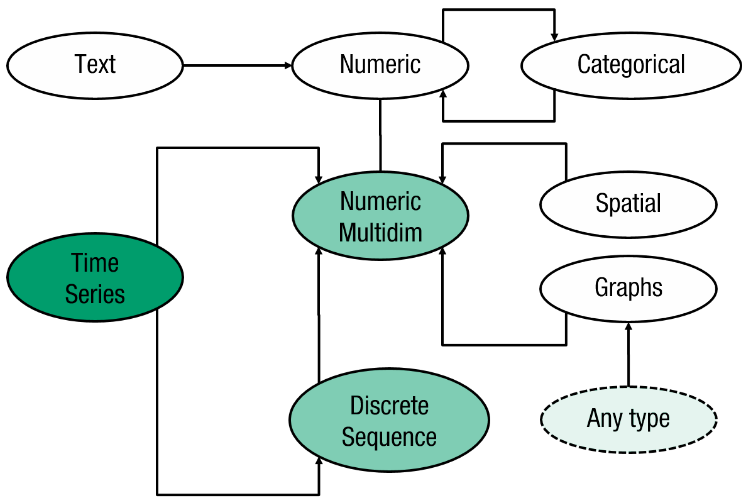


Fig. 2.4: Data type portability, based on textual description from [Agg15].

Fig. 2.4 shows the data types and possible transformations between them. Various reasons for such transformation exist, such as data size reduction or availability of more diverse algorithms, which helps to gain different insights into the data. In this thesis especially the transformation from the time series data type into the numeric multidimensional data type is of interest. Time series summarization (Ch. 2.3.3) allows to significantly reduce the dataset size and the resulting numeric multidimensional vector can easily be used as input in standard machine learning frameworks.

2.2 Quality in Manufacturing and Data Mining

An introduction about different approaches of quality in manufacturing is given and after that the advantages of data mining in manufacturing are shown.

In manufacturing there are usually three product development stages as seen in Fig. 2.5. At first the product and its process are designed. After that follow the manufacturing stage and the customer usage stage. In each of the phases quality plays an important role. Different quality tasks include process quality description, the prediction of quality, the classification of quality and parameter optimization [Kök+11].

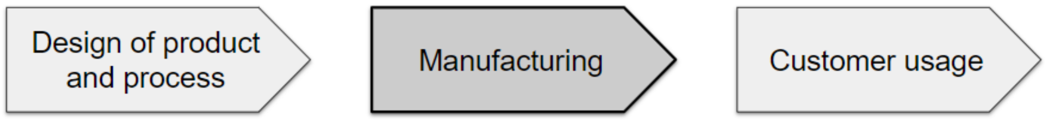


Fig. 2.5: Product development stages [Kök+11].

The specific use case presented in this thesis revolves around an already implemented process in the manufacturing stage. More specific to the product manufacturing phase it is important to determine "...the quality levels for a given set of input parameters"[Kök+11]. On the basis of this quality analysis the process itself can be improved and other corrective quality related actions can be taken [Kök+11]. The tasks, classification of quality and prediction of quality, take a set of input parameters and connect them to quality related output parameters [Kök+11]. With this, additional knowledge about outputs (like defect or no defect) can be gained [Kök+11].

Quality improvement programs such as six sigma [Sch+08; YH03] or kaizen [Bru17], that make use of data to solve quality problems, already exist [Kök+11]. Furthermore, "due to advances in data collection systems and analysis tools, data mining (DM) has widely been applied for quality improvement in manufacturing"[Kök+11]. Advantages of machine learning methods in comparison to traditional methods include:

- Handling of high dimensional problems, as the available data becomes more complex and less transparent [Wue+16].
- "Continuous quality improvement in large and complex processes"[Wue+16].
- Reduction of cycle time and scrap [Wue+16].
- Discovering of rare "couplings between different equipment and inefficient procedures"[Tao+18] or discovery of formerly unknown implicit knowledge and identification of implicit relationships in datasets [Wue+16].

- Automatic identification and removal of failed products [Tao+18].
- Elimination and control of quality defects [Tao+18].

Depending on the specific use case more or less advantages can be utilized during the analysis.

Notes on quality in manufacturing and data mining

The applications of data mining used for quality improvement in the manufacturing industry is reviewed in [Kök+11] and [Ge+17] gives an overview about the methods used for data mining in the process industry. In [Wue+16] the role of machine learning in manufacturing is highlighted and in [Wue+14] a two step approach of clustering and classification is used for quality monitoring.

The "Bosch Production Line Performance" challenge [Rob] initiated papers with the aim to improve the production line performance using classification based approaches [MK16; Pav16; Zha+16].

2.3 Time Series Data Mining

A general overview about the disciplines in the field of time series data mining is given in this chapter. At first, different definitions and notions of the considered literature are harmonized. Based on this the main disciplines are described with more focus on representation and distance measures as they are needed during classification of time series.

2.3.1 Definitions

In the following are some of the important definitions found in the literature. They have slightly been adapted for the sake of comparability.

The first differentiation is given by [Agg15]: "Continuous temporal data sets are referred to as time series, whereas discrete temporal data sets are referred to as sequences."

Definition 2.3.1. *Univariate Time Series.* A time series $T \in \mathbb{R}^n$ is an ordered sequence of $n \in \mathbb{N}$ real-values variables $y_i \in \mathbb{R}$: $T = (y_1, y_2, \ldots, y_i, \ldots, y_n)$. The measurements y_i are made at equally spaced time instants according to a given sampling rate. [EA12]

If the time series is not sampled in equally spaced intervals for all instances of time series, timestamps must be added to the measurements.

Definition 2.3.2. *Univariate Time Series Data.* Univariate time series data $T \in (\mathbb{R} \times \mathbb{R})^n$ is defined as a sequence of $n \in \mathbb{N}$ tuples (y_i, t_i): $T = [(y_1, t_1), (y_2, t_2), \ldots, (y_i, t_i), \ldots, (y_n, t_n)] \wedge (t_1 < t_2 < \ldots < t_i < \ldots < t_n)$, where each $y_i \in \mathbb{R}$ is the measurement, and each $t_i \in \mathbb{N}$ is the timestamp at which the corresponding y_i occurs. [Wan+13]

This leads to a similar definition for multivariate time series data similar to [Agg15]:

Definition 2.3.3. *Multivariate Time Series Data.* Multivariate time series data $Y \in (\mathbb{R} \times \mathbb{R})^{d \times n}$ of length $n \in \mathbb{N}$ contains $d \in \mathbb{N}$ many univariate time series data $T \in (\mathbb{R} \times \mathbb{R})^n$.

The Definition of multivariate time series data (Def. 2.3.3) describes a matrix with dimension $d \times n$ with a *tuple* $\in (\mathbb{R} \times \mathbb{R})$ at each entry. The set of tuples $\overline{y}_i \in (\mathbb{R} \times \mathbb{R})^{d \times 1}$ recorded at timestamp t_i consists of one entry for each of the individual time series, $\overline{y}_i = ((y_{1i}, t_i), (y_{2i}, t_i), \ldots, (y_{di}, t_i))^T$. The information about the timestamp t_i

occurs multiple times. To make this easier to read the depiction $\overline{Y}$ of a multivariate time series data Y stores the timestamp (only once) in an additional row. It looks the following:

$$\overline{Y} = \overline{y}_1 \cdot \overline{y}_2 \cdot \ \ldots \ \cdot \overline{y}_i \cdot \ \ldots \ \cdot \overline{y}_n = \begin{bmatrix} y_{11} & y_{12} & \ldots & y_{1i} & \ldots & y_{1n} \\ y_{21} & y_{22} & \ldots & y_{2i} & \ldots & y_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ y_{j1} & y_{j2} & \ldots & y_{ji} & \ldots & y_{jn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ y_{d1} & y_{d2} & \ldots & y_{di} & \ldots & y_{dn} \\ t_1 & t_2 & \ldots & t_i & \ldots & t_n \end{bmatrix}, \ \overline{Y} \in \mathbb{R}^{(d+1)\times n}$$

No assumption is made about equal sampling rates for the different time series and there may be vectors $\overline{y}_i$, in an extreme case, that contain only the time t_i and one value y_{ji}. This occurs if only one measurement is done at this specific timestamp t_i.

Following the definition of [EA12] in combination with [Agh+15] the representation of a time series can be defined as:

Definition 2.3.4. *Time Series Representation.* Given a univariate time series $T \in \mathbb{R}^n$ of length $n \in \mathbb{N}$, a representation of T is a model T' of reduced dimensionality $n' \in \mathbb{N}$ with $(n' << n)$ such that T' closely approximates T [EA12].

A more specific case of representation is segmentation (Def. 2.3.6), in which the time series is divided into consecutive approximating segments. Following [Din+08] and [Wan+13] a segment of a time series can be defined as:

Definition 2.3.5. *Time Series Segment.* The portion of a time series between timestamp $t_i \in \mathbb{N}$ and timestamp $t_j \in \mathbb{N}$ (inclusive) is called a segment and is denoted as s_{ij} . In particular, the segment $s_{i(i+1)}$ between two consecutive points is called a line segment. [Wan+13]

The description of segmentation is found in [KK03]:

Definition 2.3.6. *Time Series Segmentation.* Given a time series $T \in \mathbb{R}^n$ containing $n \in \mathbb{N}$ data points, construct a model T', from $k \in \mathbb{N}$ piecewise segments $(k << n)$, such that T' closely approximates T. [KK03].

Segmentation is a special kind of representation, which makes explicit use of segments. Thereby the length is reduced, as the number of segments is considerably smaller than the number of data points in the original time series [MS11]. Nevertheless the segmentation task still aims to create an accurate approximation of the time series by retaining its essential features.

The distance between two time series is defined as:

Definition 2.3.7. *Time Series Distance.* Let $T_m \in \mathbb{R}^n$ and $T_n \in \mathbb{R}^n$ be time series of length $n \in \mathbb{N}$. If the distance between two time series is defined across all points in time $t_i \in \mathbb{N}$, then $dist(T_m; T_n) \in \mathbb{R}$ is the sum of the distances between the individual points. [Agh+15]

Definition 2.3.8. *Time Series Summarization.* Global characteristics are used to represent a time series, which leads to the creation of a feature vector. [Mit10]

Be aware that [EA12] denotes segmentation (Def. 2.3.6) also as summarization, which is not the summarization defined above.

2.3.2 Disciplines

Compared to data mining the more specific field of time series data mining requires more specialized methods. Therefore a short overview about the different disciplines of time series data mining seems reasonable. The disciplines of representation and distance are omitted here, but found in the next two chapters.

Query by content or *indexing*: Informally said, "...it is based on retrieving a set of solutions that are most similar to a query provided by the user"[EA12]. From a database containing time series an ordered list of specific length with the most similar time series, regarding the query time series, are retrieved [EA12; KK03]. "The indexing scheme must allow to efficiently manage and query evergrowing massive datasets"[EA12].

Clustering and *classification*: The main difference to normal one lies within the definition of the representation (Ch. 2.3.3) specific distance measure (Ch. 2.3.4) and its corresponding properties. Usually a distinction regarding whole sequence and subsequence algorithms is done. The latter one extracts subsequences from a single or multiple longer time series. The challenge is to find meaningful overlapping or non-overlapping sequences as wrongly (overlapping) chosen sequences may result in less meaningful results [EA12].

Motif or *pattern discovery* is the task of finding subsequences that appear recurrently in a longer time series [EA12]. These patterns may be of interesting or even surprising nature [Fu11]. During analysis these motifs may carry important information which could be used for e.g. for clustering [EA12].

The task of *forecasting* or *prediction* of a time series is to model "variable dependencies to forecast the next few values of a series"[EA12] or in other words the history of values is used to predict future values [Agg15]. Time series forecasting is a current and active field of research and is applied to a wide range of applications [EA12; Agg15].

Outlier or *anomaly detection*: Outliers may be individual data points that are different [Agg15] or even whole subsequences that do not fit the model as they contain anomalies [EA12]. In case of time series data, an outlier may be the result of a preceding event of interest, so outlier detection is also called event detection [Agg15].

2.3.3 Time Series Representations

Univariate time series data (Def. 2.3.2) is an usual used data format for storing time series related data in a database as it carries all the raw information available, e.g. sensory discretised measurements of a continuous signal. Furthermore, while working with time series data, it is common to extract (non)overlapping windows (example in Fig. 2.6) of this univariate time series data to use them as input for other disciplines such as classification, clustering or motif discovery.

