Using Machine Learning to Reduce the False Call Problem in Electronics Manufacturing

Steeghs, R.H.J.

Award date:
2021

Eindhoven University of Technology

Industrial Engineering & Innovation Sciences
Operations Management & Logistics

Using Machine Learning to Reduce the False Call Problem in Electronics Manufacturing

Master Thesis Report

Author:

R.H.J. Steeghs (Rik) 0961490

Supervisors:

Dr. M. Firat (Murat)	TU/e
Dr. T. Martagan (Tugce)	TU/e
D. van Driel (Dirk)	AME
A. Schröder (Arjen)	AME

October 3, 2021

Abstract

Electronics manufacturing faces great challenges not only in product design, but also in production quality due to the complex sequential processes. In an enterprise with high customization and make-to-order electronics production, automated optical inspection systems are commonly used to preserve the highest product quality. However, high quality control standards during this automated inspection preventing error slip invoke false error calls on approximately 60% of the products produced. Each false flag requires a manual inspection by a machine operator, reducing the production line efficiency and increasing the error slip probability in case of excessive amounts of false flags. This research utilizes product and process data to enhance the automated optical inspection system of a surface mounted device production line. The data serves as input for a machine learning model which can be generalized among different product types, and is particularly developed to handle the class imbalance in manufacturing data well. Results show that the balanced bagging random forest classifier performs best for the problem at hand, reducing the number of manual checks by approximately 50%. Implementing the model as part of the production line thus improves the line efficiency, and reduces the operator workload which enhances the production quality.

Executive Summary

This thesis proposes a machine learning method to reduce the number of false error flags in the electronics manufacturing environment. Each industrial revolution over the past centuries caused great shifts in the technological paradigm. Mechanization, electrical energy and digital systems were all introduced during these technological leaps. Today, another exceptional technology-push entered industrial practice. Increased digitalization of manufacturing plants and the corresponding increase of data generation and availability are the main drivers of the fourth industrial revolution, also known as Industry 4.0. This thesis is conducted at Applied Micro Electronics (AME), an independent developer and manufacturer of high-quality electronic products in a high-mix, make-to-order production environment driven by specific and fluctuating customer demand. The goal of AME is to create innovative products that meet or even exceed customer expectations. Driven by technology, AME strives for the best solution combining the disciplines of electrical, mechanical, software and industrial engineering. In the scope of this project, machine learning models are implemented in order to improve the production control of AME, contributing to further implementation of the fourth industrial revolution. This research may lay the foundation of a transition to smart manufacturing, which is identified by using data collected by sensors serving as input for 'smart technologies' such as data-driven models, possibly leading to autonomous decision systems. Due to growing data collections and the technology driven mindset, AME provides an adequate environment to push the industrial paradigm to the next level.

The electronics manufacturing plant of the company consist of several work stations, each serving a different purpose in the process. For this study, the surface-mount device (SMD) production line is examined. This choice is twofold, it serves as the main workstation in the electronics manufacturing process and the data availability enables extensive research in the scope of the fourth industrial revolution. During the SMD production, electrical surface-mount components are attached on an empty printed circuit board, using soldering paste, an automated pick and place process and a reflow oven system. The product quality is measured using automated inspection systems, also known as the API (automated paste inspection) and the AOI (automated optical inspection). After applying the paste, the API checks the quality of this paste by measuring its volume, area, height and offset using a laser system. At the end of the process, the quality of the soldering and placement is assessed by the AOI using computer vision. Each component on the printed circuit board assembly is checked individually, using the quality settings set by the process engineers. The default inspection tolerances are based on industry wide standards. Due to the high customization at the manufacturer, it is required to fine tune these tolerances, optimizing the quality control. This tuning is mainly about finding the right balance between error slips and false calls. Error slip is the undesired result of the AOI where the quality of an inspected component is wrongly classified as good. The vast amount of different components causes it to be infeasible to fine tune all the different inspection programs. This and other (external) factors result in false error flags on approximately 60% of the products produced at the SMD production line. The occurrence of many false calls at the automated optical inspection system is a common problem in electronics manufacturing. Whenever only one component on the board does not work as it is intended, it is possible that the whole board does not function. Therefore every error call requires additional attention from an operator. To avoid scrapping or repairing good components each call is manually checked by the operator. Checking all the calls during the production of the printed circuit boards can be a time costly action, leading to decreased line efficiency. Moreover, high false call percentages during production increases the chance that an operator will miss a real defect due to negligence, potentially leading to an increase in the error slip ratio.

The goal of this research is to increase the overall production control and throughput of AME by utilizing (a combination of) data-driven methods, to predict the product quality and possibly investigate relations or dependencies between production process variables and the process outcome. This is done in an online fashion, by using online process data in order to enhance the quality control, which improves the production throughput. Product and process data related to the SMD production line serve as an input for a machine learning model which tries to predict whether an incoming error call concerns a real error or a false call. Current literature related to the topic showed some previous advances regarding false error flag prediction in electronics manufacturing. The papers propose (machine learning) methods to enhance the automated optical inspection, improving the image recognition algorithm or reinforcing the automated

decision with a prediction based on process data. However, the models proposed in the literature are limited to only one product type or use the data of only one sub process, which prevents them from being generalizable over the entire product set. This thesis extends the current literature by using the complete SMD production process data, including sensor data, machine settings and product characteristics for multiple product types. This enhances the generalizabilty of the model thus the overall usefulness from a business perspective. Further extension of business utilization is done by adding explanations of the modeling results in terms of process features, increasing the potential to capture expert knowledge.

Process data in the form of process parameters and sensor data from each of the sub processes (screen-printing, pick and place, reflow oven) is combined with static product and component type data. Data gathering, data cleaning and feature engineering to reduce the data dimension resulted in a data set consisting of approximately 7.4 million instances, with 35 product and process features. Each instance corresponds to an inspection result of a individual component on a specific printed circuit board, with the product features, process parameters, process sensor values and quality inspection results as features for that instance. The quality inspection results target classes are either sufficient, false error flag or real error. False error flags and real errors are distinguished by an operator, as currently each error call requires a manual inspection which result is stored in a database. The target class is highly imbalanced, it contains 36,347 false call instances and only 750 real error instances, the rest of the instances are components with a sufficient quality. Exploratory data analysis lead to several findings useful for modeling. Firstly, reoccurring problems (troublemakers) during production can be associated with either component types and board locations. Secondly, process deviations happen between batches, but also within a specific panel. For instance, when a panel consists of two boards, location A on board 1 can cause more problems than the same location A on board 2. Thirdly, the reflow zones are highly correlated and thus transformed into a lower dimension representing all relevant information. Lastly, it seems that the time interval between two products may affect the problems happening on a product. Therefore this features is created and included when developing the machine learning model.

The imbalanced nature of the target class was addressed with two different methods. First, various sampling techniques (e.g. random under-sampling, random over-sampling, SMOTE Tomek links and balanced bagging) were developed and tested with different machine learning models. Balanced bagging in combination with class weighting and a random forest classifier performed best. The second classifier proposed used the output of multiple autoencoders as a feature engineering method for a support vector machine model. The advantage of this method is that it uses both the good components and the false components majority sets so no information is lost by under-sampling. Both methods, the balanced bagging classifier and the hybrid autoencoder method incorporated a way to handle uncertain predictions, in order to reduce the error slip caused by the model. If the model is not certain enough, it predicts that the error flag still requires a manual check by an operator. After hyperparameter tuning, both methods were compared based on the error slip, precision and manual check ratio. For the problem at hand, the balanced bagging random forest outperformed the ensemble hybrid autoencoder method.

Evaluating the model from a business perspective was done by adding costs to each misclassification. Based on the company's input, the most severe misclassifications are error slips. Thus, the main goal of the hyperparameter tuning was finding a model which minimizes the error slip as much as possible, but also minimizing the number of machine calls which still required a manual check. Based on evaluation on the test set, the proposed model is able to reduce the number of manual inspections by an operator with approximately 50%. Yearly this means a reduction of 183,000 manual inspections on component level. Furthermore, test set results indicate that this saves checking half of the panels, saving approximately 66,500 panels to inspect each year. AME estimates that a manual inspection takes about three seconds, thus saving approximately 153 hours every year, which means three hours each week and somewhat more than half an hour per day. When calculating costs only from an operator perspective, the marginal improvement is only minimal, as the costs of an operator is one euro cent per second. However, the saved time also provides opportunity costs as more products can be produced in this time, which indirectly increases the revenue of the company. Determining the height of these benefits is not trivial as it depends on many external factors. Less false calls also enhance the product quality as the operator will be more attentive if there are less machine calls during a production batch. This potentially reduces the error slip, although currently there are no estimations available regarding these metrics. This makes it infeasible to

do any statements about potential benefits related to quality improvements. Overall, reducing the false calls will improve the working conditions of the operator as less time is consumed by repetitive manual inspection tasks. Lastly, SHAP values were used to analyse the relation between the process and product features and the product quality. It was found that the component package, the printed circuit board length, the time interval between two products, and the screenprinting environment features were the most important features used during prediction. Overall, there was not found a set of features related to a sub process which contributed the most to the model. Potentially the strength of the model lies in the fact that it combines both product, and process features of all the sub processes in the surface-mount device production line.

In order to proceed with the development and implementation of advanced data-driven techniques such as machine learning, it is recommended that AME adjusts the current framework to gather the process data of different sub processes. For this research many different sources were consulted, resulting in missing data and problems when merging the different sources. Developing a centralised database storing all the relevant machine learning information is highly recommended. This eliminates the recurrent challenges of scraping the log files for the relevant information and the dependency on local data files or third parties. Developing and testing a machine learning model will be easier, if it is convenient to extract all relevant data needed for the machine learning model. Furthermore, when the model is implemented the pretrained model makes predictions using the live process data which has just been gathered during the process. To ensure this, the data of a given product must be processed parallel to the production instance. The data set of one product is relatively small thus performing these preprocessing tasks is not computationally heavy. In short, the trained model is stored on a server which is accessed by software at the AOI station. If there are error calls, the preprocessed process data of a given instance is given to the model to predict whether these calls are false or not. After the prediction, the result is shown to the operator and all relevant data is stored in the machine learning database. Future research to extend the current research can have multiple directions. First, the proposed idea of the autoencoder classifier can be further researched and tested on other baseline machine learning problems. Another potential research direction is to add manufacturing system analysis regarding the current line performance to the evaluation of the model in order to generate a more complete overview of the model's business performance. To conclude, expert knowledge in combination with the SHAP values can be used to further improve the production line. Variables which can be controlled such as printing speed or the heating coefficient can be used to further fine tune the process parameters. After validating the model interpretation with domain experts, the adjustable features can be optimized with respect to the quality. This process can be automated with metaheuristics (e.g. simulated annealing) which can be used for global optimization in a large search space such as a manufacturing environment. By automating the adjustment of process variables relative to the production quality, AME can further progress in the paradigm of the fourth industrial revolution.

Contents

List of Figures

List of Tables

Abbreviations and Concepts

Table 1: Table of abbreviations and important concepts

Term	Description
API	Automated Paste Inspection. Inspection of the past applied to the PCB.
AOI	Automated Optical Inspection. Inspection of components attached to PCB.
Batch	Production instance with certain amount of panels.
BI	Business Intelligence system
ERP	Enterprise Resource Planning (SAP)
MES	Manufacturing Executing System. Storing all the manufacturing data.
Panel	Product instance produced at the SMD line, each panel can have multiple boards.
PCB	Printed Circuit Board. A blank board without components.
PCBA	Printed Circiut Board Assembly (or board). PCB with components attached, the final product of the SMD process.
PN	Product Number. Unique number used for products and components.
PO	Production Order. Label added to an order of one specific PN ordered by customer.
PRD	Production workcenter for electronics.
PTH	Pin-through-hole. Components attached by going through the PCB.
RefDes	Reference destination. Specific location on a board, unique per product type.
SN	Serial Number. Label added to a panel produced.
SMD	Surface-mounted-device. Components attached to surface of the board.

1 Introduction

Since the beginning of industrialization, technological leaps have lead to shifts in the industrial paradigm, so called industrial revolutions (Lasi, Fettke, Kemper, Feld, & Hoffmann, 2014). The first industrial revolution was characterized by the use of mechanization. An increasingly and more intensive use of electrical energy lead to the second industrial revolution. When technology enabled the widespread use of digitalization in industries, the third industrial revolution began. Today, another exceptional technology-push entered industrial practice, which heralded the fourth industrial revolution (Lasi et al., 2014). This revolution is also known as Industry 4.0, the term being a reminiscence of software versioning. One of the identifiers of the technology-push related to Industry 4.0 is the increasing digitalization of all manufacturing equipment and manufacturing supporting tools, and the corresponding data generation and availability. This is translated into the use of advanced models to control and analyse the manufacturing process. Applied Micro Electronics (AME) is a developer and manufacturer of electronics and products related to electronics, driven by technology. Quality is one of their strategic pillars and thus of great importance for the company. In the scope of this project, machine learning models will be implemented in order to improve the production control of AME, contributing to the further implementation of the fourth industrial revolution. This research may lay the foundation of a transition to smart manufacturing, which is identified by using data collected by sensors serving as input for 'smart technologies' such as data-driven models, possibly leading to autonomous decision systems (Lasi et al., 2014). AME states that a vast amount of data is available containing possible interrelations between the process and product quality, but that this data is not yet analyzed, hence utilized to improve the production quality. Due to growing data collections and the technology driven mindset, AME provides an adequate environment to push the industrial paradigm to the next level. The remainder of this section will provide a further company introduction, followed by a brief introduction to the field of managing manufacturing quality. Finally, the business question is concisely touched upon.

1.1 Company Introduction

This thesis is conducted at Applied Micro Electronics (AME), an independent developer and manufacturer of high-quality electronic products in a high-mix, make-to-order production environment driven by specific and fluctuating customer demand. Due to the high demand in the market, the company steadily grows approximately 20% every year. Roughly 200,000 panels with printed circuit board assemblies are produced each year, resulting in a yearly production turnover larger than €35 million. The goal of AME is to create innovative products that meet or even exceed customer expectations. Driven by technology, AME strives for the best solution combining the disciplines of electrical, mechanical, software and industrial engineering. The company is responsible for both the design and the manufacturing process. This also means that AME tries to conduct production steps in-house as much as possible. One example is the fact that they design and create their own machines parts, adapted to the customer's demand. With this design of the manufacturing process, the production system of AME is a flexible job shop system. This means that there are production units, so-called (interconnected) workcenters, dedicated to producing certain components or executing certain operations. Every workcenter is flexible by having alternative resources to carry out the operations.

The workstations which belong to the business activities of AME are electronics manufacturing, system assembly, injection moulding, machining, cable and wiring and product cleaning. Electronics manufacturing is the core of the production process, as this is where the printed circuit board assemblies are produced. This workcenter is divided in two production phases surface-mount device (SMD) production and plated through-hole (PTH) production. The former is almost fully automated and the latter requires manual work. When products need special treatment, they are handled at the cleaning workstation, where for instance additional glue is applied. During the injection moulding process liquid plastic is moulded to a certain shape, which can be used as a final product or as a product part. AME also produces the moulds in-house, so they are not dependent on third party suppliers thus providing more flexibility towards customers. Creation of the moulds is done at the machining work station. Cabling and wiring provides the cables and wires for both external customers as internal use. Finally, the sub assemblies or final products are built at the system assembly workstation. Products from different workstations may come together here to form the final product for the customer.

1.2 Quality in manufacturing

Manufacturing quality is a board concept and researched from many different perspectives, with the improvement or control of the product quality at its core. The most well known data-driven method of controlling the quality is by the use of statistical process control methods (Rokach & Hutter, 2012). Machine learning enables deeper analysis of these control charts by analyzing a mixture of charts in order to find deviating patterns (Zhang, Yuan, Wang, & Cheng, 2020; Verron, Li, & Tiplica, 2010). Another common objective in the manufacturing quality literature is creating understanding between the process features and the product quality (Kusiak & Kurasek, 2001). Doing so (either with machine learning or other methods) enables the discovery of root-causes for quality deviations and the capturing of expert knowledge (Tseng, Jothishankar, & Wu, 2004; Du, Lv, & Xi, 2012). High level goals such as achieving an overall high equipment efficiency, and analyzing the causes of non-satisfactory equipment efficiency are also a common objective (Natschlager, Kossak, & Drobics, 2004). The dimension of interest for this research is not the quality of individual products, but the overall production yield (or quality rate) as a whole. T. Tsai (2012) uses a classification model to properly interpret the defect patterns and uncover cause-and-effect relationships between the process parameters and the production yield. Besides improving the efficiency of the complete line, also much research is conducted regarding the analysis of individual products. This ranges from improving the design of the process and product (Linn & Lam, 1998; C. Tsai, Chiu, & Chen, 2005), to predicting rare quality deviations or system failures (Kim & Kang, 2019; Escobar & Morales-Menendez, 2019). Next to the broad range of available knowledge related to manufacturing quality in general, the electronics manufacturing in particular is also highly relevant. The increasing dependency of the global economy on electronics is one of the drivers of this momentum. Quality research regards different topics such as printed circuit board design (Linn & Lam, 1998), process quality analysis (T. Tsai, 2012; Chang, Wei, Chen, & Hsieh, 2019), product quality prediction (Schmitt, Bönig, Borggräfe, Beitinger, & Deuse, 2020) and quality inspection using image recognition (Richter, Streitferdt, & Rozova, 2017). Many of these advances contribute to the new industry paradigm of Industry 4.0, and provide a foundation for a new standard in manufacturing. This thesis builds on this previous quality research and provides further insights in electronics manufacturing quality analysis using data-driven techniques.

1.3 Business Question

Although all of the work centers have interesting challenges related to product and production quality, the SMD production process is the scope of this research project. The choice for this work center is twofold. First, the SMD line is the core of the production plant and is therefore an important chain in the production process of AME. Previous research found that SMD is a bottleneck in production, thus improvement in process efficiency will have direct impact on the efficiency of the company as a whole. Secondly, each sub process of the SMD line is equipped with data gathering tools, leading to much data availability. Thus, solving problems in this process may lead to major benefits from a business perspective and the data availability provides a decent environment for extensive research.

In terms of quality there are a few problems occurring at the surface-mount device production line. When automatically inspecting the products, large amounts of false calls happen as a result of minimizing the error slip. Both the efficiency of the production line and the operator workload suffer due to this issue. It is estimated each error flag costs a line operator 3 to 5 seconds. Besides the salary of the line operator, which is 1 eurocent per second, error flags also reduce the throughput so false calls indirectly reduce the potential turnover. Also, having many false calls during a production reduces the operator's attentiveness thus increases the chance of error slips. Each error slip is estimated to cost €20 to €30. Furthermore, it is not always clear why quality deviations (resulting in false calls) happen during the process as the interrelations between the quality and the process are not exactly known. A comprehensive description of the business case is given in Section 4. These issues in combination with the vast amounts of data AME is gathering lead to the following business question:

How can process data be utilized to improve production throughput, be more efficient in dedicated workforce, and decrease the false call rate of the automated optical inspection?

This thesis tries to answer this question by exploring and analysing process data of the surface-mounted device production line of AME using data-driven methods. In the next section the research objective related to this business question is described by further defining the scope, research relevance and the research questions.

2 Research Objective

The goal of this research is to improve the working conditions and increase the overall production control and throughput of AME by utilizing (a combination of) data-driven methods, predicting the product quality and possibly investigating relations or dependencies between production process variables and the process outcome. This will mainly be done in an online fashion, using online process data in order to enhance the quality inspection which improves the production throughput. Achieving this goal will result in several benefits for AME, which can be divided into internal and external motivators. First the internal motivators are described. In general, this project will help AME internally by improving their product process via the reduction of quality deviations. This results in an increase in efficiency and a reduction of waste (and thus costs), both in terms of materials and labour. By analyzing the production process in relation to the product quality, expert knowledge can be captured. This could lead to an increase of the understanding regarding the interrelations between the process and the product. Important process parameters that affect product quality can be discovered, helping with a deeper understanding of the process by opening the process black box. Process design might thus be improved when introducing new products or adjusting existing ones, due to that important process features which relate to the product quality are known. From an external point of view, the results of this project could enlarge the adaptiveness of AME related to production planning and execution. This increases the flexibility of AME to changing market conditions, leading to more competitive advantage.

The data-driven approach is chosen due to the fact that AME stores a vast amount of data without using this to its full potential. If built and implemented correctly, the model will improve the production throughput and potentially provide additional insight in the production process by exploring relations in the production data. Different data granularity levels are used to build the model, such as the product type level, batch level and serial number level. Another goal of this thesis is to incorporate as many perspectives as possible to detect potential causes of product quality deviations. A combination of different data-mining techniques will potentially contribute to an adequate quality prediction model. A more extensive description of the relevant data concepts can be found in Section 4.4. The following section will describe the research scope, research relevance and finally present the research questions.

2.1 Research Scope

The scope of this research will be the work center using the surface-mount technology, also known as the SMD process. This choice is based on several reasons. First, the SMD process was (and has been) the core of AME's production plant from the first day of production. Almost all products which go through the electronics manufacturing process at AME include components which require surface-mount technology. As this research is about the possible advancements of manufacturing quality in the Industry 4.0 era, data availability is an important factor when defining the scope. Of all work centers, the SMD process holds the largest amount of data in terms of both automated quality inspections (API & AOI) and log data of the machines (more about this in Section 4.4). This also makes the SMD process very relevant for this research. The Manufacturing Execution System (MES) data will be the main data resource of this research. From a business perspective the large amount of data does naturally lead to many potential insights. Production employees state that it can be hard to reduce the number of false calls during the SMD process without increasing the error slip. This research encounters the false call problem from a new (data) perspective and expectedly lead to the enhancement of the production process by reducing the false calls and increasing the process knowledge.

This research fits in the collaboration agreement between AME and the TU/e, which goal is to optimize the overall manufacturing production control. Within this coherence of projects different topics are researched in order to maximize AME's production control. Currently, research is conducted regarding the optimization of production planning. Potential throughput improvements as a result of this research project will reduce the production time for each order and therefore enhance the production planning. The synergy of the different projects combined will lead to an increased added value from both a research and business perspective. The complete production system enables more automated control in the planning and the execution of that planning, increasing the responsiveness of the company to changing external conditions. An overview of the topics researched in the projects are shown in Figure 1. This

project will focus on the quality part shown in the overview.

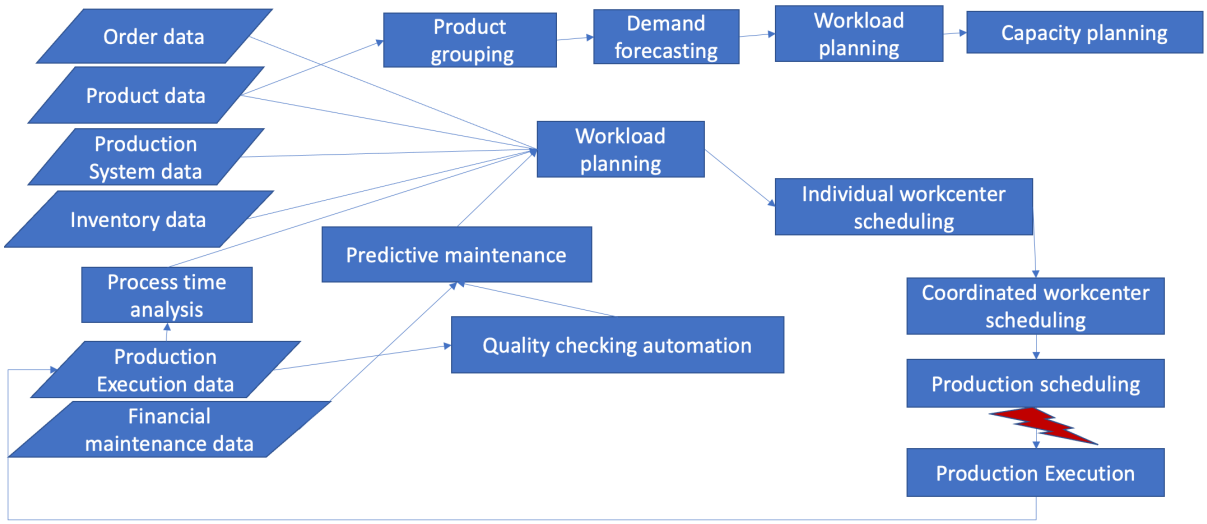


Figure 1: Overview research project topics

As previously stated, the quality of a printed circuit board assembly is determined by a lot of factors. Therefore, it is important to define what the definition of product quality during this research. For this project quality is defined as the component placement quality inspected by the AOI at the end of the SMD line. The quality of all components on a PCBA must be sufficient. A component's quality is considered insufficient if both the automated inspection and the operator reach consensus about the unsatisfactory quality, meaning the product requires a repair. This is both costly in time and money and reduces the overall sustainability of AME's production plant. What is furthermore important for this research are the components for which the machine and operator do not reach mutual agreement, as these are falsely labeled as erroneous by the machine. Thus the quality of a placed component can fall into three categories: good, insufficient or false call. Whenever the quality is insufficient, an additional label with the error type is added to the inspection result of the placed component.

2.2 Research Relevance

The application of machine learning in manufacturing is hardly a novel theme, more than two decades ago various papers proposed applying machine learning to improve production in the manufacturing field (Preuveneers & Ilie-Zudor, 2017). However, due to the digital transformations which take place in the 4th Generation Industrial Revolution, the increasing availability of data boost the potential of machine learning in manufacturing environments. Much of the research regarding PCBA quality is devoted to a single sub process or is not generalizable for a wide range of PCBA's. This research will try to improve the product quality inspection by including the whole SMD production process and incorporating different product types. In terms of business relevance the thesis will contribute to a more efficient production line by reducing unnecessary manual inspections. Futhermore, there is a vast amount of (yet unconnected) process data available which is not yet analysed or used in order to control the production. Thus from both an operational and a data standpoint, there is room for improvement.

2.3 Research Questions

Based on the research objectives and the problem description the main research question is defined. This question runs like a thread through the project. The main research question is formulated as:

What (explainable) data-driven model can be developed with product and process data to reduce the false calls during the quality inspection, in order to improve the efficiency of production operations?

In order to thoroughly answer the main question and provide some structure during the research the problem is divided into several sub questions. Each sub question is addressed by a section in this research. The questions are as follows:

1. What is AME's current practice for monitoring the quality of printed circuit board assemblies?
2. Which state-of-arts methods are proposed in the literature regarding data-driven product quality control in the electronics manufacturing field?
3. What product and (online) process data features are available and can be extracted, forming the most convenient conceptual data set to analyze the product quality?
4. Which relations can be explored in the data using exploratory data analysis in order to find a correct subset of features for predictive modeling?
5. How to select and train a data-driven model that is robust and insensitive to imbalanced manufacturing data, to predict whether error flags are false or correct?
6. How is the model evaluated and interpreted from a business perspective and what are the potential benefits?

3 Theoretical Background

Before the rise of data-driven analytics, physically based modeling served as the status quo for process optimization in the production quality domain (Krauß, Frye, Beck, & Schmitt, 2019). This modeling technique uses physical dependencies to describe the current and future state of a product or system. A potential downside of this method is the need for a deep understanding of these physical interdependencies. With the increasing complexity of production processes and the rising employee turnover, it is harder to acquire and retain this knowledge within companies. Digitalization has lead to a steady growth of data in the recent years, which resulted in an increase in the use of data analytics in a wide range of domains (Krauß et al., 2019). One of the reasons to use analytics in business decision making is the possible avoidance of subjectivity (Banerjee, Bandyopadhyay, & Acharya, 2013). Furthermore, data-driven models use information from observed data to identify system characteristics and predict the future without requiring a deep understanding of interdependencies (Krauß et al., 2019). With the advancements in collecting data with rich content and the ease of access to this data, the use of machine learning (ML) algorithms also increased. Apart from the increase in data acquisition, other reasons for this trend are the higher computing power, increasing reliability of algorithms, and the increase of accessible programming libraries which enable the implementation of complex methods (Krauß et al., 2019). Advanced analytics, such as machine learning, enable businesses to initiate proactive decision making which can be a major competitive advantage (Banerjee et al., 2013). Thus, the development of data-driven models shows a high potential for improvements of production processes (Krauß et al., 2019). According to Filipič and Junkar (2000), machine learning methods are an appropriate tool for incorporating expert knowledge into decision making procedures for machining. Machine learning could help in clarifying complex interrelations among parameters and features involved in the machining process, thus enabling performance prediction and enhancing control. Another advantage of data mining is that the data needed for the analysis can be collected during the normal operation of the process being studied (Kusiak & Kurasek, 2001). This is in contrast with other approaches such as the design of experiment approach, where costly experimentation is essential. Due to these facts a data-driven approach, machine learning in particular, is a suitable method for this thesis and will be used in order to improve the production control of AME's production process. In this section, data project methodologies are described, an overview is given regarding data mining in electronics manufacturing, followed by an explanation of relevant methods and algorithms.

3.1 Methodology

When the data mining industry entered the main stream markets around 2000, the need for a standardized process model increased (Wirth & Hipp, 2000). The CRISP-DM (Cross Industry Standard Process for Data Mining) methodology was especially designed to provide a structured approach for all data mining projects. The model is independent of both the industry sector and the data mining technologies used, providing a strong guidance for research projects. While the methodology explicitly states data mining, it can also be applied to machine learning projects as these two areas have a strong overlap and are often used synonymously (Seidel et al., 2019). According to the methodology, a data mining project is organized into six phases, as depicted in Figure 2. Note that the sequence of the phases is not strict and the process is iterative. In practice, the outcome of a certain phases determines what the next phase will be.

Business understanding is the initial phase, which focuses on understanding the project objectives and requirements purely from a business perspective (Wirth & Hipp, 2000). This knowledge is then translated into a data science or data-driven problem definition. The subsequent phase, data understanding, naturally progresses from the first one. Starting with the initial data collection and proceeding with activities in order to get familiar with the data, identify data quality problems, and discover first insights. These insights could lead to detection of interesting data subsets resulting in new business understanding or hypotheses (Wirth & Hipp, 2000). Data preparation covers all activities to construct the final dataset used for the data-driven model. During the modelling phase of the CRISP-DM framework, various modelling techniques are selected, tested and tuned. As some problems only arise during the modelling phase, there is a close link between the data preparation and the modelling phases, resulting in a highly iterative process. Before proceeding to the deployment stage, it is important to thoroughly evaluate

the built model from a data-mining perspective, which is done during the evaluation phase. Another key objective is to determine if there is some important business issue that is not sufficiently considered (Wirth & Hipp, 2000). Finally, during the deployment phase the data-driven model is implemented in the business environment. This might be as simple as generating a report or as complex as a built-in machine learning model in a production plant.

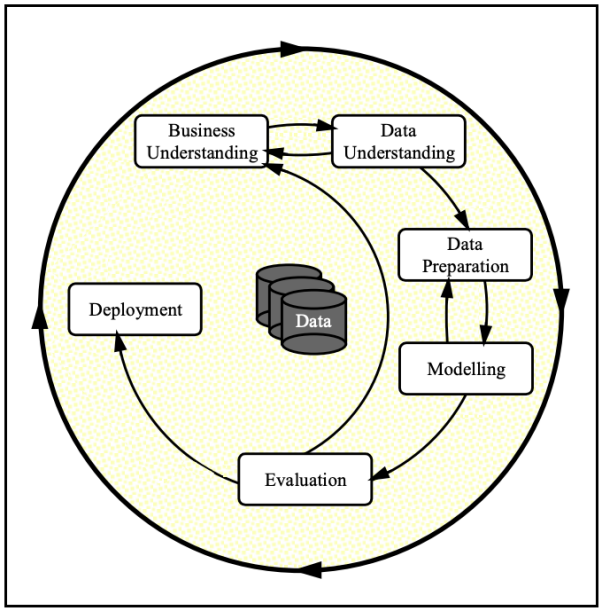


Figure 2: CRISP-DM Phases

The main focus of this project is creating a state-of-art machine learning model which will serve as a proof of concept for AME's manufacturing environment. Therefore, the deployment phase of the CRISP-DM model is less relevant for this thesis. However, all other phases of the methodology provide a solid structure to successfully perform a data-driven research withing AME's production environment. The phases of the CRISP-DM framework are covered in the chapters of this study. Business understanding is mainly done in Section 4, data understanding and data preparation is covered in Section 5, modeling is examined in Section 6, the evaluation phase is described in Section 7. In the conclusion of this research, the deployment phase will be briefly touched upon by providing implementation recommendations.

When tackling a data project, a wide range of analytical techniques exists. Methodologies such as CRISP-DM are used for the general project planning. During the data preparation and modeling phase, the research problem can be divided into different types of analytical problems namely, descriptive, diagnostic or inquisititive, predictive and prescriptive analytics (Belhadi et al., 2019). An overall framework of data analytics capabilities in manufacturing processes with these sub-problems integrated is presented in Figure 3. Machine learning approaches are traditionally divided into three categories, depending on the feedback of the learning system (Alpaydin, 2020). Learning is called supervised when the data contains example inputs and corresponding labels, with as goal to learn a general set of rules that maps the input to the output. Predicting a categorical target based on input consisting of historical examples is an instance of supervised learning. With unsupervised learning there are no labels provided to the model and the model must find structure given only the input. Clustering analysis is an example of unsupervised learning. When using reinforcement learning, the model interacts with an environment in order to learn a given task or goal (without human interference). Making a computer learn how to play chess is an example of reinforcement learning. These categories of machine learning (or a combination thereof) can be used in all four analytical levels. However, they are most commonly used during the inquisitive and predictive phase.