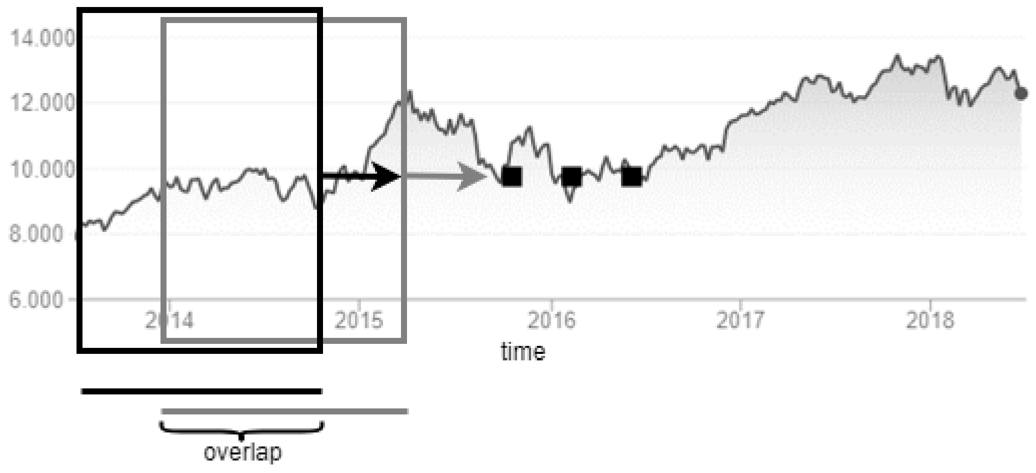


Fig. 2.6: Window approach for one time series: The black window (step zero) is shifted along the time dimension, which results in the grey window (step one).

The extraction of overlapping windows implies a duplication of information, which again makes it hard to work efficiently with. Therefore the need arises to reduce the dimensionality of each univariate time series data window – dimensionality in this case refers to the length of the individual time series (Def. 2.3.4). In other words it is essential to apply a representation schema while working with time series data, as it allows to significantly reduce the dimensionality of the problem for further processing [EA12; Mit10].

Each transformation of original data involves some kind of information loss and so the choice for a representation also influences the visibility of characteristics of a time series. Some representations may be a good fit to find similarities in frequency (discrete fourier transformation or discrete cosine transformation) and others are more suited to find similarities in the underlying structure (markov models or auto regressive moving average processes). Therefore it is of interest what representation approaches are principally available as described in the following.

Additionally to the raw time series data format, the further representations in general are categorized into *non-data adaptive*, *data adaptive* and *model based* approaches [EA12; Mit10]. At the end of this subsection an even more compact way of describing a time series, the *summarization* (Def. 2.3.8), is explained.

While using *non-data adaptive* approaches, the parameters of the transformation are the same for every time series [EA12]. Examples for such transformations are discrete fourier transformation (DFT), discrete wavelet transformation (DWT) or piecewise aggregate approximation (PAA).

Data-adaptive approaches however, adjust their parameters depending on the underlying data [EA12]. Commonly used methods, among others, are singular value decomposition (SVD), symbolic aggregate approximation (SAX) or the clipping technique, where a binary string represents the time series. These methods try to minimize the global reconstruction error of the time series [Agh+15]. In general it is possible to change many non-data adaptive approaches into a data-adaptive one by adding a data sensitive step before [EA12].

In case of *model based* approaches the goal is to find the underlying model that describes the time series [EA12]. Examples for model based approaches are hidden markov models (HMM) or autoregressive models like autoregressive integrated moving average (ARIMA) [Mit10].

Furthermore, if a representation fulfils the lower bounding property, it does not allow any false dismissals [Mit10] or in other words "... if two series are to be found similar in the original space, they should also be found similar in the transformation space"[Mit10; Agh+15]. This means that the distance defined in the chosen representation is less or equal than the one in the original data space [Mit10]. The authors in [Din+08] and [Wan+13] do an extensive experimental comparison of representation and distance measures of time series.

Lastly, time series *summarization* [Mit10] is an even more compact way of describing time series. This can be done with various methods ranging from summarative statistical properties of a time series, like mean, median and variance ranging to histogram based summarization [Mit10]. An advantage of summarization over representation is the the even more significant data reduction.

2.3.4 Time Series Distances

The distance measure for the raw representation of a time series (Def. 2.3.7) comes with different challenges such as the distortion of the time series (warping in time or scaling of the measurement amplitude), dropped readings of the sensor or the presence of noise and outliers.

Specific to the chosen representation of the time series different distance measures are available, which are invariant to some of the mentioned challenges (some distance measures with regard to their invariances are compared in [EA12]). Furthermore, while defining a distance measure it is important to consider what kind of similarity is of interest [Agh+15]. Therefore, the choice of the distance function is constrained by the chosen representation and the similarity of interest.

In general, distance or similarity can be categorized into *shape based*, *edit based*, *feature based* and *structure based* approaches [EA12] and each approach considers a different kind of similarity.

The *shape based* approach tries, as its name implies, to find similarities in the shape of time series, whereas the time of occurrence is not important [Agh+15]. Popular methods are L_p norms or elastic measures such as dynamic time warping (DTW) while the latter one handles scaling along time better [EA12; Agh+15].

The *edit based* approach uses the idea of counting the number of operations required to transform two strings into each other [EA12]. Here, methods include the Levenshtein distance or longest common subsequence algorithms.

Feature based distances "... rely on the computation of a feature set reflecting various aspects of the series"[EA12]. Possible approaches may include the use of the coefficients of a DWT and DFT or to use a set combined features such as maximum, number of local maxima, number of ascending intervals, quantiles, etc. Be aware of the close connection between the summarization of time series in chapter Ch. 2.3.3 by [Mit10] to the feature based distance approach here of [EA12].

Finally, the *structure based* distance approach goes hand in hand with the model based representation of time series. In this case the distance is defined by the differences in the parameters of the corresponding model.

Notes on Time Series Data Mining

The book [Agg15] includes a concise overview about time series data mining and [Mit10] is entirely about temporal data mining. In addition [Agg17] is about outlier analysis including time series approaches. The authors in [Xu+15] write about the different steps of data cleaning in general with regard to time series specific methods.

The field of time series data mining is reviewed by [EA12] and [Fu11], the more specific field of time series classification by [Bag+17] and time series clustering by [Agh+15]. An comparison of different similarity or distance measures of time series is given by [EA12] (evaluating for invariances) and by [Agh+15] (including an evaluation of representation methods). The authors in [Din+08] and [Wan+13] do an extensive experimental comparison of representation and distance measures of time series.

2.4 Dataset Partitioning Approaches

The initial domain dependent question needs to divided into smaller easier to handle data mining tasks and every subtask contributes to answering the initial question. Data partitioning therefore is essential to structure the available data as one task does not need the entire available dataset as input.

Therefore it is one of the first steps to find a natural partitioning that is present in the data. This section introduces different possible approaches to divide the dataset of a manufacturing process – time series based ones and state based ones.

2.4.1 Manufacturing Process Dataset

For partitioning it is crucial to be aware of the characteristics of the specific manufacturing process at hand, as shown in Fig. 2.7. The manufacturing process is realised as a production line, while the production line itself consists of multiple machines and each machine has got its own meta information, like tool numbers etc. Furthermore, additional meta information is associated for the entire production line such as momentarily produced article number, variant number of the article and so on.

The considered manufacturing process is a linear continuous production process. Linear describes the property that the input material always flows along the process from one machine into another, i.e. there is no variation of the ordering of the machines. Moreover, continuous means the processed material never changes order among itself. The input material at time t_i, compared to the input material at time t_{i+1}, finishes processing first at all times.

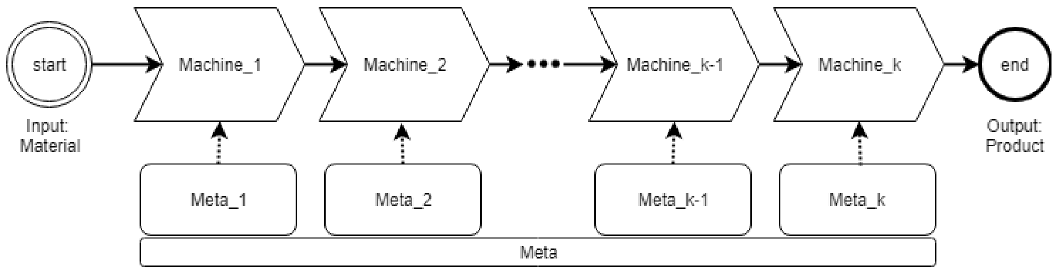


Fig. 2.7: Linear continuous production line.

2.4.2 Time Series based

The first approach for partitioning is to divide the dataset along the time dimension – horizontal approach considering Fig. 2.7. The depiction of multivariate time series data (Def. 2.3.3) can be used to describe the dataset from a sensor based view. Each one of the multivariate time series is actually a sensor that is described by $y_j(t)$ and this way, a $SensorMatrix$ can be described that contains all the sensors of the production line:

$$SensorMatrix = Y(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_j(t) \\ \vdots \\ y_d(t) \end{bmatrix} = \begin{bmatrix} y_{11} & y_{12} & \dots & y_{1i} & \dots & y_{1n} \\ y_{21} & y_{22} & \dots & y_{2i} & \dots & y_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ y_{j1} & y_{j2} & \dots & y_{ji} & \dots & y_{jn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ y_{d1} & y_{d2} & \dots & y_{di} & \dots & y_{dn} \\ t_1 & t_2 & \dots & t_i & \dots & t_n \end{bmatrix}$$

Each sensor y_j is part of only one machine M and one machine encompasses one or more sensors. Therefore the $MachineSensorMatrix$ can be written as:

$$MachineSensorMatrix = MS(t) = \begin{bmatrix} \vdots \\ y_{j-1}(t) \\ y_j(t) \\ y_{j+1}(t) \\ \vdots \end{bmatrix}$$

The informal "aggregation" of all machine sensor matrices is equivalent to the sensor matrix $Y(t)$ defined before.

The $MetaInformationMatrix$ is described in a similar way as the sensor matrix. At the different timestamps t_i different values of the meta information are present. Although the meta information is not strictly sensor based, they can be interpreted in an analogous way, dependent on the time dimension:

$$MetaInformationMatrix = MI(t) \begin{bmatrix} mi_1(t) \\ mi_2(t) \\ \vdots \\ mi_o(t) \\ \vdots \\ mi_p(t) \end{bmatrix}$$

This leads to the idea of a $MachineMatrix$, which contains all of its sensors as well as its relevant meta information:

$$MachineMatrix = M(t) = \begin{bmatrix} \vdots \\ y_{j-1}(t) \\ y_j(t) \\ y_{j+1}(t) \\ \vdots \\ mi_{o-1}(t) \\ mi_o(t) \\ mi_{o+1}(t) \\ \vdots \end{bmatrix}$$

With these definitions the production line PL can be written as a combination of different machine sensor matrices MS_k and the meta information matrix $MI(t)$:

$$ProductionLineMatrix = PL(t) = \begin{bmatrix} MS_1(t) \\ MS_2(t) \\ \vdots \\ MS_k(t) \\ MI(t) \end{bmatrix}$$

2.4.3 State based

Another approach for partitioning is to divide the dataset along one point of time – vertical approach considering Fig. 2.7. Each one of the already described matrices can be evaluated at one specific point in time, also called a timestamp. The resulting $MachineState$ is defined as following:

$$MachineState = M(t_i) = \begin{bmatrix} \vdots \\ y_{j-1}(t_i) \\ y_j(t_i) \\ y_{j+1}(t_i) \\ \vdots \\ mi_{o-1}(t_i) \\ mi_o(t_i) \\ mi_{o+1}(t_i) \\ \vdots \end{bmatrix}, \; t_i \in \{t_1, \ldots, t_n\}$$

Together with the analogously defined $MachineSensorState\ MS(t_i)$ and $MetaInformationState\ MI(t_i)$ the $ProductionLineState$ is:

$$ProductionLineState = PL(t_i) = \begin{bmatrix} MS_1(t_i) \\ MS_2(t_i) \\ \vdots \\ MS_k(t_i) \\ MI(t_i) \end{bmatrix}, \ t_i \in \{t_1, \ldots, t_n\}$$

Lastly the definition of an $ArticleState$ can be formulated. It can be seen as a feature vector that contains all sensor values and values of meta information present for the article. During production an article passes each sensor directly or indirectly. At the moment the article passes by, the corresponding value of the sensor or the meta information is stored into the article state. Hence the resulting article state contains information about all the possible sensors and meta information. The concept of the article state can be found in a similar way in [Wue+14] that describes a $ProductState$, which is used for monitoring the quality in manufacturing in a supervised manner.

3 Data Mining in an Industrial Process

The purpose in this thesis is to complement the existing quality control by finding a suitable algorithm for automatic determination of the article quality during production. Essential steps of data mining are dataset description, data partitioning, preprocessing and analytical processing and therefore in this chapter their application on the analysed data is described.

3.1 Dataset Description

The first step is to create a dataset that can be analysed with machine learning methods. This takes a considerable large amount of time and effort as the data lake of the company contains structured and semi structured data. After pre-selection more than 100 various potential usable features are documented and analysed regarding their usability to determine the quality. A method is developed to reorganize the storage of the data and to load it into a NOSQL database. At this point the raw data finally is available for further analysis. The rough layout of data processing is shown in Fig. 3.1.

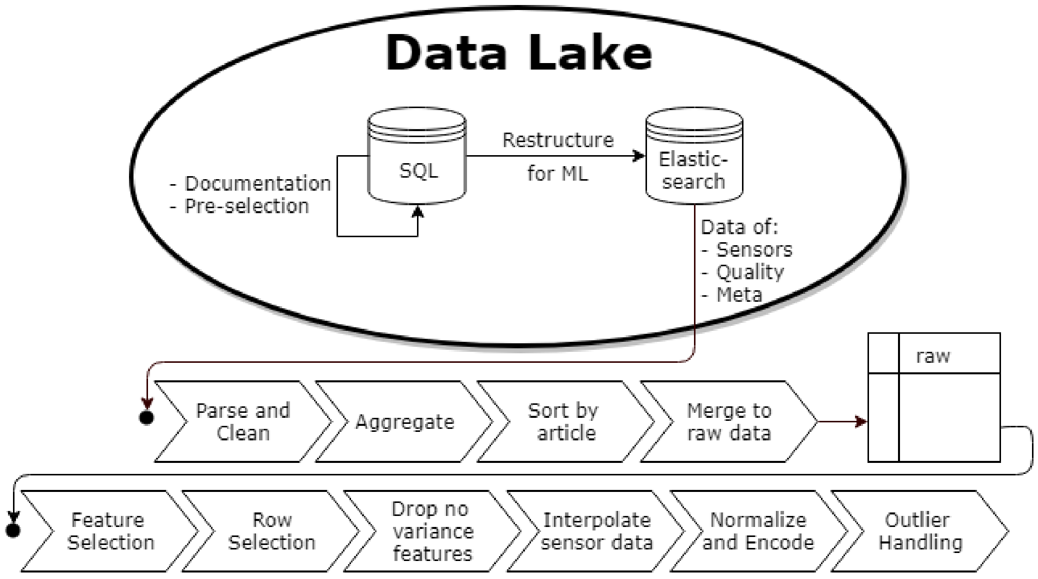


Fig. 3.1: Data processing: From the data lake to a machine learning usable dataset.

3.1.1 Dataset Detail

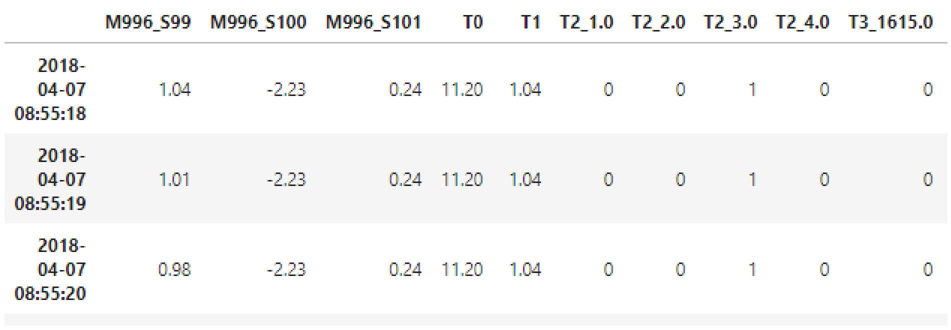
Data understanding is an essential step during the analysis, which includes the initial description of the dataset. In this case it is carried out on a normalized dataset with renamed features and as a consequence no inference can be made about the actual data. The secrets regarding the production of the company are kept safe, but nevertheless necessary insight into the structure of the data is given.

The manufacturing line of interest is "L42" which consists of eight different machines. The machine ids are correspondingly "M994, M995, M996, M1195, M1391, M1392, M1470 and M1490". Out of the available features around 20 can be considered meta information while the rest consist of sensor related values.

The meta information contains information about the specified time intervals during the manufacturing process. This includes information about the internal assigned material number, variant numbers of produced articles, garbage production, amount of recycled material, aggregated quality information and more. In the following these are named Tj, e.g. T1, T2, T3, ... The column names T2_1.0 ... T2_4.0 represent an one hot encoded categorical feature with name T2 and the categorical values of 1.0 ... 4.0 (Tab. 3.1).

For the sensor related values each feature column represents one sensor Si that is assigned to a machine. Among others, information about temperature, pressure, weight, speed, effective power and position of different machine elements is given. To keep track which machine the feature belongs to, these features follow the naming scheme M995_Si. All together the dataset can be seen as a big flat table containing all information at one time index, schematically shown in Tab. 3.1.

Tab. 3.1: Excerpt of normalised data.



	M996_S99	M996_S100	M996_S101	T0	T1	T2_1.0	T2_2.0	T2_3.0	T2_4.0	T3_1615.0
2018-04-07 08:55:18	1.04	-2.23	0.24	11.20	1.04	0	0	1	0	0
2018-04-07 08:55:19	1.01	-2.23	0.24	11.20	1.04	0	0	1	0	0
2018-04-07 08:55:20	0.98	-2.23	0.24	11.20	1.04	0	0	1	0	0

A company offers various articles, which customers, both internally as well as externally, can buy. Depending on the actual demand for one specific article more or less batches of different sizes are produced. The production time intervals of batches for one specific article must not be consecutive. For example at first some article A, then some article B and then again article A is produced. This is reflected by the time index of an instance of a dataset for one article. There may be temporal gaps of different sizes between one specific time index and the next. The temporal index of the dataset is not uniformly sampled.

3.1.2 Representative Article

Quality determination parameters are specific to the article and to the production line it is produced on and therefore one representative article is chosen to do a meaningful comparison of algorithms. In case of the analysed production line L42 different articles are produced and one article is chosen as a representative. Nevertheless, the used approach for this specific article can analogously be adapted to the other articles on other production lines as well.

Out of the articles produced by line L42 the one with article number 22333 will be the representative for all articles. This specific article is chosen because it has the most samples available. Because of this, this article is more likely to represent a more general case with regards to the data it produces. Further application of methods and discussion is solely based on this one article with normalized values.

3.2 Data Partitioning

Data partitioning is essential to structure the available data as one task does not need the entire available dataset as input (Ch. 2.4). Here two of the possible approaches of partitioning for the dataset of a manufacturing process are described.

3.2.1 State based

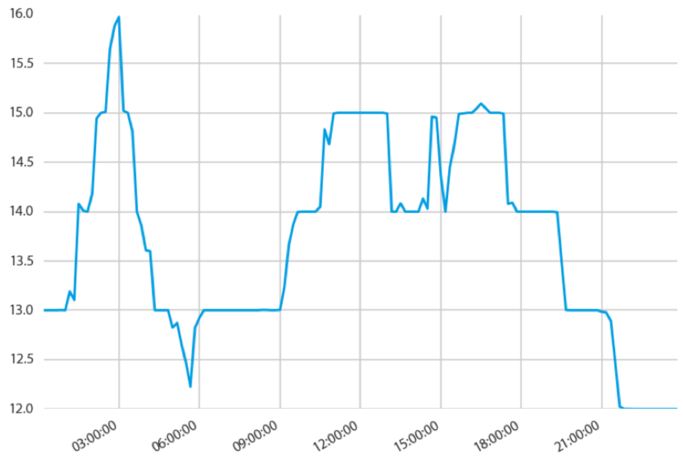
The final quality of the produced article is influenced by the parameters during production and this is also reflected in the overall state of the production line itself. Therefore the $ProductionLineState$ (Ch. 2.4.3) is a good fit as a state based approach for determining the quality of the article as it incorporates a wide range of available information.

3.2.2 Time series based

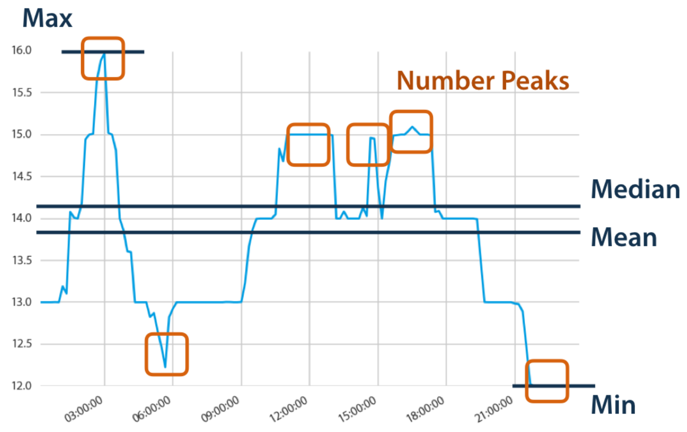
Nevertheless, a state based approach suffers the drawback of not including the implicit time dependency, which occurs in time series data. The time dimension also carries important information and so the $ProductionLineMatrix$ (Ch. 2.4.2) is utilised in conjunction with a window approach (Fig. 2.6).

In a time series, the value $y(t_i)$ is related to the values before $y(t_{i-1}),\ y(t_{i-2}),\ y(t_{i-3})$, ... and this property can be utilised by the window approach. Each window represents the preceding time series that contains the implicit information which leads to the value $y(t_i)$. The windows have a fixed size and overlap.

Therefore it is a plausible idea to map the resulting time series windows onto the $ProductionLineState$ present at time t_i. This way, instead of only one single value, the entire preceding time series determines the article quality at one timestamp t_i. The combination of values $y_j(t_i)$ of each time series j make up the $ProductionLine$-$State$. Instead of only using the single (state based) value $y_j(t_i)$ the resulting j different time series windows (one for each time series at the specific timestamp) now determine the article quality at time t_i.



(a) Original time series [Chr+18].



(b) Extraction of time series features [Chr+18].

Fig. 3.2: Example of a time series feature based representation.

Using the entire matrix of values is impractical as this would lead to an multiplication of data size in case of overlapping windows. The initial raw representation (Fig. 3.2a) of the time series is used and a set of relevant characteristics is extracted for further processing (Fig. 3.2b). This way, for each time series a substitutional feature vector is calculated and used instead of the entire time series window, which results in a significant dataset size reduction. This is a feature based distance measure (Ch. 2.3.4), which is closely connected to summarization (Ch. 2.3.3) of a time series.

3.3 Preprocessing

The overall achievable possible result of the final analysis steps are indirectly determined by the correct preprocessing of the data. At first the data is explored to validate the initial data description and to deepen the data understanding. Next is explicit type casting and feature encoding of categorical values. After that follow the arising challenges of preprocessing and its possible solutions for the more common cases like missing data and normalization. Next is outlier removal as well as the case of noise removal.

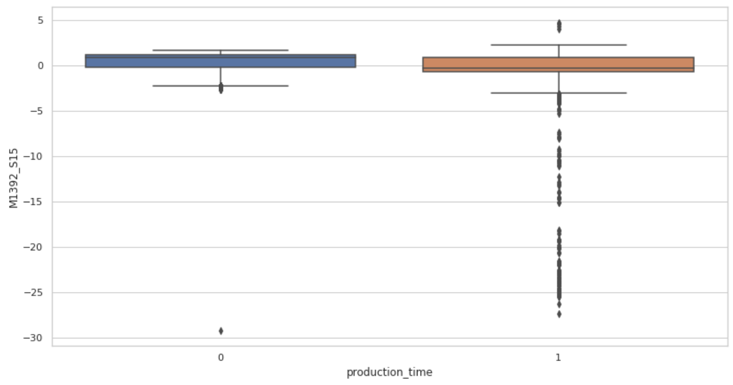
3.3.1 Data Exploration

Data exploration follows no standardized approach and hence the most important applied methods and findings are shown. This includes summarative statistics, information about quantiles, duplicate values, types of time and class imbalance.

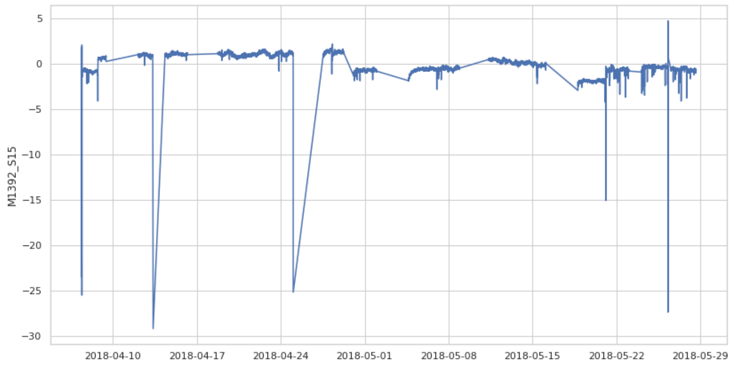
Summarative Statistics

First important insights into the data are provided by visual summarative statistics. For each feature a combination of boxplot and lineplot diagramm, as seen example wise for the sensor M1392_S15 in Fig. 3.3, is used. During exploration all of the available features for data mining are examined in this way, but not depicted in this thesis due to the non-disclosure agreement.