Descriptive analytics provide hindsight on the current business situation using business intelligence tools (Belhadi et al., 2019). These analytics are regarded backward looking and help explain the question *what happened?*. Examples of descriptive analysis are dashboards or reports with visualizations and statistics. Diagnostic analytics answers the *why did it happen?* question. This type of analysis frequently requires input from the descriptive phase. Generally, diagnostic (or inquisitive) analytics seek to reveal potential rules, characteristics or relationships that exist in the data (Belhadi et al., 2019). Examples of techniques are clustering analysis, decision trees, sequence pattern mining or generalization. Predictive analytics aim to provide a glimps into the future based on historical data, answering the *what is likely to happen?* question (Belhadi et al., 2019). Cheng, Chen, Sun, Zhang, and Tao (2018) divide predictive analytics into two categories: statistical oriented analytics techniques and knowledge discovery techniques. The first techniques (which for example include multinomial logit models and logistic or linear regression) uses mathematical models to induce and analyse the data as well as predict unknown future information. These methods are bound to statistical assumptions in order to be sound. The second category does not require these assumptions and is sometimes able to learn more complex data relations. This category mainly includes machine learning techniques such as artificial neural networks and support vector machines (Belhadi et al., 2019). Prescriptive analytics improve the process or task at hand based on the output information of the predictive models. The techniques are concerned with the definition of the set of decisions that should be done in order to improve the business process (Banerjee et al., 2013). During these analyses, the *what should be done?* question is answered. Section 3 summarizes how these techniques have already been applied in the manufacturing field.

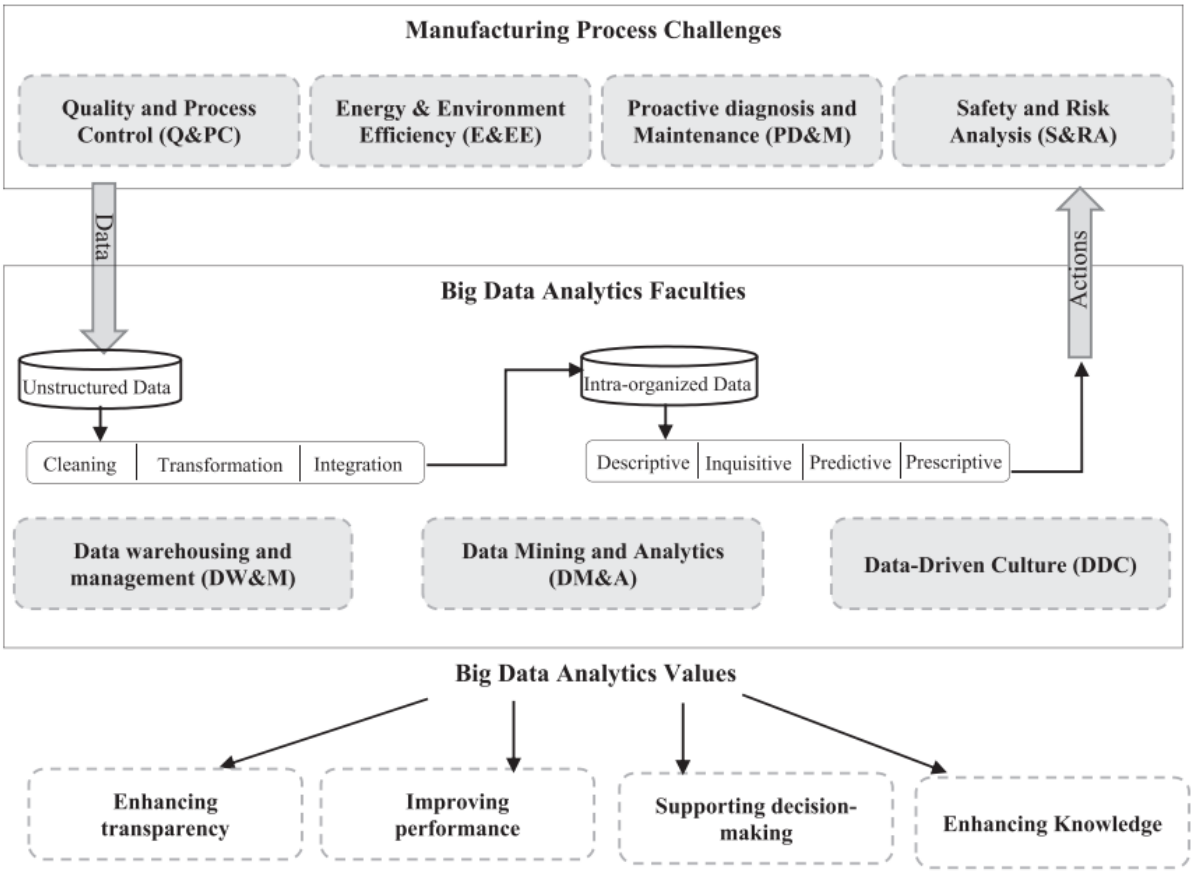


Figure 3: General framework of data analytics capabilities in a manufacturing process (Belhadi et al., 2019)

Furthermore, identifying machine learning use cases for the SMD production process requires a highly interdisciplinary background, i.e. knowledge in computer science, statistics and (business specific) production engineering (Seidel et al., 2019). As this knowledge is rarely combined in one single person, cooperation between different experts is essential. To facilitate this cooperation, Seidel et al. (2019) pro-

pose a process-oriented use case identification methodology using the analytical levels described above. Use cases are divided into technology-push or problem-pull categories. For technology-push, first the use case question has to be defined. Secondly, the required input data sources containing relevant information have to be figured out. Thirdly, the respective output data that shall be predicted or classified need to be defined. In order to identify machine learning uses cases for the problem-pull category, practical problems must be broken down in such a way, that they can be solved by one or more machine learning models (Seidel et al., 2019). This thesis mainly uses the technology-push method, as it tackles a problem with state of the art methods using data from the process. When observing the framework, this thesis can be placed at the AOI level in the process chain and, reaches for the predictive level. The framework is visualized in Figure 42, Appendix A.

3.2 Surface Mount Technology Literature

Production quality is a very broad concept and has been studied extensively. As the scope of this research is electronics manufacturing, a brief overview of the available quality research in the surface mount production environment is provided in this section. Note that the papers related to computer vision or image recognition are not part of the scope of this project. Each paper will is shortly covered in terms of its objective, method and results. The section is concluded by a brief summary and the potential literature gap which can be filled by this thesis.

Linn and Lam (1998) research the in-process errors of the production process related to a single process step, namely the placement of components. Their objective is create more understanding regarding the interaction between the product design and the manufacturing process, so that less placement errors occur during the production process. The focus is on which (and how) product characteristics can be changed during the design phase to reduce the amount of quality deviations. A deeper understanding is sought between the design of the product and the product quality deviations occurring due to the production. The researchers used physical modeling to model the component placement process in a mathematical way. In order to do so, they heavily depend on expert knowledge to find sources of process errors. Components on the PCBA are treated as random variables with a failure probability distribution (assumed to follow a normal distribution). With these properties added to the mathematical model, Monte Carlo Simulation is used to do in-process error analysis and refine the design of the product in order to improve the product quality. Besides enhancing the design process to reduce quality, there are also methods available in the literature which automatically adjust the process to reduce errors. Khader and Yoon (2021) propose an optimal adaptive control system to maintain quality parameters within given limits during real-time production. Khader and Yoon (2018) implement a Q-learning reinforcement model in order to design optimal parameter values (printing speed, squeegee pressure and separation speed) for the screen printing process of a PCB, finding the best possible volume deposited on the PCB. Thus, if the printing volume is not sufficient, the agent will take action based on the optimal policy related to that process state, changing the process parameters in real-time.

Kusiak and Kurasek (2001) analyze the occurrence of a specific solder ball defect on the PCB by taking the entire production process into account, especially features during the production process which could influence the error researched. They try to find rules which describe the process in relation to the production quality, so expert knowledge is captured more easily. Kusiak and Kurasek (2001) use the rough set theory to find rule sets which describe the relationship between solder ball defects on a PCB and production features of the production chain. The rough set approach identifies unique features of an object and sees whether they are shared with other objects. This method offers straightforward interpretation of the obtained results thus enabling to increase the understanding of the production process in relation to the product quality or specific errors (Suraj, 2004). Tseng et al. (2004) also analyze the solder ball defects by building on the idea of Kusiak and Kurasek (2001). Their main goal is also to find a cause for the quality deviations related to a specific error during production. Tseng et al. (2004) extend this rough set theory by applying additional weights to features and objects in order to give some objects more importance and using an heuristic to solve the problem. This reduces the computational time without lowering the predictive accuracy. Tseng et al. (2004) find that the types of solder paste, (conveyor) speed of the machine, stencil characteristics, use of vacuum to pick up components, frequency of stencil cleanings, oven temperature and the component type are important for the PCBA quality control process.

T. Tsai (2012) use a classification model to properly interpret the defect patterns and uncover cause-and-effect relationships between the process parameters and the production quality. The research utilizes decision tree learning to formulate the relationship between the process parameters and the product quality. K-means is used to derive the definite clusters which each represent the soldering quality profiles, serving as the target variables during the supervised learning (T. Tsai, 2012). The sub processes which are used include paste printing, pick and place, and reflow. They express their output as a set of "IF-THEN" rules involving the relevant input variables (process parameters) and the product quality classes. Their main finding is that for their problem, 50% to 70% of the soldering defects are related to stencil printing process (T. Tsai, 2012).

Chang et al. (2019) propose a method to lower the number of false calls which are made by the detection system of a manufacturing process, in order to increase the product yield rate and improve the equipment effectiveness. During the research the automated paste inspection data at the beginning of the production line is utilized to enhance the automated optical inspection at the end of the production line, in order to reduce the number of false calls at component level (Chang et al., 2019). This can lead to a slight change in the routing of a PCBA during production as not all PCBAs require an inspection if the model provides solid predictions. This improves the production yield rate, equipment effectiveness and reduces production costs and handling time. Incorporating different data sources related to the product and process can increase the performance and thus the effectiveness of the model. A deep neural network with two hidden layers is used to predict whether components are real defects or false calls. Chang et al. (2019) find that many problems later in the SMD process are caused by the screen printing process, as the SPI values of this process step can explain much of the later errors which occur (e.g. tombstone). Schmitt et al. (2020) also try to reduce the number of product inspections during the production, which reduces the production time and enhances efficiency by use gradient boosted trees. The paste inspection data is also used in this research, reducing approximately 30% of the volume which requires a test. Thielen et al. (2020) use the AOI inspection data in combination with the component information to enhance the machine call detection. Using an artificial neural network they create a proof of concept for one product type which is able to reduce 25% of the false calls having zero error slip. Each paper addressing the false call problem uses online manufacturing data in their post-hoc analysis, which is a common thing to do in monitoring and controlling the quality in electronics manufacturing (Lv, Kim, Zheng, & Jin, 2018).

In summary, the papers regarding electronics manufacturing which are relevant for this thesis can be divided into several topics: improving the PCBA design, analyzing the soldering quality, analyzing the relations between the process and the quality, and solving the false call problem as this is a common difficulty in the industry. Several methods are used, from physical modeling to machine learning methods, including but not limited to clustering, neural networks, tree based methods and support vector machines. For the false call problem the first literature gap which this thesis can fill is the fact that the models are either trained on only one product type, or that the models do no take the complete SMD process data into account. Schmitt et al. (2020) and Thielen et al. (2020) both state that in order to generalize their models over a wide range of products, it is necessary to add data related to product characteristics and the complete process. This thesis fills this gap as it will try to find decision rules for the false call problem which have the ability to generalize over multiple product types. This will be done by including different product types with their corresponding features in the input data. A second research gap is the lack of explainable models in the false call studies for electronics manufacturing. To the best of our knowledge, there is no electronics manufacturing false call literature available which elaborates on explaining the results of the machine learning model. Explaining the outcome of the models potentially enhances the expert knowledge of the process engineers in the electronics manufacturing industry. Adding process features of the complete SMD production line to the input data will complement to this explainability.

3.3 Classification methods for imbalanced data

Many manufacturing quality research has to cope with imbalanced target distributions in the production data, as quality deviations occur less frequent than sufficient production outcomes. When major class imbalance exists in a data set, the learning system may have difficulties to learn the concept given in the

minority class (Batista, Prati, & Monard, 2004). This section includes the different state-of-art methods for classification with imbalanced target classes. First the different sampling methods are described. Then, an ensemble method is depicted which is especially developed to handle class imbalance. Finally, the chosen algorithms are described and the reasons for selecting these algorithms are discussed.

3.3.1 Sampling methods and class weights

Readjusting the target class distribution can be done by either over and under-sampling. Over-sampling enlarges the minority class and under-sampling reduces the number of samples in the majority class. The techniques which can be used for this method can be non-heuristic methods or heuristics.

Non-heuristic methods are relatively simple. Random over-sampling is such method and aims to balance the class distribution by randomly replicating minority class examples (Batista et al., 2004). Random under-sampling, also a non-heuristic method, does the opposite as it balances the class distribution through random eliminating majority class samples (Batista et al., 2004). The non-heuristic methods have some drawbacks. Random over-sampling increases the likelihood of overfitting because the decision rules learned by an algorithm are then very much dependent on these copied samples. This could lead to problems with generalization when evaluating the model over a different data set. For random under-sampling the major drawback is the fact that potentially useful information in the majority set is lost when removing the samples. This could hinder the learning process as these samples might be important for the induction process (Batista et al., 2004). Heuristics can also be used for over and under-sampling and can overcome these limitations.

SMOTE (synthetic minority over-sampling technique) (Chawla, Bowyer, Hall, & Kegelmeyer, 2002) is an over-sampling technique which is commonly used to balance a data set before machine learning (Fernández, Garcia, Herrera, & Chawla, 2018). The method developed by Chawla et al. (2002) chooses random minority samples and calculates the Euclidean distance between the sample and its k nearest neighbors. This distance is then multiplied by a random number between 0 and 1, and added to the minority sample, forming a new synthetic sample. Unlike random over-sampling, SMOTE adds new information to the data set which enhances the learning process of classification models.

Tomek et al. (1976) proposed an under-sampling method based on nearest neighbors, called Tomek links. Two data samples form a Tomek link if both points are each other's nearest neighbors and both observations belong to a different target class (i.e. one to the majority class and one to the minority class). Tomek links define majority data points which are close to the minority class data, making it ambiguous to distinct. The majority class samples which are Tomek links are then removed from the data to create a more balanced data set. This results in a data set with less edge cases which enhances the learning of general decision rules.

Both SMOTE and Tomek links solve the drawbacks of non-heuristic methods but still not always result in better model performance. In imbalanced data sets, class clusters may not be well defined. It can be the case that majority class examples invade the minority class samples, even after using Tomek links to remove cases. SMOTE can also result in synthetic minority cases which are too deeply in the majority class space (Batista et al., 2004). Batista et al. (2003) propose a method combining both methods, producing a balanced data set with well-defined class clusters. An example of the method is given in Figure 4. The original data set (a) is over-sampled using SMOTE (b), then the Tomek links are identified (c) and removed, resulting in a balanced data set with well defined clusters (d) (Batista et al., 2003).

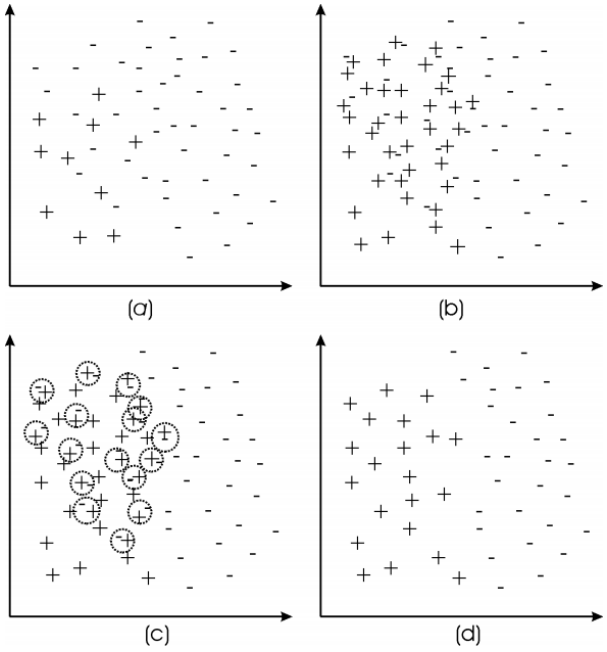


Figure 4: SMOTE Links (Batista et al., 2003)

The problem of skewed target classes can also be solved by applying different weights to the majority and minority classes. The purpose of the class weights is to penalize misclassifications in the minority class more strictly and the majority when training the model. Applying class weights during model training is not dependent on a specific algorithm. To reduce the bias in the class imbalance as much as possible, the sampling methods can be combined with class weights to mitigate the effect of the class imbalance during model development.

3.3.2 Balanced Bagging Classifier

Ensemble methods use multiple machine learning algorithms to enhance the predictive performance compared to a single algorithm (Polikar, 2006). Although some algorithms can be well-suited for a given problem space, finding the best one might be difficult. Combining (preferably diverse) models can result in better predictive performance and less overfitting. The downside of ensembling models is that it is computationally heavier than non-ensemble models. Therefore, fast algorithms such as decision trees are commonly used in ensemble methods. Nevertheless, slower algorithms can also benefit from ensembling in terms of model performance. In manufacturing, the most commonly used ensemble learning methods are bootstrap aggregating (bagging) and boosting.

Bootstrap aggregating (abbreviated as bagging) is an ensemble method which combines models based on votes of equal weights. In order to obtain a wide range of diverse models, the method trains each model in the ensemble with a randomly drawn subset of the training set. Training these models can be done in parallel. Each model has a different sample although replacement is permitted during sampling. The final output is determined by a majority vote or an average of the outputs of all models. Bagging can be used for a wide range of methods including (but not limited to) decision trees (Sankhye & Hu, 2020) and neural networks (Du et al., 2012). In order to find the best "candidate" models, several techniques can be used. A possible, relatively simple, method is to order the candidate models based on the mean squared error and create an ensemble with the models having the lowest error (Perrone & Cooper, 1993). Another common techniques besides bagging is boosting. Boosting is a meta-algorithm that builds an ensemble incrementally and emphasizes misclassified training examples of the previous model to train the new model. Weak learners (e.g. small decision trees, also known as stumps) are added to the ensemble to concentrate on these misclassified observations to compensate for the areas where the existing model did not suffice. Each training example has a weight assigned which increases if the instance is misclassified. In order to make a prediction, the results of the models are combined with a

voting mechanism. Boosting techniques can improve the accuracy of models without depreciation of the other useful capabilities (Martinek & Krammer, 2018). In general, boosting techniques may enlarge the computational effort. However when using a gradient boosting approach, it is possible to overcome this problem. Gradient boosting adjusts the original boosting algorithm so that it minimizes a differentiable loss function via gradient descent by adding models to the ensemble. This means that the training of new models is based on the residuals of the previous model, which speeds up the process as the weights do not have to be calculated.

Both bagging and boosting can be used to mitigate the effect of an imbalanced target class. However, bagging ensembles are more common due to its simplicity and good generalization ability (Galar, Fernandez, Barrenechea, Bustince, & Herrera, 2011). When using bagging for dealing with the class imbalance problems, it is not required to recompute weights or change computations in the algorithm itself, which is the case for boosting. The main challenge of using a balanced bagging ensemble method is finding a good way to collect the balanced subsamples which serve as an input for the ensemble models. One way of doing this is by using an under-sampling method to reduce the size of the majority class in each replication (Barandela, Valdovinos, & Sánchez, 2003). Each bootstrap replica then consists of all the minority class samples and an under-sampled majority class which is different in each iteration to form a diverse ensemble. An example of the balanced bagging classifier method in combination with under-sampling is provided in Figure 5. Over-sampling techniques such as SMOTE can also be used when forming the bootstrap replications in each iteration (Wang & Yao, 2009). Then the set of majority instances is again bootstrapped in each iteration, but the SMOTE algorithm generates additional minority class samples in each iteration. Note that the same sampling method limitations as described in Section 3.3.1 apply to each individual model in the ensemble, but the balanced bagging technique reduces these effects.

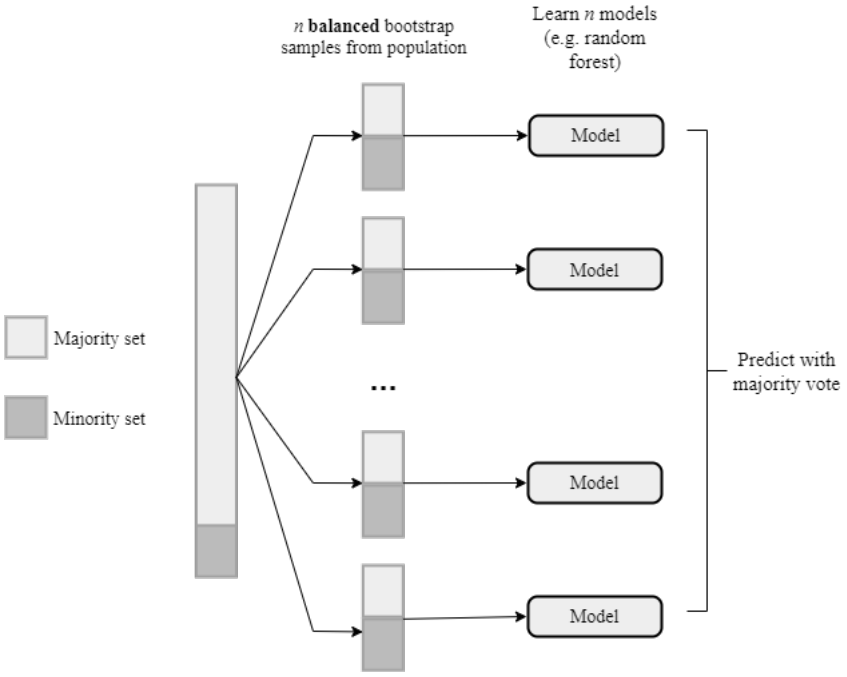


Figure 5: Balanced bagging classifier with under-sampling

3.3.3 Choice of machine learning models

According to the *no free lunch theorem* by Wolpert and Macready (1997), there is no such algorithm having a priori distinction compared to other algorithms for a given problem. Stated otherwise, any two optimization algorithms are equivalent when their performance is averaged across all possible problems (Wolpert & Macready, 1997). Choosing an algorithm for the problem at hand is not trivial, therefore three different supervised classification methods are examined based on the differences in underlying learning techniques. These include a linear method (logistic regression), a nonlinear method (support

vector machine), and a tree based method (random forest). The methods are briefly described in the remainder of this section.

Logistic Regression
Logistic regression is a linear statistical method to model the relationship between the log odds of a dichotomous variable and a set of explanatory variables (Kleinbaum, Dietz, Gail, Klein, & Klein, 2002). Production features and corresponding quality measures nowadays are often described in a non-linear fashion, making linear regression less suited for the quality prediction task. However, due to the fact that it is a relatively simple method, it can be used as a base model to explore any first relations among variables. For the logistic regression, the equation is comparable with the equation of multiple linear regression, and is shown in Equation 1.

$$\log \frac{P(y=1)}{1-P(y=1)} = \alpha + \beta_1 x_1 + \ldots + \beta_n x_n \tag{1}$$

The coefficients β_n of the logit model can be interpreted as the change in the log odds of an event when x_n increases by one, all other variables held constant. These coefficients can be transformed to odd ratios calculating e to the power of β_n (Kleinbaum et al., 2002). These odd ratios can then be interpreted as the increase (or decrease, if the odd ratio is smaller than 1) in the probability of the dependent variable being present when a certain variable increases or is present. Logistic regression performs badly when there are outliers in the data or when there is multicollinearity present.

Random Forest
A Random Forest is a bagging method and combines several individual weak learners (decision trees), also known as an ensemble model. Every tree uses different subsamples of the data and splits the tree nodes with different random subsets of features in order to reduce the bias in the model (Sankhye & Hu, 2020). The main purpose of this method is to add randomness to generate de-correlated trees (Garcia-Ceja et al., 2019). By averaging the output of all generated trees, the final prediction is obtained. An example of the random forest concept is depicted in Figure 6. Random forest models are (relatively to other well known methods) not prone to overfitting, have a good tolerance for outliers and noise, and is not sensitive to multicollinearity in the data (Chen et al., 2020). Furthermore, it can handle nonlinear high-dimensional data both in continuous and discrete form (Chen et al., 2020). These properties make random forests a suitable method for predicting product quality in manufacturing. As random forests generate multiple different random trees, it is possible to find the variable importance by averaging the differences in prediction error based on predictor variable permutations over all trees (Garcia-Ceja et al., 2019). This feature importance can then be used as feature selection tool prior to another machine learning model (Leng et al., 2020). Besides the variable importance, it is also possible to further analyse the relationship between the prediction variable and the outcome of the model, which can be visualized with a partial dependence plot (Molnar, 2018). For the random forest the most important hyperparameters are the maximum depth, the minimum samples in a split and the number of estimators. Maximum depth limits the depth of each trained tree and thus to which extend each decision tree trained is prone to overfitting. The minimum samples in a split determine how many samples are needed to make another split in the tree. If this number is small, it is possible to create leafs with only a few samples dependent on many decision rules. The number of estimators defines how many trees are trained to create the forest.

Support Vector Machines
Support vector machine (SVM) is a supervised learning algorithm which was originally introduced to classify discrete multidimensional data. Support vector machines are particularly useful in small sample, non-linear classification problems with high dimensional data (Wei, Feng, Hong, Qu, & Tan, 2017). This makes this method useful the manufacturing industry due to the high dimensional data sets. The algorithm maps training examples in a space and the optimization goal is to maximize the width (or margin) of a hyperplane (or decision boundary) which (linearly) separates training examples of different categories or classes. In most real data sets it is not possible to linearly separate training examples in a given feature space, requiring a transformation of the data into a higher dimensional feature space. The goal of this transformation is trying to find a dimension in which the classes are linearly separable. The kernel trick provides a solution to this problem by using a function which represents the data only by a

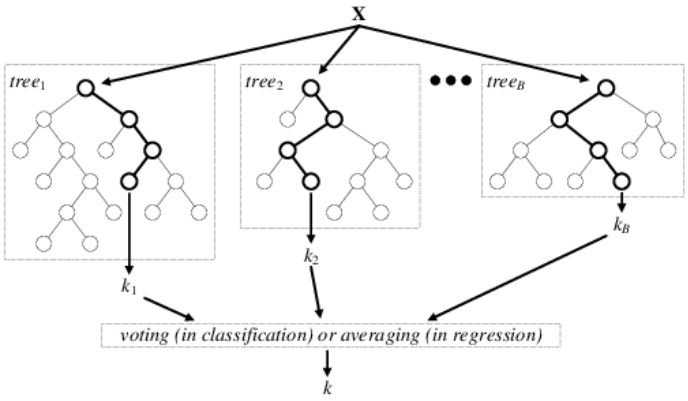


Figure 6: Random forest concept

set of pairwise similarity comparisons between the observations. The data then does not need a explicit transformation but it can be represented by these coordinates in a higher dimensional feature space, which saves computational effort. New cases which are to be predicted are mapped into this space, and based on their position in that space relative to that learned hyperplane the new cases are classified (Vapnik & Learner, 1963). Vapnik and Learner (1963) show that chance on overfitting and thus the probability of misrepresenting untrained data is minimized when using SVM, resulting in a decent generalizing ability. Due to the fact that support vector machines are distance based, it is recommended to standardize the input data before learning. Support vector machines have two main hyperparameters (C and gamma) which can be tuned to find the most suitable model for a problem. The C parameter depicts the penalty for a misclassified data point (directly proportional to the distance to the hyperplane) when training the decision boundary. The size of C determines the size of the penalty, and larger values result in smaller margins thus an increased probability of overfitting. When using the radius basis function as the kernel function to create a linearly separable data set, the gamma parameter determines the influence of a single training point. It determines how close points should be in order to be considered as the same group or class. For small gamma values, the points can be further away which results in bigger groups. Large gamma values require closer points to be in the same class. Both hyperparameters can be used to tune the model in a way that it generalizes well for all data related to a problem.

3.4 Anomaly detection with autoencoders

Another method to tackle the problem of an imbalanced target class is anomaly detection. This machine learning method tries to identify rare items in a data set by comparing these items with the majority of the data, treating the minority class as outliers or anomalies (Zimek & Schubert, 2017). Both supervised and unsupervised anomaly detection methods are available. In general, supervised anomaly detection at its core is classification in an imbalanced environment as the normal and anomaly cases are labeled (Chandola, Banerjee, & Kumar, 2009). As these techniques are already described, supervised anomaly detection methods are not further explained. Unsupervised anomaly detection however, provide another potential modeling methodology. In this case, the observations are considered to be unlabeled and it is assumed that anomalous samples differ significantly from the normal samples (Chandola et al., 2009). By doing so, the model learns what normal behaviour is by only looking at the majority class data. When classifying a sample, it compares the sample with this normal behaviour. If this sample does differ to a certain extend from the normal behaviour, it is considered an outlier thus likely part of the minority class (Chandola et al., 2009). This method utilizes the large size of the majority class by learning as much as possible from this sample. The advantage of this method is it is not necessary to learn the behaviour of the minority class, which is a hard task if only a few minority samples are available. A subspace of anomaly detection are deep anomaly detection methods, which have several advantages over traditional algorithms (Chalapathy & Chawla, 2019). First, traditional algorithms require extensive feature engineering which results in the performance being sub-optimal for complex structures in the manufacturing data. Deep anomaly detection methods can also be used to learn a latent space from

the complex data structure. This output can be used as input for traditional methods (Chalapathy & Chawla, 2019). Furthermore, deep anomaly detection methods can handle increasingly data volumes generally well. Traditional methods can have convergence challenges when finding outliers in large scale data sets. The complexity in the manufacturing data due to the many different product types and the increasing data volumes make deep anomaly detection methods a suitable choice for this research.

Autoencoders are based on artificial neural networks and are considered as the fundamental unsupervised deep architecture for anomaly detection (Baldi, 2012). Models such as autoencoders need to met several assumptions in order to work properly (Chalapathy & Chawla, 2019; Goldstein & Uchida, 2016):

- Normal data can be distinguished from anomalous data in the original data space or the learned data space. This assumption is met, because the error flags found by the machine have some sort of deviation from normal component placements in order to be an error flag at all.
- The vast majority of the samples in the data set are considered to have normal behaviour. Data imbalance is a well known problem in manufacturing environments, in most cases this results in an imbalanced target class containing a large majority class and a small minority class. Therefore this assumption is also met for this research.
- Unsupervised anomaly detection methods produce a outlier score based on the intrinsic data properties which need to be present in the data in order to find outliers. Finding whether this assumption holds is one of the main goals of the research: trying to find if it is possible to capture the intrinsic properties of a manufacturing data set by a model, which helps explaining quality behaviour.

Autoencoders are learning circuits in the form of artificial neural networks aiming to transform inputs to outputs in the best possible way, with $\hat{x}$ as reconstruction of the original input x (Baldi, 2012). It is an unsupervised learning method because the input of the neural network is identical to the target of the neural network. The general framework of an autoencoder consist of encoding layers, a hidden layer representing a latent space, and decoding layers. A common practice is to constrain the number of nodes in the hidden layer(s) compared to the input and output layers. This limits the amount of information which can flow through the network thus learning an encoded representation of the data in a latent space. Generally speaking, the number of neurons in the hidden layers p are smaller than the number of neurons n in the input and output layers ($0 < p < n$). A rule of thumb for a basic architecture when determining the number of neurons in a layer might be that each layer is half the size of the previous layer for encoding, and twice the size of the previous layer for decoding. An example of an architecture is given in Figure 7.