The boxplot displays information about the distribution of the data such as the minimum, 1st quartile, median, 3rd quartile, maximum and outliers. The boxplot diagramm contains both class labels as a boxplot. The left/blue boxplot of Fig. 3.3a represents error time. The right/orange boxplot of Fig. 3.3a represents production time.



(a) Boxplot diagram: Class label error time on the left/blue and class label production time on the right/orange.



(b) Lineplot diagram: The time dependent changes of the sensor values are visible.

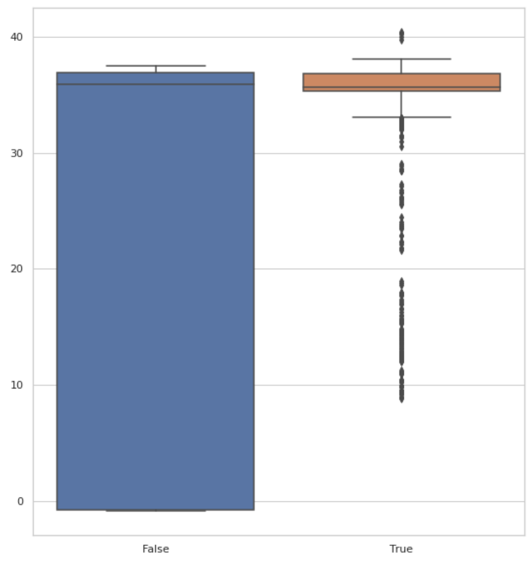
Fig. 3.3: Exploratory graphics. The y-axis displays the normalized values of the sensor.

The lineplot diagram (Fig. 3.3b) on the other hand takes the dimension of time into account. It gives a rough overview about the development of each variable and allows a first intuitive visual exploration. Information about drifts, periodicity, shifts, noise and more of the signal is provided in a visual way.

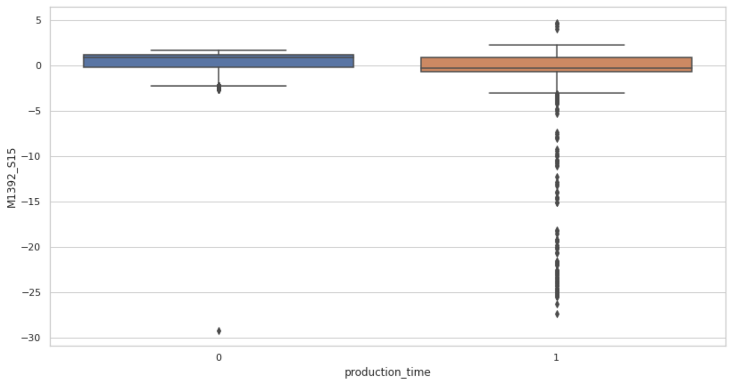
Quantiles

Investigation of the quantiles gives a first summarative insight into the distribution of the data values. The error time as already seen in Fig. 3.3 before (depicted in Fig. 3.4b again) is actually the more cleaned version of data as it is originally available for preprocessing. In Fig. 3.4a the originally available data is shown and the boxplot of the error time in left/blue is a distorted one as the 25 %-quantile reaches until the lowest values.

Further examination of the quantiles reveals that the value for some sensors does not change in the slightest until the 32 %-quantile. While investigating this phenomenon, the conclusion is made, that the error time needs further categorization. The two arising error times are error time before a corrective event and error time after a corrective event.



(a) Samples with screw speed equal zero are part of the dataset.



(b) Samples with screw speed equal zero are not part of the dataset.

Fig. 3.4: Dataset comparison of screw speed.

Two options are available if an error occurs. Either the production can be stabilized to continue with producing or a corrective event takes place that interrupts the entire production. The corrective event involves stopping of the input flow of material into the production line so that no more waste is generated. The stop of input material is indicated by the variable of screw speed being zero, which is reflected in the distribution of data and seen in Fig. 3.4a as a distorted boxplot.

This clear behaviour of only one value being present until the 32 %-quantile could not be observed for all the sensors. The reason lies in what a sensor actually measures. One sensor measures the very quickly adjustable screw speed while another one measures the more inert temperature. The change in temperature lags behind the rapid change of screw speed because the temperature physically can not be influenced or does not change as fast as the screw speed can be adjusted.

There is no advantage in learning the samples of error time that are after a corrective event because an appropriate action is already taken. Therefore they will no longer be part of the dataset as the goal is to classify error time and production before the corrective event happens.

As a side note: The sensors that show a clear behaviour until the 32 %-quantile have an unusually high correlation towards the predicted label too. This is one more indicator for the need to analyse these features in more detail.

Duplicates

As seen in Fig. 3.3 many of the measured values are distributed within a close interquartile interval and this holds for the values during production time as well as error time. Even more, the two different intervals error time and production time overlap to a large degree. As this overlap can be seen in many of the sensors, it might be of interest to analyse for the presence of duplicates in the dataset. For example, the absence of duplicates indicates that there may be a cumulative value present in the dataset that needs further processing.

At first for each feature the fraction of unique values (=uniqueValues/allValues) specific to all values is calculated and compared in a boxplot as seen in Fig. 3.5.

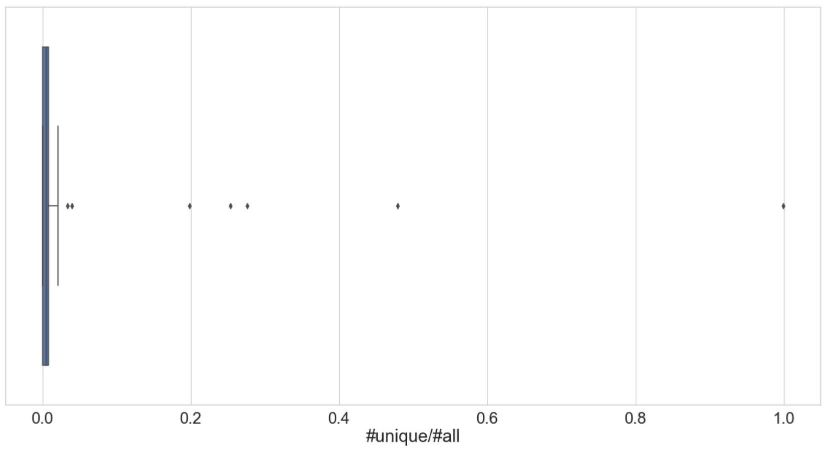


Fig. 3.5: Boxplot over fraction of unique values on all variables.

The mean of the fraction of unique values is at 2.7 % and the median is at 0.5 %. So most of the features seen alone contain a very low fraction of non-unique samples as indicated by the small interquartile borders near 0.0. But some outliers and even one near the 1.0 are detected.

The one feature containing only unique values (fraction of unique values=1.0) leads to the entire dataset being duplicate free. The Fig. 3.6 shows the sensor that does only contain unique values. This sensor represents a value that cumulates over time and the feature can not be used during learning like this. Either feature engineering needs to be applied (using the differences) or the sensor is no longer part of the dataset.

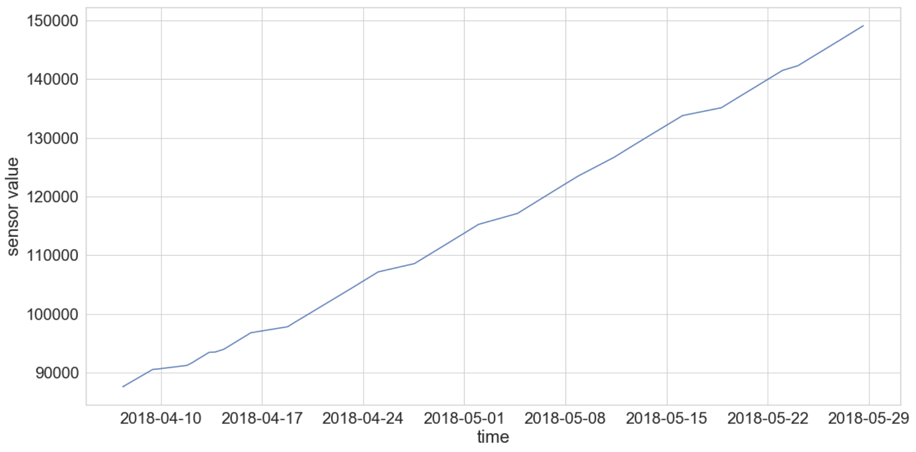


Fig. 3.6: Sensor that does not contain any duplicate values as it measures a cumulative value.

Types of Time Intervals

The meta data of the production line reveals that there are different types of time in which the overall state of the line can be categorized in. Among others there are production time, error time, set-up time, idle time, dead time. Every time that is not production time or error time will be referred to as "other-time". The other-time is not part of the used dataset, as it is of interest to differentiate between error time and production time only.

Class Imbalance

One important property of the dataset is class imbalance as it implicitly influences the selection of a scoring function for classification and an appropriate splitting strategy during cross validation needs to chosen. In case of production time and error time the class imbalance is present. The samples of the dataset consist of 4.6 % error time and 95.4 % production time.

3.3.2 Type Casting and Feature Encoding

The data may contain pseudo numerical features that are actually categorical features and it is important whether a feature is numerical or categorical. A feature being numerical implies an ordering of the values while it being categorical mostly does not. Lastly, categorical values usually need to be encoded to be usable as input for machine learning algorithms.

Type Casting

During loading of the data the features are implicitly parsed dependent on their values. For example there exist variant numbers or tool numbers which are parsed as a numeric type, but in reality they represent categorical information. These numbers in particular are categorical names or unique identifiers. Within these names or identifiers no ordering is actually present and without type casting these pseudo numerical features would implicitly contribute to the distance measure in an unwanted way. Therefore explicit type casting of these pseudo numerical features into categorical ones needs to be ensured.

Feature Encoding

Many machine learning algorithms do not allow categorical features as input and therefore they need to be encoded [sci]. One hot encoding is a standard approach and is a good fit for the data in the exemplary dataset for this thesis. The cardinality of the categorical features is low and therefore the resulting amount of new columns or dimensions is low also. Another advantage is that one hot encoding does not introduce an implicit ordering of the categorical values like the ordinal encoding does.

3.3.3 Missing Data

One challenge in data analysis is the high probability for missing values to occur. Algorithms for analytical processing often do not allow missing values in the data and therefore a suitable method to handle these in the dataset needs to be chosen. Goals during missing value handling are to preserve the structure of the underlying data, the shape of the distribution probability function and to keep the information loss at a minimum.

At first the type of the missing data needs to be determined [Xu+15]. It is important to know whether the data is missing at random and whether one or multiple values are missing at time t_i. This gives a first impression about the missing data and quality of the data itself. Depending on the dataset, possible solutions for missing data are deletion, replacement by mean, regression or interpolation, model-based approaches or machine learning approaches such as clustering, classification and regression. More about interpolation of time series data can be found in [Xu+15; Lep+17] and working with missing data in [LR02].

Application on the dataset

The dataset of the studied production process contains missing values. The sensors transmit their measurements into the database, but they are not all sent at the same time. Instead, there exist different groups of sensors that transmit together with the same frequency. All the sensor groups however are shifted among each other. This results in periodically occurring missing values for each of the sensors in the merged dataset.

For the data at hand deletion of datapoints with missing values would lead to heavy losses of useful data. Simple replacement by mean values does introduce too much bias as it does not relate to the implicit time dependency in the time series data type.

A commonly used method is the imputation of missing values in time series with linear interpolation. In comparison to more complex interpolation methods it often does give competitive results [Agg15]. Furthermore recent study by [Lep+17] came to the conclusion that there are no comprehensive comparative studies which help to decide in detail which interpolation method to choose. Linear interpolation will always "overestimate the minima in profiles and underestimate the maxima"[NL13]. This can advantageous as other methods such as spline interpolation lead to unrealistic low minima and high maxima [NL13].

In case of continuous production the goal is to always keep the production parameters within a certain range to produce good quality articles. Extreme values during the production usually lead to poor quality articles. Therefore, to avoid the artificial introduction of extreme values, using the linear interpolation seems a justified fit.

3.3.4 Normalization

Normalization overcomes the downside that one feature dominates another one in distance computation, simply because the measurement of the different features is conducted on different ranges. Therefore in many cases it is essential to normalize the different feature dimensions into comparable ranges to be able to calculate meaningful distances. The lack of normalization can result in a complete disregard of an actually relevant feature. Nevertheless, the transformation of the original data comes with the risk of losing information and sometimes using other methods that do not need distances at all like decision trees maybe favourable.

Popular approaches for normalization are range based (scaling the measurement of the jth-sensor to its corresponding minimum and maximum values $y_{j,i,n} = (y_{j,i} - min_j)/(max_j - min_j)$) or standardization (mapping the measurement of the jth-sensor to its corresponding z-values $y_{j,i,n} = (y_{j,i} - mean_j)/standard\_deviation_j$) [Agg15].