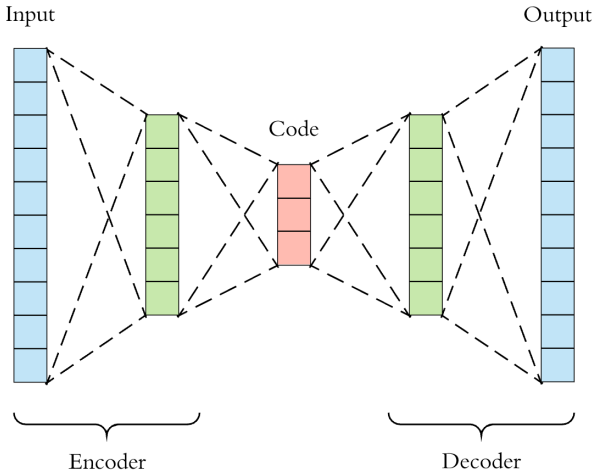


Figure 7: General autoencoder architecture

Learning happens in the same manner as other artificial neural networks, evaluating the output (which is the result of the combination of activation functions in the neurons) using a loss function, and back-propagation with the gradient descent algorithm to learn the correct model parameters (weights and

biases) (Rumelhart, Durbin, Golden, & Chauvin, 1995). The learning process is very briefly explained to form a general understanding of how the autoencoder model learns. First, the architecture of the model is designed, defining the number of layers, the number of neurons per layer, the activation functions, and the loss function. To differentiate from PCA, it is required to use non-linear activation functions such as the Rectified Linear Unit (ReLU) or the Exponential Linear Unit (ELU) (Clevert, Unterthiner, & Hochreiter, 2015). When initializing the network, the weights between the neurons and the biases related to these neurons are assigned randomly. The loss function evaluates how well the output vector $\hat{Y}_i$ reconstructs the input vector Y_i by comparing the input and output with the mean squared error (mse), see Equation 2. In short, the backpropgation algorithm then calculates the gradient of the loss value in the solution space relative to the weights, biases and activation functions of the network. A gradient is defined as the direction in which a scalar function (the combination of weights and biases) has to move for the greatest decrease of the loss. The weights and biases are then adjusted according to that gradient in a backwards fashion, corrected by a learning rate. By providing many samples to the neural network in an iterative fashion, the backpropagation algorithm slowly converges to a function (described by the parameters of the network) which minimizes the loss function. An elaborate mathematical explanation can be found in the article of Rumelhart et al. (1995).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2}$$

When classifying new data samples with an autoencoder, the vector of the sample is given as input to the trained model. The model outputs a reconstructed vector of the same size as the input. Again, the mean squared error is used to calculate the average reconstruction error for a sample. Then, in order to define whether a sample is an outlier or not, it is compared with a given threshold value. If the reconstruction error is larger than the threshold, the sample is considered as an anomaly data point, if it is smaller than the threshold, the sample is considered as a normal data point (Massi, Ieva, Gasperoni, & Paganoni, 2021). Defining the value of the threshold is problem dependent and can be done manually, with statistical distributions defining an outlier, or based on (supervised) machine learning methods.

There are several advantages and disadvantages of using unsupervised deep anomaly detection methods for analysing imbalanced data (Chalapathy & Chawla, 2019). One of the main advantages is that the method learns inherent data characteristics of the data set to separate normal points from anomaly points, identifying commonalities in the data set. This method has several implementations such as dimension reduction and feature engineering. The reconstruction error for each feature can also be used for feature selection or finding feature importance when differentiating between majority and minority classes. Lastly, the reconstruction errors can also be used in a classification problem if the labels of the data samples are known. Setting the threshold can be seen as an advantage as well as a disadvantage. It is useful as it brings a certain freedom when classifying with the model. However, this requires additional tuning. Finding the right threshold for unsupervised problems is not always a trivial task, as unsupervised techniques are very sensitive to noise in the data (Chalapathy & Chawla, 2019). Another disadvantage of autoencoders is the fact that it is not easy to find the right network architecture for a problem at hand, as it requires a lot of experimentation to find the right degree of compression.

3.5 Explainable machine learning

Generally, the main goal of a machine learning model is to find a set of decision rules which lead to the highest possible performance. However, lately there is an uprising high demand for understanding why a model makes certain decisions (Roscher, Bohn, Duarte, & Garcke, 2020). Model transparency and interpretability of the output results enhance the adoption of models in business and increase the overall trust in the models. Domain or expert knowledge can help explain the model or further improve the model's outcomes, but the complexity of well performing models hinders transparency and interpretability by design. Therefore Lundberg and Lee (2017) proposed SHAP values, an unified approach to interpret any machine learning model.

SHAP values stands for Shapeley Additive Explanations and is a state of the art method to explain the result of a machine learning model (Lundberg & Lee, 2017). Shapley values originate from game theory

and assumes there is a game with players. Shapley values quantify the contribution of each player to the game, or the value of each player in a given game (Hart, 1989). SHAP values are an extension of Shapley values and quantify the contribution of each feature to a prediction done by the model, where one sample is assumed to be the game and the features assumed to be the players. To calculate the SHAP values the algorithm trains a model on all different combinations of features. These combinations are represented by the power set F of the features. A power set of a set S is the set of all subsets of S, including the empty set and S (Vardi, 1991). Thus to compute the SHAP values, 2^F distinct models are trained with the same hyperparameters and training data, but with different feature sets. The marginal contribution of a feature is calculated by comparing models in the power set. For instance, model A is built with features x_1 and x_2, model B is built with features x_1, x_2 and x_3. If model A predicts that the target is 0.7 and model B predicts that the target is 0.6, the marginal contribution of x_3 is -0.1. These marginal values are calculated between each model which adds x_3 to the feature set, also known as the f-models. The marginal contributions of these f-models are aggregated per feature and weighted by the binomial coefficient $C(n, k)$ with $n = f, k = F$, times f. F is the total number of features in the power set and f depicts the number of features in the f-model of a feature. Thus, the formula to calculate a sample's SHAP value for a given feature is depicted in Equation 3 (Lundberg & Lee, 2017).

$$\text{SHAP}_{feature}(x) = \sum_{set: feature \in set} [|set| \times \binom{F}{|set|}]^{-1} [Predict_{set}(x) - Predict_{set \setminus feature}(x)] \tag{3}$$

The use of SHAP values is twofold, it can be used for global interpretability and local interpretability. The former uses the collective SHAP values to show how much each feature positively or negatively behaves relative to the target variable. This can also be interpreted as the feature importance for each variable. The latter increases the transparency per predicted observation, using the SHAP values of that sample. A prediction for a case can be explained by showing the contributions to that prediction of each feature.

4 Application Background and Data Concepts

Since the research is being conducted at AME, it is important to develop the problem statement within the framework of the company. This section belongs to the business understanding phase of the CRISP-DM methodology, and will also touch upon the data understanding phase. First of all, a complete overview of the business activities is provided. Secondly, the problem definition will be developed, consisting of AME's quality vision (their call for quality), the state of current practice for monitoring quality, the problem statement from a company's perspective, and process measures to support the problem statement. Next, the important data sources are described followed by the definition of potentially relevant data concepts for the problem at hand. Eventually the data gathering process is described including the feasibility in terms of data availability, leading to the raw data set used for analysis.

4.1 General Process Description

When AME was founded the main focus was the production of electronics at the electronics manufacturing work center. Over the years, the company has grown, which emerged into new business interests due to customer demand and the company's urge for development. This means that some work centers nowadays are mainly used to fill AME's own need for machine parts or components, decreasing the dependence on other companies. The following section will describe the work centers in more detail, to provide a complete overview of the company's activities. AME divides its manufacturing plants into six work centers, each serving a different production purpose, contributing to the overall business of AME:

- Electronics Manufacturing
- System Assembly
- Injection Moulding
- Machining
- Cable & Wiring
- Product Cleaning

At the Electronics Manufacturing (PRD) the core products of AME are produced, which are the printed circuit board assemblies (PCBA). AME divides this work center into two distinct subprocesses: the surface-mounted devices (SMD), which is the workcenter of importance for this thesis, and post surface-mounted device (post-SMD). During the SMD process, small components (which are not connected via holes in the PCB) are attached to the printed circuit board with solder paste. To start production, an operator places a batch of the correct blank printed circuit boards (PCB) at the beginning of the process. First, soldering paste is added to the PCB's solder pads with a stencil (a mold to apply the paste on the right places) and a squeegee (a flat smooth rubber blade to apply the paste) and automatically inspected. Then, the PCB is moved through the pick-and-place machine, where all surface-mounted devices are picked from the component feeders and placed on the PCB. Before the production of a batch starts, the operator loads the correct tape & reels onto feeders mounted to the machine, these tape reels supply the components to the machine. After the components are placed on the PCB, the paste is hardened by going through a heating and cooling process. Then, the PCB is cleaned from dust and the component placement is automatically inspected. The operator removes the PCBA from the machine and places it on a tray. For some products, the bottom of the PCB also requires components. In this case the tray is moved to the beginning of the process. AME has three SMD lines which can work in parallel. Figure 8 schematically shows the steps during the SMD process. If a product needs plated through-hole components attached, it also moves through the post-SMD process. In this process there are several product dependent production steps, meaning that different products can move differently through the production plant. However, the main process during post-SMD is attaching the plated through-hole components. These components are placed through the holes of the PCB's by hand and then automatically soldered and depanelized by a machine. Dependent on the product, it is either boxed for storage or moved to other post-SMD production steps such as kitting and coating, in-circuit programming or a manual inspection. During both the SMD and post-SMD processes, tests take place to assess the quality of both the process (e.g. how many errors occurred when placing the components) and the products (e.g. are components firmly attached to the PCB).

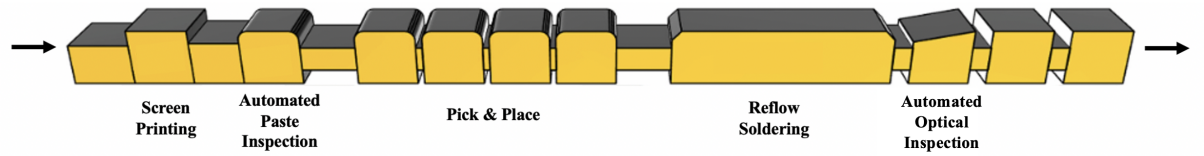


Figure 8: SMD process overview

During the Injection Moulding (IM) process, liquid plastic is moulded into a certain shape, serving as a product component or final product. The work center has five machines, all of different size. A larger machine means that it is able to produce larger products. Some products are boxed and tested automatically by the machine. Other (primarily larger) moulded products require some manual inspection and are not boxed automatically. Larger products require visual inspection due to the fact that there are simply more aspects of the product that could have been moulded wrong. Besides, as these larger products are more costly than the smaller products, the manufacturing process requires more attention. The moulds for the IM process are made by AME at the Machining (MILL) work center. Each product which is made at the IM work center has its own mould design which is designed in collaboration with the customer. During this design phase AME tries to use material as efficient as possible throughout the whole company. One example is the integration of a PCB connector to the housing of a product in order to save components during the Electronics Manufacturing work center.

The Cable & Wiring (WIRE) work center provides cables and wires for external customers but the main bulk of the production is for internal use at the System Assembly work center. Producing cables and wires is a relatively fast and (semi)automated process. After the wires are produced they are checked (and sometimes assembled) manually by the manufacturing operators. The Product Cleaning (CLEAN) work center is used to clean and glue products which require special treatment. Only a small proportion of the produced products go through this work center.

At the System Assembly (SA) work center a sub assembly or a final product is manually assembled. Often this is the final step before packaging and shipping. Products from different work stations may come together here to form the final product for the customer. After assembling the products, products can be tested automatically or manually, dependent on the product. Due to the fact that the work center is based on manual labour, the quality of products may vary with the skill of the operator which assembles the product. After SA, the product either is shipped to the customer or stored in the warehouse.

4.2 Problem Definition

This section describes the problem statement and outlines the importance of product quality from a business perspective. Also, methods for assessing the product quality of the SMD production process are described in detail which serves as a building block for the problem statement. Then the section is concluded with an overview of the quality indicators that empirically support the problem statement.

4.2.1 Call For Quality

One of the key strategic pillars of AME is the quality of their products. They state that quality in every discipline of the organization is on the top of the agenda. The company estimates that the direct costs of overall poor quality is in the order of 1 million euro every year, which directly influences the profit. Customer satisfaction is another reason why the quality of AME's products is important. A higher customer satisfaction will lead to more recurrent customers and a better reputation within the competitive market. Lastly, increasing quality is a key aspect for reducing waste thus increasing sustainability over the whole value chain. Waste reduction is important from a material, component, human and energy perspective, as well as working in a lean and efficient way. This project is in line with AME's *Call-for-Quality* of

2021, and will contribute to their quality vision by enhancing the process related to the printed circuit board assembly production.

Printed circuit board assemblies play a large role in the technology people use every day, which is why the quality of these products is of great importance. Smart phones, coffee machines, assembly lines or smart fridges depend on the functioning of their printed circuit board assemblies. Therefore, it can be seen as the core of most technological products. By manufacturing PCBA's, AME fulfills an important role in this technological product space. Improving the production process quality leads to several advances. If the production stability is improved, the overall performance will become more reliable. For the PCBA production process, full production stability means that the error slip is nullified, the First Pass Yield (FPY) is 100% and the Defects Per Million Opportunities (DPMO) is as low as possible, preferably 0. These indicate that no defect products are unnoticed and almost no reparations or scraps occur due to insufficient production quality.

The SMD production process consist of several sequential steps as described in Section 4.1. Placing the components on the panel requires a precise and controlled process. Product engineers and operators claim that they believe it is hard to find out why quality deviations occur during the process as the behaviour of the production line is considered relatively stable. As of today, it is unknown whether these deviations are incidents or whether they can be assigned an explanation. The quality of the PCBA's may be influenced by many (external) manufacturing features such as the component types and the process parameters. Quality can be seen as individual quality deviations on products or the total number of quality deviations between distinct production orders. This project will only focus on the component placement quality deviations on individual products (printed circuit board assemblies). Although there are presumptions regarding what production variables influence the quality of the printed circuit board assemblies, no extensive research is conducted on this topic within AME yet. Furthermore, the company also states that there is much data availability regarding the production process but that currently this information is not used to find explanations for quality deviations, as it is hard to structure these vast amounts. The latter problem will partly be tackled by this project as providing more insight in the production process through data analysis is one of the main objectives.

4.2.2 Production Quality Assessment

There are many factors of a printed circuit board assembly which could influence its quality (e.g. soldering issues, missing components or wrongly placed components). Due to this fact, it is not always obvious what determines the quality of a PCBA. During the SMD production process there are several moments of quality inspection, as briefly mentioned above. This subsection describes the general quality checks applicable to the surface-mount production of printed circuit board assemblies.

Before a product goes through the SMD production process, product engineers are responsible for adjusting the settings related to the automated inspections as described in Section 4.1. The fundamentals for the settings of these automated inspections are based on industry quality standards. Product and process specific knowledge of AME's engineers is used to further specify the inspection settings for each product. Examples of inspection settings are the minimum and maximum amount of paste volume margins on a board during the paste application, or the maximum rotation in degrees a component may have during the optical inspection. As PCBA's may have a vast amount of SMD components, finding the right quality settings for every part of the board can be a tedious work. Trade-offs must be made between the error sensitivity during the process and the strictness of the quality check. For example, smaller components are harder to place, which increases the chance of a wrong placement thus requiring a more precise quality check.

During the production process the quality checks can be divided into two distinct groups: single product quality checks and general process quality checks. The single *product quality* checks are described first. After applying the paste to the pads on a blank panel, the paste on each pad is inspected by sensors (automated paste inspection, or API) which measure the paste volume, area covered and the height. These values are instantly compared with the quality standards as set by the product engineers. Each paste location on the board automatically receives a quality assessment and based on the combined

outcomes, an overall paste quality assessment is given by the system. If the quality is good or there are no major flaws, then the product proceeds to the next production step. Quality messages are board location specific and could be *position* (indicating there is paste on a wrong place), *bridging* (occurs when two pads are accidentally connected by excess paste) or *insufficient* (meaning that there is not enough paste added to a pad). In case there are too much warnings or even errors, the production line is stopped as the product's quality measurements needs revision of an operator. Each operator which controls a production line is well trained and is able to judge whether the quality of the product is truly insufficient. If (despite the inspection warning) the paste quality seems acceptable, the operator overrules the automated quality inspection and the product may continue. However, if there is indeed a problem with the paste quality, then the product is taken out of the production line, cleaned from left over paste, and placed back at the start of the paste application process.

The next quality check for the printed circuit board assembly happens after the SMD components are attached to the board and the paste is hardened. This is an Automated Optical Inspection (AOI) which visually checks whether the placement of **each component on the board** is done correctly, in terms of both soldering quality and component positioning. Example warnings or error messages are *solder-fillet* (indicating that the soldering is not done correctly) or *polarity* (indicating that the rotation of the component might be incorrect). Again, the automated inspection can accept the quality without interference of an machine operator if it is acceptable. However, if the overall quality is uncertain, the operator must manually check the components for which the quality might not be sufficient. When the operator decides that the quality is acceptable, the machine's decision is overruled, resulting in a false machine call. Whenever an operator is also convinced that there is a quality problem related to one or more of the components, the product is taken out of the production line and brought to a separate repair station. An overview of the process flow is depicted in Figure 9.

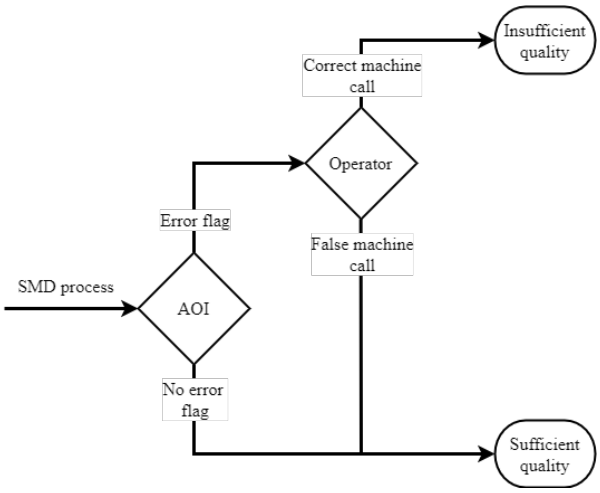


Figure 9: Component quality check process flow

The general *process quality* is measured by aggregated process quality measurements which are visualized in a quality monitoring system. AME created a dashboard with visualizations of traffic lights per subprocess in their production plant. A green light means no possible quality problems, a orange light means that there might be a possible problem and a red light indicates that an operator should intervene with the process in order to reduce the chance of quality deviations. The color of the lights are based on statistical process control charts, measuring different process quality parameters per subprocess. In some cases, the lower and upper bounds of the control charts automatically adjust to a potential shift (and sometime structural) shift in a trend. Finally, products which do not meet the correct quality are repaired at repair stations after the production process. During this repair process additional information regarding the quality deviations is stored, such as the type of component and the type of error. Eventually, all quality information gathered during the production process is stored in the BI system of AME. This BI system aggregates all information regarding the different inspections and repairs during the process

to a production order level. This summary of the overall production quality includes several performance measurements per product type, batch or production order.

4.2.3 Business Problem Statement

Error slip is the undesired result of the AOI where the quality of an inspected component is wrongly classified as good (Lin & Su, 2006). To ensure the highest possible product quality, error slip must be minimized or even nullified. By adjusting the inspection toleration boundaries of the AOI sharply, error slip can be prevented in almost all cases. Due to the fact that error slip is undesired in electronics manufacturing, using sharp tolerance boundaries is common practice in this manufacturing area (Thielen et al., 2020). A downside of this choice is that by reducing the number of error slips, the number of false calls increase. Components which are not erroneous are falsely classified as being so due to the strict inspection tolerances. It is virtually impossible to fine tune each inspection program so that no false error flags are raised, as the tuning is based on trial and error and thus a time intensive task. Furthermore, several other difficulties are present when using an automated inspection of the components and their solder joints. First, the reflective surface of the solder paste does not always reflect the light of the inspection machine in the same manner, leading to misclassifications as a result of differences in lightning. Second, there exists variety in the solder amounts applied to the pads which can lead to variations in soldering shapes per product. This is also a barrier to developing an automatic solder joint inspection system (Mar, Yarlagadda, & Fookes, 2011). Third, different component types (even when they have the same purpose) may also induce problems when designing an automated optical inspection. Each product type (printed circuit board assembly) has a unique board layout therefore requiring a unique automated inspection program. Due to the fact each component inspection requires its own settings, reducing error slips without causing false calls is a very tedious task for the product development engineers.

The occurrence of many false calls in the inspection system is a common problem in electronics manufacturing. In case of AME, on average 62% of all panels produced yearly during contain one or more false calls. Whenever only one component on the board does not work as intended, it is possible that the whole board does not function. Therefore every error call requires additional attention from an operator. To avoid scrapping or repairing good components each call is manually checked by an operator. Checking all the calls during the production of the printed circuit boards can be a time costly action. Operators are not always present at the review station because other locations of the production line require attention as well. For instance, if an operator is checking the screenprinting calls or changing a component reel the operator cannot check the machine calls at the automated optical inspection. AME also states that it should not be the main task of an operator to check all the machine calls as this distracts from controlling the rest of the process. Moreover, high false call percentages during production increases the chance that an operator will miss a real defect due to negligence (Ellenbogen, 2006). Besides the increased chance on reduced product quality in case of many false calls, the efficiency of the manufacturing line is also reduced. Whenever a panel requires an inspection, it is stopped at the review station which can cause congestions at the production line, reducing the throughput per hour. Hence, false calls are the result of reducing or nullifying the error slip to increase the product quality but a vast amount of false calls are not desired for two reasons. Too many false calls can result in the opposite of the desired quality objectives as the negligence of operators may again lead to error slip, and it furthermore reduces the throughput of a production line as it may cause congestions.

4.2.4 False Call Measurements

This section provides a brief overview of the magnitude of the false call problem. False call rates from the most recent year are summarized and some example production orders with a large false call percentage are given. It also briefly elaborates on the losses in terms of efficiency and costs. The Business Intelligence (BI) environment encapsulates much data regarding different topics such as finance, service, sales and operations. The aggregated production quality measurements are stored in the operations category of the BI system. Different quality measurements are stored in different data aggregation levels: quality per test location, per production order, per product type, per month, etcetera. Test locations are places in the process where the quality of products is assessed (API and AOI). Production orders (PO) are created by production planners based on demand generated by sales orders placed by a customer. A PO

is unique and belongs to a certain product (PN). Thus, a PN can have multiple unique POs, but a PO can only belong to one single unique PN.

In the recent year AME produced 210,505 panels on the SMD production line and 63.34% of those panels contained one or more false calls at the AOI. This means that 133,324 panels in the last year required a manual inspection without a real error being present. With an average of 2.75 calls per erroneous panel, 366,641 false calls occurred in the last year. For the API, the paste inspection earlier in the process, approximately 65.57% of all panels contained a false call. Although this inspection location is outside the scope of this research project, it shows that these additional manual checks induce avoidable load on the production line operators. An overview of these values can be seen in Table 2.

Table 2: False Call Statistics September 2020 until August 2021

Panels with False Call(s)	Average calls per panel	Total False Calls	Total Panels	Panel Call Rate
133,324	2.75	366,641	210,505	63.34%

When inspecting the false call ratio per month over the recent year (Figure 10), it seems that the average false call ratio per month is increasing over the months. The false call rate depends on the number of panels which have one or more false calls. A possible reason could be that in the most recent months, products are produced having more components per board, thus increasing the chance of false calls. To take this into account the Defects Per Million Opportunities (DPMO) is used. An opportunity is defined as the placement of a single component on a board. This way the false call measurement is corrected by the total components placed. As can be seen in Figure 10, the DPMO also increases when the false call rate increases. This indicates that the incremental false call rate is not caused by the fact that more components are placed. These statistics confirm that AME is experiencing problems with the false call rates at their SMD production lines.

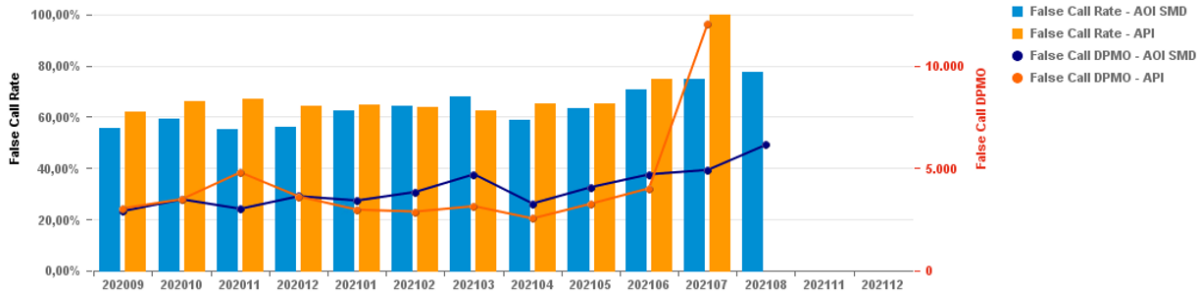


Figure 10: Average false call rate per month

Furthermore, it is not an exception that a panel has more than one false call, with the average false calls per panel being 2.75. Table 3 shows the product types with, on average, the most false calls per panel. These values show that false calls on a panel might require quite some work from the operator. One can imagine that when a panel has 18 false calls that the operator will be less attentive when checking all the calls on a panel. Especially not when this is the case for more than half of all panels which is produced.

Table 3: Top 5 most false calls per panel

Product Number	Average false calls per panel
6661-1900-0701	18.31
6649-1900-0600	17.56
6045-2000-4101	17.31
6880-2000-4001	16.76
6045-1904-5700	16.05

The 63.34% of panels with false calls is the yearly average, there are product types with a much higher percentage. Figure 11 shows the product types with the most panels with false calls in the last year.

In case many panels require more than one manual inspection due to multiple false calls per panel the attentiveness of the operator is likely to decrease. As an example, the false call statistics of the production orders of product 6047-1800-9204 are shown in Table 4. For each production order, almost all panels require manual inspection by an operator. In case of production order 3, almost 11,700 manual inspections were required during the production order. Although these are extreme cases, it confirms why reducing the number of false calls can be beneficial both in terms of operator load and production line efficiency.

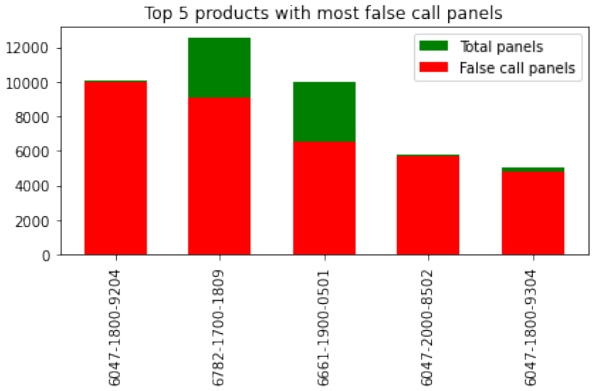


Figure 11: 5 product types with most false calls in the recent year

Estimating the time it takes to check a machine call is not trivial as it is depending on many factors. Ellenbogen (2006) estimated that when the operator is seated at the reviewing station, it takes approximately two seconds to review a single call. AME states that a false call is lost time per definition, as every error call congests the production line. The amount of time it takes to resolve an error is dependent on the operator and the number of problems occurring during a production batch. From an operator perspective it is dependent on how keen the operator is during its shift. This depends on the fatigue, working hours, training amount, experience, and certainty related to a given error. Furthermore, when many (false) machine calls are raised for each board in a production batch the attentiveness of an operator will likely decrease. The number of calls in a batch can depend on for instance the size of the board, but also how many times a product is already produced before. AME assumes that whenever the average calls on a board is less than five, the operator is more likely to check these. If there are more than five calls per board, the operator becomes less attentive after a few boards and just ignores all other errors. However, closing each individual flag without further inspection also takes time. Even if an operator does not do any manual checks, time is lost by clicking on the button to close each call. If the operator is less attentive it is also possible that real errors are assessed as false error flags, which can result in insufficient quality for the customer. A customer returning a defect product is very time consuming and costly thus not desired. It is however unknown what proportion of false calls actually are real errors. Concluding, AME estimates that a decent manual check of a component requires approximately five seconds. As not all flags are inspected well and due to the buffers in the production process they estimate that a machine call results in a congestion time of two to three seconds, which is in line with the literature related to this topic. This rough estimation will therefore be used in the remainder of this research.

Table 4: False call statistics for product 6047-1800-9204

Production Order	**Panels with false calls**	**Total panels**	**False call rate**	**Average false calls per panel**
1	98	100	98%	5.4
2	397	398	99.75%	5.51
3	1800	1805	99.72%	6.47
4	603	603	100%	5.73
5	1196	1202	99.50%	6

4.3 Data Sources

Data is one of the main drivers of this project which is why it is important to have a clear view of what data is relevant and available. Besides the application background it is important to define the data concepts within AME relevant to the research project. First a brief overview of the different information systems is given, serving as data sources. Then the identified data concepts related to the product quality of the SMD production line are defined based on interviews with product and process engineers. After defining the relevant data concepts, the availability of the concepts and the data gathering process is addressed.

The manufacturing process of AME is supported by several information systems. Besides the daily operational function of these systems, the data stored here may also serve as an additional value for the process control. Most important are: the Product Data Management (PDM) system, the Enterprise Resource Planning (ERP) system, the Manufacturing Executing System (MES) and the Business Intelligence (BI) system. PDM is done with Orion Client, serving as the *definition* layer. All relevant information of products, tools and equipment is stored in this system. This layer also provides information regarding the components (bill of material) per product to the ERP sytem. The ERP system serves as the *planning* layer, which is done with SAP. This layer is used on a daily operational bases to support the production, store information regarding the demand, the need for material, maintenance or service and support the finance department. The MES serves as the *execution* layer and is executed via the Orion Board Administration software. This layer includes data regarding the traceability of the products, product quality tests and machine log files. Finally, SAP Business Intelligence serves as the *reporting* layer, used for presenting the data in an readable manner for the entire organisation. An overview of the data sources can be seen in Figure 12.

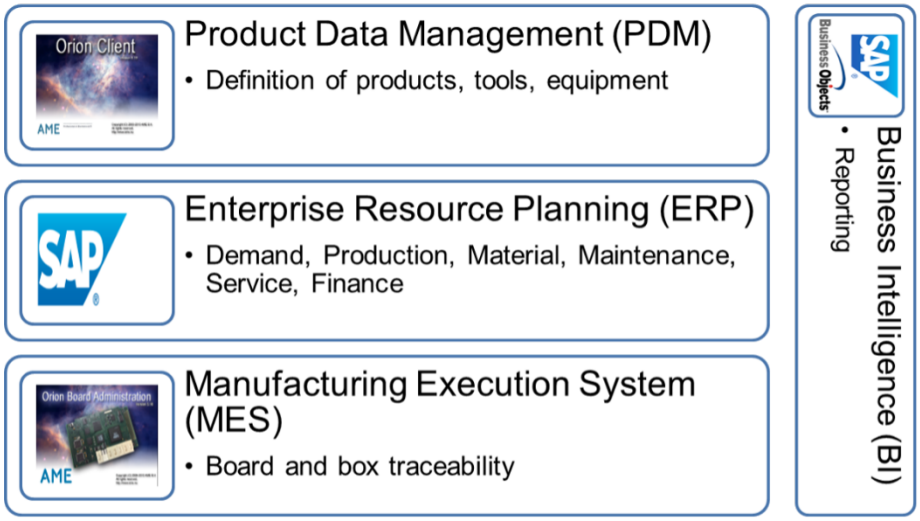


Figure 12: Overview data sources

Due to the fact that this thesis is about analyzing production execution data, the most important data source for this research is the MES, as this includes data for the product quality and possible relevant process features. In the next section a brief description of the relevant data is given.

4.4 Relevant Data

Besides the different data sources, (online) production data related to the product quality can conceptually be divided into several hierarchies or levels: the product number (PN) level, the production batch level (PO), the serial number (SN) level, the component level and the board location level (RefDes). Dividing the data into these separate levels is only needed to create additional data understanding. It provides a conceptual framework to show how features differ between product types, batches or individual products. Before diving into the conceptual levels an example is given for the above definitions

to enhance the general understanding. Figure 13 depicts different concepts related to the PCBA production. The left most image shows a PCBA from an example product type (PN) with i components attached on unique board locations (RefDes). When these PCBA's are produced, the empty boards (without components) enter the production line in panels so they can be handled in standardized sizes. The panel is identified by a serial number (SN) and contains m boards. After production these boards are depanelized (removed from the panel) so the PCBA's can be assembled or sold individually. The products are produced in batches of n panels and a batch is produced on one production line. However, a production order (which is the total quantity sold to the customer and related to a sales order) may consist of several batches which can be produced at different production lines and on different moments in time.

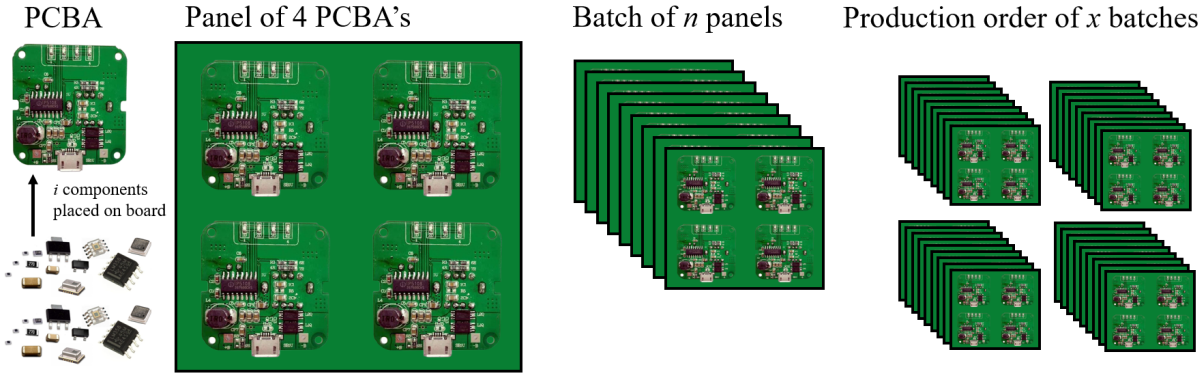


Figure 13: PCBA Concepts

In the remainder of this section all relevant product and process features are described and assigned to a data level. The relevance of these variables is based on the domain knowledge of the product engineers, gathered via interviews. Data withing the scope of this project is related to the the surface-mount device production lines of AME. Each sub process, as stated in 4.1, logs data related to that sub process. No centralized or readable data sources are in place for the machine settings of each sub process for each product. Therefore, gathering this data is only limited to the process parameters which are stored in the log files related to the sub processes. The remainder of this section will provide an overview of the data concepts which might influence the product quality (as appeared from interviews with process and product engineers) and the data availability of these concepts. Note that not all described data concepts could be gathered for this research. Nevertheless, as some might be useful for future research it is chosen to describe them anyway. Before describing the levels and corresponding features, an overview is provided in Table 5. Descriptions of the features described are found in Table 28, Appendix B.

Table 5: Data levels and associated features

	Features	
Data level	**Available**	**Not available**
Product type		Panel thickness Soldermask finish Pad surface Stencil thickness Transfer efficiency Squeegee width
Batch	Print speed Print force Snap off distance Snap off speed Paste type Zone temperature settings Conveyor speed	Number of strokes Stencil cleaning interval Gluing Placement speed Pick up method Component rotation Bottom support
Serial number	Position in batch Time interval products Printing temperature Printing humidity Measured reflow temperatures	Board calibration
Component	Component package Moisture sensitivity Supply form Placement errors	
RefDes	API features AOI quality assessment Error type Operator review	

4.4.1 Product type level

Product type level features define a given product type (PN) and only differ between product types. In terms of placement quality, there are several features on product type level which are considered important, related to the panels, the PCB and the components on the product. Panel features can influence the product quality in several ways. Deviations in the thickness (in millimeters) of panels may result in placement problems due to wrong calibrations. Each panel has pads where soldering paste is applied in order to attach the components on these specific places. The remaining area on the board (where not paste is required, thus no components are placed) is soldermask, a material that does not mix with soldering paste (tin). The soldermask determines the color of the PCB, and is green by default. However, when PCB's deviate from this standard color, the soldermask finish can be thicker than in the standard case, resulting in screen printing deviations. Not all pads are the same for each product type. In the standard case the pads on a PCB are a copper surface, but sometimes there is a pre-applied layer of tin present on the pads. This can lead to a skewed placement or printing deviations.

Other product type specific features are replaceable machine parts, designed especially for specific product types. A stencil is used to apply the paste on the right locations on the board during the screen-printing. It is a metal plate with apertures corresponding to the pad locations of the printed circuit board. When the paste is pressed over this stencil, these apertures ensure that the paste ends up on the pads. Paste application is thus greatly influenced by stencil thickness and the transfer efficiency of the apertures (Khader & Yoon, 2021). When a stencil is thicker, more paste can be pushed on the board, possibly influencing the paste volumes. The thickness also affects the transfer efficiency of each aperture, which is calculated by using the aperature's circumference and its height (which is the bound to the stencil thickness). The transfer efficiency value determines how much of the paste is transferred through the holes in the stencil. In some cases, when the transfer efficiency is not 100%, leftover paste

gets stuck in the apertures which can negatively influence the paste application. In order to sweep the paste over the stencil, a squeegee is used. This is a smooth rubber blade which controls the flow of liquid over the stencil surface. The width of this squeegee is product type specific and may also influence the paste application process.

4.4.2 Production batch level

The production batch or production order level regards the available data related to a given production batch. Each product type can be produced during multiple production batches. The machine settings and process parameters belong to the batch data level. In spite of the fact that machine settings or production parameters seem to be product type specific, for some cases the manufacturing settings or production parameters are changed in between batches by the process engineers. Therefore, these features may be different for the same product type. Each sub process has its own set of machine settings and process features. Screen printing requires several machine settings. Printing speed and printing force determine the squeegee speed in millimeters per second and the squeegee force in kilograms. The amount of strokes the squeegee makes also influences the paste quality. After applying the paste, the board and the stencil are separated. The separation speed (in meters per second) and distance (in millimeters) can be controlled and might influence the paste quality on the board (e.g. when leftover paste sticks to the stencil). During production, the stencil is cleaned after a set interval of products. This cleaning interval can also be controlled and when this interval is too large, left over paste might negatively influence the outcome of the paste printing process and thus the placement quality. Process features related to the screen printing process are the paste type and whether the board requires additional gluing to strengthen the attachment of components.

During the pick & place process, the process engineer must design the settings for each placed component separately. These settings are fixed for each batch. Features which can influence the quality are the placement speed, the pick up method, whether the component requires rotation and the board bottom support. Placement speed determines how fast the component is placed on the board. Larger components are placed with less speed to reduce placement accuracy issues. Components are picked up with a vacuum grip or mechanical grip. Slight differences can be present when comparing both methods on picking stability. Component rotation is needed when the component is not placed linearly on the board. Whenever this is the case, there is more room for error due to an extra machine setting and handling. Bottom support is the force applied to the bottom of the board when the components are placed. Having a board support which is too low might reduce the stability thus the placement quality. Reflow is a relatively simple process, consisting of ten heating zones and three cooling zones. During the heating and cooling zones the temperature respectively increases and decreases in a linear fashion. The shape of the soldering joints may vary when the heating zones are not stable or too hot. The conveyor speed in the reflow process also influences the heating process. Moving too fast might cause the temperature to shift between the heating zones as the board moves the air through the different zones. All the above settings and parameters influence components and products on an individual level but as they are constant for the entire batch they fall into the batch data level.

4.4.3 Serial number level

The serial number level can also be defined as the panel level, which contains data on the individual product level. Each panel which passes the production line has a unique serial number, based on the batch number, the year and the place in the batch. Features related to this data level are mainly measured by sensors during the production process. These features are thus constant for the entire panel. AME did some research related to product quality within a production batch and found out that the sequential position of a product in a batch might influence the quality of the product. For instance, they found that paste quality might deteriorate when the machine is idle (e.g. during breaks). Therefore, features regarding the product's position in a batch and the time interval since the last produced product in a batch can influence the product quality. During screenprinting process the environmental features temperature and humidity are measured within the machine. Both influence the paste fluidity thus the ease of applying the paste to the board. When a individual panel enters the pick & place process, the calibration of the board is measured. This feature provides information about whether the board is well aligned relative to the nozzles. Incorrect calibration can lead to misaligned component placement,

reducing the quality of the product. To measure the temperature during the reflow process, sensors are installed at each zone. Since the actual temperature may differ from the set temperature, it is interesting to include this feature as this can influence the soldering quality. The temperatures are constant for a panel but can differ between panels due to the movement of air through the machine.

4.4.4 Component level

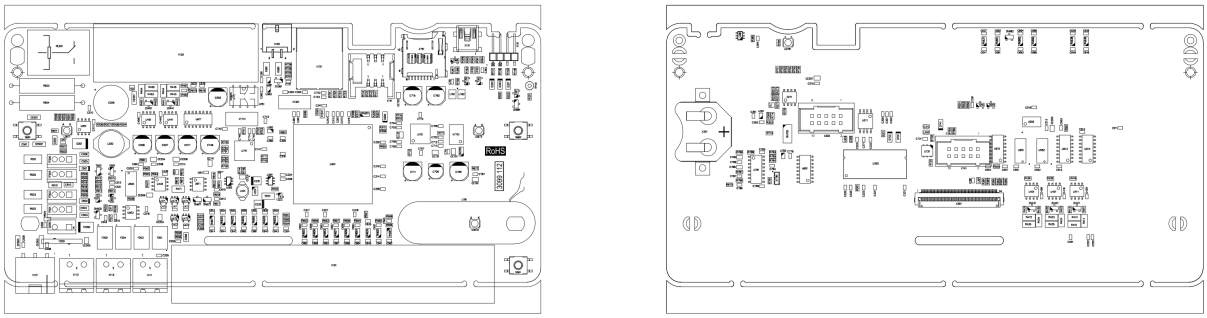
Each product type has its own unique combination of different components, which can be used to describe each product type. However, a component type can be placed multiple times one the same product type and on multiple product types. Therefore the component characteristics are described on the component data level. The data related to each component can be divided into component characteristics and process information related to the component. Features describing a component type are its package, whether the component is moisture sensitive and the supply form. Component packages are industry wide type categories for surface-mount devices. The package code encapsulated information related to the component types (e.g. resistors, capacitors, coils, diodes), geometrical dimensions or number of leads (the pin which connects the component with the pad through the soldering paste). These component types can influence the placement quality. Smaller components are harder to place as there is less tolerance for mistakes. If a component has more leads, placing the component is also more prone to errors. Another component feature is whether components are sensitive to moisture, therefore require special handling procedures. The supply form of a component can also affect the placement quality. Most components are supplied to the machine via tape, which is the most stable supply form. Some components are picked from a tray, this form is less stable as these components can easily shift or rotate while on the tray. Besides the process characteristics, information (e.g. pick up errors) is gathered for each panel during the process related to the placement scores of the component types on that panel. For example, during production of product type *6736-1504-2007*, on panel *A* there occurred 8 pick up errors when placing component *2000-3003-1004* (which is placed on 12 locations on the panel), but on panel *B* only 1 error occurred for that same component type. These placement error features do thus not belong to the serial number (panel) data level, as they differ for each component on a specific panel.

4.4.5 Board location (RefDes) level

The board location data level has the lowest data granularity as it describes a specific location on a particular printed circuit board assembly. Each PCB has its own set of locations where components can be placed, and all of these locations are identified by a unique identifier (which is called a reference destination, or RefDes), see Figure 14 for an example. In most cases a RefDes is described by a letter and one or more numbers, and it is bound to a product type. Meaning that for product type *A*, the RefDes R102 can be located in the upper right corner, but for product type *B* RefDes R102 might be located in the middle of the board. Thus, these names do only serve as a unique identifier per product type and do not provide any information related to the location on the product type. Figure 15 shows that for each board of a given product type, the RefDes defines a unique location on that product type. Hence, when there are 4 boards in a panel, the panel has 4 RefDes with the same name, which are distinguishable in combination with the board identifier in the panel (e.g. U301_1, U301_2, U301_3, U301_4).

Both the API and AOI operate on the board location (or RefDes) level. After the screenprinting process, the API checks the paste on every board location based on several features: paste volume, paste height, paste area, offset in the X direction and offset in the Y direction. The latter two features describe whether the paste is applied at the correct coordinates related to the pad. The volume, height and area features provide information related to the amount of paste applied on the pads. At the end of the SMD production process, the AOI checks the quality of each component on the separate board locations. If the automated inspection evaluates the component placement on a RefDes as insufficient, the error flag type is added to the call. For each error flag found by the machine, the operator must manually check the call, deciding whether the machine made a false call or a correct call. If both the machine and the operator agree on the fact that the quality is insufficient, the whole board is brought to the repair station. Whenever the repair station operator thinks no repair is required after checking the error, the correct call will be replaced by a false call for that component. A false call can thus occur when the machine and the line operator disagree, and when the repair station operator overrules the decision of the AOI

and the line operator. Consequently, the earlier described product quality and the false call indication is defined at the lowest data granularity level, which is the board location (or RefDes) level.



(a) Top of PCB

(b) Bottom of PCB

Figure 14: PCB design with reference destinations

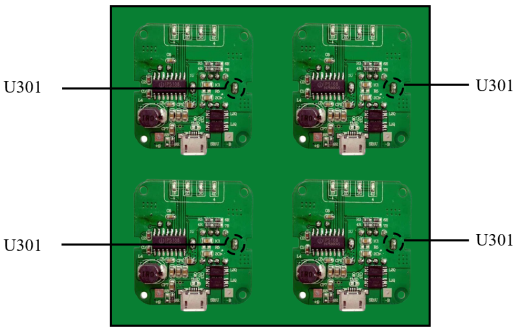


Figure 15: RefDes example

4.4.6 Process and data

In order to provide a better view of the data related to the process, Figure 16 depicts the data by linking it to the relevant sub processes. A distinction is made between the log data (which is gathered during the process) and the process parameters (which are set before the production starts). If a real error is found at the AOI, the product is brought to the repair station. Thus, data regarding the type of error and whether a call is false or are generated at both the AOI and the repair station.

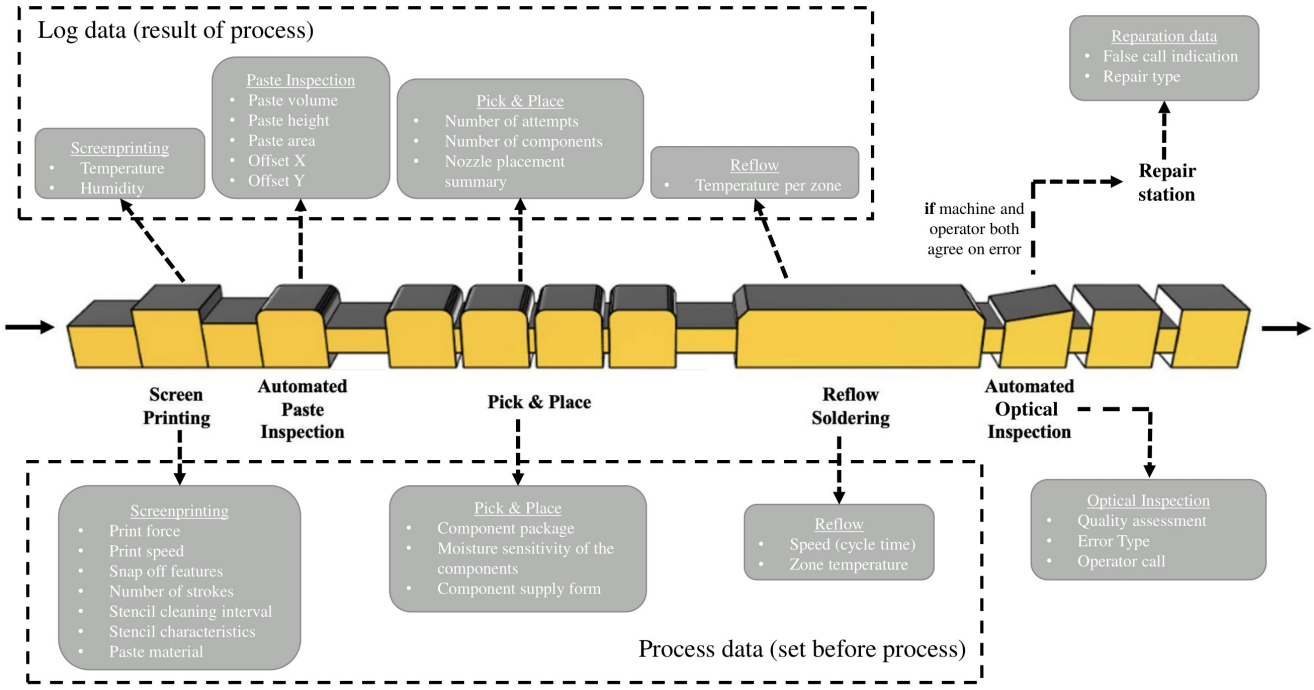


Figure 16: Data concepts linked to the SMD line

4.5 Data Gathering

After defining the data concepts it is required to gather a representative data sample in order to conduct analyses. This section briefly elaborates on the data gathering process, data sources, data availability and provides an example of the gathered data set.

A notation is used to introduce the different data set concepts. Let $\mathbf{x}_i$ denote the feature vector of all product and process features for sample i. The categorical target variable for sample i is depicted as $\mathbf{y}_i$, with $\mathbf{y}_i = \{\text{good}, \text{false call}, \text{real error}\}$. Each sample represents the inspection of a placed component on a board, given all the product, board, production and component features of that instance. All the gathered instances combined form the complete data set $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}$. The training, validation, and test sets are denoted by $\mathcal{D}_{train}$, $\mathcal{D}_{val}$, $\mathcal{D}_{test}$ respectively, where each of these sets is a subset of $\mathcal{D}$. Ten different PN's were randomly chosen to start the data gathering. Choosing randomly reduces the potential bias in the data set and increases the probability on a representative sample. Before March 2021, there were no log files available for the reflow production process, which is why this date is chosen as a starting point. The chosen product types were produced in totally 52 batches from March 2021 to July 2021. Some products have a top and a bottom side (see Figure 14, and due to the fact that each product side has its own settings and features, a PCB side (top or bottom) is treated as a single production batch instance. This results in 83 batch instances of 10 different product types.

During the gathering process, several problems occurred related to the data availability. First, in case of some variables, the data is not available at all or not available in any utilizable form (e.g. information related to the soldermask finish is only stored in the text of PDF files). Second, other features, such as the panel thickness or the gluing specifications, are only gathered by process engineers in local data files (e.g. Excel sheets) without being stored in a company wide database. This does not contribute to the completeness and quality of the data. Third, the machine settings of the pick and place process are embedded in a software environment which cannot easily return the settings in a readable format. Furthermore, the (large) database connected to this process is developed and configured by the machine supplier. This leads to difficulties when there is a need to extract the correct data from the system. Due to this fact only summarized error messages per component per panel could be extracted for the pick & place process. Fourth, some data is supplied by third parties. For example, the stencil data related to

the stencil thickness and the transfer efficiency of the apertures is supplied by the stencil manufacturer. Lastly, the above mentioned issues not only result in incompleteness of the data but also bring forth merging issues. In case of the transfer efficiency per aperture related to a board location (which can be very useful for analysing the quality of the product), it is not possible to merge the aperture's efficiency to the board location because the stencil location identifiers do not match with the PCB location identifiers (RefDes) of AME. Besides the lack of correct primary and secondary keys in the different data sources, the missing data in some of the sources also results in highly incomplete and therefore useless data records. Features for which there was too much missing data were disregarded for further data gathering.

The result of exploring the data gathering possibilities regarding the availability of each feature can be found in Table 5. It is interesting to see that the product type features are the least available. All features with sufficient quality or amounts were incorporated in the data gathering to create the raw data set. Most gathered data is scraped from log files as a result of the production process. Each panel has its own log file per sub process. For the API and the AOI these log files contain information per pad, and multiple pads can be associated with a RefDes. There are a few assumptions made when reading the log files. The API checks each pad on the board, but the individual pads are not within the scope of the research. Therefore, the paste inspection values of multiple pads belonging to the same RefDes are aggregated by taking the mean values. For the AOI, whenever an error is found at a given pad, it is assumed that this error relates to the associated RefDes. When multiple pads of a RefDes contain an error, the most occuring error type as proposed by the machine is assumed as the main error type associated with the RefDes. Information related to the components are extracted from the ERP system and merged via the component identifier.

After collecting and merging all the data, a set is created representing 9 product types and 53 production batches. The raw data contains approximately 7.4 million instances (or rows). Each row represents a board location on a specific panel which is checked by the AOI at the end of the process. The features describing the row are all the process features as described in Table 28, Appendix B. A unique row is described by a combination of primary keys namely the PN (product type), SN (serial number), batch identifier, board identifier in a panel, and the RefDes (unique board location identifier). The previously described data levels associated with each feature describe whether variables are constant between rows or not. For instance, if a specific panel contains 200 components then there are 200 rows associated with that panel. As each placed components has information related to the paste inspection, the rows differ for these features. However, all components on the panel went through the reflow zones at the same time so the reflow temperature features are constant for all these 200 rows. Figure 17 provides a general example of the variability in the data levels for one given product type. The colors in the columns depict how the data varies in and between the data levels. The most variability can be found on the board location level, and the least on the product type level. Before building any machine learning models it is first required to understand the data well. In the next section an exploratory data analysis is conducted which provides useful information regarding constructs like the data distributions, multicollinearity and relations with the target.

Batch identifier	Board identifier	RefDes	Product type level	Production batch level	Serial number level	Component level	Board location level
A	1	C100					
A	1	D300					
A	2	C100					
A	2	D300					
B	1	C100					
B	1	D300					
B	2	C100					
B	2	D300					

Figure 17: Data set dummy example for one product type to show variability within and between data levels

5 Data Exploration

Exploratory data analysis is an important facet in the comprehensive world of data science. This important step of the data understanding phase in the CRISP-DM framework summarizes the gathered data sets by using statistical measures and data visualizations. Besides describing the main characteristics of the data, exploratory data analysis is also useful for formulating hypotheses. These hypotheses can lead to either new ideas for further data collection or insights which are useful during the following phases of the CRISP-DM framework (e.g. feature engineering or data modeling). This section will first elaborate on the data understanding by performing an exploratory data analysis. After the data understanding, the data preparation phase of CRISP-DM is described, including data preprocessing and feature engineering.

5.1 Exploratory Data Analysis

Initially the data gathering started with 10 PN's, consisting of approximately 83 production batches. However, due to merging different data sources together, missing data in one or more of the sources resulted in a dataset of 9 PN's and 59 production batches. These batches include 8795 produced panels which can have one or more PCBA's (see Figure 13). As each panel can have a top and a bottom (which are produced separately), the data contains 15,253 panel production instances. After depanelization (cutting the boards from the panel), the data contains 22,846 individual PCBA's which are sold to the customers. On all these boards approximately 7.4 million components were inspected, each row representing a component with its corresponding process features. These inspections include 36,347 false calls and only 750 real errors, see Table 6. Besides the error flag, the machine also provides an error type to the data. These error types and their false call ratio is shown in Table 7. For the most occurring error types, coplanarity has the most real errors. This error occurs when not all leads are connected well to the board due to component quality deviations, soldering or placement issues. Other interesting errors are polarity and OCROCV because relatively much error flags are raised here by the machine but almost none are real errors.

Table 6: Target class imbalance

Inspection result	**Instances**	**Ratio**
Good	7,388,342	0.9951
False call	36,347	0.0048
Real error	750	0.0001

Table 7: Error types and false call rates

Error type	**Total**	**Real errors**	**False calls**	**False call percentage**
Pad overhang	13,153	119	13,034	99%
Coplanarity	7365	279	7086	96.2%
Solderfillet	7018	42	6976	99.4%
Polarity	5954	7	5947	99.9%
Missing	1654	194	1460	88.3%
OCROCV	1525	1	1524	99.9%
Dimension	237	40	197	83.1%
Bridging	180	66	114	63.3%
Absence	11	2	9	81.8%

These values might not really endorse the problem stated earlier. Nevertheless, 58% of all the panels produced in the sample had one or more false calls which required a manual check, see Table 8. Relative to the number of inspections performed only few false calls occur. Still more than half of the panels need manual inspection, thus the false calls form a real issue during SMD production. An overview per product type can be found in Table 29, Appendix C.

Table 8: Distribution of false calls over panels

One or more false call(s) on panel	Number of panels
Yes	8,918
No	6,335
Total	**15,253**

5.1.1 Errors in batches over time

Exploring the data starts from a high level, finding whether there are differences between product types or batches. As can be seen in Figure 10, Section 4 the false calls increase over time during the last months. As time might be important, the batches are chronologically sorted and visualized in Figure 18. Each bar represents the correct error flag ratio. The same trend is seen, as there is an increase in the number of false calls in the most recent weeks, having a correct call ratio of almost zero. Furthermore, no clear relation is found regarding the product types and the correct calls. Therefore it might be interesting to see which other factors (e.g. component types or board locations) cause the error flags in the batches.

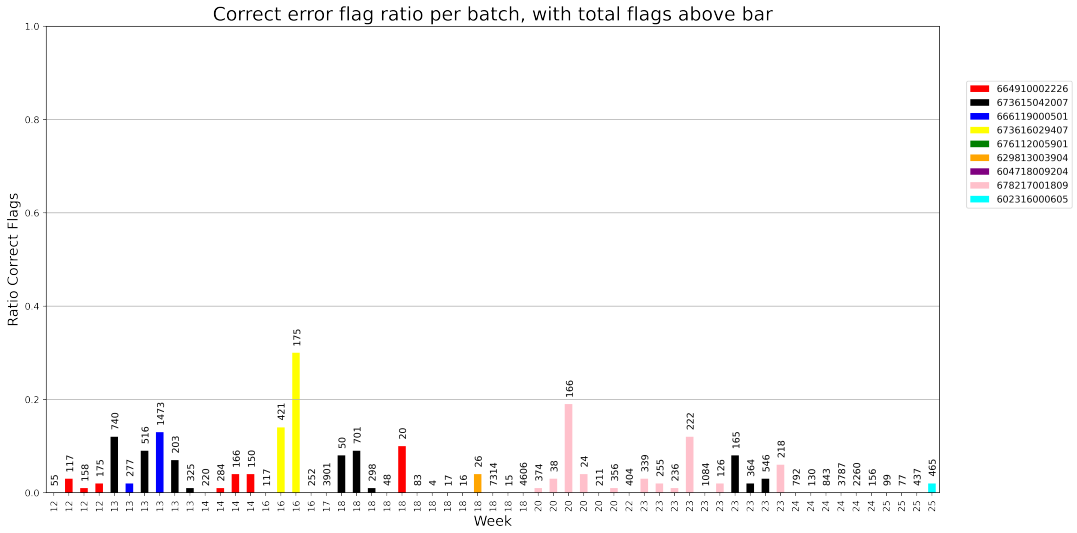
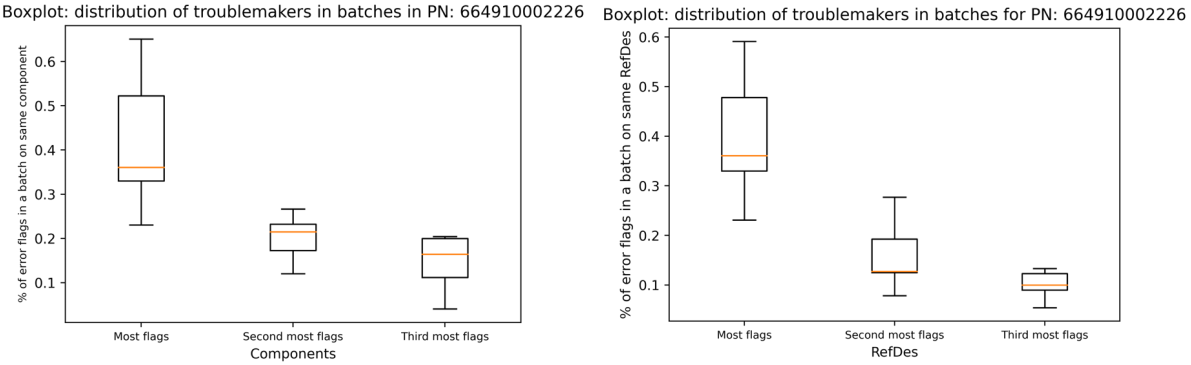


Figure 18: Correct flag ratio for batches over time

5.1.2 Troublemakers

Line operators and production engineers state that during some batches, error flags are caused by specific components or board locations, which are defined as *troublemakers*. A troublemaker is a component package or board location which causes the most (false) error flags. To check whether this observation is true, the error flags are analyzed per batch. For each batch the error flags are counted per component type and board location, providing the troublemaker and its error ratios for the batch. These ratio's can be compared between batches to find in which extend a troublemaker is causing problems. As an example, the comparison of these distributions for product type 6649-1000-2226 is visualized with boxplots in Figure 19. For this PN the component distributions (Figure 19a) show that 25% to 65% of the error flags are caused by the same component package (which can be on multiple board locations). Then there is a sharp decrease in terms of the distribution domain for the second largest troublemaker (approximately 10% to 30%). The domain of the third largest component type troublemaker is even smaller but decreases less sharply. This is an indication that each batch has a large troublemaker accompanied by smaller troublemakers. Although it cannot be generalized that all error flags are caused by only one or two component types in a batch. When looking at the RefDes distribution, as seen in Figure 19b, we see a similar pattern. Nevertheless, the component type and RefDes distributions differ a bit, which might indicate that error flags are either influenced by components or the board locations. To further substantiate the hypothesis about troublemakers, a t-test is conducted comparing

the distributions of the trouble makers. The result of this test also indicates that there is a statistically difference ($p<0.05$) between troublemakers and non-troublemakers. The above findings were also found when analyzing the other product types so can be generalized for the entire sample.



(a) Component type

(b) RefDes (board location)

Figure 19: Boxplots of troublemaker distributions over batches

After the confirmation that batches contain troublemakers it is interesting to see which component types and board locations are troublemakers. The biggest component type troublemaker per batch is depicted in Figure 20. There is a clear distinction between troublemakers of product types when comparing the top or bottom of the products. Furthermore, the troublemakers can reoccur between similar batches within the same week but also in later produced batches. Problems can thus be both product type specific or batch specific. Either way, the data shows that taking the component types into account during the modeling phase can be fruitful for the modeling results. Reoccurring component type troublemakers can also be the result of board location problems (e.g. due to stencil quality deviations). Therefore the same exploration is also done for the board locations on a panel, as shown in Figure 21a. When comparing both Figure 20 and 21a, it is notable that for some batches the ratio decreases and for some batches the ratio stays relatively the same. Indicating that respectively, for some cases the problems are caused by the component types and in other cases the problems are caused by the board locations. Furthermore, the reoccurring troublemaker pattern over batches for the same product type as seen for the component type, is also present for the board locations.

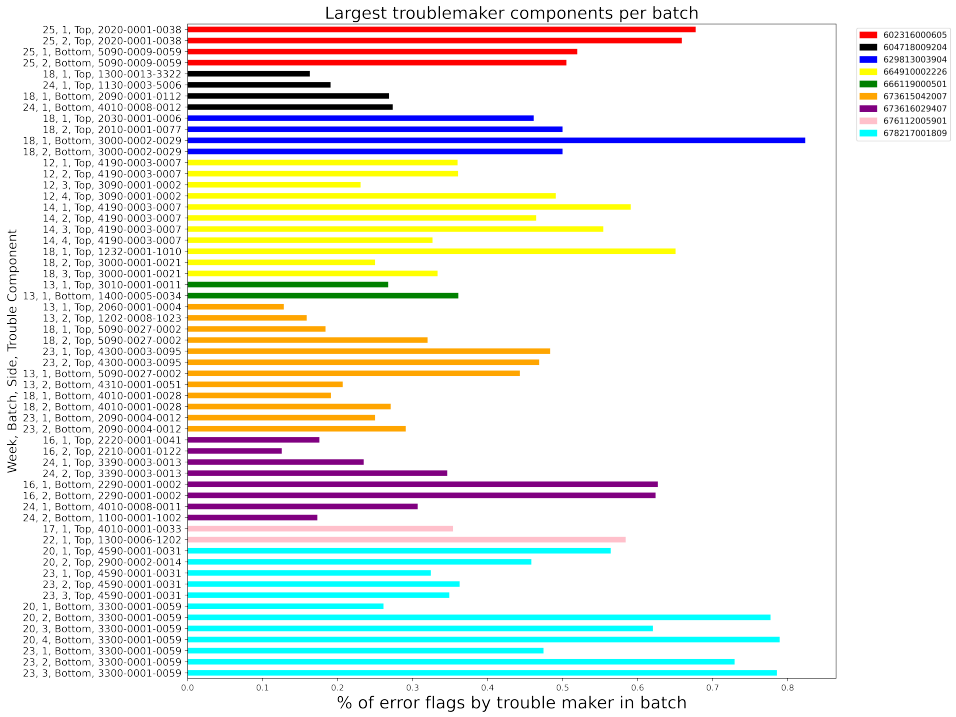
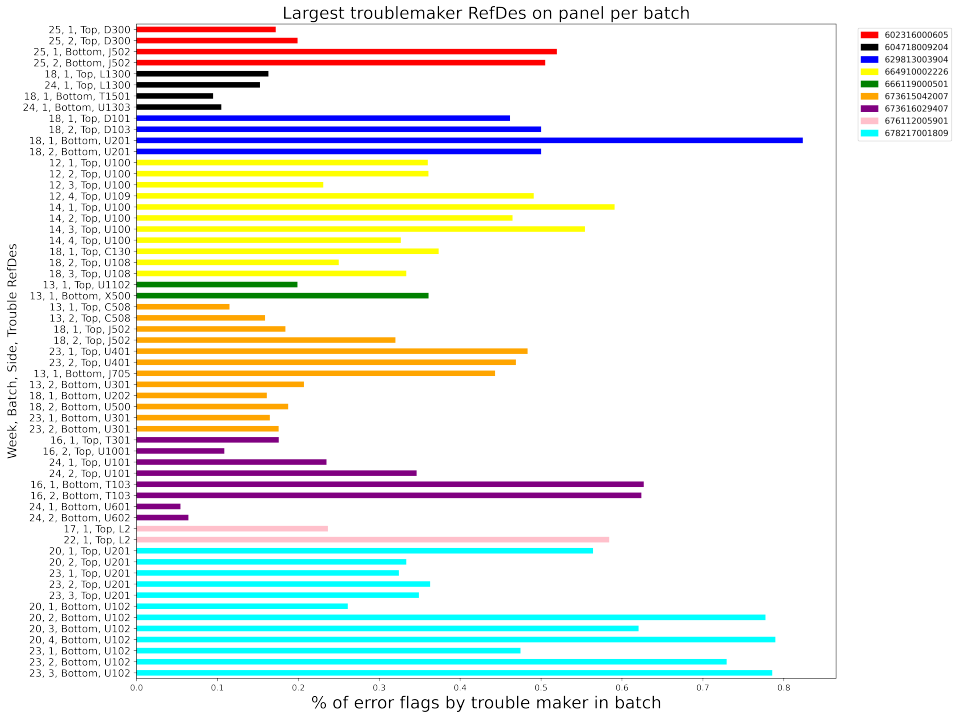
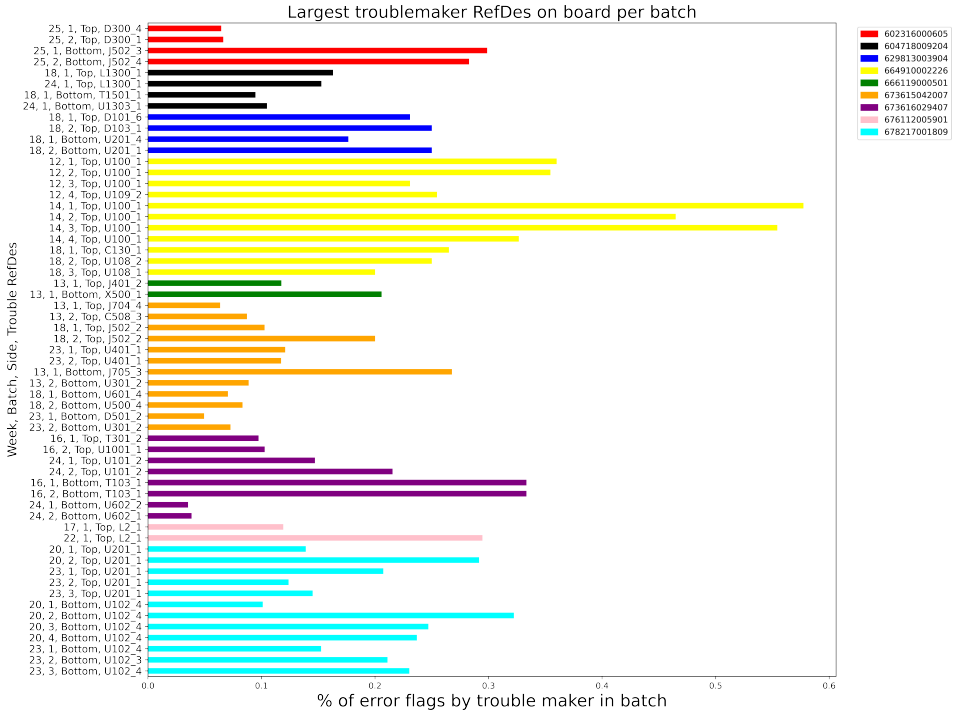


Figure 20: Component type troublemakers per batch

A panel can have multiple board locations with the same name due to the fact that multiple boards can be in a panel. To further investigate whether problems occur at panel level or board level, Figure 21b depicts the troublemakers based on the board specific locations. When the ratio of the biggest troublemaker decreases, there is an indication that it is a panel problem. If the ratio does not decrease the problem can be ascribed to a unique board location thus it is a problem on board level. Both situations seem to happen when comparing figures 21a and 21b. Therefore, it is interesting to further look into the data variance on the RefDes levels.