Application on the dataset

The sensors of a production line measure a variety of parameters. Among others, the exemplary production line has sensors for temperature, screw speed or pressure. Temperature is measured in $°C$ with a range from 10^1 to 10^2. Screw speed is measured in revolutions per minute (rev/min) from 10^1 to 10^2 and pressure is measured in bar in a range from 0 to 10^1. The different ranges of measurements make a normalization necessary so that the features are weighted the same in the distance computation. The investigated dataset contains outliers and therefore the range based min-max-scaling is not applicable as it is sensitive to outliers. Therefore, the standardization by z-score is the normalization method of choice.

3.3.5 Outlier Removal

Outlier handling is an important step during the preprocessing as only a subset of machine learning algorithms is robust against them. Moreover, outlier handling is highly dependent on the questions asked during the analysis. On one hand it might be advisable to remove the outliers for training a machine learning algorithm, while on the other hand for some tasks like event detection important information might be lost. Furthermore, to decide if an outlier captures an actual extreme value of the machine state or if it is an erroneous measurement, due to sensor failure that should be deleted, is hard.

There is no general rule that can be applied for outlier removal and a lot of different methods for specific data types and for specific types of outliers have been developed. Time series methods are categorized into univariate outlier detection and multivariate outlier detection. Further details regarding outliers are presented in [Xu+15] and [Agg17].

Application on the dataset

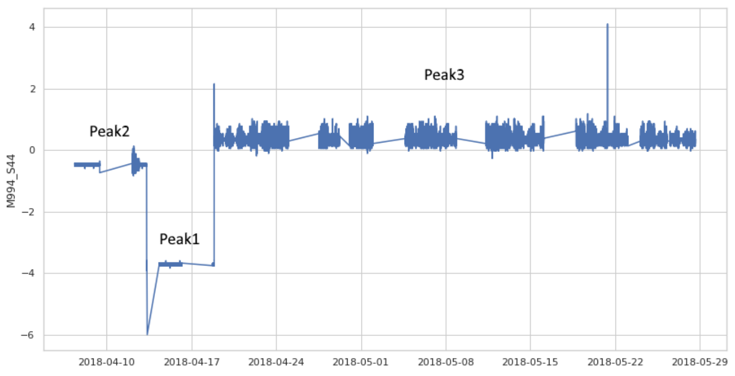
The removal of outliers comes with the risk of losing information about the system that generated the data in the first place. A detailed investigation of the outliers is simply not possible during this thesis. This investigation would require many time consuming discussions with the operating staff of the machine. Therefore the aim is to manipulate only the extreme outliers to lessen the impact of uncommon artefacts that are irrefutably not part of the underlying system.

During exploration summarative diagrams provide a concise overview about the distribution of the examined data as shown example wise in Fig. 3.3. A more detailed look at the other diagrams and statistics reveals that depending on the sensor, values with a maximum up to 300 times the standard deviation are present. These extreme values are uncommon artefacts that should not be learned during training of the machine learning algorithm.

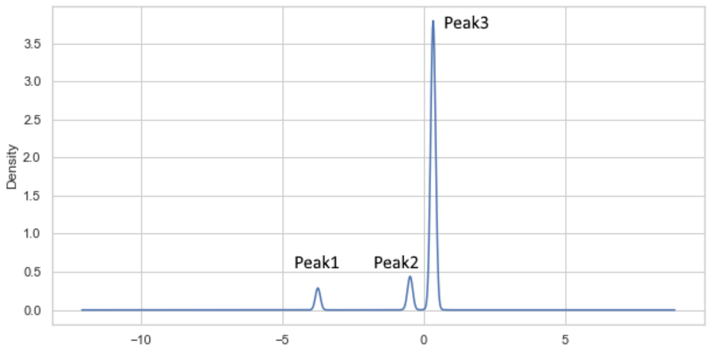
Every other-time interval, not production time or error time, has been removed and this leads to gaps in time in the dataset. Examining the extreme outliers reveals that they occur near the borders of intervals at which such a gap appears. This leads to the conclusion that an extreme change of values is happening close to the interval borders that may have not been correctly captured.

Handling of outliers is needed to lessen the unwanted bias introduced by these uncommon artefacts. Deletion of the outliers would completely conceal the appearance of an unusual value. Simply replacing these measurements by the value of the mean does not account for the temporal dependency. The time series are dynamic and include characteristic behaviour such as the shift of a sensor value over time, see for an example Fig. 3.7. Therefore other common approaches for outlier handling like the 3σ rule or interquartile range based approaches can not directly be applied, either.

The shift of sensor measurements in time is reflected in the probability distribution, see Fig. 3.7b. The location of the peaks correspond to the means of the different shifted intervals. Applying a common approach for outlier detection directly to the



(a) Lineplot of a sensor than contains shifts of measurements over time.



(b) Schematic probability density plot of the sensor measurements. The position of a peak corresponds to the mean of a shifted interval. The height of a peak correlates to the length of intervals.

Fig. 3.7: Dynamic time series have specific properties, such as shifts in time.

example in Fig. 3.7b would detect much more outliers in the left peak. This does not compare well to reality as the values can not be assumed as normally distributed.

As mentioned before the goal here during outlier manipulation is to influence only the extreme outliers, which in this case are global outliers. Based on the widely used 3σ rule outliers are within outermost 0.3 % values of the normal distribution. This implies that the extreme outliers in the analysed sensor data are found within that 0.3 %.

The general idea is to find the extreme global outliers within the distribution of values in the lower and upper tail of the distribution. A modified approach includes that each of the sensors is first analysed for the presence of outliers. If none are present nothing happens and otherwise the extreme outliers are manipulated in the following described way.

Here the case for the upper tail is described, while the case for the lower tail works analogously. At first it is checked if outliers occur at the upper tail considering the entire set of values x by using the quartile based method. This method is the same as applied in boxplots to identify outliers. The first quartile is Q_1 (=25 $\%$-$quantile$), the third quartile is Q_3 (=75 $\%$-$quantile$) and the interquartile range is $IQR = Q_3 - Q_1$. An value $y_j(t_i)$ is considered an outlier $y_{outl}(t_i)$ if the following is fulfilled:

$$y_j(t_i) \in \{x \mid \{x > Q_3 + 1.5 \cdot IQR\}\} \rightarrow y_{outl}(t_i)$$

If an outlier $y_{outl}(t_i)$ exists at the upper tail then the uppermost 0.15 % values of the upper tail x_{tail} are used to determine the extreme outliers by applying the quartile range approach again:

$$y_j(t_i) \in \{x_{tail} \mid \{x_{tail} > Q_{3,tail} + 1.5 \cdot IQR_{tail}\}\} \rightarrow y_{outl,tail}(t_i)$$

Outliers $y_{outl,tail}(t_i)$ are then not removed as they still contribute information. The outlier values $y_{outl,tail}(t_i)$ are scaled towards the whisker $a = Q_{3,tail} + 1.5 \cdot IQR_{tail}$ that indicates outlier values:

$$y_{outl,tail,new}(t_i) = a + log_{10}(|y_{outl,tail}(t_i) - a| + 1) \; with \; a = Q_{3,tail} + 1.5 \cdot IQR_{tail}$$

In case of outliers at the lower tail it holds analogously:

$$y_{outl,tail,new}(t_i) = b - log_{10}(|y_{outl,tail}(t_i) - b| + 1) \; with \; b = Q_{1,tail} - 1.5 \cdot IQR_{tail}$$

This procedure allows to find global extreme outliers and reduces the bias of the extreme outliers and it keeps the occurring temporal difference of the extreme values, that can be incorporated into the learning of the algorithms. Compared to the normal standard approach it affects overall less values and takes the distribution of the values at the tails itself into account to determine the outliers.

3.3.6 Noise Removal

Data that originates from sensors is likely to be contaminated with high frequency noise [Xu+15] and noise removal may play an important role to improve the overall quality of the data. Filters are applied to the data before analysis with the aim to preserve useful information.

During filtering, effects such as information loss and the introduction of lag can occur and therefore their exact tuning is crucial. In case of a lossy filter the result is a more compressed representation of the time series. While it is useful for fast distance computations [Agg15] it is not in the case of outlier detection. Once again, it depends on the questions of interest how the data should be processed. Examples for noise removal are also found in the traditional field of signal processing, like digital filters or wavelet thresholding [EA12].

Noise removal is done if the values of the sensors contain high frequency noise. The sensors of the analysed dataset are only measured periodically in 30 second intervals. Therefore the task of finding high frequency noise has not been carried out as the sampling rate of the sensor is low.

3.4 Analytical Processing

Data Mining is categorized into the different disciplines such as classification, clustering, outlier detection etc. In the case of quality determination labelled information about the quality of the final product is available. Classification algorithms make use of these labels and therefore this discipline is chosen. The first subsection explains the selection of potentially useable classification algorithms in general and the second subsection the importance to choose well suited scoring functions, that are later used for evaluating the classification performance in Ch. 4.

3.4.1 Data Mining in Manufacturing

The No Free Lunch theorem states that there is no "master-algorithm" that is the best for all possible problems ever encounterable. This is due to the fact that there are infinitely many possible problems and over that space one single algorithm can not perform better than all other problems [Dud+12]. In essence, if an algorithm is shown to be superior (if compared to the set of all possible other algorithms) this is only due to the nature of the problem and the distribution of the data. Therefore it is a reasonable approach to evaluate the performance of different algorithms to determine which algorithm is the best suited for the analysed dataset. In other words "the best fitting algorithm has to be found in testing various ones in a realistic environment"[Wue+16] as each problem is unique in its own way.

An overview of popular methods used for data mining in the process industry is given by [Ge+17] and Tab. 3.2 shows an excerpt of methods used for fault classification, quality prediction or dimensionality reduction.

The selection of principally suitable classification algorithms is based on the recent overview of [Ge+17] and combined with the expert knowledge of [Rom19]. Furthermore, regarding the already available algorithms with stable implementations in major frameworks leads to the decision for the following specific algorithms. The

Tab. 3.2: Excerpt of popular machine learning methods in the process industry [Ge+17].

Method Abbreviation	Description	(Un)Supervised
PCA [Jol02]	Principal Component Analysis	Unsupervised
ICA [JH91]	Independent Component Analysis	Unsupervised
SVM [Bur98]	Support Vector Machine	Supervised
NN [Das91]	Nearest Neighbour	Supervised
RF [Bre01]	Random Forest	Supervised

selection of algorithms also considers the explainability of the models and therefore artificial neural networks are not considered during algorithms selection.

For dimension reduction, principal component analysis (PCA) and independent component analysis (ICA) were chosen. Gaussian naive bayes [Zha04] (GNB), k nearest neighbour (KNN), logistic regression [HL00] (LR), linear support vector machine (SVM) and random forest (RF) are used as classification algorithms. Each classification algorithm is optionally combined with a dimension reduction method. The chosen methods are evaluated in Ch. 4 for applicability on the analysed dataset.

3.4.2 Scoring functions for Classification

As mentioned earlier, a wider range of algorithms is evaluated and therefore a performance measure for comparison of these algorithms is needed. Dependent on the type of classification task, there are different scoring functions for evaluating and comparing the algorithms available. Following [SL09] classification tasks are either of the binary, multi-class, multi-labelled or hierarchical type and the performance measures are categorized analogously. Often the definition of a scoring function is based on the abbreviations of the confusion matrix as shown in Fig. 3.3.

Tab. 3.3: Confusion matrix, analogously to [SL09].

Data Class	Classified as Positive	Classified as Negative
Positive	True Positive (TP)	False Negative (FN)
Negative	False Positive (FP)	True Negative (TN)

Furthermore, the choice of a particular scoring function depends on the properties of the data and the goal of the analysis. Properties of the dataset such as class imbalance (found during exploration in Ch. 3.3.1) need to be considered. In binary classification measures like accuracy, precision, recall, etc. are used [SL09]. Moreover, the Matthews correlation coefficient [Mat75] is a popular alternative method that can be directly calculated from the confusion matrix and takes class imbalance into account [Tha18].

In general, scoring functions help to decide if the algorithm is suitable for the dataset. Additionally, the specific choice of the scoring function must take the question of the application domain into account. Depending on it (the specific question) the importance and interpretability of different scoring functions varies significantly.

The goal of analysis in this thesis is to find a machine learning model that assists or complements the existing routines for error detection that are currently implemented. Therefore $precision = TP/(TP + FP)$ is of utmost importance as the model should only trigger an alarm during production if bad quality is detected. The decision how to handle this alarm needs the consideration of a working staff member and this decision time itself is costly as well as the downtime induced through the halt of the production line. Too many false alarms additionally lessen the trust of the working staff in the future decisions of the model.

Nevertheless it is advisable to consider more than one scoring function during the evaluation of the chosen algorithm. The precision of a model may be very good, but as this measure only considers the true positives fraction of all positives it does only give insight about the positive class label and neglects the negative class label entirely. A good complementation to the precision score is the recall score $recall = TP/(TP + FN)$, that allows to determine the fraction of actually found samples. Precision and recall can be visualized in the precision recall curve analogously to that of the receiver operating characteristic (ROC).

Experiments 4

At first the general model parameter selection and data subset selection are explained. Then the algorithms are explored on the subset of the data to determine what kind of algorithms are suitable for the problem at all, which is done for the corresponding cases of the state based and time series based partitioning approach. After that the best selected algorithms are determined and trained on the entire dataset to evaluate their final performance. Lastly, the two most precise algorithms are combined to see if further enhancing of the performance can be achieved.

4.1 Model Parameter Selection

To determine the parameters of a model, the dataset is usually divided into a training, a validation and a test dataset. The training dataset is used to fit the parameters of the model onto the data. The validation dataset is used for tuning to the parameters and the unseen test dataset determines the final score of the model [Dud+12].

It is common practice to use the validation dataset in conjunction with cross validation as it allows a more robust evaluation of the model [Van16]. During model selection, there is always a tradeoff between overfitting and underfitting a model onto the data. More complex models tend to learn the data better but have the drawback of overfitting more easily, which implies low bias and high variance (Fig. 4.1).

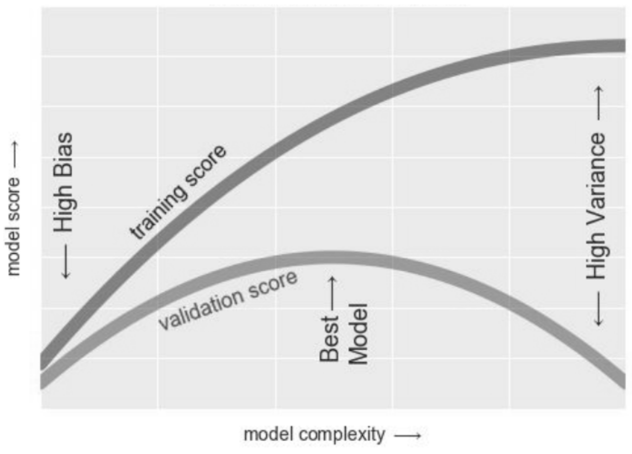


Fig. 4.1: Schematic validation curve [Van16].

Using cross validation during parameter determination allows to find a generalizing model – a not overfitting model. In more detail, the score during parameter determination by cross validation is similar to the final score on the test dataset, which means a model with the ability to generalize is found.

This tradeoff in model complexity is also called bias-variance tradeoff [Van16] or bias-variance dilemma [Dud+12]. Fig. 4.1 displays a schematic validation curve that captures the schematic relation between the training and validation score.

4.2 Data Subset Selection

An exhaustive grid search over each of the algorithm combinations and their parameters for the entire dataset is simply not feasible due to memory usage limitations on the exploring machine. During the algorithms exploration (Ch. 4.3) it is not yet the goal to train the algorithms on the entire dataset. The aim of the algorithm exploration is to recommend an algorithm type that outperforms the others. Based on that recommendation further more detailed analysis with this algorithm can be done. Therefore it is reasonable to only use a subset during exploration.

The dataset originally contains 400.000 samples with 100 columns which equals 40 million cells. For the state based approach (case one and case two in Tab. 4.1) a subset of data is generated by a stratified shuffle split [sci]. It retains the class imbalance and the shuffling allows to make the training more robust against the time dimension. The final dataset size for exploration on the state based partitioning is 40.000 samples large.

For the feature based partitioning approach the goal is to retain the 40 million cells that are present in the original dataset. Heuristics determine that a window size with five minutes and a shift of one minute in the time direction is good fit. This results in 40.000 windows and for each of the 100 columns another ten time series features are calculated. This again results in a total of 40 million cells. In case of the time series based partitioning approach (case three and case four in Tab. 4.1) the size of the subset for exploration is analogously and so is 4.000 samples large.

4.3 Algorithm Exploration

Various techniques for exploring the parameters of the algorithms exist. These include exhaustive grid search, randomized parameter search or auto machine learning approaches and each of them comes with high computational cost. A large variety of parameter combinations with internal cross validation are considered for each the different analysed algorithms.

In the following the algorithms are explored for the state based and time series based partitioning approach on their respective subsets. For each evaluated algorithm the score is determined by five times cross validation and for each algorithm type 100 different parameters settings of the algorithms are evaluated. For each of the cases (one to four in Tab. 4.1) the dataset is fitted 2500 times.

Tab. 4.1: Cases for algorithm exploration.

Case	Approach	Scoring Function
1	state based	precision
2	state based	recall
3	time series based	precision
4	time series based	recall

4.3.1 State based

The following pages discuss the results of the exploration of the state based partitioning approach. They are rated with the scoring functions precision and recall respectively (case one and case two in Tab. 4.1).

State based: Precision

In Fig. 4.2 the result for the exploration of the different algorithms with the precision score is seen. At first sight random forest (RF) and k nearest neighbour (KNN) are the algorithms types with the best precision. Their median values on the score precision is higher than the others. Also the interval between the 25 %- and 75 %-quantile is the smallest.

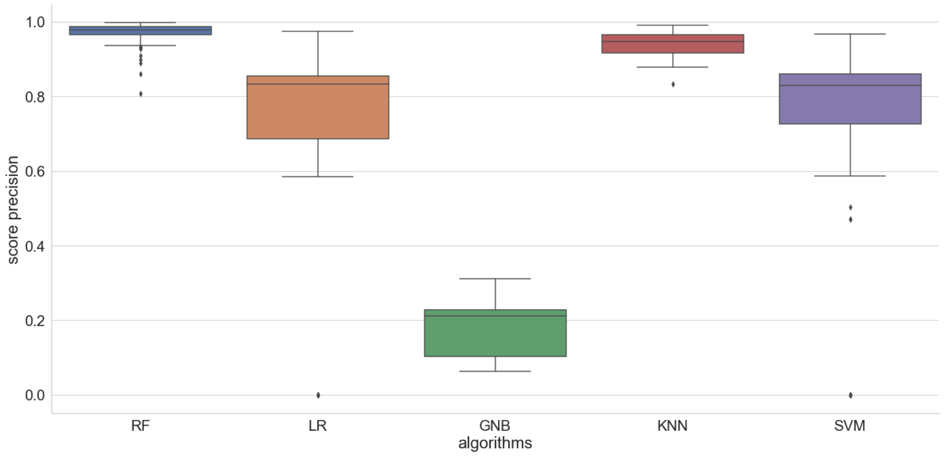


Fig. 4.2: Algorithm comparison with score precision for the state based approach (pipelines optionally use PCA/ICA).

Next is the table Tab. 4.2, that shows the characteristic values for each of the boxplots seen in Fig. 4.2 before. It shows that the RF and the KNN algorithm types have an maximum score of precision above 99 %.

Tab. 4.2: Algorithm comparison with score precision for the state based approach (pipelines optionally use PCA/ICA) by general statistical measures.

	RF	LR	GNB	KNN	SVM
mean	0.9705	0.6917	0.1736	0.9398	0.7156
min	0.8076	0.0000	0.0632	0.8340	0.0000
25%	0.9662	0.6860	0.1021	0.9161	0.7263
median	0.9790	0.8326	0.2112	0.9478	0.8287
75%	0.9870	0.8554	0.2278	0.9661	0.8608
max	0.9977	0.9742	0.3110	0.9916	0.9664

State based: Recall

In Fig. 4.3 the result for the exploration of the different algorithms with the recall score is seen. Here the median of the RF algorithm type is the highest again. The KNN is pushed back at the third position by the gaussian naive bayes (GNB).

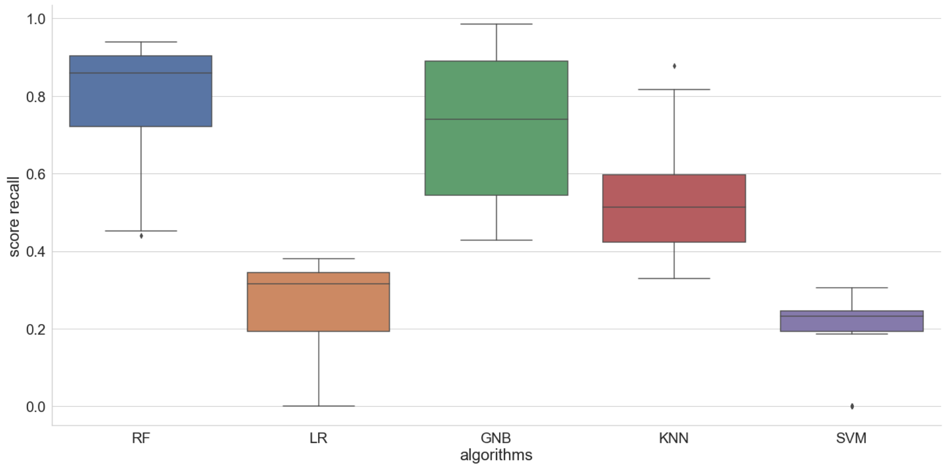


Fig. 4.3: Algorithm comparison with score recall for the state based approach (pipelines optionally use PCA/ICA).

Next is Tab. 4.3, that shows the characteristic values for each of the boxplots seen in Fig. 4.3 before in more detail. The maximum score is achieved by a GNB algorithm but the median of the RF type beats the median of the GNB type by 10 %.

Tab. 4.3: Algorithm comparison with score recall for the state based approach (pipelines optionally use PCA/ICA) by general statistical measures.

	RF	LR	GNB	KNN	SVM
mean	0.8019	0.2497	0.7353	0.5221	0.2053
min	0.4397	0.0000	0.4276	0.3300	0.0000
25%	0.7204	0.1924	0.5439	0.4232	0.1924
median	0.8586	0.3152	0.7404	0.5132	0.2316
75%	0.9032	0.3443	0.8897	0.5965	0.2462
max	0.9397	0.3805	0.9846	0.8772	0.3059

4.3.2 Time Series based

The exploration of time series features based algorithms is again carried out with the scoring function precision and recall (case three and case four in Tab. 4.1).

Time Series based: Precision

In Fig. 4.4 the result for the exploration of the different algorithms with the precision score is seen. The best scores are achieved by the RF algorithm type followed by the KNN algorithm type.

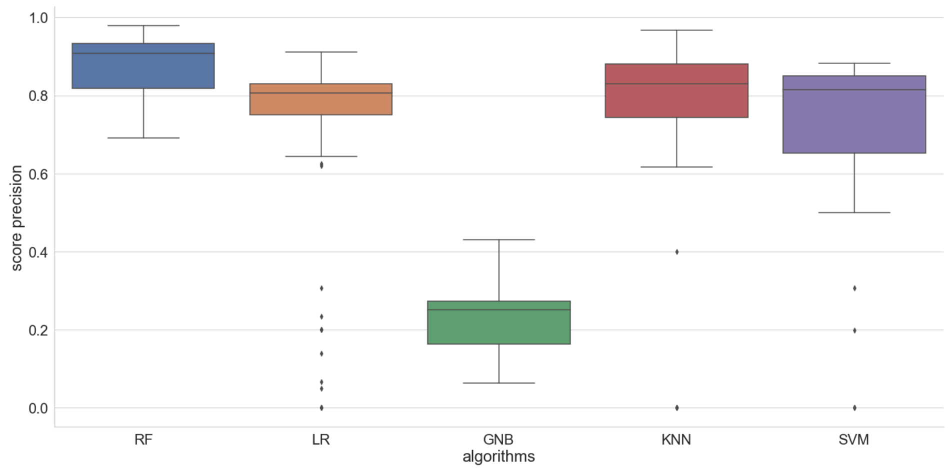


Fig. 4.4: Algorithm comparison with score precision for the time series based partitioning approach (pipelines optionally use PCA/ICA).

Next is the table Tab. 4.4, that shows the characteristic values for each of the boxplots. It shows that the maximum of the RF and KNN algorithm types on precision is 98 % and 97 % respectively. The median of RF type beats the one of KNN by 7 %.

Tab. 4.4: Algorithm comparison with score precision for the time series based partitioning approach (pipelines optionally use PCA/ICA) by general statistical measures.

	RF	LR	GNB	KNN	SVM
mean	0.8753	0.7006	0.2277	0.7705	0.7284
min	0.6901	0.0000	0.0640	0.0000	0.0000
25%	0.8179	0.7503	0.1631	0.7440	0.6515
median	0.9078	0.8060	0.2517	0.8300	0.8146
75%	0.9322	0.8301	0.2739	0.8797	0.8501
max	0.9780	0.9106	0.4297	0.9667	0.8816

Time Series based: Recall

In Fig. 4.5 the result for the exploration of the different algorithms with the recall score is seen. The best median value is from the GNB, followed by the RF.