(a) Troublemakers per RefDes (can be multiple) on panel



(b) Troublemakers per unique RefDes on a board

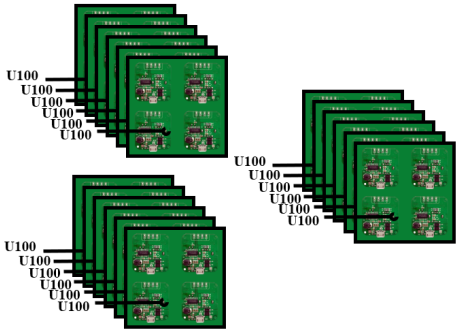
Figure 21: Troublemakers for board locations

5.1.3 RefDes Variance

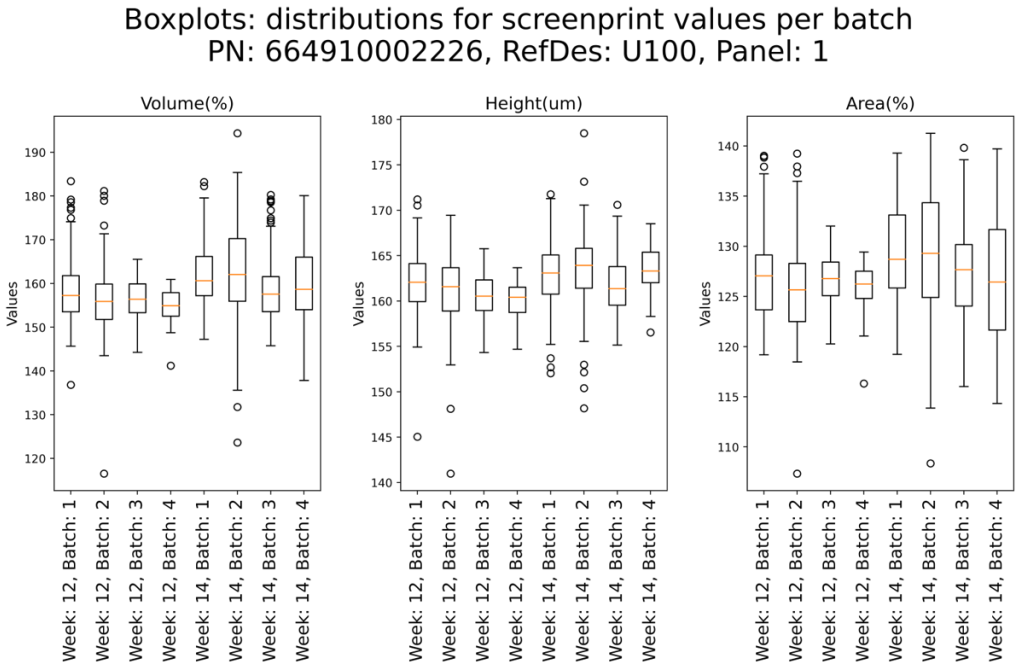
So far the exploration of the data showed that problems can occur due to components or board locations. AME states that the process features for a given RefDes can vary between batches or even within the same panel. The variance between batches is the result of overall varying conditions during different

production instances. Varying process features within the same panel is the result of fluctuations during the screenprinting process. For instance, a board on the upper left side of the panel can have other paste distributions than a board on the lower left side of the panel. This can have several causes such as minor deviations for each board in the stencil, or because less paste is applied at certain stencil areas. To see whether these assumptions related to the board level variance hold, the distributions of the same board location between batches are compared, and the distributions of the same board locations within a panel are compared.

Both explorations are most easily described by using a specific RefDes on a product type as an example. First, the paste feature variance between batches is compared by comparing the RefDes U100 for a set of eight batches of product 6649-1000-2226. Four of the batches are produced in week 12 and the other four batches are produced in week 14. An conceptual example of how the data distributions were constructed is shown in Figure 22a. The distributions are compared by using boxplots, which are shown in Figure 22b. Each boxplot represents the distribution of paste feature values for one specific RefDes in a batch, unique to a board in the panel. What is seen is that batches produced in the same week behave somewhat the same, although there is also variation when comparing those batches. For batches produced in other weeks, the distributions differ a lot more, which confirms the statement of AME that the process can vary between production orders and even between batches within the same production order.



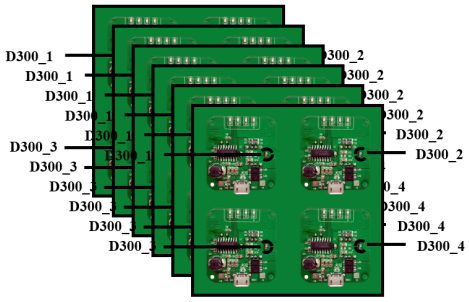
(a) Conceptual example of data distributions between batches



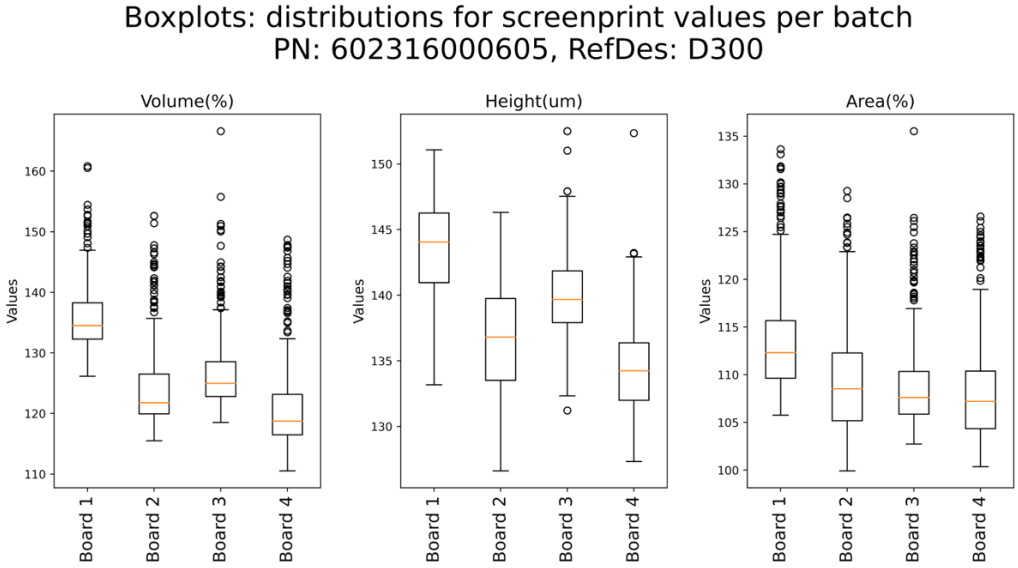
(b) Paste feature distributions per batch

Figure 22: Example of paste feature variation between batches

Variance can also be present for the same board locations on different boards in a panel. A random batch is picked from a random product type, in order to compare the distributions for the same reference destinations on a panel. Product 6023-1600-0605 is produced in a panel with four boards, resulting in four reference destinations with name D300 in a panel. Again, a conceptual example is provided in Figure 23a. Each boxplot in Figure 23b depicts the paste feature distribution in a batch for a given board. What can be concluded is that even on same board locations within a panel, there is variation in the screenprinting process. Both comparisons of the variation between batches and within panels lead to the conclusion that it is worthwhile to treat each board location as a separate data point during the data analysis and model building.



(a) Conceptual example of data distributions within a panel



(b) Paste feature distributions per batch

Figure 23: Example of paste feature variation within a panel

5.1.4 Errors over time within batch

It might be the case that errors happen during the start of the batch or after operator breaks. Therefore the error flag data relative to the time in a batch is explored. Here again, due to the varying production time lengths, examples are used to find whether hypothesis can be formed based on the data. These examples can be found in Figure 43, Appendix C. The examples do not show any relationship between the false calls and the time aspect in a batch. In order to compare all the batches in the data, the relative place of a product in a batch is considered by standardizing all time intervals of the batches to a range between 0 and 1. Then, the distribution of error flags can be plotted to show whether on average more flags happen in the beginning of a batch, see Figure 24. Apart from the large spike, which is caused by a product with many errors, only a small increase in errors is found in the beginning of a batch. Furthermore the wavy pattern might indicate that after production breaks more false calls occur.

Due to these observations, two features related to the time in a batch can be relevant for the modeling phase. First, to find whether there is a relation between the start of a batch and error flags, a sequential index is added to the data, referring to the location in the production batch of each panel. The first panel which passes the SMD line has index 1, the second index 2, and so on. Second, the time interval in seconds between the last product and the current product is added as a feature to catch possible quality deviations after production breaks. If there was a 5 minute break between two panels, then the feature value for the current panel will be 300 seconds.

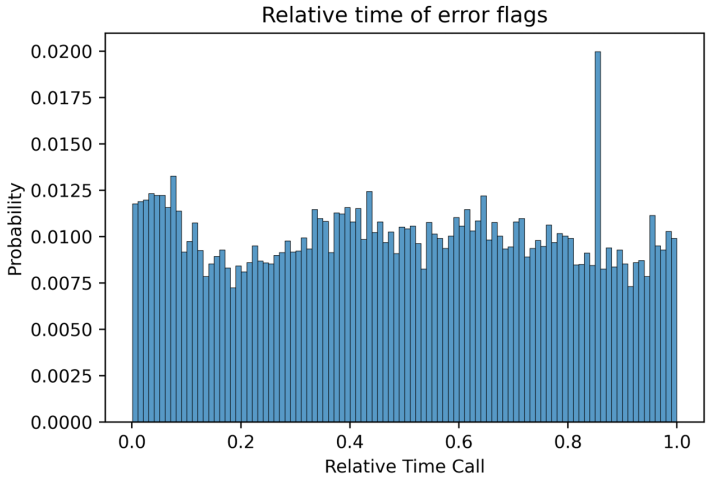


Figure 24: Distribution of all error flags relative to the time of the production case

5.1.5 Predictive features in relation with target

To get a feeling for the relationships between the process variables and the target variable the distributions of the features are explored for each target category (good component, false machine call, correct machine call). This is done with visualizations, descriptive statistics and statistical tests to check whether there is a difference between the distributions of the targets. Welch t-test is a statistical test for comparing means between two groups of unequal sample size (no assumption regarding equal variances as with the student t-test) (Ahad & Yahaya, 2014). As the Welch t-test is not robust when the data is not normally distributed, a Kolmogorov-Smirnov test is also executed. This test is based on the maximum difference between a hypothetical and empirical distribution function (Massey Jr, 1951).

Generally the process features can be divided into several categories, related to the sub process and machine parameters or features as a result of the process. As it is needless to deal with each variable individually, the relationships between the process and the target variable are discussed per category. The results are depicted in Appendix C, Section C.3. For the screen printing environment variables no large differences are found when comparing the targets. When comparing the descriptives of the screen printing paste features a downward deviation is found for the real error flags. In case of the screen printing parameters, it is found that the real errors have higher settings, indicating that there might be some problems related to the settings. The pick and place features related to the components show some deviations in the descriptives but when comparing the visualizations of the distributions it is hard to tell any real differences. The pick and place error features show that this data is very sparse which might indicate that there is a lack of information there. When looking at the reflow features, no unexpected differences are found. Thus, when comparing all the descriptives of the process features only in some cases deviations are found. Furthermore, all statistical test show that there are significant differences between the target categories. Note the robustness of these test might be decreased due to the great imbalance in the target class. Further analysis in the next section will provide more insights in the data distributions for each quality class.

As seen earlier, component characteristics can also influence the process quality. Therefore the relations between the categorical component variables and the target variable are also explored. These are depicted

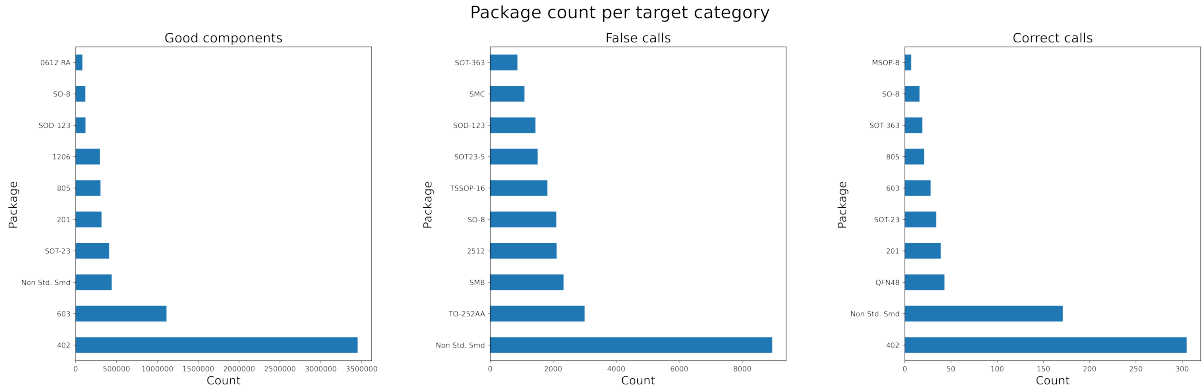


Figure 25: Component packages for each target class

by using bar charts for each target class, comparing the distributions of these bars. Most interesting is the component package, as this feature has the most variation. The bar charts related to the component package feature are depicted in Figure 25. The differences between the target classes might indicate the component package relevance when assessing the process quality with the component features. Further component characteristic features can be found in Appendix C, Section C.3. The above univariate comparisons showed that there are variations between target classes when comparing the process data. However, multivariate analysis (during the modeling phase) must find the complete set of relations between the process features and the target.

5.1.6 Correlating process features

Multicollinearity occurs in a data set whenever the predicting features are highly correlated. Models derived based on this data might have reduced predictive performance or lead to wrong system analysis (Garg & Tai, 2013). Therefore a correlation matrix of the process variables is visualized in Figure 26. Remarkable but not unexpected are the high correlations between the reflow features, as these represent the linear increase and decrease of the temperature in the oven. For the other variables there are no worrying high correlation values found. In case of the reflow features, a different representation of the data is required in order to be used during the modeling phase. This and other preprocessing actions are described in the following section.

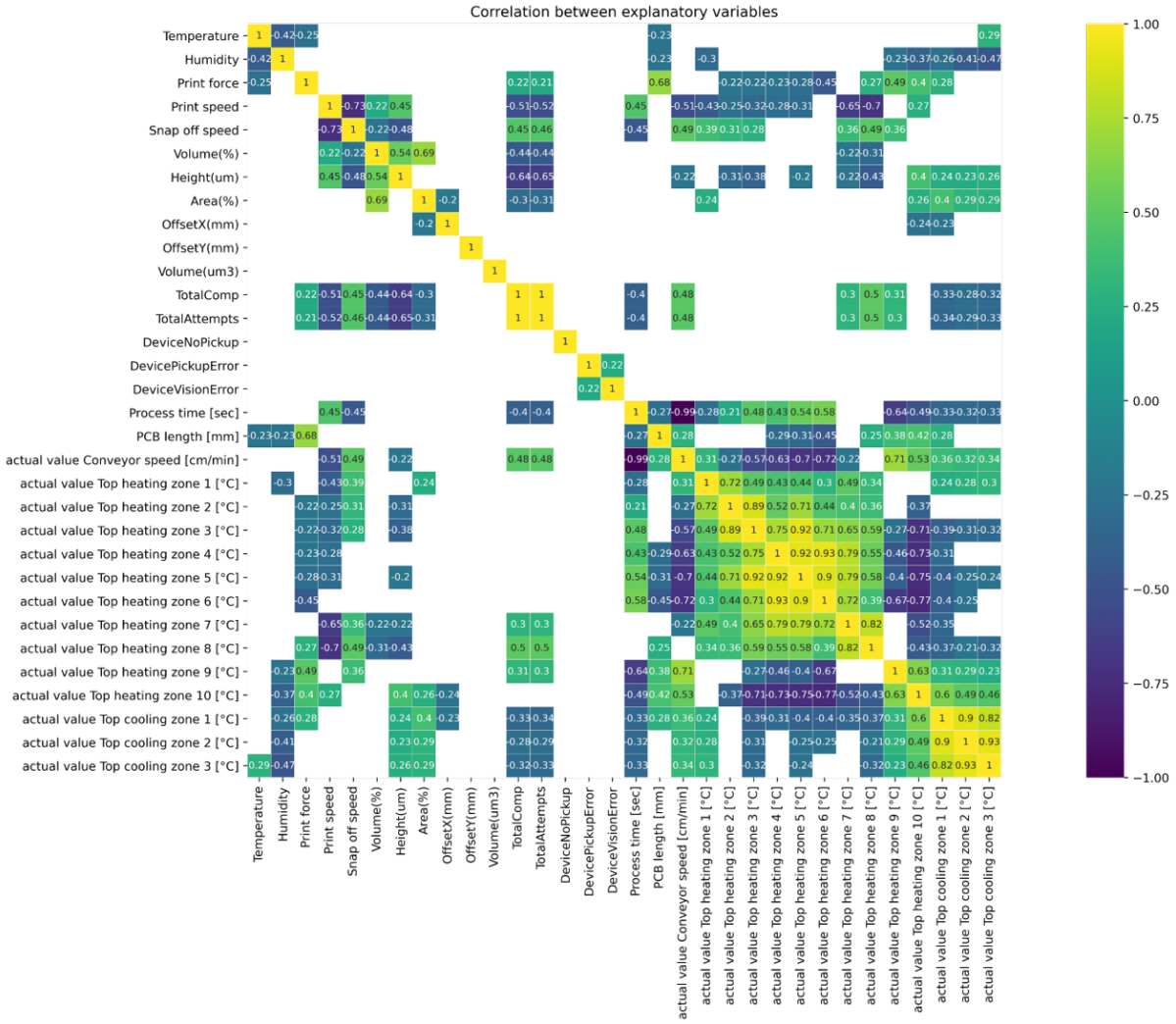


Figure 26: Correlation table of process features

5.2 Preprocessing & feature engineering

After getting a first feeling for the data it is important to further preprocess the data prior to the modeling phase. Preprocessing consists of data cleaning and transformation (Lv et al., 2018). Cleaning involves techniques for filling or removing missing data, reducing noise by handling outliers and duplicate data, and dropping redundant columns. Transformation includes both dimensionality reduction and encoding features to the right data format for modeling.

Missing data

Not all modeling techniques can handle missing data equally well which is why it is important to reduce the missingess in a data set. Most missingness is occurring at the reflow features, as approximately 25% of the conveyor speed variable is missing and 2% for each heating zone, which results in approximately 530,000 rows with missing values in on or more of the heating zones. The conveyor speed feature is removed from the data set as this feature can be replaced by the reflow process time, which is a direct result of the conveyor speed. This relation can also be seen in the correlation matrix, having a very strong correlation of -0.99 (see Figure 26). Interpolation is a mathematical method to create new points based on the values of surrounding existing points (Steffensen, 2006). In the case of the reflow temperature zones, linear interpolation is an appropriate method, since the zones become warmer and colder in a linear fashion. Only missing values at the beginning of the end of the sequence cannot be estimated

with this method. However when using linear interpolation, approximately 85% of the rows containing missing values can be filled. The other instances are removed from the data set. Furthermore, the wiper frequency variable related to the screen printing process also contains many missing values, but this features is dropped as it is a constant for all rows. After removing the missing data, there are 7,346,253 complete data rows in the data set. For the target class distribution of this complete data set see Table 9.

Table 9: Target classes after removing missingness

Inspection result	**Instances**	**Ratio**
Good	7,297,070	0.9950
False call	36,096	0.0049
Real error	747	0.0001
Total	**7,333,913**	**1.0000**

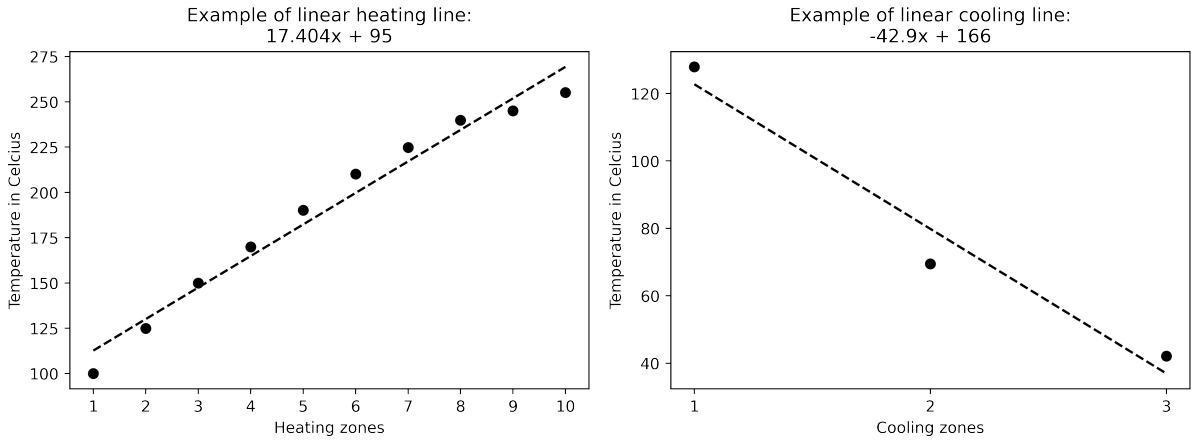
Product time intervals
As stated earlier, the time interval in seconds between two products in a batch can be relevant for the detection of quality deviations. In order to create this variable, the panels are sorted per batch based on the production time in the AOI log file, and the interval between two consecutive products is calculated. In order to remove any outliers related to this variable, all the rows having a time interval greater than 7 days (604,800 seconds) are removed from the data. It was found that these outliers are duplicates in the data, as the serial number of these products were all found earlier in the data. These duplicates can be a result of a manual scan after the reparation in the data or a new test after the production batch. Note that removing these instances also reduces the noise in the data as only the quality during the first test (without reparations or other adjustments) is relevant for this research. The size of the data set after removing these outliers or duplicates is shown in Table 10.

Table 10: Target classes after removing duplicates

Inspection result	**Instances**	**Ratio**
Good	7,287,670	0.9950
False call	36,011	0.0049
Real error	739	0.0001
Total	**7,324,420**	**1.0000**

Categorical encoding
Machine learning models require a numerical vector representation of the data in order to work well. Representing categorical variables (i.e. the component characteristics) in a numerical form can be done in several ways. A common practice is using one-hot encoding, which transforms each category of a categorical feature in a dummy variable. Whenever a feature has seven unique categories, seven dummy variables are added each describing one category. However, when there are many categories present in a variable the method of one-hot encoding create high-dimensional feature vectors. This leads to sparse input vectors which decreases the machine learning model performance as the hypothesis space increases (Altendorf, Restificar, & Dietterich, 2012). Therefore when having many categories, it is preferred to find a low-dimensional representation of these high-cardinality string categorical variables (Cerda & Varoquaux, 2020). A simple yet effective method is target encoding, because this does not increase the dimensionality of the data to a large extend. The method accounts for the prior probability of a category feature relative to the target. This probability is calculated based on the given (training) data. For instance, when a component package occurs 10 times of which 7 times as good call, 2 times as false error flag and 1 time as correct error flag, the encoded values for that component package are 0.7, 0.2 and 0.1 respectively. New variables are added per target category and not per categorical value, reducing the increase of the data dimension greatly. A potential downside of the method is the decrease of the model's outcome interpretability. To prevent data leakage, the encoding is done after splitting the data and fitted on the training set. The method is robust against imbalanced data and therefore useful for this research.

Feature engineering
The curse of dimensionality theory states that the difficulty of problems rapidly grows as the number of dimension increase (Bellman, 1966). Some state that the cost of an algorithm grows exponentially by adding a dimension (Kuo & Sloan, 2005), besides the fact that adding more variables increases the noise in the data set thus the ability to generalize well. As stated, it is preferred to transform the correlating variables related to the reflow process in a way that captures the most important information in the least amount of dimensions. Several things are important when considering the heating parameters related to the soldering quality: what is the maximum temperature of the zones and how fast do the temperatures rise and fall (Mar et al., 2011). Therefore, the temperature features can be captured in three variables: the maximum temperature, the average temperature, and the linear slope of the temperature zones. The latter is done by fitting a linear line to the data points related to the heating and cooling zone values, then using the line coefficient as feature for the model. An example of this method is depicted in Figure 27, the coefficients for the heating and cooling zones are 17.40 and -42.90, respectively. Using this method captures all relevant information of the heating zones and reduces the noise and dimensions in the data set. Furthermore, using only the total attempts as a indicator for how well the pick and place process is dependent on the number of components placed. Therefore the total components placed is divided by the total picking attempts, to create a placement score which is independent of the total components placed. Finally, to get the relative place of a product in batch, the product index is divided by the total products in a batch.



(a) Heating zones

(b) Cooling zones

Figure 27: Lines fitted to the heating and cooling zones

5.3 Concluding remarks

Before proceeding the the modeling phase of the CRISP-DM framework, a brief summary of the data exploration findings and feature engineering is provided. During the data exploration several useful modeling insights are gained. First of all, the exploratory data analysis show that the data is highly imbalanced, and that some error types happen more frequently than others. The number of panels in the data set with one or more false calls is 58%, this is comparable to the complete population which is 62%. Furthermore, no time dependency was found for the real error ratio when comparing the different batches over time, the results only show an increase in false calls over time. Reoccurring problems (troublemakers) during production can be associated with either component types or board locations, thus both should be taken into account during modeling. Process deviations can happen between batches, but also within a specific panel. For instance, when a panel consists of two boards, RefDes U100 on board 1 can cause more problems than the same RefDes U100 on board 2. Also, it is also shown that the production position of a product in a batch may slightly influence the quality, as well as the time interval between two produced panels. Analysis regarding the interrelations of features show that the distributions for the targets behave slightly different for some features. Interpolating the missing values and cleaning the data lead to a complete data set, which can be used during the modeling phase. It was also shown that the reflow process features are highly correlated, requiring an aggregation to reduce

the dimensionality of the data. The linear increment and decrement of the temperature in the reflow heating zones are transformed to a linear coefficient, representing the temperature change in one variable. The average and maximum temperature are also added so no information regarding the process is lost. Lastly, the categorical features related to the components have a high cardinality. Therefore, target encoding is used to transform these categorical features making them eligible for the modeling phase. To conclude this section an overview of the features used for the modeling phase is given in Table 11. In the next section the modeling phase is described, defining the data splitting methods and the modeling development.

Table 11: Modeling features

Identifiers	**Screen printing**	**Pick & place**	**Component (encoded)**
PN	Temperature	Total components	Package type
SN	Humidity	Total pick attempts	Supply form
Board ID	Print force	No pick up error	Moisture sensitive
Side ID	Print speed	Pick up error	
Week Nr	Snap off distance	Vision error	**Quality result**
Batch Nr	Snap off speed	Placement score	Machine error call
Component ID	Volume (%)		False call target
RefDes	Height (um)	**Reflow**	
	Area (%)	Process time (s)	
Panel	Offset X (mm)	Max. heating zone	
Panel interval (s)	Offset Y (mm)	Max cooling zone	
Relative batchpos.		Heating & cooling coefficients	

6 Quality Modeling

This section describes the development of a predictive model that aims to use online process data to enhance the outcome of the quality control system, which is part of the modeling phase in the CRISP-DM framework. Doing so helps reducing the required manual checks at the SMD production line. If an error flag is raised by the AOI, the model uses the process data to assess the error flag using binary classification. Whenever the model predicts that it is a real error or a false error with a given certainty, there is no manual check required. If the model is not certain enough, then the operator still has to do a manual check to prevent error slips. An overview of the process flow with the model is given in Figure 28. The highly imbalanced target class requires specific modeling approaches which are able to handle the minority classes well. Different types of models are developed: multiple classification methods using various methods to tackle the imbalance data problem, and an anomaly detection method which treats the false error flags and correct error flags as abnormal data. Firstly, the different data splitting methods of the models are explained. Secondly, the general evaluation metrics are described and explained briefly. Thirdly, for each model an explanation of the model development is given followed by a numerical evaluation of the models. Furthermore, for each approach an uncertainty mechanism is developed and numerically tested. This mechanism ensures that a model is as certain as possible about a prediction to reduce the numbers of unwanted errors from a business perspective.

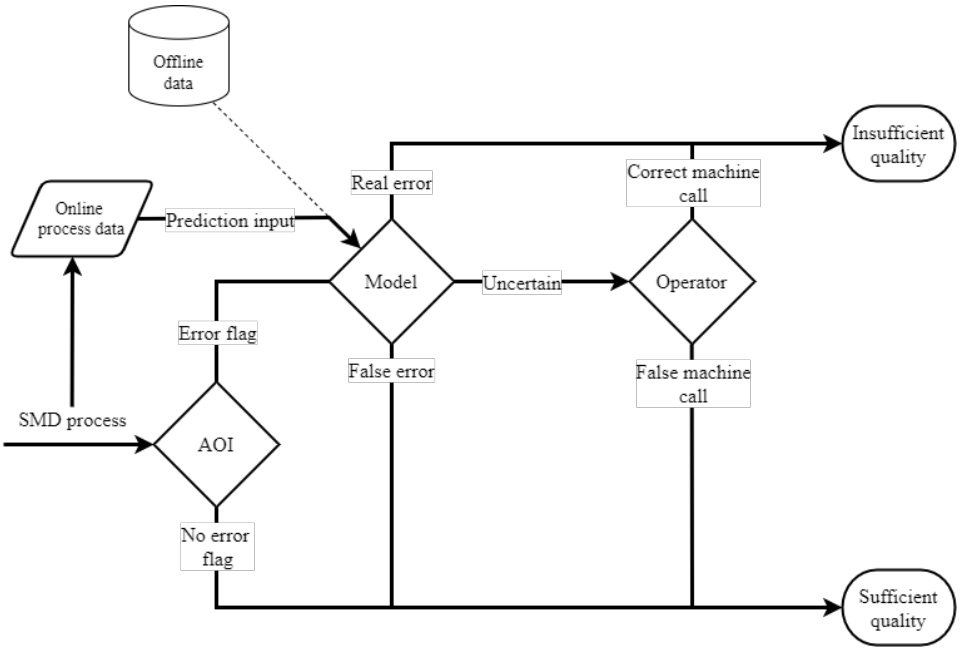


Figure 28: Component quality check process flow with model

6.1 Splitting method

During the model development there is a constant tradeoff between bias and variance. The tradeoff is the conflict trying to simultaneously reduce these two constructs in order to let the model generalize well beyond the training set (Kohavi, Wolpert, et al., 1996). Bias can also be described as underfitting, which happens when the model's assumptions are too simple which results in missing relevant relations between the input data and the target. Variance is the opposite of bias and can also be defined as overfitting. A model overfits if it is too strictly fitted to the training set, which results in learning random noise from that set. When this happens the model does not generalize well on other data sets. Splitting the data in a proper way helps with evaluating the model well and helps in finding the best tradeoff between bias and variance. The data is split in a training, validation and test set. Training data $\mathcal{D}_{train}$ is used to train the model in order to learn from the data. Validation data $\mathcal{D}_{val}$ is used to tune the hyperparameters of the model in the best possible way. Testing data $\mathcal{D}_{test}$ is eventually used to test the real performance of the model, as this data is never shown to the model thus considered as 'new'

data. Both modeling methods (classification and anomaly detection) require a different splitting method.

When using standard classification, the training, validation and testing data is split into proportions of respectively 60%, 20% and 20%. Group splitting is used when splitting the data, meaning that a data group can either be in the training or the test set to prevent data leakage. For the problem at hand, an individual panel (SN) is considered as one group because SN data level features are the same for all samples belonging to that panel. If no group split is used, the algorithm can learn information about a group which can be used during the evaluation on the test set. Using the group split ensures that no panel specific information is learned during training which is then also present in the test set. This concept is called data leakage, and is a common problem in machine learning problems with groups in the data. It can lead to higher performance measures during the development phase than expected in the real world, resulting in unwanted outcomes when deploying the model (Ayotte, Banavar, Hou, & Schuckers, 2021). K-fold cross validation is a resampling technique to evaluate models with different samples when only limited data is available. Due to the small minority classes, K-fold cross validation is an effective method to evaluate the models. An example of 5-fold group cross validation for the classification models is given in Figure 29. To ensure that the performance metrics are less biased due to the class imbalance, the validation and test sets are balanced (2 majority class samples to 1 minority class sample). Doing so facilitates a better evaluation regarding how well the model can separate the different classes. It further enables comparison with the anomaly detection method performance metrics, as the test and validation set for an autoencoder classification problem are balanced by design. Splitting the data for the anomaly detection is described in the following paragraph.

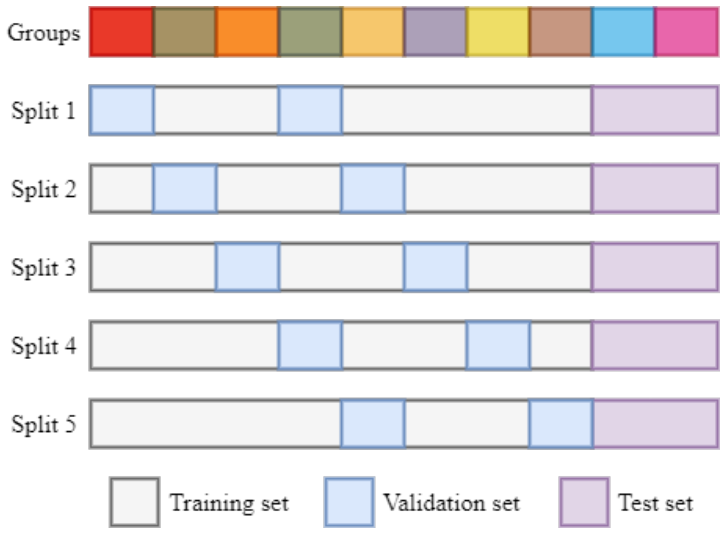


Figure 29: Classification K-fold group split

For the anomaly detection a different splitting method is required. As described in Section 3.4, anomaly detection methods learn the distribution of what is considered normal data. After learning what is considered normal data, it tries to predict the anomaly cases (the minority class in our problem). If the model cannot predict these instances well, they are labeled as anomaly data. The training set for this problem thus only contains normal data (the majority target class), which is size N. Then, the anomaly data (minority class) is split into a validation and a test set. If the anomaly data is size A, then the validation set and the test set both have $A/2$ anomaly instances. The validation set and test set also require normal data in order to evaluate the model for both classes. Therefore, the the training set is size N-A, and A normal cases are split into the validation and test set. Thus, both the validation and test set have $A/2$ normal instances and $A/2$ anomaly data instances. Again, data belonging to a given panel (SN) may not occur in both the training and test set to prevent data leakage. An example of the splitting method is given in Figure 30.

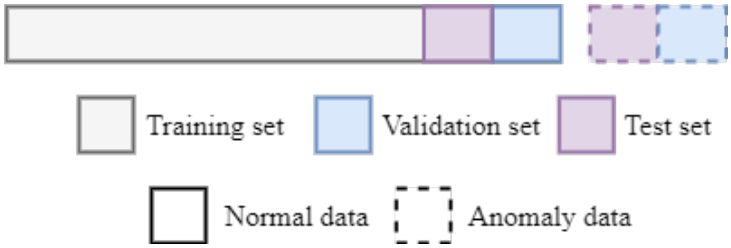


Figure 30: Anomaly detection split

6.2 Evaluation metrics

Interpreting the metrics and their significance is an important step to correctly evaluate different learning algorithms (Tharwat, 2020). For this research it is important to use metrics which are not sensitive to imbalanced data. For instance, accuracy is a commonly used metric for classification performance and is defined as the ratio between the correctly samples to the total number of samples. However, when the majority class contains 99% of all the data and the model predicts everything as the majority class, the accuracy will also be 99%. More robust metrics for imbalanced data are sensitivity (recall), specificity (inverse recall), and the Geometric Mean (GM) (Tharwat, 2020). Classification metrics are constructed by using a combination of true positives, true negative, false positives and false negatives. These concepts are dependent on the predicted values in relation to the real values of the test samples. An visual explanation of these concepts is given in Figure 31.

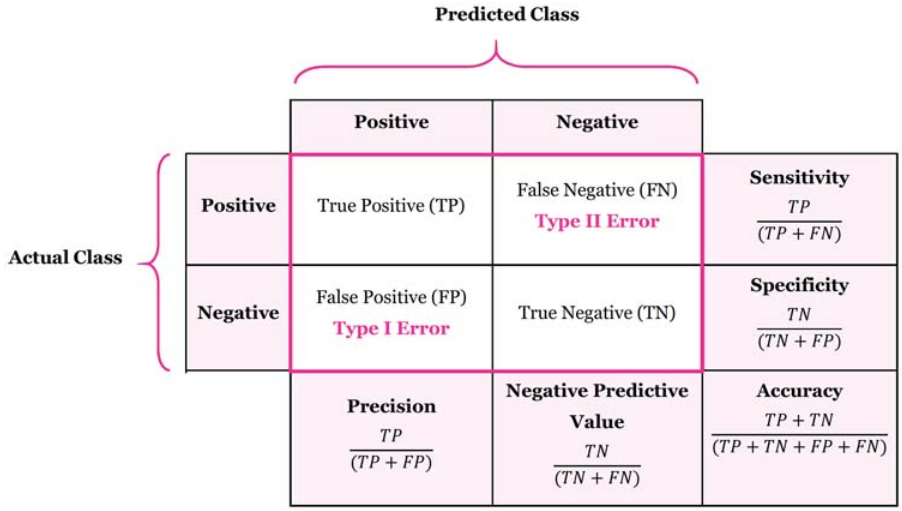


Figure 31: Confusion matrix concepts

Sensitivity & Specificity

Recall or sensitivity represents the positive classified samples to the total number of positive samples (Tharwat, 2020). Stated otherwise, it shows how many of the samples are detected. Specificity or inverse recall is the same, but then for the negative class. These constructs can also be viewed as the accuracy for a given class and are therefore not sensitive to the class imbalance. See Figure 31 for the equations related to both concepts.

Geometric Mean

Most classifiers aim to improve both the sensitivity without sacrificing the specificity. The problem with this goal is that, especially with imbalanced data, these constructs are often conflicting (Tharwat, 2020). The Geometric Mean (GM) metric uses both the sensitivity and specificity so that it is not dependent on the class imbalance (Boughorbel, Jarray, & El-Anbari, 2017). The formula of the GM is showed in Equation 4.

$$\sqrt{\frac{TP}{TP+FN} \times \frac{TN}{TN+FP}} \tag{4}$$

Precision & Negative Predictive Value
Precision represents the proportion of positive predicted sample which really is a positive predicted sample. Thus it represents the correctness of predicting a given class. The negative predictive value is the inverse of the precision and relates to the same construct for the negative class. These metrics, although sensitive to imbalanced data (Tharwat, 2020), is also an interesting metric for this research because it is important how well the model can predict a certain class if it finds one. The equations of both precision and negative predictive value are found in Figure 31.

Metric importance for the false call detection problem
Prior to developing any models it is important to define the importance of each performance metric based on the business implication. By taking the business implementation into account when building the model, the most gains can be made if the model is used during production. The best case scenario would be that the model does not make any mistakes and each error flag is classified correctly as false or real error, removing all manual checks. However, this is an unrealistic assumption as there will always be wrong classifications due to noise in the data. For each wrong classification there are different business implications. The worst misclassification is when a real error is predicted to be a false call because then the error flag is wrongly dismissed and the component is assumed to be of sufficient quality (resulting in error slip). There is a probability that this failure will be found later during the functional test but as not every board undergoes this functional test, it is not safe to assume that the erroneous component is captured at this workstation. The misclassification of a false error flag as real error is less of a problem because at the repair station it is then found that no repair is required for the component. However, this is also not preferred as sending too much false calls to the repair stations may create new inefficiencies at the repair station. It is necessary to be quite sure about the predictions, especially when a false call is predicted. This means that it is preferred to develop models for which the precision is as high as possible for both classes. All other uncertain predictions still require a manual inspection to nullify the error slip as much as possible. Due to the imbalanced target class the most business value is in correctly classifying false calls. This is simply because only correctly classifying the real error only solves a small part of the problem in absolute sense. Say a model is only certain about classifying the real errors, then for the given sample only 739 of the 36,750 error flags do not require a manual check.

For the problem at hand it is important that no misclassifications are made, therefore an additional class which can be predicted is added when the model is not sure, which is referred to as the uncertain class from this point forward. The proportion of the predicted uncertain class should be as small as possible. Nonetheless, the error slips (insufficient quality deemed as sufficient) and redundant repairs (products at repair station without error) should also be minimized in parallel. It is likely that these metrics will contravene as predicting less uncertain classes will lead to more misclassifications. Both the uncertain class ratio and the misclassification ratio (error slip and redundant repairs) equations are shown in Equation 5 and 6 respectively. For instance, when there are 10 cases of class A, of which 7 are predicted correctly, 2 are predicted as uncertain and 1 is predicted as class B, the misclassification ratio is 0.1. The uncertainty ratio for class A in this case is 0.2.

$$Uncertainty\ ratio = \frac{Uncertain\ predictions}{Total\ predictions} \tag{5}$$

$$Misclassification\ ratio = \frac{Class\ misclassifications - uncertain\ predictions}{Total\ class\ occurrences} \tag{6}$$

6.3 Machine learning for imbalanced data

In this section different machine learning methods are developed for classifying false error flags and real errors. As described in Section 3.3.3, these models are logistic regression, support vector machine and random forest. To tackle the target imbalance in the manufacturing data, different sampling techniques

and the balanced bagging method are examined. For all these methods balanced class weights are used to penalize minority class misclassifications, in order to learn as much as possible from the minority class. First, the cross validation results are discussed after which the most suitable method is chosen. For this method an additional uncertainty mechanism is added to ensure the model only predicts a class when it is really sure. Finally, the hyperparameters of the model are tuned and evaluated on the test set.

6.3.1 Selecting the classification model

Each modeling method is tested with the different sampling techniques and validated via 5-fold cross validation. The performance metrics for both classes are important, as it is not preferred to have error slips and it is also not preferred to create congestion at the repair station. As described, it is required to find a model with the highest possible precision for both classes. The performance metrics of the logistic regression for the different sampling methods are given in Table 12. It is notable that the sampling techniques have little effect on the logistic regression results. Only when using balanced bagging with under-sampling, the performance metrics differ slightly. However, the performance metrics show that logistic regression only predicts false calls, as the recall is almost one and the precision is exactly the distribution of the majority class relative to the minority class. These findings might be the result of the fact that logistic regression is very prone to outliers. Overall, the linear method is not suitable for this problem, which was already expected due to the distributions of manufacturing data in general.

Table 12: Logistic regression average validation results with standard deviation

	Real error			**False call**		
Sampling method	**Recall**	**Precision**	**GM**	**Recall**	**Precision**	**GM**
None	0.03 (0.06)	0.18 (0.37)	0.08 (0.15)	0.99 (0.00)	0.67 (0.01)	0.82 (0.01)
Random over-sampling	0.03 (0.06)	0.18 (0.37)	0.08 (0.15)	0.99 (0.00)	0.67 (0.01)	0.82 (0.01)
Random under-sampling	0.03 (0.06)	0.18 (0.37)	0.08 (0.15)	0.99 (0.00)	0.67 (0.01)	0.82 (0.01)
SMOTE Links	0.03 (0.06)	0.18 (0.37)	0.08 (0.15)	0.99 (0.00)	0.67 (0.01)	0.82 (0.01)
Balanced Bagging SMOTE	0.03 (0.06)	0.18 (0.37)	0.08 (0.15)	0.99 (0.00)	0.67 (0.01)	0.82 (0.01)
Balanced Bagging under-sampling	**0.11 (0.17)**	**0.30 (0.37)**	**0.17 (0.23)**	**0.98 (0.03)**	**0.69 (0.04)**	**0.82 (0.01)**

The performance results of the support vector classifier are depicted in Table 13. In general, the method can separate the false call class relatively well, which is expected due to the class imbalance. The false call recall is high for the methods which do not incorporate under-sampling, leading to a lower recall in the real error class. This indicates that most of the samples are predicted as false call, which might result in error slips. Using under-sampling for support vector machines enables the method to better separate both classes, as the recall for the real error class tends to be higher in these cases. Compared to logistic regression, support vector machines perform much better as the method enables to separate the classes in a non-linear way.

Table 13: Support vector machine average validation results with standard deviation

	Real error			**False call**		
Sampling method	**Recall**	**Precision**	**GM**	**Recall**	**Precision**	**GM**
None	0.64 (0.24)	0.88 (0.09)	0.74 (0.19)	0.97 (0.01)	0.85 (0.08)	0.91 (0.04)
Random over-sampling	0.66 (0.24)	0.88 (0.08)	0.75 (0.2)	0.97 (0.01)	0.86 (0.08)	0.91 (0.05)
Random under-sampling	**0.77 (0.12)**	**0.77 (0.09)**	**0.77 (0.09)**	**0.89 (0.05)**	**0.89 (0.05)**	**0.89 (0.04)**
SMOTE Links	0.63 (0.26)	0.87 (0.12)	0.73 (0.22)	0.97 (0.01)	0.85 (0.08)	0.91 (0.05)
Balanced Bagging SMOTE	0.62 (0.25)	0.89 (0.11)	0.73 (0.22)	0.97 (0.01)	0.84 (0.08)	0.91 (0.05)
Balanced Bagging under-sampling	0.72 (0.21)	0.76 (0.12)	0.74 (0.16)	0.89 (0.04)	0.87 (0.08)	0.88 (0.05)

When consulting the performance metrics of the random forest models in Table 14, several things stand out. First, the over-sampling techniques perform relatively bad compared to the under-sampling techniques. The precision might be relatively high for the real errors, indicating that when it predicts an error it is most likely to be correct. However, there are also many false call predictions for which the precision is not that high, indicating that there are relatively much real errors predicted as false calls, which is not preferred. For the under-sampling cases the results seem more promising, especially for the balanced bagging with under-sampling. The method has a relatively high precision for both classes combined with a decent recall. Compared with the support vector classifier, the random forest performs better over

all folds, as the standard deviation of the metrics is much lower. The fact that balanced bagging suits random forest better might be explained by the fact that this algorithm already uses a bagging method. Overall, the results show that under-sampling is a more suitable method for the problem at hand than over-sampling. The random forest classifier in combination with the balanced bagging under-sampling is assumed to suit the false call problem best. The following section will further elaborate on the model by adding uncertainty, further tuning the model to the problem at hand.

Table 14: Random Forest average validation results with standard deviation

	Real error			**False call**		
Sampling method	**Recall**	**Precision**	**GM**	**Recall**	**Precision**	**GM**
None	0.21 (0.09)	0.99 (0.02)	0.44 (0.12)	1.0 (0.00)	0.72 (0.02)	0.85 (0.01)
Random over-sampling	0.26 (0.11)	0.99 (0.02)	0.49 (0.13)	1.0 (0.00)	0.73 (0.03)	0.85 (0.02)
Random under-sampling	0.72 (0.15)	0.87 (0.07)	0.79 (0.11)	0.95 (0.02)	0.87 (0.06)	0.91 (0.04)
SMOTE Links	0.34 (0.15)	0.97 (0.03)	0.56 (0.14)	0.99 (0.01)	0.75 (0.04)	0.87 (0.02)
Balanced Bagging SMOTE	0.30 (0.13)	0.99 (0.02)	0.53 (0.14)	1.0 (0.00)	0.74 (0.04)	0.86 (0.02)
Balanced Bagging under-sampling	**0.74 (0.10)**	**0.91 (0.05)**	**0.82 (0.06)**	**0.97 (0.02)**	**0.88 (0.04)**	**0.92 (0.02)**

6.3.2 Adding uncertainty to the model

As described in Section 6.2, misclassifications of both target classes may result in unwanted business results. Therefore, the model should be very sure about a prediction before accepting the outcome. The production process design allows that the model can be uncertain about a prediction. For these cases the production operator should just check the machine call, as is already required for every error flag currently. The aim of the model is to be as confident as possible about a prediction but also to limit the number of uncertain predictions. In order to find the prediction confidence of the random forest model, the predicted class probabilities can be used. These probabilities are computed as the mean predicted class probabilities of the trees in the forest. The fraction of class samples in a leaf determines the class probability of a single tree. When predicting, the random forest model then can provide a probability indicating how certain the model is regarding a particular prediction. To add uncertainty to the model, a probability cutoff value must be chosen. When the class probability of a prediction is smaller than that cutoff value, the model is not certain enough about a prediction thus the error flag requires a manual check. When the model is 50% sure about a prediction, the prediction is just as good as a random guess. Therefore, the cutoff value should always be between 0.51 and 1, preferably as close to 1 as possible. For instance, if the cutoff value is 0.7 and the class probability is 0.75, the model predicts that particular class. If the class probability is only 0.6, the model predicts that the error flag requires a manual check.

6.3.3 Model evaluation

To find the best model configuration for the balanced bagging random forest model, different sets of hyperparameters are evaluated via grid search. This regards the maximum depth of the trees, the minimum samples used for a split, and the number of estimators in the ensemble. Furthermore, different cutoff values for the classification probability are also examined, to check what level of certainty suits the problem best. In total, 630 different configurations are tested and assessed on different performance metrics. Most importantly, the misclassification ratio's of both the false call class and real error class should be minimized. These metrics are depicted by the error slip and the redundant repair ratio, which are associated with the precision. However, having a high precision but also many uncertain cases (which then still require a manual check) does not solve any business problems. It is therefore key to balance both the precision the classes and the number of predictions which still require a manual check. Reducing the number of manual checks is comparable to finding a recall which is as high as possible. Table 15 shows three model outcomes which each serve a different goal, from low misclassification ratio's to a low manual inspection percentage with a high recall for both classes.

Table 15: Hyperparameter search results balanced bagging random forest classifier

				Real error			False call			
Max. depth	**Min. samples split**	**Estimators**	**P. cutoff**	**Recall**	**Precision**	**Error slip**	**Recall**	**Precision**	**Redundant repair**	**Manual ratio**
10	2	2000	0.90	0.19	1.00	0.02	0.61	0.99	0.00	0.53
25	5	100	0.80	0.25	0.98	0.04	0.78	0.98	0.001	0.39
10	5	100	0.60	0.52	0.94	0.17	0.93	0.92	0.01	0.14

In general, the precision of each class is relatively high, probably due to the fact that the uncertainty factor is added to the model. The higher the probability cutoff, the more predictions require a manual check, reducing the recall of both classes. This shows that the probability cutoff used for classification might serve as a useful method to control the model's certainty. The algorithm is a bit better at separating the false call class as the recall for this class is higher than for the real error class. This is also indicated by the high error slip (when a real error is predicted as false call), compared to the low redundant repairs (when a false call is predicted as real error). Selecting the best model in terms of business perspective purely based on these performance metrics is not feasible as they do not take costs and saved time into account. However, these metrics can be used to compare the model with the anomaly detection method of the following section. Both methods will be compared in Section 6.5 and the most feasible model will be evaluated in Section 7.

6.4 Autoencoder

This section describes the development of the autoencoder model suitable for the binary classification task as described in Figure 28. Firstly, the common way of using the output of an autoencoder for binary target classification is explained. Secondly, the most suitable input data sets (normal and anomaly) are discovered using this classification method. Then, the most suitable network architecture is defined and tested. After defining the input data and architecture, a method is proposed and tested to improve the autoencoder classification task for the problem at hand, using multiple autoencoders in combination with supervised learning.

As an additional introduction for this section, the chosen default model parameters are described briefly. The shape of the input data is $n \times 35$, where n is dependent on which data points are considered as normal behaviour. The number of features define the shape of the input layer, which is 35 in this case. As stated in Section 3.4, halving and doubling the number of neurons during encoding and decoding respectively is a good rule of thumb when developing a first model. Therefore, this architecture serves as the default architecture used in the development of the autoencoder. The default structure is depicted in Figure 32. For each layer the activation function is the Exponential Linear Unit (ELU) as this alleviates the vanishing gradient problem, speeds up the learning, and has better generalization performance compared to the ReLU function (Clevert et al., 2015). For each model which is trained during the development phase, column wise standardization is done. This enhances the learning process because it prevents that large feature values overshadow smaller features when computing the gradients.

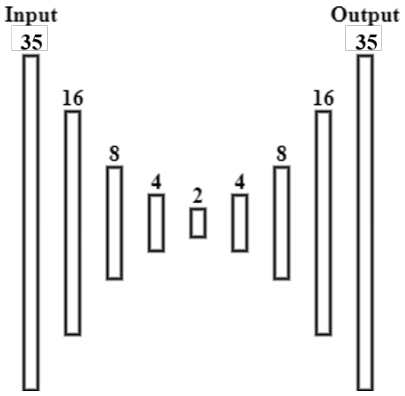


Figure 32: Default architecture

6.4.1 Binary autoencoder classification using τ

Despite the unsupervised nature of the autoencoder method, it is possible to use the model as a binary classifier. After training the model on the normal data, the validation set (containing normal data and anomaly data) enables evaluating to which extend the model is able to distinguish normal data from anomaly data. After reconstructing each sample in the validation set with the trained autoencoder, it is possible to calculate the mean reconstruction error for each sample. It is expected that for the different target classes the reconstruction error distributions will be different. For the normal set the reconstruction errors should be low while the errors for the anomaly set should be higher. To classify new data samples, a threshold τ must be set which serves as a reconstruction error cutoff point for normal samples and anomaly samples. The validation set is used to decide the value of τ, and the test set is used to evaluate the performance of the model and its threshold. τ can be determined based on the reconstruction errors but it is also possible to use the standardized distribution of the reconstruction errors (e.g. the Z-score). This enables the use of percentiles which makes it is more trivial to set a threshold detecting outliers. Instead of using percentiles based on the standard deviation, the Median Absolute Deviation (MAD) is more resilient to outliers and therefore better scalable and more robust. MAD is defined as the median of the absolute deviations from the data's median $\tilde{X}$, see Equation 8 (Sematech, 2006). The modified Z-score is then calculated with the MAD instead of the standard deviation, see Equation 9.

$$\tilde{X} = median(X) \tag{7}$$

$$MAD = median(|X_i - \tilde{X}|) \tag{8}$$

$$M_i = \frac{0.6745(X_i - \tilde{X})}{MAD} \tag{9}$$

These Z-scores can then be used to determine when a sample is an outlier or not, setting the threshold τ based on the standardized distribution. Note that using a threshold which is too high causes all the data points to be normal. Setting a threshold which is too low causes all the data points to be classified as anomaly. In summary, a higher threshold means more precision for the anomaly class, and a lower threshold means more recall for the anomaly class (although the value of the classifier vanishes when all samples are classified as anomalies). Whilst the threshold when classifying brings a certain flexibility to the model, choosing the best threshold value is a non-trivial task and requires decent validation to prevent over- or underfitting.

6.4.2 Normal and anomaly input data

When training the autoencoder, there are several possibilities regarding what is considered as normal data and anomaly data due to the different target classes. In order to explore how the different data samples behave when trained on an autoencoder, several autoencoders with the default architecture are trained on multiple data sets. Before training the neural network and using it for the binary classification, the data is split as described in Section 6.1. The performance of the different models using $\tau = 3$ is given in Table 16.

Table 16: Default AE performance for different input sets, $\tau = 3$

			Normal class			**Anomaly class**		
Model	**Normal data**	**Anomaly data**	**Recall**	**Precision**	**GM**	**Recall**	**Precision**	**GM**
1	Good	False	95.3%	63.8%	78.0%	46.0%	90.8%	64.6%
2	Good	Real	93.0%	59.5%	74.4%	36.9%	84.0%	55.6%
3	Good & False	Real	94.0%	59.4%	74.7%	35.8%	85.7%	55.4%
4	False	Real	98.4%	55.6%	73.9%	21.4%	92.9%	44.6%

When inspecting the performance metrics of the different models, the most interesting are the metrics of the anomaly class. From these values it can be concluded that the model is better able to distinguish good and false call data points than good and real error data points. When treating the false calls as anomaly data (Model 1), the model has a precision of 90.8%, indicating that when it finds a anomaly class it is

right in most of the cases. When the real errors are treated as anomaly data, the most promising normal data set based on the GM is the good components subset (Model 2), with a GM of 55.6%. However, as stated in Section 6.2, it is important to perform well on precision for both the false error and the real error class. When taking this into account, model 1 and model 4 perform best when classifying the false calls and the real errors respectively. These results show that using a single autoencoder might not be sufficient when solving the problem at hand, as the classifiers perform not well enough to be reliable.

6.4.3 Network architecture and parameters

In order to find the best model architecture, different configurations are tested with data sets considering false error flags and real error flags as normal and anomaly data, respectively. This choice is made as this is these are the smallest subsets of data which reduces the computational time during the tests. Both shallow (only having one hidden layer) and different levels of deep (having more than one hidden layers) configurations are tested and evaluated on the anomaly precision and recall. Furthermore, the effect of using the ReLU activation function is also considered. All configurations with performance metrics are found in Table 17.

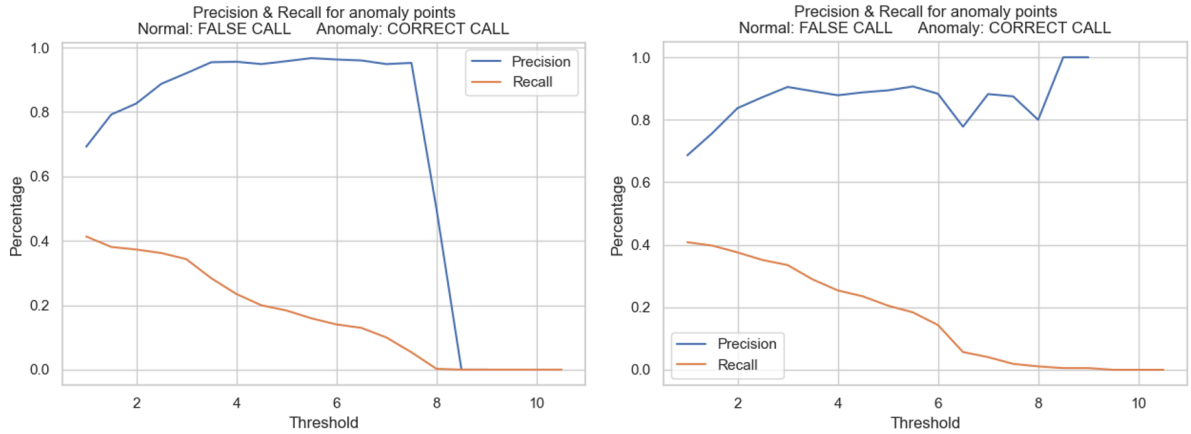
Table 17: Network parameters and anomaly evaluation metrics for false calls and real errors

Model	**Hidden Layers**	**Activation function**	**Recall**	**Precision**
1	4	Elu	30.89%	82.61%
2	8	Elu	33.60%	83.22%
3	16	Elu	26.83%	84.62%
4	8, 4, 8	Elu	31.44%	89.32%
5	16, 8, 16	Elu	24.93%	86.79%
6	16, 8, 4, 8, 16	Elu	31.44%	81.69%
7	16, 8, 4, 2, 4, 8, 16	ReLu	3.79%	60.87%
8	16, 8, 4, 2, 4, 8, 16	Elu	30.89%	91.20%
9	16, 8, 4, 2, 1, 2, 4, 8, 16	Elu	23.04%	90.43%
10	24, 18 ,12, 6, 2, 6, 12, 18, 24	Elu	21.68%	84.21%

First of all, it can be confirmed that the ELU is the preferred activation function when compared to the ReLU (Clevert et al., 2015). Surprisingly, the shallow neural network do not perform badly compared to the deep neural networks. Nevertheless, the deeper variants seem to perform a bit better as probably more information can be captured by each layer. The most promising configurations are Model 4 and Model 8, having comparable anomaly performance metrics. These configurations investigated further by analysing how the models perform for different threshold values τ.

Threshold analysis

The configurations of Model 4 and 8 from Table 17 are tested for different values of τ by using a grid search. Figure 33 shows the results of these analyses. When comparing both figures, it can be seen that for Model 4 the metrics both drop to zero after a threshold of 8, which means that no data points are classified as anomaly after this threshold. The graph of Model 8 shows that there is a moment when all of the anomaly samples are classified correctly, although the recall is very low at this point. The slope of the recall line is a bit less steep for Model 8 compared to Model 4. However, in terms of performance metrics there is not a significant difference between the two models. Since more layers increase the likelihood that more information is captured, Model 8 is chosen as the most suitable configuration for the problem at hand.

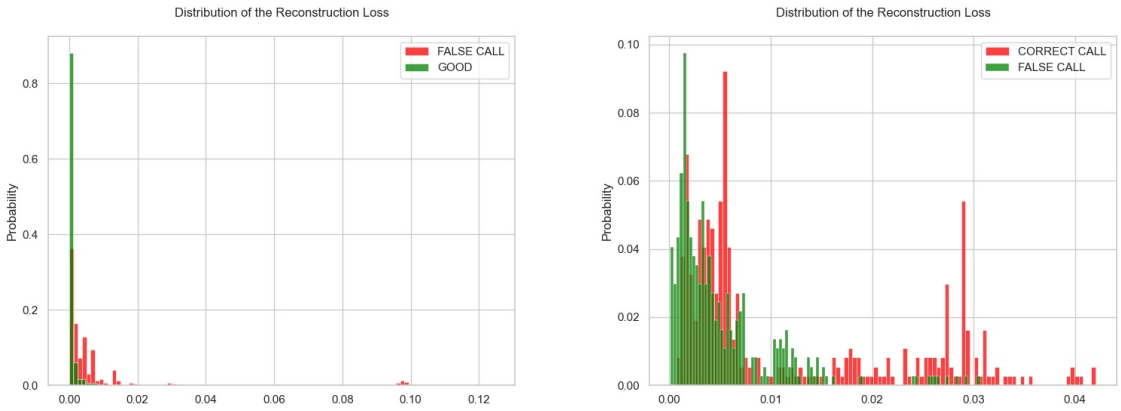


(a) Model 4 (b) Model 8

Figure 33: Model performance for different values of τ

6.4.4 Autoencoder ensemble for target classification

After determining a suitable network configuration, the model can be used to classify the target variables. For the problem at hand, it is important to be really sure about a prediction before overruling the machine call with an algorithm. Table 16 and Figure 33 show that only learning the false call data as normal behaviour and classifying the real errors as anomaly data is not sufficient to solve the business problem. Using the hard threshold as a decision boundary for the problem is too sensitive to the edge cases. Too substantiate this claim, the reconstruction errors of Models 1 and 4 in Table 16, are depicted in Figure 34.



(a) Good - False flag (b) False flag - Real error

Figure 34: Reconstruction error distributions

The figures show that there are too many uncertain cases (overlapping reconstruction errors) when only using one autoencoder. Therefore, using multiple autoencoders which learned representations of different normal data sets can help to further differentiate the target classes. Additionally, uncertainty can be added to the model in the form of a third target class. Model A learns the representation of the good components and Model B learns the representation of the false error flags. These models can collectively classify new data instances by using boolean logic. This logic is schematically represented in Figure 35. The input sample is an error flag, as explained in Figure 28. When both the models classify the incoming sample as normal data it is more likely that it is indeed a false call, as models A and B classify it as a good component and false call respectively. Whenever both models classify it as an anomaly, models A and B both classify it as a false call and a real error respectively, increasing the likelihood that it is

indeed a real error. When there is no agreement between the models regarding the sample, the sample (machine call) requires the additional manual check to prevent error slip.