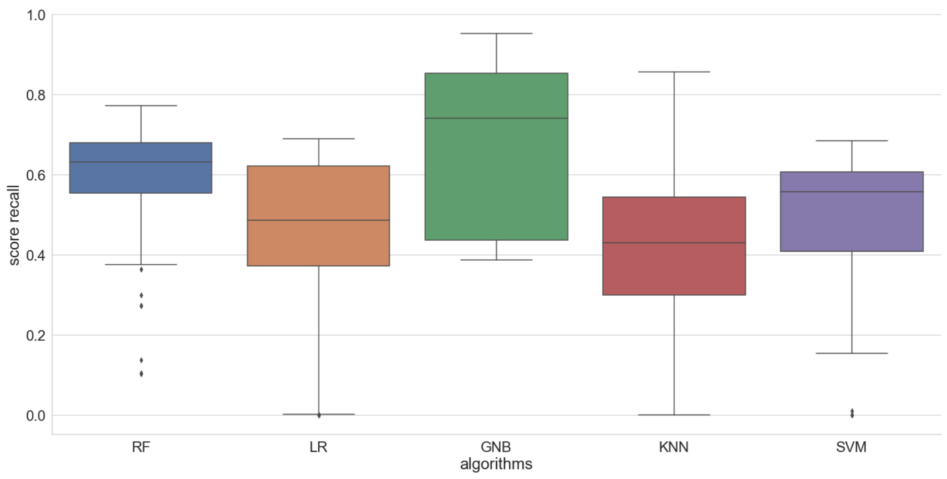


Fig. 4.5: Algorithm comparison with score recall for the time series based partitioning approach (pipelines optionally use PCA/ICA).

Next is the table Tab. 4.5, that shows the characteristic values for each of the boxplots seen in Fig. 4.5 before in more detail again. The median of the GNB algorithm type is better by 10 % compared to the RF algorithm type.

Tab. 4.5: Algorithm comparison with score recall for the time series based partitioning approach (pipelines optionally use PCA/ICA) by general statistical measures.

	RF	LR	GNB	KNN	SVM
mean	0.5905	0.4619	0.6618	0.4186	0.4919
min	0.1025	0.0000	0.3871	0.0000	0.0000
25%	0.5551	0.3734	0.4368	0.2999	0.4087
median	0.6326	0.4860	0.7404	0.4310	0.5578
75%	0.6808	0.6217	0.8537	0.5443	0.6077
max	0.7733	0.6906	0.9530	0.8565	0.6850

4.4 Best Algorithms

In this chapter the best algorithms of the state based and time series based partitioning approach are inferred and evaluated in more detail. During selection of the best algorithm precision is of utmost importance to minimize the amount of false alarms as demanded by the initial domain task. Nevertheless a good precision score alone is not enough. There is no use in selecting an algorithm that perfectly predicts error time but only selects a very small percentage of the overall error time. The first deciding property is to have a precision near the best possible. The second one is to have reasonably high recall score.

A recommendation for the best two algorithm therefore can be inferred by including both scores. Each of the best algorithms is then scored again in relation to precision and recall on the entire dataset. The determination of the 95 %-p interval was done by a ten times cross validation. The final score of the algorithm is based on the unseen test dataset.

Tab. 4.6: Evaluation cases for the two best algorithms.

Case	Approach	Scoring Function
5	Best of state based	precision
6	Best of state based	recall
7	Best of time series based	precision
8	Best of time series based	recall

Best of State based

The best algorithm for the state based approach is now trained and evaluated on the entire dataset for precision and recall (case five and case six in Tab. 4.6). RF algorithms outperform the other algorithm types in the state based approach, their median is the highest and the interquartile range the lowest in the exploration (Ch. 4.3.1). Therefore, many high scoring possible choices of RF algorithms are available and they will be the basis to select the best algorithm. The best algorithms for the state based approach are shown in Tab. 4.7. They are ordered by descending precision which is the most important criteria.

The additional information whether PCA or ICA is used is not relevant now, but it is during the discussion later. Compared to RF0 and RF1 the RF2 only has 0.28 % less precision but the recall falls above the 25 %-quantile. Therefore the best algorithm of choice is RF2.

Tab. 4.7: The best ten RF algorithms of the state based approach sorted by precision.

	RF_precision	RF_recall	pca_ica_used
RF0	0.9977	0.4397	True
RF1	0.9951	0.5482	True
RF2	0.9949	0.7494	False
RF3	0.9944	0.6749	False
RF4	0.9920	0.8838	False
RF5	0.9918	0.8722	False
RF6	0.9911	0.4885	True
RF7	0.9910	0.9024	False
RF8	0.9899	0.8657	False
RF9	0.9895	0.6201	True

The final evaluation for the chosen algorithm is done on the entire dataset. The cross validation results for precision is 99.83 % ±0.17 % expressed by the 95 %-p interval which is equivalent to [99.66, 100] %. For the recall the 95 %-p interval is 98.83 % ±0.63 %, equalling [98.20, 99.46] %. The final score of the best algorithm on the test data for precision is 99.97 % and for recall 98.99 %. Both final scores fall into their respective intervals that indicate that the algorithm does not overfit.

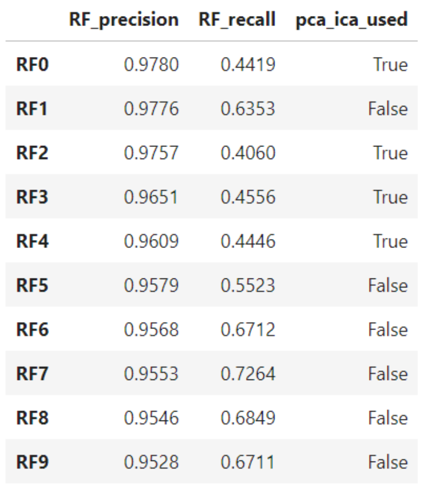
Best of Time Series based

The best algorithm for the time series based approach is now trained and evaluated on the entire dataset for precision and recall (case seven and case eight in Tab. 4.6). The selection of the best algorithm in the of time series based case is analogous to that of the state based approach. RF algorithms outperform the other algorithm types in the time series based approach as well, their median score on precision is the highest (Ch. 4.3.2). Again many high scoring possible choices of random forests are available. The best algorithms for the time series based approach are shown in Tab. 4.8. They are ordered by descending precision which is the most important criteria.

The additional information whether PCA or ICA is used is not relevant now, but it is during the discussion later. Compared to RF0 the difference in precision is neglectable for RF1, but the recall again falls above the 25 %-quantile. The best algorithm of choice is therefore RF1.

The final evaluation for the chosen algorithm is done on the entire dataset. The cross validation results for precision is 99.20 % ±1.74 % expressed by the 95 %-p interval

Tab. 4.8: The best ten RF algorithms of the time series features based approach sorted by precision.



	RF_precision	RF_recall	pca_ica_used
RF0	0.9780	0.4419	True
RF1	0.9776	0.6353	False
RF2	0.9757	0.4060	True
RF3	0.9651	0.4556	True
RF4	0.9609	0.4446	True
RF5	0.9579	0.5523	False
RF6	0.9568	0.6712	False
RF7	0.9553	0.7264	False
RF8	0.9546	0.6849	False
RF9	0.9528	0.6711	False

which is equivalent to [97.46, 100] %. For the recall the 95%-p interval is 86.45 % $\pm$4.19 %, equalling [82.26, 90.64] %. The final score of the best algorithm on the test data for precision is 99.35 % and for recall 84.53 %. Both final scores fall into their respective intervals that indicate that the algorithm does not overfit.

4.5 Combination of Algorithms

The approaches for finding a good algorithm are either state based or time series based and only the latter one incorporates the time dependency of the time series. In [Bag+15] it is shown that a combination of data representation methods improves the performance on time series classification problems. This basic idea behind, an combination of algorithms, is picked up on and done with one algorithm of each partitioning approach.

The general idealized concept behind the combination is that two perfect precise algorithms exist. They do have a small recall score and the respective set of recalled samples does not overlap. The precision score of the two combined algorithms is still perfect, but the new recall is $sum(recall_1,\ recall_2)$.

At first each most precise algorithm RF0 of the state based and time series based approach is trained on the entire dataset. Their individual and combined scores for precision and recall are calculated on the intersection of common timestamps in the validation datasets. For each sample of the intersection a predicted value of the algorithm is available and the combination by the logical OR is possible.

In reality an acceptable improvement in performance will be hard to achieve as a perfect precision is rare and the recalled sets may overlap. There will always be a tradeoff between precision loss and recall gain. Nevertheless this combination is of interest as it may lead to promising results – an acceptable loss in precision with a considerable recall gain. The evaluation of the combined algorithm is done on precision and recall again. The cross validate scores of the different algorithms are shown in Tab. 4.9.

Tab. 4.9: Comparison of the combined algorithms RF0 of the state based and time series based approach.

Scoring Function	$RF0_{state}$ [%]	$RF0_{tsfeatures}$ [%]	Combined [%]
Precision	99.85 ± 0.46	97.87 ± 1.50	97.94 ± 1.52
Recall	98.62 ± 0.84	89.11 ± 2.23	99.00 ± 0.94

The scores during exploration of $RF0_{state,\ ex}$ in the state based approach are 99.77 % for precision and 43.97 % for recall (Tab. 4.7). In the time series based approach the scores of $RF0_{tsfeatures,\ ex}$ during exploration are 97.80 % for precision and 44.19 % for recall (Tab. 4.8). In both cases the training the algorithms on more data leads to a minor improvement in precision compared to the near doubling of the recall score (Tab. 4.9).

The remarkable improvement of the recall score in this case may lead to the idea that training the algorithm on more samples leads to an improvement in recall. This is good if the algorithms are evaluated alone but contradicts the initial idea in evaluating the combination of two algorithms with a low initial recall score. If the overlap of the recalled sets is high then the precision will drop faster towards the minimum precision of the two algorithms and the combined recall gain is lower. The same can be seen in the example of Tab. 4.9. Because of the high recall scores of the individual RF0 algorithms, the precision is nearly the same as that of the time series based approach and at the same time the recall only improves by around 0.4 %.

Notes on frameworks used during Implementation

The mainly used packages are scikit learn [Ped+11] and pandas [Wes] which both are de facto standard libraries for data science in the Python language. The TPOT package [Ols+16a; Ols+16b] is used during exploration and the TSFRESH package [CB16; Chr+18] for automatic time series feature extraction.

Discussion and Conclusion

5

At first the core question to determine the quality of articles in a manufacturing process is taken up. Then the results are discussed from the perspective of the data mining domain.

Quality in Manufacturing

The continuous improvement of quality control for the manufacturing processes is of central importance in industrial companies. Therefore data mining based methods can be utilised to handle complex processes that are challenging to analyse by conventional methods. The goal to find a precisely working algorithm to determine the quality of the final product is met. For an representative article the lack of quality can be detected with precision of 99.83 % ($\pm$0.17 %) and a recall rate of 98.83 % ($\pm$0.63 %). The used methodology to determine the best performing algorithm can analogously be adapted to other articles and other production lines as well.

Data Mining Domain

The basis for selecting the best working algorithms is the exploration of different popular algorithms on a subset of the available data. The two data partitioning approaches are the state based and the time series based one and for each of them the random forests are the best working algorithm types.

The state based approach outperforms the time series based approach. Further improvement of the time series features based approach can be achieved by using a bigger set of relevant time series features. Moreover the focus in selecting the features may be to determine features that are invariant to the characteristics of the sensor data, such as shifts or drifts over time.

The combination of the state based and time series based approach does not yield a significant improvement in the recall accompanied by a reasonable low precision loss. This is due to the already high individual recall scores of the combined algorithms.

Random forest algorithms are a good fit to the data of the analysed manufacturing process. In both approaches they dominate over the other algorithm types. One reason may be that they are relatively robust to outliers [Bre01]. Another desirable advantage of random forests is the ability to get "internal estimates of error, strength, correlation and variable importance"[Bre01] as it gives further clues to understand the data even more. Random forests do belong to the class of ensemble methods that combine multiple decision trees with bootstrap aggregation (bagging). The bagging approach may be used together with the other algorithms also to do a comparison solely on ensemble methods for each of the algorithm types.

The exploration of the algorithms is carried out on a subset of the available data for the representative article. The subset size of 10 % of the original dataset is chosen due to the restrictions for computing time and memory usage limitations on the exploring machine. Using more data during the exploration leads to more stable results when the performance of the algorithm is compared to that on the entire dataset.

Furthermore, it is optional during the exploration whether the algorithm pipelines include a dimensionality reduction step (PCA or ICA). During comparison of the ten most precise algorithms of each approach lead to the conclusion that the dimension reduction step leads to a lower possible recall rate. The application of an initial dimensionality reduction step implies an irreversible information loss. The following applied machine learning algorithm must learn with less information available and this therefore may result in an overall less achievable recall.