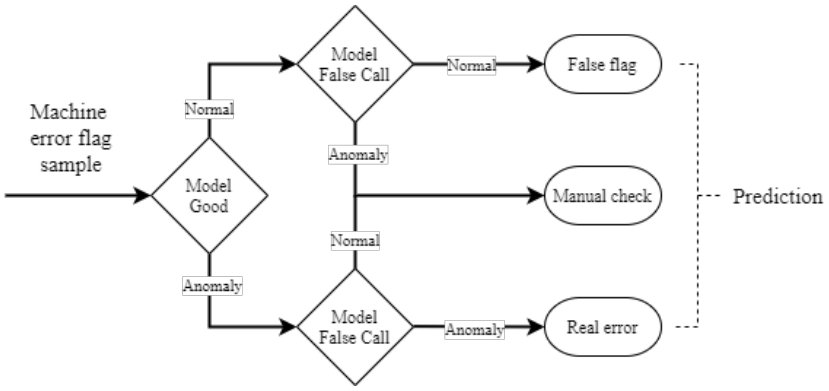


Figure 35: Boolean logic autoencoder ensemble classifier

The concept can also be seen as dividing the reconstruction error of each sample per model in a two dimensional space. Each dimension represents the reconstruction errors of a model, which are then classified by using a linear threshold τ for each model. This other representation is somewhat similar to dividing a search space with a decision tree. A dummy example of classification using the boolean logic in as explained in Figure 35, is depicted in Figure 36.

Figure 36: Dummy example of the boolean logic

Multiple models potentially increase the amount of information which can be learned, but setting the threshold for each model becomes even less trivial. The thresholds interact with each other to come to a classification, which is why it is not as easy as finding the best threshold for each model. An feasible solution is to do a grid search over a range of threshold combinations. The goal is to find a combination of threshold values which minimizes the misclassification ratio's. Besides, it is also important to minimize the number of manual check predictions especially for the false calls, as this will greatly reduce the number of manual checks. Two metrics can be considered using the Pareto frontier representing the metrics graphically on a 2D space. This enables finding a set of feasible solutions. The goal of the grid search is to find a combination of threshold values which minimizes the ratio of real errors classified as false flags (to prevent error slip) and maximize the ratio of false flags classified as false flags (to decrease the manual checks). Figure 37 visualizes the Pareto frontier respective to the two chosen performance metrics.

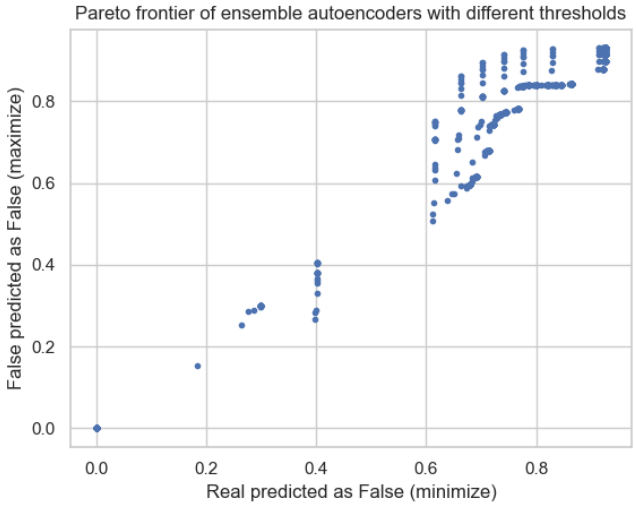


Figure 37: Pareto frontier result of threshold gridsearch

What can be seen in the frontier is that there are not any feasible solutions. Whenever the correct false error flag predictions are maximized, there are also many real errors predicted as false error flags (as seen in the top right corner of the graph). When the misclassified real errors are minimized then also almost no false calls are classified as false calls. Both findings indicate that the boolean logic with hard threshold values as described in Figure 35, lead to many false flag predictions, both correct and incorrect. In order to see how the predictions behave over the different target classes, the point where both the axes are approximately 0.4 is further investigated. The threshold values for this point are 4.5 for Model A and 0.5 for model B. The confusion matrix of this model is shown in Table 18.

Table 18: Confusion matrix with $\tau_A = 4.5$, $\tau_B = 0.5$

Actual \ Predicted	False flag	Real error	Manual check
False flag	141	47	182
Real error	149	80	141

First of all, the matrix shows that by using the boolean logic of the ensemble autoencoders an additional target class is added. For these threshold values the two models do not agree in 43.6% of the cases. This indicates that there are many edge cases in the data for which the distributions do not clearly tell whether it is a real error or a false call. Furthermore, there are 149 real errors classified as false flag which would result in a high error slip. Also 47 real errors were actually false error flags, which may cause unnecessary workload at the repair stations. The results of both the Pareto frontier and the chosen example show that the boolean logic of combining autoencoders based on hard τ values does not provide the wanted results. Thus, using hard threshold values to classify the reconstruction errors is not a suitable method for the problem at hand, due to the fact that the classifier must be really sure when a target is predicted. If not this is not the case, then a sample must be classified as uncertain, leading to a manual check. The following section will propose methods to define a threshold which can include more flexibility in the decision making.

6.4.5 Learning a threshold function in a multidimensional space

An additional layer of uncertainty can be added to the model by using a threshold range over each reconstruction error space instead of a hard threshold. In that case, all points smaller than the minimum of the threshold range are classified as normal samples and all points larger than the maximum range value are classified as anomaly samples. The samples which fall in the threshold range are then classified as uncertain points, requiring a manual check. However, there are several challenges when using this method. First of all, setting the threshold range for the best model performance is even less trivial

than using a hard threshold. Even when only using one autoencoder, searching the solution space of the minimum and maximum range values can be a computationally heavy task when done by brute force. If this problem is simplified by choosing a standard range and moving that range over the search space as a sliding window new challenges emerge (e.g. finding the right standard range and defining the optimal step size for optimization). Combining multiple autoencoders enlarges the hyperparameter search space of the method, increasing the number of possible parameter sets with the power of 2 each time an additional model is used.

One of the reasons for the misclassifications with the boolean logic is probably that using a linear decision boundary for each autoencoder seperately limits the search space, see Figure 36. Therefore a method is proposed which combines the reconstruction errors from the two model in order to create a multidimensional search space. Still, each autoencoder learns a separate representation of a different normal data set, but the reconstruction errors of the validation set are then used to learn a decision function. It is assumed that the reconstruction errors of the false calls are closer to the origin of the space than the real errors. This creates a potential to divide the space in normal and anomaly samples with a decision boundary. Samples which are close to that boundary are assumed to be edge cases, and predicted as uncertain cases. A dummy example of a two dimensional space divided by a learned decision boundary is shown in Figure 38. The search space is the result of the unsupervised autoencoders which can be sensitive to noise (Chalapathy & Chawla, 2019). To reduce this sensitivity, supervised learning methods can be used to create decision rules dividing the subspace (Chalapathy & Chawla, 2019). These decision boundary can then be interpreted as a flexible threshold dividing the results of the autoencoders in false error flags, real error flags, and manual checks.

Figure 38: Dummy example of learned decision threshold

Support vector machines are an obvious choice for this problem, as the goal of the algorithm is to find a (non-linear) decision boundary separating samples in a multidimensional space, as explained in Section 3. This way a flexible threshold is learned in the form of a decision boundary separating the classes in the best possible way. The soft margin theory used for learning the support vectors serves as an inspiration for defining a margin around the decision boundary in which samples are classified as uncertain. If data points in the multidimensional space cannot be separated easily, the model classifies it as an uncertain class. Furthermore, the algorithms hyperparameters (C & gamma) provide a compact search space to find a decision boundary which generalizes well on new data other than the validation set. When using this hybrid machine learning method, the latent representation of the trained autoencoders is used as a feature engineering step. The imbalanced target class distributions are presented to the supervised learning methods in a different way than in Section 6.3. The reconstruction error samples which are used as input for the supervised model are treated as a separate machine learning problem. This means

that K-fold group splits are used to train, validate and test the model, again making sure that no data leakage takes place. A complete conceptual overview of the hybrid machine learning method is depicted in Figure 66, Appendix D.

Average reconstruction error results
For each sample x_i which is predicted by the trained autoencoder, the reconstruction error of each feature in x_i is returned. This is the squared difference of the real feature value and the predicted feature value. In other words, the autoencoder provides information related to how well it can reproduce each feature in x_i. Finding the mean reconstruction error for x_i is done by calculating the mean of all feature reconstruction errors. Doing so for both autoencoders creates a two dimensional search space, as depicted in Figure 38. The results of the default support vector machine separating the two dimensional reconstruction error space, are depicted in Table 19. The model predicts a manual check if the distance of a sample to the decision boundary is smaller than 1. Compared with Table 18, the method is significantly better in separating the classes.

Table 19: Confusion matrix learned decision boundary for average reconstruction errors

Actual \ Predicted	False flag	Real error	Manual check
False flag	87	14	137
Real error	2	16	107

Feature reconstruction error
To enable further learning, the feature reconstruction errors are used to create a multidimensional space. Each of the two autoencoders returns a reconstruction error for each feature, resulting in a total of 70 dimensions. Again, a decision boundary is trained with 5-fold cross validation. The result of the classification based on the feature reconstruction errors is depicted in the confusion matrix in Table 20. Compared to the average reconstruction error model, it can be seen that the model is better at correctly detecting false flags in the data. It does not directly lead to better results for the real error class, as more real errors are predicted as false flags which is unfavourable. However, this model seems to have the most potential due to the information increase which is brought by the higher dimensional data. Therefore, a grid search is done for the model's hyperparameters in order to find the best model.

Table 20: Confusion matrix learned decision boundary for feature reconstruction errors

Actual \ Predicted	False flag	Real error	Manual check
False flag	113	3	121
Real error	8	20	96

Hyperparameter tuning
The hyperparameters tuned for the support vector machine are C and gamma, combined with the margin to the decision boundary to predict the uncertain class. In total 378 different configurations are tested. Just as for the balanced bagging random forest in Section 6.3.3, the misclassification ratio's for both classes and the manual check ratio should be minimized. Again, three model outcomes which each serve a different goal are chosen for comparison, from high precision in both classes to a low manual inspection percentage. The results of the models are shown in Table 21.

Table 21: Hyperparameter search results hybrid machine learning

			Real error			**False call**			
C	**gamma**	**Margin**	**Recall**	**Precision**	**Error slip**	**Recall**	**Precision**	**Redundant repair**	**Manual ratio**
10	10	2	0.03	0.96	0.01	0.17	0.98	0.00	0.97
1	10	1	0.20	0.83	0.06	0.5	0.93	0.01	0.58
10,000	10	1	0.54	0.86	0.29	0.84	0.83	0.04	0.13

The results show that when the precision of the real error class is high, the recall is relatively low. The low error slip and the high manual ratio indicate that much of the real errors are classified as edge cases and thus predicted as the uncertain class. When the recall of both classes is higher, the number of manual checks also decrease. However, the precision of both classes is reduced and the error slip is relatively high, which is not preferred. To choose the most feasible method, the next section will compare the random forest classifier with the classifier based on the reconstruction errors.

6.5 Concluding Remarks Modeling Phase

The final section of this chapter will conclude the section by briefly comparing the balanced bagging classifier and the anomaly detection classification method. The methods are compared through the various performance metrics. The best method will then be used to find the most suitable model from a business perspective in Section 7.

When comparing the results of the hyperparameter tuning of the methods in Tables 15 and 21 it turns out that adjusting the hyperparameters brings quite some flexibility to both methods. Comparing the first model in both tables, where both methods have a high precision for the classes, the balanced bagging random forest seems to outperform the anomaly detection method. Both the recall and the number of required manual checks show better results, indicating that the method is better at detecting real errors without classifying them as uncertain predictions. For the second set of configurations the balanced bagging random forest performs better for all the performance metrics. The recall is higher for both classes, the precision is also very high compared to the anomaly detection method and the manual check ratio is also lower. Both models seem to handle uncertain predictions relatively well, as the error slip and redundant repair values are not very high. For the final configuration, with the lowest manual check ratio, the error slip for both models increases to a certain extend. However, the balanced bagging classifier method again outperforms the autoencoder method as it still has a smaller error slip and a higher precision for both classes.

Overall, it seems that the balanced bagging random forest method is a more suitable method to predict whether a machine call is a real error or not. This result is somewhat unexpected because the anomaly detection method also could learn from the good cases, providing an additional dimension to the data. However, an important assumption regarding this method is that the distributions of the normal and anomaly data sets are substantially different. Most probably there is not enough difference in the set of process features of the good inspection results, false calls and real errors. A reason for this could be the fact that an error flag can occur due to the a slight deviation in a certain feature. It is possible that the autoencoder was not able to detect major differences between the data distributions of the classes, providing a somewhat noisy reconstruction error input for the support vector machine classifier. Furthermore, the random forest classifier was able to learn more from the false call majority set in a supervised manner, as it had more samples to learn from. For the autoencoder method, these majority samples were used to learn the autoencoder what normal behaviour is. Then only a slight subset was used to train the supervised method with the learned representations of the autoencoders. The results show that for the problem at hand it, is better to provide more (majority) data to the supervised learning method than using this data to create a different representation, in order to tackle the heavily class imbalance problem. Furthermore, it is not trivial to decide which balanced bagging classifier configuration performs best in terms of business goals. Just picking the model with the highest precision does not solve any real problems. It is necessary to evaluate the balanced bagging random forest in relation to the business case. In the next section real world costs are used to estimate which model leads to the most business benefits.

7 Business Evaluation

This section covers the evaluation phase of the CRISP-DM methodology by evaluating the selected model from a business perspective. As shown in Figure 2, this phase is done in close collaboration with the company as it requires deeper business understanding related to the different model outcomes. First, the evaluation method is proposed by attaching the model results to business outcomes. Then this evaluation method is used to find the best model configuration in terms of hyperparameters. The results of the best model are examined, and explained using SHAP values. Eventually, the final model is then evaluated on the test set which enables estimation of the added business value of the model.

7.1 Evaluation method

Evaluating the balanced bagging random forest classifier from a business perspective requires certain business assumptions. Based on these assumptions regarding the costs of the model's outcome, the most feasible model is chosen. The company states that the error slip is the most important performance measure because it is both undesired in terms of costs but also from a customer satisfaction viewpoint. Due to the importance of the competitive advantage on the market, AME cannot permit to have a bad reputation in terms of product quality. They aim to have an error slip ratio which is smaller than 0.3%. The current error slip for the surface-mount device production line is however unknown, as it is complicated to directly assign a defect on a returned PCBA to a problem in the production line. The costs when a customer returns an insufficient product are approximated at €20. Redundant repairs (products which go to the repair station but do not require a repair) are the result of predicting a false call as a real error. This error is less severe compared to an error slip, but still not preferred as it can congest the repair station. The actual redundant repair rate is approximated to be negligible, which should also be the aim of the model. There are no direct costs associated to these repairs. AME expresses all time related inefficiencies in terms of operator costs. The salary of an operator is €36 per hour, which means each second costs 1 eurocent. It is approximated that a redundant repair costs 45 seconds, thus €0.45 per redundant repair. This logic can also be used to determine the savings related to reduced manual inspections. As mentioned in Section 4, each machine call takes approximately 4 seconds, which means that each manual inspection costs €0.04. The costs of the error slip, redundant repairs, and manual inspections can be used to create a linear cost function from a business perspective. The outcome of the 5-fold cross validation combined with this linear function, results in a cost value to determine the most feasible hyperparameters. The linear function is depicted in Equation 10.

$$c = 20 \times \textit{error slip} + 0.45 \times \textit{redundant repair} + 0.04 \times \textit{manual inspection} \tag{10}$$

7.1.1 Soft constraints

The skewed costs of the manual inspection will probably result in a large bias towards having as much manual inspections as possible, which is not desired. During the assessment of the hyperparameter search results, it indeed appears that the models with the lowest costs have a manual inspection rate of 100%. For these cases, no misclassifications are done and only manual inspections are required. These results make it seem as if using no model is more preferred due to the lack of mistakes. However, when all machine calls require a manual inspection it is still very plausible that error slips and redundant repairs happen. The fact that the current costs related to these human errors are unknown, is why it is required to add additional soft constraints for choosing the model. Otherwise, the current situation could have been compared with the model outcome to find the configuration which provides the largest improvement. First, the reduction of manual inspections should be as high as possible in order to solve issues related to the false call problem. Unfortunately, these issues cannot be directly related to monetary values (e.g. operator workload and line efficiency), which is why it is necessary to assess the best model in a subjective way. Second, the error slip ratio is preferred to be smaller than 0.3%, and should be as low as possible at all times. Lastly, the redundant repair rate also cannot be too high, due to the fact that this results in repair station congestions.

7.1.2 Choose model with cost function

The minimum cost value of all 630 hyperparamter configurations is €77.08, and the maximum cost value is €2442.30. When taking the soft constraints into account, the most feasible model for the problem at hand is €184.36 based on the 5-fold cross validation. Note that due to the fact that cross validation is used, it is possible that some folds perform relatively bad due to the small minority sample in these folds. This can particularly result in overestimations of the error slip. The performance metrics and the confusion matrix are given in Table 22 and 23 respectively. These results show that the model is very good at detecting false error flags relative to detecting real errors. This is probably caused by the fact that this minority class is less present in the 5 folds on which the model was trained. The relatively high model probability cut off of 0.90 in combination with the model configurations, show that approximately 45% of the manual inspections can be reduced. Although there are no redundant repairs, the error slip of 2% is a bit too high for the business case of AME.

Table 22: Most feasible model cross validation performance metrics

				Real error			**False call**			
Max. depth	**Min. samples split**	**Estimators**	**P. cutoff**	**Recall**	**Precision**	**Error slip**	**Recall**	**Precision**	**Redundant repair**	**Manual ratio**
None	10	250	0.90	0.18	1.00	0.02	0.58	0.99	0.00	0.55

Table 23: Confusion matrix 5-fold cross validation most feasible model

Actual \ Predicted	False flag	Real error	Manual check
False flag	722	0	564
Real error	7	91	545

7.1.3 Company benefit estimations

To assess the final performance of the model, the model configuration is retrained on all training data and examined on the test set. It is expected that these results are slightly better as the model has more minority data to learn from. As stated in Section 6.1, the test set is balanced to reduce any bias in the evaluation. The test set includes 8 product types, 281 panels and has 408 samples. Among these samples, there are 291 false call instances and 117 real error instances. The test set performance metrics and the confusion matrix of the model trained on the complete training set are respectively given in Tables 24 and 25. As expected, performance on the test set is a bit better, probably due to the increase minority samples during training. Most noticeable is that the error slip approximates zero. This indicates that having more data to learn from enhances the separation ability for the minority class.

Table 24: Confusion matrix test set

Actual \ Predicted	False flag	Real error	Manual check
False flag	173	1	117
Real error	0	24	93

Table 25: Test set performance metrics

				Real error			**False call**			
Max. depth	**Min. samples split**	**Estimators**	**P. cutoff**	**Recall**	**Precision**	**Error slip**	**Recall**	**Precision**	**Redundant repair**	**Manual ratio**
None	10	250	0.90	0.21	0.96	0.00	0.59	1	0.01	0.51

It is not trivial to estimate the monetary savings when implementing the tool in the production line, as the current costs and ratio for error slips and redundant repairs are unknown. Nevertheless, the results

show that it potentially 50% of the manual inspections can be reduced. As shown in Section 4.2.4, there are 366,641 false error flags occurring in the recent year. Using the model would approximately reduce the number of manual inspections by half, saving 183,000 inspections yearly. Each day, approximately 30 minutes of manual inspections can be saved. This leads to benefits both from a human capital perspective and a production line efficiency perspective. From a human capital perspective this means that less manual checks are required, making the work less tedious which reduces the probability of error slips. Reducing the number of manual checks also indirectly increases the throughput.

Besides the enhancements on a component level, it is also possible to create an estimation for the benefits from a panel perspective. This estimation regards the reduced number of panels which require a manual inspection. Reducing the manual inspections per panel rather than per component provides additional benefits, as not stopping a panel at all (compared to checking less calls) further reduces operator workload and increases throughput. To estimate the reduction in panel inspections the following assumption applies: if there is no manual check class prediction done by the model for a given panel, then the panel does not require a manual inspection thus is considered as a saving. The results show that 147 of the 281 panels in the test set do not require any manual checks if the model is used, which results in approximately 52% less panel inspections. To further investigate this on a product type level, the panel savings per product type are given in Table 26. There seems to be a slight relation between the average calls per panel for a product type and the reduced manual inspections. This is expected as panels with only a few falls error flags require less predictions, so there is less chance that the model predicts the uncertain class. Lastly, the potential monetary savings are also shown in the table. These values only indicate the direct savings of the model, which relates to the reduction of manual inspections (being €0.04 per reduced manual inspection). Note that the values are small due to the relatively small amount of panels. Other potential euro savings regard the increase in throughput and efficiency, leading to an increase in turnover due to an increase in the production amount. Next, an explanation of the model's outcome regarding is given in the following section.

Table 26: Panel ratio without manual inspection

Product type	Total Panels	Mean Calls per Panel	Panel Savings	Saving Percentage	Direct Euro Savings
6023-1600-0605	8	1.88	5	62.5%	€0.38
6047-1800-9204	113	1.30	84	74.3%	€4.38
6649-1000-2226	9	1.11	0	0%	€0
6661-1900-0501	9	1.11	0	0%	€0
6736-1504-2007	56	1.31	19	33.9%	€1.00
6736-1602-9407	22	1.24	10	45.5%	€0.50
6761-1200-5901	31	1.10	26	83.9%	€1.14
6782-1700-1809	33	2.18	3	9%	€0.26

7.2 Explaining the model

Besides the savings related to the manual inspections, the machine learning model's outcome can also be used to create additional process understanding. Increased interpretability of a machine learning model leads to easier adoption within the business setting, which is why this is an important evaluation step. Explaining the model's outcome can be done by analysing the samples predicted as the uncertain class, and utilizing SHAP values to interpret the impact of a certain value for the target feature. Both serve as new input for further modeling improvements and can create additional business understanding related to the false error flags.

Global interpretability

The collective SHAP values of the training set are used to examine the feature importances of the model, also known as the global interpretability. For each balanced bootstrap sample in the balanced bagging classifier, the SHAP values are calculated with the random forest trained on that bootstrap sample.

These values are aggregated per feature and the 20 most important features are shown in Figure 39. The plot is generated by all the samples and their SHAP values and includes information about the feature importance, impact and original value. Importances are shown by the vertical ordering of the variables, decreasing from top to bottom. The impact is given by the horizontal position on the axis. For the problem at hand, a prediction of 0 means a real error and a prediction of 1 means a false call. Thus, samples with a negative SHAP value lead to real error predictions and positive values lead to false call predictions. Based on the colors of the samples, information regarding the original value of a sample is provided. Red means the value of the sample is relatively high and blue means the value of the sample is relatively low. Combining both the impact and the original values provide insights in the correlation with the target variable. To test the robustness of the method, the SHAP values are compared with the feature importances as computed by the random forest. These are the averages of the impurity decrease within each tree, and shown in Figure 40. Both methods show almost the same ordering of the variables, confirming the robustness of the SHAP values.

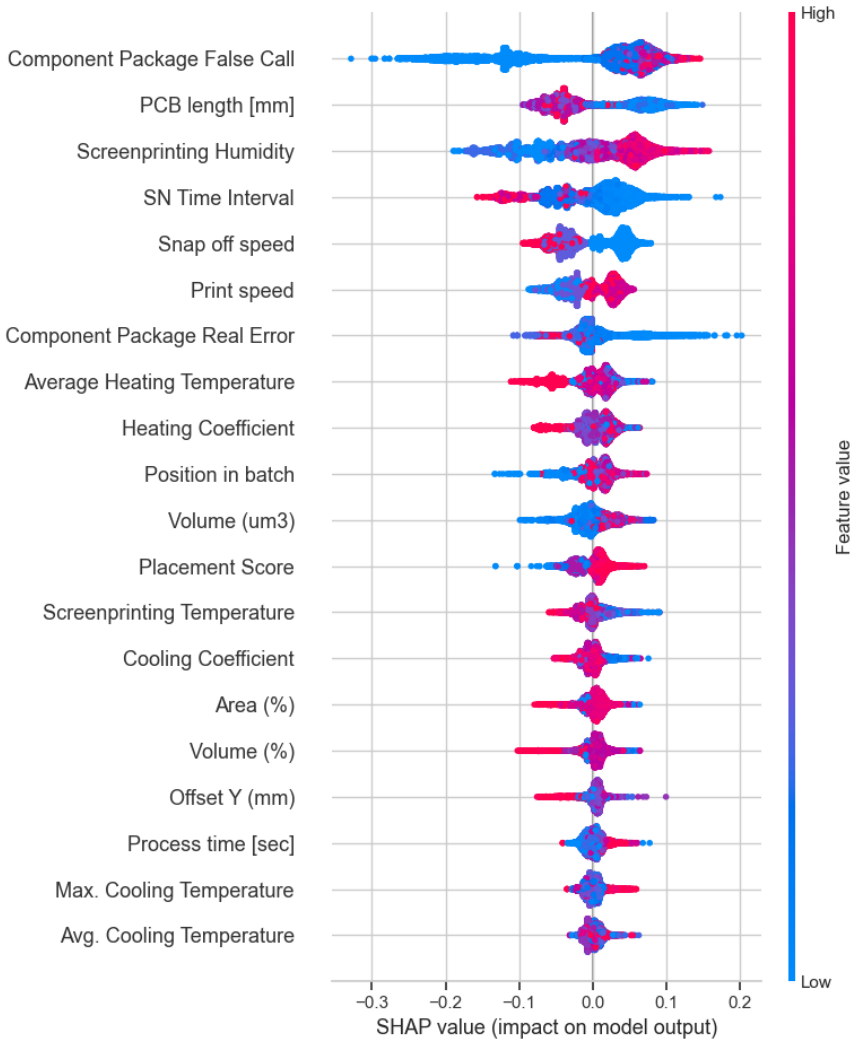


Figure 39: SHAP values for real errors (left) and false error flags (right)

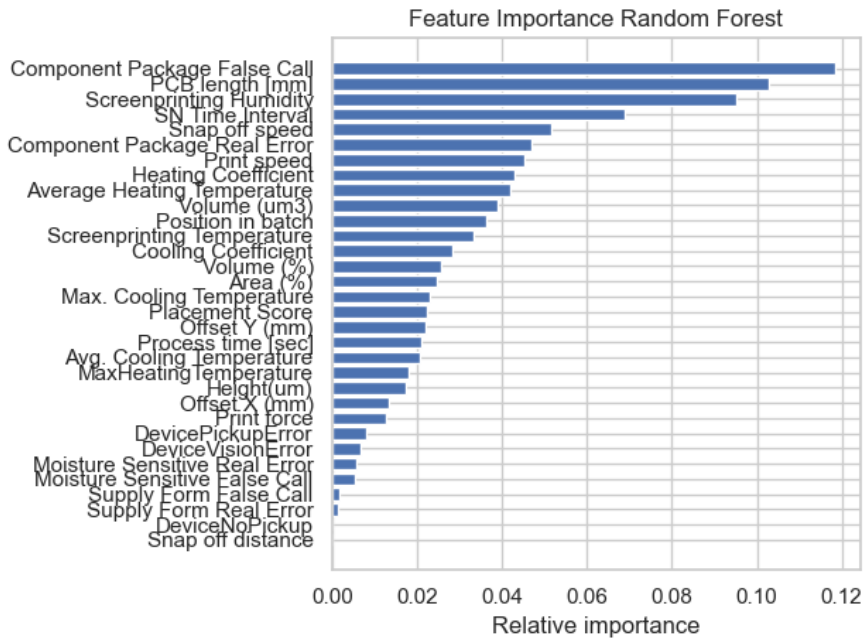


Figure 40: Feature importance random forest

The plot with the SHAP values can be used to explore the relationship between the process and product quality. Most interesting are the top features, as these provide the most information when doing a prediction. It is interesting to see that the impact and values for some feature are separated generally well, providing information about the relation between the quality inspection outcome and the feature. Before examining the results, it is important to note that these values do not serve as causal relations, but only provide insights to the associations between the process features and the target variable. Directly interpreting these results as casual relations might result in wrong conclusions due to spurious relations in the data. First, the component package encoded feature seems to have a great influence on the predictions. A low value for this feature means that the component package is not frequently involved in false calls, based on historical data. Furthermore, smaller PCB lengths seem to have a stronger relationship with false error flags and larger PCB's seem to be more involved in real errors. For screenprinting humidity, it seems that most errors happen when the humidity is low during screenprinting. Furthermore, if the time interval between products is low, less real errors happen. Increased time between products can for instance be caused by lunch breaks, shift rotations or machine setups. The company confirms that these occasions can negatively influence the product quality.

Screenprinting settings also influence the prediction of the product quality. Lower snap off speed seem to have a positive relation with the product quality. A high print speed also seems to have a positive relation with the product quality. Be aware that this result can also be found due to the fact that products which are prone to more errors, have slower printing speed setting. Paste values measured with the API are less important but show that high values for area, volume and offset are slightly associated with real errors. The placement score, which captures the quality of the component placement shows that a higher score is associated with false calls thus better quality. For the reflow temperature features, the plot shows that higher average temperatures and larger heating coefficients (faster temperature increase) are associated with real errors. Note that many of these samples are centered around zero, indicating that if the reflow values are not extreme there is not much information in these features. Overall, no set of features related to a specific SMD sub process contributed most to the model. Potentially, the strength of the model lies in the fact that it combines both product, and process features of all the sub processes in the surface-mount device production line.

Local interpretability

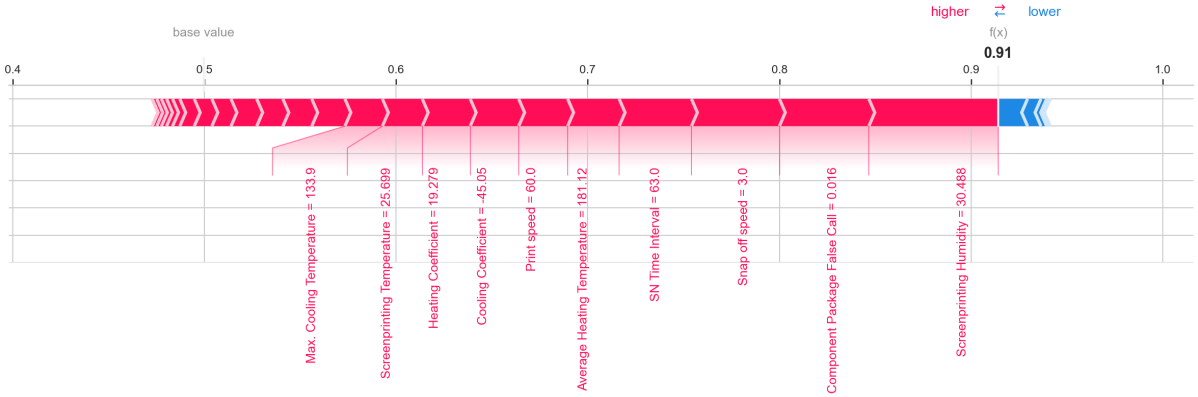
Local interpretability regards the analysis of individual samples which are predicted by the model. It

could be the case that specific error categories are harder to detect well. Table 26 already provided information related to the product types and model predictions. For each of the misclassifications in the test set, it interesting to see the error message distributions. This information could provide further insights in why the model makes certain decisions to potentially improve the performance. An overview of the error message distributions is given in Table 27. The error types are ranked vertically in decreasing order from most occurring to least occurring in the complete data set (both training and test). For the false error flag classes, there is no clear distribution between the predicted classes and the error type. However, for the real error class it is shown that the model distinguishes the coplanarity error category best. It is also able to find some of the pad overhang classes but is not very successful in classifying the other real error types.