During preprocessing the choice of methods follows the commonly used approaches in data mining. Their respective properties are considered regarding the dataset at hand and the best fitting one is chosen. A detailed evaluation of the methods in each step may lead to a more specific solution, which is more robust and results in better final scores of the algorithms.

Working with the dataset of an industrial company involves the challenge to process structured and semi-structured data. The ETL processing of the data is very time consuming but needed to generate a first raw data file which can be used as input for the data mining methods. Only after that the preprocessing can be applied on the data. The amount of time needed for ETL and preprocessing should not be underestimated.

The availability of class labels for error time (poor quality) and production time (good quality) allows to make use of the classification discipline in data mining. Nevertheless other approaches like motif discovery or anomaly detection can be utilised to solve the initial question to determine the article quality. Moreover, the theoretical background in this thesis intends to motivate for using other possible approaches to solve the initial problem. Various combinations of a discipline of time series data mining with a representation and distance approach are possible.

Main Conclusions

The goal to design a precisely working data mining based method to further improve the quality control in a plastic profile extrusion process is met. An appropriate algorithm for determining the quality of a representative article is found and the used approach is analogously transferable onto other articles and other production lines as well. The random forest algorithm type achieves the best results for the analysed representative article. The state based approach outperforms the time series based approach while further improvement of the latter one may be promising as it incorporates the implicit time dependency of the values.

Bibliography

[Agg15] Charu C. Aggarwal. *Data Mining*. Cham: Springer International Publishing, 2015 (cit. on pp. 4, 5, 8, 11, 14, 29, 30, 33).

[Agg17] Charu C. Aggarwal. *Outlier Analysis*. Cham: Springer International Publishing, 2017 (cit. on pp. 14, 31).

[Agh+15] Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, and Teh Ying Wah. „Time-series clustering – A decade review“. In: *Information Systems* 53 (2015), pp. 16–38 (cit. on pp. 9, 10, 12–14).

[Bag+15] Anthony Bagnall, Jason Lines, Jon Hills, and Aaron Bostrom. „Time-Series Classification with COTE: The Collective of Transformation-Based Ensembles“. In: *IEEE Transactions on Knowledge and Data Engineering* 27.9 (2015), pp. 2522–2535 (cit. on p. 46).

[Bag+17] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. „The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances“. In: *Data Mining and Knowledge Discovery* 31.3 (2017), pp. 606–660 (cit. on p. 14).

[Bre01] Leo Breiman. „Random Forests“. In: *Machine Learning* 45.1 (2001), pp. 5–32 (cit. on pp. 35, 50).

[Bru17] Franz J. Brunner. *Japanische Erfolgskonzepte: KAIZEN, KVP, Lean Production Management, Total Productive Maintenance, Shopfloor Management, Toyota Production System, GD3 - Lean Development*. 4., überarbeitete Auflage. Praxisreihe Qualitätswissen. München: Hanser, 2017 (cit. on p. 6).

[Bur98] Christopher J.C. Burges. „A Tutorial on Support Vector Machines for Pattern Recognition“. In: *Data Mining and Knowledge Discovery* 2.2 (1998), pp. 121–167 (cit. on p. 35).

[CB16] Maximilian Christ and Blue Yonder GmbH. *https://tsfresh.readthedocs.io/en/latest/*. 2016 (cit. on p. 47).

[Cha+00] Pete Chapman, Julian Clinton, Randy Kerber, et al. „CRISP-DM 1.0 Step-by-step data mining guide“. In: *The CRISP-DM Consortium* (2000) (cit. on p. 3).

[Chr+18] Maximilian Christ, Nils Braun, Julius Neuffer, and Andreas W. Kempa-Liehr. „Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package)“. In: *Neurocomputing* 307 (2018), pp. 72–77 (cit. on pp. 22, 47).

[Das91] Belur V. Dasarathy, ed. *Nearest neighbor (NN) norms: Nn pattern classification techniques*. Los Alamitos and Washington: IEEE Computer Society Press and IEEE Computer Society Press Tutorial, 1991 (cit. on p. 35).

[Din+08] Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. „Querying and mining of time series data: Experimental Comparison of Representations and Distance Measures“. In: *Proceedings of the VLDB Endowment* 1.2 (2008), pp. 1542–1552 (cit. on pp. 9, 12, 14).

[Dud+12] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. 2. Aufl. s.l.: Wiley-Interscience, 2012 (cit. on pp. 34, 37, 38).

[EA12] Philippe Esling and Carlos Agon. „Time-series data mining“. In: *ACM Computing Surveys* 45.1 (2012), pp. 1–34 (cit. on pp. 8–14, 34).

[Fu11] Tak-chung Fu. „A review on time series data mining“. In: *Engineering Applications of Artificial Intelligence* 24.1 (2011), pp. 164–181 (cit. on pp. 11, 14).

[Ge+17] Zhiqiang Ge, Zhihuan Song, Steven X. Ding, and Biao Huang. „Data Mining and Analytics in the Process Industry: The Role of Machine Learning“. In: *IEEE Access* 5 (2017), pp. 20590–20616 (cit. on pp. 1, 7, 34, 35).

[HL00] David W. Hosmer and Stanley Lemeshow. *Applied logistic regression*. 2. ed. Wiley series in probability and statistics Texts and references section. New York: Wiley, 2000 (cit. on p. 35).

[JH91] Christian Jutten and Jeanny Herault. „Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture“. In: *Signal Processing* 24.1 (1991), pp. 1–10 (cit. on p. 35).

[Jol02] I. T. Jolliffe. *Principal Component Analysis*. Second Edition. Springer Series in Statistics. New York, NY: Springer-Verlag New York Inc, 2002 (cit. on p. 35).

[KK03] Eamonn Keogh and Shruti Kasetty. „On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration“. In: *Data Mining and Knowledge Discovery* 7.4 (2003), pp. 349–371 (cit. on pp. 9, 10).

[Kök+11] Gülser Köksal, İnci Batmaz, and Murat Caner Testik. „A review of data mining applications for quality improvement in manufacturing industry“. In: *Expert Systems with Applications* 38.10 (2011), pp. 13448–13467 (cit. on pp. 6, 7).

[Lep+17] Mathieu Lepot, Jean-Baptiste Aubin, and François Clemens. „Interpolation in Time Series: An Introductive Overview of Existing Methods, Their Performance Criteria and Uncertainty Assessment“. In: *Water* 9.10 (2017), p. 796 (cit. on p. 29).

[LR02] Roderick J. A. Little and Donald B. Rubin. *Statistical analysis with missing data*. 2nd ed. Wiley Series in Probability and Statistics. Hoboken: Wiley, 2002 (cit. on p. 29).

[Mat75] B. W. Matthews. „Comparison of the predicted and observed secondary structure of T4 phage lysozyme“. In: *Biochimica et Biophysica Acta (BBA) - Protein Structure* 405.2 (1975), pp. 442–451 (cit. on p. 35).

[Mit10] Theophano Mitsa. *Temporal data mining vs. time series analysis vs. machine learning*. 2010 (cit. on pp. 10, 12–14).

[MK16] Ankita Mangal and Nishant Kumar. „Using big data to enhance the bosch production line performance: A Kaggle challenge“. In: *2016 IEEE International Conference on Big Data (Big Data)*. IEEE, 2016, pp. 2029–2035 (cit. on p. 7).

[MS11] Marina Milanović and Milan Stamenković. „Data mining in time series“. In: *Economic Horizons* 13 (2011), pp. 5–25 (cit. on p. 10).

[NL13] Ryan P. North and David M. Livingstone. „Comparison of linear and cubic spline methods of interpolating lake water column profiles“. In: *Limnology and Oceanography: Methods* 11.4 (2013), pp. 213–224 (cit. on p. 29).

[Ols+16a] Randal S. Olson, Ryan J. Urbanowicz, Peter C. Andrews, et al. „Automating Biomedical Data Science Through Tree-Based Pipeline Optimization“. In: *Applications of Evolutionary Computation*. Ed. by Giovanni Squillero and Paolo Burelli. Vol. 9597. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2016, pp. 123–137 (cit. on p. 47).

[Ols+16b] Randal S. Olson, Nathan Bartley, Ryan J. Urbanowicz, and Jason H. Moore. „Evaluation of a Tree-based Pipeline Optimization Tool for Automating Data Science“. In: *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference - GECCO '16*. Ed. by Frank Neumann, Tobias Friedrich, and Andrew M. Sutton. New York, New York, USA: ACM Press, 2016, pp. 485–492 (cit. on p. 47).

[Pav16] B. Pavlyshenko. „Machine learning, linear and Bayesian models for logistic regression in failure detection problems“. In: *2016 IEEE International Conference on Big Data (Big Data)*. IEEE, 2016, pp. 2046–2050 (cit. on p. 7).

[Ped+11] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, et al. „Scikit-learn: Machine Learning in Python“. In: *Journal of Machine Learning Research (2011)* (2011) (cit. on p. 47).

[Rom19] Sandra Romeis. *Ph.D. in statistics, Senior Data Science Expert: Discussion*. Ed. by Andreas Braun. 2019 (cit. on p. 34).

[Sch+08] Roger G. Schroeder, Kevin Linderman, Charles Liedtke, and Adrian S. Choo. „Six Sigma: Definition and underlying theory⋆“. In: *Journal of Operations Management* 26.4 (2008), pp. 536–554 (cit. on p. 6).

[sci] scikitlearn. *https://scikit-learn.org/stable/documentation.html* (cit. on pp. 28, 38).

[SL09] Marina Sokolova and Guy Lapalme. „A systematic analysis of performance measures for classification tasks“. In: *Information Processing & Management* 45.4 (2009), pp. 427–437 (cit. on p. 35).

[Tao+18] Fei Tao, Qinglin Qi, Ang Liu, and Andrew Kusiak. „Data-driven smart manufacturing“. In: *Journal of Manufacturing Systems* 48 (2018), pp. 157–169 (cit. on pp. 6, 7).

[Tha18] Alaa Tharwat. „Classification assessment methods“. In: *Applied Computing and Informatics* (2018) (cit. on p. 35).

[Van16] Jacob T. Vanderplas. *Python data science handbook: Essential tools for working with data*. First edition. Sebastopol, CA: O'Reilly Media Inc, 2016 (cit. on pp. 37, 38).

[Wan+13] Xiaoyue Wang, Abdullah Mueen, Hui Ding, et al. „Experimental comparison of representation methods and distance measures for time series data“. In: *Data Mining and Knowledge Discovery* 26.2 (2013), pp. 275–309 (cit. on pp. 8, 9, 12, 14).

[Wue+14] Thorsten Wuest, Christopher Irgens, and Klaus-Dieter Thoben. „An approach to monitoring quality in manufacturing using supervised machine learning on product state data“. In: *Journal of Intelligent Manufacturing* 25.5 (2014), pp. 1167–1180 (cit. on pp. 7, 18).

[Wue+16] Thorsten Wuest, Daniel Weimer, Christopher Irgens, and Klaus-Dieter Thoben. „Machine learning in manufacturing: advantages, challenges, and applications“. In: *Production & Manufacturing Research* 4.1 (2016), pp. 23–45 (cit. on pp. 1, 6, 7, 34).

[Wue15] Thorsten Wuest. *Identifying product and process state drivers in manufacturing systems using supervised machine learning*. Springer theses. Cham: Springer, 2015 (cit. on p. 1).

[Xu+15] Shu Xu, Bo Lu, Michael Baldea, et al. „Data cleaning in the process industries“. In: *Reviews in Chemical Engineering* 31.5 (2015), p. 93 (cit. on pp. 14, 29, 31, 33).

[YH03] Kai Yang and Basem El Haik. *Design for Six Sigma: A roadmap for product development*. New York, NY: McGraw-Hill, 2003 (cit. on p. 6).

[Zha+16] Darui Zhang, Bin Xu, and Jasmine Wood. „Predict failures in production lines: A two-stage approach with clustering and supervised learning“. In: *2016 IEEE International Conference on Big Data (Big Data)*. IEEE, 2016, pp. 2070–2074 (cit. on p. 7).

[Zha04] Harry Zhang. „The Optimality of Naive Bayes“. In: *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference* 2 (2004) (cit. on p. 35).

[Rob] Robert Bosch GmbH. *Bosch production line performance challenge* (cit. on p. 7).

[Wes] Wes McKinney. „Data Structures for Statistical Computing in Python“. In: vol. Proceedings of the 9th Python in science conference, pp. 51–56 (cit. on p. 47).

List of Figures

Declaration

Hiermit versichere ich, Andreas Braun (Matrikelnummer 1200197), dass ich die von mir vorgelegte Arbeit *Data Mining in Industrial Processes: Evaluation of different machine learning models for product quality prediction* selbstständig verfasst, keine anderen als die angegebenen Quelle und Hilfsmittel verwendet und die Arbeit nicht bereits zur Erlangung eines akademischen Grades eingereicht habe.

Bayreuth, 28.03.2019

Andreas Braun