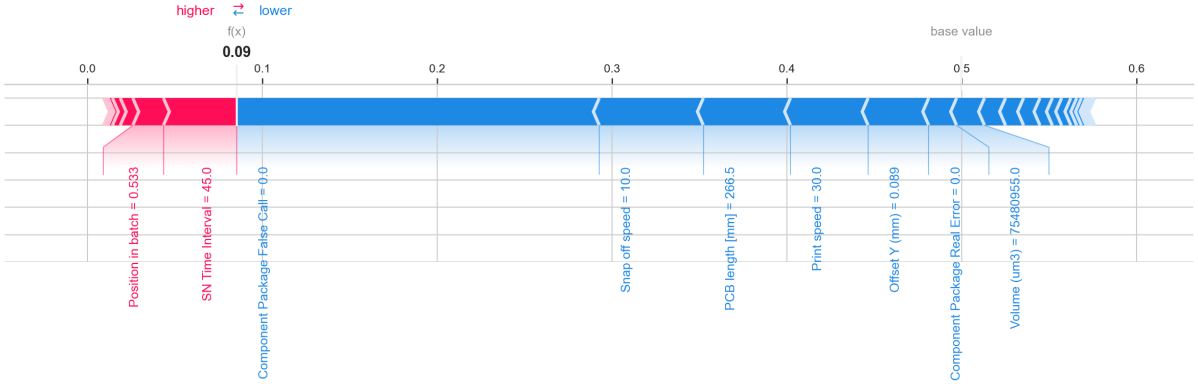
Table 27: Error types and predictions

Actual	**False flag**				**Real error**			
Predicted / **Error type**	**False flag**	**Real error**	**Manual check**	**Absolute total**	**False flag**	**Real error**	**Manual check**	**Absolute total**
Pad overhang	61%	1%	38%	110	-	8%	92%	37
Coplanarity	54%	-	46%	54	-	64%	36%	33
Solderfillet	57%	-	43%	58	-	-	100%	8
Polarity	67%	-	33%	49	-	-	-	0
Missing	78%	-	22%	9	-	-	100%	21
OCROCV	40%	-	60%	10	-	-	-	0
Dimension	-	-	100%	1	-	-	100%	1
Bridging	-	-	-	0	-	-	100%	15
Absence	-	-	-	0	-	-	100%	2

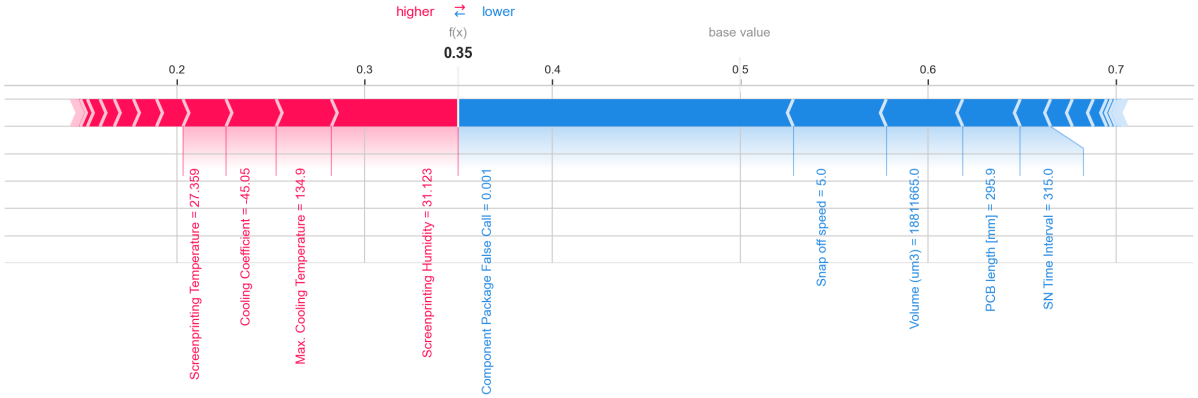
To further investigate these findings, the individual SHAP values of these points can be used to see which features were used to predict these cases. Three samples are inspected namely, pad overhang correctly predicted as false error flag, correctly predicted as real error, and a real error predicted as manual check. The first two show the difference between a false call prediction and a real error prediction, and the latter depicts a sample for when the model is not certain enough. These instances as shown in Figure 41 serve as an example for how the model's predictions can be explained relative to their feature values.



(a) True False Error Flag



(b) True Real Error



(c) Manual Check - Real Error

Figure 41: SHAP values: feature contributions to sample predictions

A predicted value $f(x)$ close to zero means that the model predicts a real error, and close to one means a false error flag prediction. If the model is uncertain as in Figure 41c, depicting an uncertain real error prediction, the manual check class is predicted. The base value is 0.5, and the arrows indicate how strongly each feature is pushing the prediction in a certain direction. When examining Figure 41a, the main drivers of the false call prediction are the humidity during the screenprinting process, the fact that the component package which is checked often occurs as a false call in the entire data set, and the low snap off speed. Another interesting feature is the small time interval between the previous product and this product, which also pushes the prediction to the false call class. Figure 41b also shows that the small time interval pushes the prediction to the false call, but the impact of the features pushing the prediction to a real error are much stronger. The main driver is that the component package does not seem to be a package which results in many false calls. For this prediction, contrary to the latter, the

high (instead of low) snap off speed pushes the prediction to a real error. For the uncertain prediction, depicted in Figure 41c, the prediction is pushed in both directions without a dominant set of features for each class. What is interesting to see is that the sample has both a high screenprinting humidity pushing the prediction to the false call, but also a component package which is associated with real errors. Due to these contrary features, the model is not certain enough and a manual inspection is required. The first two samples showed that the small time interval between the samples and previous products pushed the prediction to the false error flag prediction. The large time interval for the last sample presents the opposite behaviour, it pushes the prediction to a real error. Combining the local SHAP values with the plot in Figure 39 creates a complete picture regarding the predictions, which also shows that large time intervals between products are more likely to be associated with real errors. In general, the figures provide a intuitive insight in how the model makes predictions. These examples can either be used to validate the results with domain knowledge, improve the domain knowledge or enhance the model by detecting potential noisy or missing features.

8 Conclusion

This research proposed a machine learning model to enhance the automated optical inspection for surface-mount devices in an electronics manufacturing environment. The model enables reducing the false error flags by predicting whether machine calls are real errors or false calls. The concluding section of this thesis summarizes the main findings of the research by answering the research question as proposed in Section 2.3. Each of the sub questions is extensively answered by the sections in this report. Therefore this section provides a brief summary of the findings. Summarizing the sub questions will collectively provide an answer to the main research question:

What (explainable) data-driven model can be developed with process data to reduce the false calls during the quality inspection, in order to improve line efficiency and decrease operator workload?

After answering the main research question by collectively answering the sub questions, the (business) recommendations, limitations, and future research are described finally.

8.1 Main Findings

AME uses an automated optical inspection as current practice for the product quality control of the surface-mount devices. The default inspection tolerances are based on industry wide arrangements. Due to the high customization at the manufacturer, it is required to fine tune these tolerances, optimizing the quality control. This tuning is mainly about finding the right balance between error slips and false calls. However, the vast amount of different components causes it to be infeasible to fine tune all the different inspection programs. This and other (external) factors result in false calls on approximately 60% of the products, leading to line inefficiencies and additional operator workload. Literature regarding the topic endorse the falls call problem in electronics manufacturing. These papers propose (machine learning) methods to enhance the automated optical inspection, by improving the image recognition algorithm or reinforcing the automated decision with a prediction based on process data. However, the models proposed in the literature are limited to the analysis of one product, error type, or solely use the data of one sub process. This prevents the model to be generalized over multiple products. This research extends the current literature by using the complete SMD production process data, including sensor data, machine settings and product characteristics for multiple product types. This enhances the generalizabilty of the model, thus the overall usefulness from a business perspective. Further extending this business utilization is done by adding explanations of the modeling results in terms of process features, increasing the potential to capture expert knowledge.

When gathering the data, issues occurred mainly due to missing data or the inability to link different data sources. Exploring the data showed that the target class was highly imbalanced. It also showed that troublemakers can be either board locations or components, and that it is not clear at which moment in a batch the most errors happen. Furthermore, (correlating) process variables required additional feature engineering. This included extracting reflow process heating zone features, such as the linear coefficient for the change in temperature and the average and maximum temperatures in the zones. To suppress the dimensionality increase, the categorical features were target encoded. The product and process features where then used to develop and test different data-driven models, based on imbalanced machine learning and anomaly detection. It turned out that for the problem at hand, the imbalanced classification model performed best. This is probably due to the fact that the data distributions of the different classes did not differ to such an extend, that the reconstruction errors could be used to detect anomalies. Furthermore, the imbalanced machine learning classifier was trained in such a way that it had more potential to learn from the majority class than the anomaly detection method, which could also explain the difference in performance. Nonetheless, there is potential in using the reconstruction errors of multiple autoencoders as a feature engineering step in imbalanced problems, as it can capture information from the complete majority set.

The balanced bagging random forest classifier with an additional uncertainty filter dependent on the class probability, is the best performing model. Based on the test set, the model is able to reduce approximately 50% of the manual component checks (approximately 183,000) by predicting whether a machine call is false or not, without increasing the error slips or redundant repairs. In terms of panels,

a first simple estimation showed that 52% less panels require a manual check. This saves approximately 30 minutes of manual inspections each day. Implementing the model in the production line improves the line efficiency (increasing the throughput) and reduces the operator workload (increasing the quality). Besides the improvements from a production line perspective, the model can also be used to create additional process understanding relative to the product quality. SHAP values can be used to interpret the predictions from both a global and local perspective. The former provides an importance for each feature related to the target variable, and the latter increases the transparency of individual predictions based on their feature values.

8.2 Business Recommendations

The recommendations resulting from this research are divided into several topics. First, an overview of the improvements related to the data storage and gathering is given. Then, general implementation recommendations are provided from a model development, data engineering and software engineering perspective.

Data is the main driver of a machine learning model and in order to make it work properly the data must be clean and complete. One of the drawbacks during this research was the fact that the data related to the different sub processes were stored on different databases, local file directories or even managed by third parties. In order to successfully develop and implement machine learning engines, the data for each important sub process should be easily accessible and up to date. It can be helpful to create a new database dedicated to this task. The database can retrieve the relevant data from other databases or directly from machines. This avoids problems with data merging and accelerates future machine learning projects. Using such a database will also prevent the use of complicated log text file scraping techniques. This scraping step of data collection is unnecessary as it only leads to a duplication of work. When the machine generates the data, the data is handled by saving it in the log file and storing this file in a directory. In addition to this handling, it is recommended to subtract the relevant machine learning data directly from the sub process in the dedicated database. Doing so prevents replicative time consuming data gathering tasks and potential missing data due to unstructured saving directories. Furthermore, the dependencies on local files, either from process engineers or third parties such as machine part suppliers is also undesirable. Machine part and product specific information is currently only stored in local directories and not easily accessible. Adding this general product information to the machine learning database will be helpful. Furthermore, the above mentioned problems (i.e. storing log files as text and having data stored in local directories) leads to merging problems (e.g. for the stencil aperture locations and the board locations) and missingness in the data. Lastly, the component packages database seems rather outdated, as many nowadays standard component packages are still categorized as non-standard components due to the technology shift over the years. This decreases the potential information in the package feature, which is why it is recommended to reassess the package categories of these components.

Implementation of the developed model in the surface-mount device production process is recommended to increase the line efficiency and reduce the operator workload. The implementation of a machine learning engine is an important task and should not be underestimated. There are few things that should be taken into account. To improve the ability to generalize well over the different product types, the training set should be maximized as much as possible. In the best case, the entire data population is used to train one or more machine learning models. When the model is developed and trained, the decision function must be stored on a server which can be used by the software of the AOI. In order to predict possible machine calls, it is necessary that process data related to the product arriving at the inspection station is quickly processed so it can be used as model input. A brief example is provided to elaborate on how the model might be used in the production process. For the example it is assumed that the model is already trained and stored at the inspection station, ready to predict incoming cases.

When product A enters the SMD process, the product is scanned, creating a data profile for the product which is temporarily stored on a server. The static features of this data profile are automatically filled by the machine learning database. During the production of product A, process data is gathered and added to the data profile at every production step. When the automated optical inspection finds three error flags, the data profile of product A is read from the server to retrieve the feature values of those

three components with error flags. This data is used as input for the machine learning model which provides a prediction per flag. If all three calls are assessed as false, and the model is certain about its predictions, the operator does not see any of these calls. However, when a manual check is required, the model shows the operator which component needs a manual check, how certain the model is about its prediction, and what variables (additional to the standard AOI information) guided the prediction. Whenever the model predicts that the board requires a repair, the data relevant for that repair is stored, and a notification is added that the board must go to the repair station. After the predictions, both the results of the predictions and the product's data profile are stored in the machine learning database. This data can then again be used when training new models. Training new models once in a while is important due to the concept of data shift. Certain trends in the production process may change over time as new products are added, reducing the performance of the model. Lastly, using the model in the above described manner minimizes any real-time data handling constraints, as the data size handled locally is only small and can be computed parallel to the process.

8.3 Limitations & Future Research

Finally, the research is concluded by describing the limitations of this thesis and specifying the directions for future research.

The conducted research is subject to several limitations. In the first place, the data set which is used to conduct the research consists of 9 randomly selected product types. Relative to the complete set of product types which AME produces this is only a small sample. This size reduces the ability to generalize the solution over all product types of AME. Furthermore, the size of the sample only provides a small test set which is why the results of the study only are a rough estimation of the performance and business benefits. Another limitation is the lack of knowledge regarding the current performance estimators of the business. Not having this information complicates the model's evaluation in terms of saved costs. Due to time limitations no extensive real world sampling was possible to create an estimation for these metrics, resulting in business evaluations which only remain an estimation. Another limitation is the fact that no sensitivity analysis is conducted for different input data sets (e.g. training the model only on one product type to see its effectiveness). Possibly, the model performs better when only trained on one product type. However, using sub sets with this minority sample size will probably not be beneficial for the general performance. The final limitation regards the lack of analysis conducted regarding the manufacturing system as a whole. Analyzing the manufacturing system provides insights in the current performance of a production line in terms of throughput and utilization. Doing so enables better estimations of the current production line performance, which improves the evaluation of the potential benefits when the model is implemented. This research mainly focused on developing the model and less on how the production line behaves, and what the exact implications of using such model are for the manufacturing system of AME.

The academical goal of the research was to develop a model able to handle imbalanced high dimensional manufacturing data, nullifying the number of misclassifications. During the research, several other directions have been found which can further improve the proposed method. First, the proposed idea of the autoencoder classifier can be further researched and tested on other baseline machine learning problems. The main assumption regarding this method is that the feature distributions of the target classes differ significantly, using the combined reconstruction errors of the autoencoders as a feature engineering method for a supervised learning method. For the problem at hand, the results showed that there are too many cases in which there is uncertainty due to overlapping process feature values. However, the ensemble autoencoder classifier potentially performs well in imbalanced environments with multiple classes, as it is able to learn from the complete majority class sets, losing as little as possible information. Instead of using the reconstruction error as input, the output of the latent space (the most inner layer of the network) can also be utilized as input data for a supervised learning method.

Another potential research direction is to add manufacturing system analysis regarding the current line performance to the evaluation of the model in order to generate a more complete overview of the model's performance. Doing so will be crucially for determining how a false call detection model fits in the surface mount device production line, potentially strengthening the need for real world implementation.

Currently the model is optimised on reducing the total number of false calls, further research can also be conducted to optimise the model in a way that it reduces the number of panels which require inspection. This would need a different approach and data aggregation level. Moreover, the output of the model in terms of line efficiency improvements can also be used to enhance the planning model as stated in Section 2.1. If there is more data available to evaluate the model on, a time saving estimation can be made for each product type in Table 26. These estimations can then be used to improve the input data of the planning tool in order to get a more reliable outcome, in case the false call detection model is implemented. Lastly, expert knowledge in combination with the model explanation can be used to further improve the production line. Not all features which are in the model can be controlled, such as the component package or the PCB length. However, variables which can be controlled such as the printing speed or the heating coefficient, can be used to fine tune the process parameters. After validating the model explanation with domain experts, the controllable features can be optimized respective to the quality. This process can even be automated with metaheuristics (e.g. simulated annealing), which can be used for global optimization in a large search space such as a manufacturing environment (Kirkpatrick, Gelatt, & Vecchi, 1983). By automating the adjustment of process parameters relative to the production quality, AME can further progress in the paradigm of the fourth industrial revolution.

References

Ahad, N. A., & Yahaya, S. S. S. (2014). Sensitivity analysis of welch'st-test. In *Aip conference proceedings* (Vol. 1605, pp. 888–893).

Alpaydin, E. (2020). *Introduction to machine learning*. MIT press.

Altendorf, E. E., Restificar, A. C., & Dietterich, T. G. (2012). Learning from sparse data by exploiting monotonicity constraints. *arXiv preprint arXiv:1207.1364*.

Ayotte, B., Banavar, M. K., Hou, D., & Schuckers, S. (2021). Group leakage overestimates performance: A case study in keystroke dynamics. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 1410–1417).

Baldi, P. (2012). Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of icml workshop on unsupervised and transfer learning* (pp. 37–49).

Banerjee, A., Bandyopadhyay, T., & Acharya, P. (2013). Data analytics: Hyped up aspirations or true potential? *Vikalpa*, *38*(4), 1–12.

Barandela, R., Valdovinos, R. M., & Sánchez, J. S. (2003). New applications of ensembles of classifiers. *Pattern Analysis & Applications*, *6*(3), 245–256.

Batista, G. E., Bazzan, A. L., Monard, M. C., et al. (2003). Balancing training data for automated annotation of keywords: a case study. In *Wob* (pp. 10–18).

Batista, G. E., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, *6*(1), 20–29.

Belhadi, A., Zkik, K., Cherrafi, A., Sha'ri, M. Y., et al. (2019). Understanding big data analytics for manufacturing processes: insights from literature review and multiple case studies. *Computers & Industrial Engineering*, *137*, 106099.

Bellman, R. (1966). Dynamic programming. *Science*, *153*(3731), 34–37.

Boughorbel, S., Jarray, F., & El-Anbari, M. (2017). Optimal classifier for imbalanced data using matthews correlation coefficient metric. *PloS one*, *12*(6), e0177678.

Cerda, P., & Varoquaux, G. (2020). Encoding high-cardinality string categorical variables. *IEEE Transactions on Knowledge and Data Engineering*.

Chalapathy, R., & Chawla, S. (2019). Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM computing surveys (CSUR)*, *41*(3), 1–58.

Chang, Y.-M., Wei, C.-C., Chen, J., & Hsieh, P. (2019). An implementation of health prediction in smt solder joint via machine learning. In *2019 ieee international conference on big data and smart computing (bigcomp)* (pp. 1–4).

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, *16*, 321–357.

Chen, C., Zhou, L., Ji, X., He, G., Dai, Y., & Dang, Y. (2020). Adaptive modeling strategy integrating feature selection and random forest for fluid catalytic cracking processes. *Industrial & Engineering Chemistry Research*, *59*(24), 11265–11274.

Cheng, Y., Chen, K., Sun, H., Zhang, Y., & Tao, F. (2018). Data and knowledge mining with big data towards smart production. *Journal of Industrial Information Integration*, *9*, 1–13.

Clevert, D.-A., Unterthiner, T., & Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*.

Du, S., Lv, J., & Xi, L. (2012). A robust approach for root causes identification in machining processes using hybrid learning algorithm and engineering knowledge. *Journal of Intelligent Manufacturing*, *23*(5), 1833–1847.

Ellenbogen, R. (2006). Cutting down on false alarms. *OnBoard Technology*.

Escobar, C. A., & Morales-Menendez, R. (2019). Process-monitoring-for-quality—a model selection criterion for support vector machine. *Procedia Manufacturing*, *34*, 1010–1017.

Fernández, A., Garcia, S., Herrera, F., & Chawla, N. V. (2018). Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. *Journal of artificial intelligence research*, *61*, 863–905.

Filipič, B., & Junkar, M. (2000). Using inductive machine learning to support decision making in machining processes. *Computers in Industry*, *43*(1), 31–41.

Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., & Herrera, F. (2011). A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *42*(4), 463–484.

Garcia-Ceja, E., Hugo, Å., Morin, B., Hansen, P. O., Martinsen, E., Lam, A. N., & Haugen, Ø. (2019). Towards the automation of a chemical sulphonation process with machine learning. In *2019 7th international conference on control, mechatronics and automation (iccma)* (pp. 352–357).

Garg, A., & Tai, K. (2013). Comparison of statistical and machine learning methods in modelling of data with multicollinearity. *International Journal of Modelling, Identification and Control*, *18*(4), 295–312.

Goldstein, M., & Uchida, S. (2016). A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PloS one*, *11*(4), e0152173.

Hart, S. (1989). Shapley value. In *Game theory* (pp. 210–216). Springer.

Khader, N., & Yoon, S. W. (2018). Online control of stencil printing parameters using reinforcement learning approach. *Procedia Manufacturing*, *17*, 94–101.

Khader, N., & Yoon, S. W. (2021). Adaptive optimal control of stencil printing process using reinforcement learning. *Robotics and Computer-Integrated Manufacturing*, *71*, 102132.

Kim, D., & Kang, S. (2019). Effect of irrelevant variables on faulty wafer detection in semiconductor manufacturing. *Energies*, *12*(13), 2530.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *science*, *220*(4598), 671–680.

Kleinbaum, D. G., Dietz, K., Gail, M., Klein, M., & Klein, M. (2002). *Logistic regression*. Springer.

Kohavi, R., Wolpert, D. H., et al. (1996). Bias plus variance decomposition for zero-one loss functions. In *Icml* (Vol. 96, pp. 275–83).

Krauß, J., Frye, M., Beck, G. T. D., & Schmitt, R. H. (2019). Selection and application of machine learning-algorithms in production quality. In *Machine learning for cyber physical systems* (pp. 46–57). Springer.

Kuo, F. Y., & Sloan, I. H. (2005). Lifting the curse of dimensionality. *Notices of the AMS*, *52*(11), 1320–1328.

Kusiak, A., & Kurasek, C. (2001). Data mining of printed-circuit board defects. *IEEE transactions on robotics and automation*, *17*(2), 191–196.

Lasi, H., Fettke, P., Kemper, H.-G., Feld, T., & Hoffmann, M. (2014). Industry 4.0. *Business & information systems engineering*, *6*(4), 239–242.

Leng, J., Ruan, G., Song, Y., Liu, Q., Fu, Y., Ding, K., & Chen, X. (2020). A loosely-coupled deep reinforcement learning approach for order acceptance decision of mass-individualized printed circuit board manufacturing in industry 4.0. *Journal of cleaner production*, *280*, 124405.

Lin, S.-C., & Su, C.-H. (2006). A visual inspection system for surface mounted devices on printed circuit board. In *2006 ieee conference on cybernetics and intelligent systems* (pp. 1–4).

Linn, R. J., & Lam, M.-M. (1998). Analysis of process errors and production yield for surface mounted printed circuit assembly. *Journal of Electronics Manufacturing*, *8*(01), 51–72.

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems* (pp. 4768–4777).

Lv, S., Kim, H., Zheng, B., & Jin, H. (2018). A review of data mining with big data towards its applications in the electronics industry. *Applied Sciences*, *8*(4), 582.

Mar, N. S. S., Yarlagadda, P., & Fookes, C. (2011). Design and development of automatic visual inspection system for pcb manufacturing. *Robotics and computer-integrated manufacturing*, *27*(5), 949–962.

Martinek, P., & Krammer, O. (2018). Optimising pin-in-paste technology using gradient boosted decision trees. *Soldering & Surface Mount Technology*.

Massey Jr, F. J. (1951). The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, *46*(253), 68–78.

Massi, M. C., Ieva, F., Gasperoni, F., & Paganoni, A. M. (2021). Feature selection for imbalanced data with deep sparse autoencoders ensemble. *arXiv preprint arXiv:2103.11678*.

Molnar, C. (2018). A guide for making black box models explainable. *Interpretable Machine Learning*, *1*.

Natschlager, T., Kossak, F., & Drobics, M. (2004). Extracting knowledge and computable models from data - needs, expectations, and experience. , *1*, 493-498 vol.1. doi: 10.1109/FUZZY.2004.1375780

Perrone, M., & Cooper, L. (1993). *When networks disagree: Ensemble method for neural networks. artificial neural networks for speech and vision, edited by mammone, rj.* Chapman & Hall, New York.

Polikar, R. (2006). Ensemble based systems in decision making. *IEEE Circuits and systems magazine*, *6*(3), 21–45.

Preuveneers, D., & Ilie-Zudor, E. (2017). The intelligent industry of the future: A survey on emerging trends, research challenges and opportunities in industry 4.0. *Journal of Ambient Intelligence and Smart Environments*, *9*(3), 287–298.

Richter, J., Streitferdt, D., & Rozova, E. (2017). On the development of intelligent optical inspections. In *2017 ieee 7th annual computing and communication workshop and conference (ccwc)* (pp. 1–6).

Rokach, L., & Hutter, D. (2012). Automatic discovery of the root causes for quality drift in high dimensionality manufacturing processes. *Journal of Intelligent Manufacturing*, *23*(5), 1915–1930.

Roscher, R., Bohn, B., Duarte, M. F., & Garcke, J. (2020). Explainable machine learning for scientific insights and discoveries. *Ieee Access*, *8*, 42200–42216.

Rumelhart, D. E., Durbin, R., Golden, R., & Chauvin, Y. (1995). Backpropagation: The basic theory. *Backpropagation: Theory, architectures and applications*, 1–34.

Sankhye, S., & Hu, G. (2020). Machine learning methods for quality prediction in production. *Logistics*, *4*(4), 35.

Schmitt, J., Bönig, J., Borggräfe, T., Beitinger, G., & Deuse, J. (2020). Predictive model-based quality inspection using machine learning and edge cloud computing. *Advanced Engineering Informatics*, *45*, 101101.

Seidel, R., Mayr, A., Schäfer, F., Kißkalt, D., & Franke, J. (2019). Towards a smart electronics production using machine learning techniques. In *2019 42nd international spring seminar on electronics technology (isse)* (pp. 1–6).

Sematech, N. (2006). Engineering statistics handbook. *NIST SEMATECH*.

Steffensen, J. F. (2006). *Interpolation*. Courier Corporation.

Suraj, Z. (2004). An introduction to rough set theory and its applications. *ICENCO, Cairo, Egypt*, *3*, 80.

Tharwat, A. (2020). Classification assessment methods. *Applied Computing and Informatics*.

Thielen, N., Werner, D., Schmidt, K., Seidel, R., Reinhardt, A., & Franke, J. (2020). A machine learning based approach to detect false calls in smt manufacturing. In *2020 43rd international spring seminar on electronics technology (isse)* (pp. 1–6).

Tomek, I., et al. (1976). Two modifications of cnn.

Tsai, C., Chiu, C.-C., & Chen, J.-S. (2005). A case-based reasoning system for pcb defect prediction. *Expert Systems with Applications*, *28*(4), 813–822.

Tsai, T. (2012). Development of a soldering quality classifier system using a hybrid data mining approach. *expert systems with applications*, *39*(5), 5727–5738.

Tseng, T.-L. B., Jothishankar, M., & Wu, T. T. (2004). Quality control problem in printed circuit board manufacturing—an extended rough set theory approach. *Journal of manufacturing systems*, *23*(1), 56–72.

Vapnik, V., & Learner, A. (1963). Pattern recognition using generalized portrait method. *Automation and remote control*, *24*, 774–780.

Vardi, I. (1991). *Computational recreations in mathematica*. Addison Wesley Longman Publishing Co., Inc.

Verron, S., Li, J., & Tiplica, T. (2010). Fault detection and isolation of faults in a multivariate process with bayesian network. *Journal of Process Control*, *20*(8), 902–911.

Wang, S., & Yao, X. (2009). Diversity analysis on imbalanced data sets by using ensemble models. In *2009 ieee symposium on computational intelligence and data mining* (pp. 324–331).

Wei, Z., Feng, Y., Hong, Z., Qu, R., & Tan, J. (2017). Product quality improvement method in manufacturing process based on kernel optimisation algorithm. *International Journal of Production Research*, *55*(19), 5597–5608.

Wirth, R., & Hipp, J. (2000). Crisp-dm: Towards a standard process model for data mining. In *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining* (Vol. 1).

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, *1*(1), 67–82.

Zhang, M., Yuan, Y., Wang, R., & Cheng, W. (2020). Recognition of mixture control chart patterns based on fusion feature reduction and fireworks algorithm-optimized msvm. *Pattern Analysis and Applications*, *23*(1), 15–26.

Zimek, A., & Schubert, E. (2017). Outlier detection. In *Encyclopedia of database systems.* Springer.

Appendix A Use case identification framework

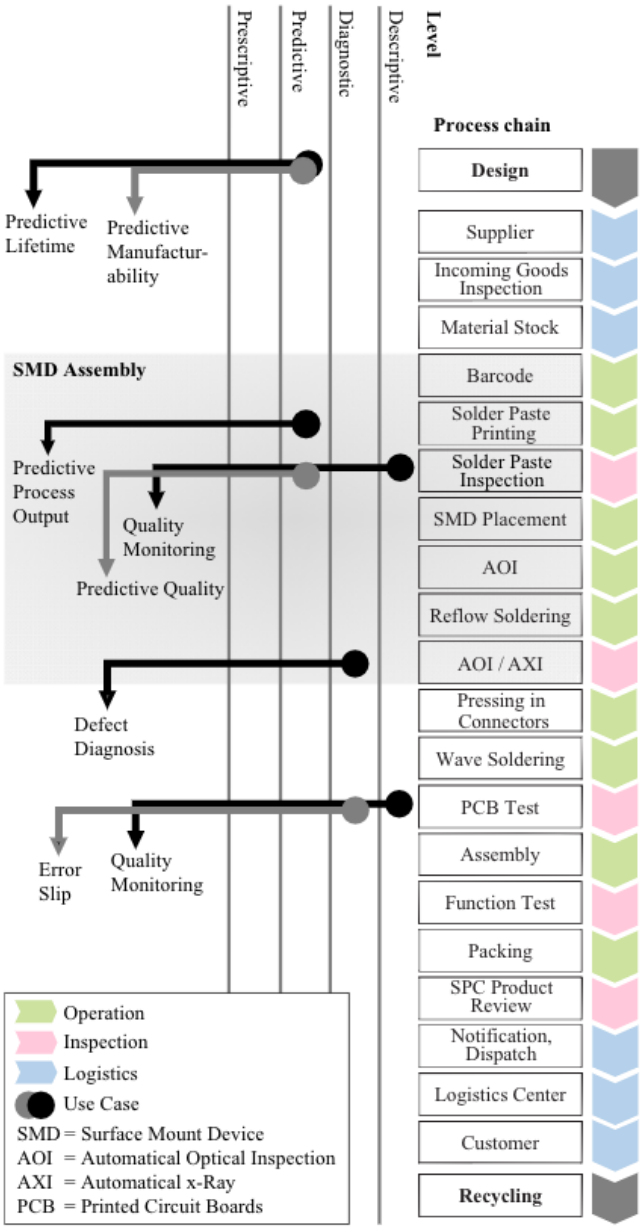


Figure 42: Use case identification matrix for SMD production from a process perspective (Seidel et al., 2019)

Appendix B Data Concepts

Table 28: Data concepts relevant for the surface-mount device quality

Feature name	**Description**	**Data level**	**Available**
Panel thickness	Thickness of the panel in millimeter	Product type	No
Soldermask finish	Material type that does not mix with tin	Product type	No
Pad surface	Components are attached on the copper pads	Product type	No
Stencil thickness	Thickness (height) in millimeters	Product type	No
Transfer efficiency	Information for each aperture of the stencil, indicating how easily paste can make its way through the hole	Product type	No
Squeegee width	Width of the squeegee (rubber blade to apply paste) in millimeters	Product type	No
Print speed	Squeegee speed in millimeters per second (screenprinting)	Batch	Yes
Print force	Squeegee force on the stencil in kilograms (screenprinting)	Batch	Yes
Number of strokes	Amount of times the squeegee moves over the stencil (screenprinting)	Batch	No
Snap off distance	Separation distance in millimeters after paste application between the printed circuit board and the stencil (screenprinting)	Batch	Yes
Snap off speed	Separation speed in meters per second after paste application between the printed circuit board and the stencil (screenprinting)	Batch	Yes
Stencil cleaning interval	Amount of products after which the stencil is cleaned during production (screenprinting)	Batch	No
Paste type	Type of paste used for the attachment of components (screenprinting)	Batch	Yes
Gluing	Binary feature whether the board requires gluing in addition to the paste (screenprinting)	Batch	No
Placement speed	How fast the components are placed from the supply form on the board (pick & place)	Batch	No

Pick up method	How components are picked up from the supply form, either with a vacuum grip or mechanical grip (pick & place)	Batch	No
Component rotation	Whether the placed component requires additional rotation when placed on the board (pick & place)	Batch	No
Bottom support	Force applied to the bottom of the board when placing the components (pick & place)	Batch	No
Zone temperature settings	Settings for the temperature in degrees Celcius for each of the heating and cooling zones (reflow)	Batch	Yes
Conveyor speed	Speed of the board when moving through the reflow heating zones (reflow)	Batch	Yes
Position in batch	Index indicating whether the product was the i-th product in a batch	Serial number	Yes
Time interval since last product	The time interval in seconds between the given product and the product produced before	Serial number	Yes
Printing temperature	Temperature in the printing machine in degrees Celcius	Serial number	Yes
Printing humidity	Humidity of the air in the printing machine in degrees Celcius	Serial number	Yes
Board calibration	Information how well the board is aligned relative to the Fuji machine (pick & place)	Serial number	No
Measured zone temperature	Temperatures in degrees Celcius for the zones may differ from the set temperatures	Serial number	Yes
Component package	Industry wide component category defining the shape, number of leads and type	Component	Yes
Moisture sensitivity level	Indicates whether a component is sensitive to moisture and requires special handling	Component	Yes
Component supply form	How the component is supplied to the machine as this can influence the picking stability	Component	Yes
Placement errors	Placement errors occurred during production for a component type on a specific panel	Component	Yes

Paste inspection features	API features measured for each location on a panel, providing information related to the paste quality	RefDes	Yes
Quality assessment	AOI call related to component placement for each board location on a panel	RefDes	Yes
Error type	Error types are provided for insufficient board locations	RefDes	Yes
Operator review	Manual check of the operator related to insufficient component placements as assessed by the AOI	RefDes	Yes

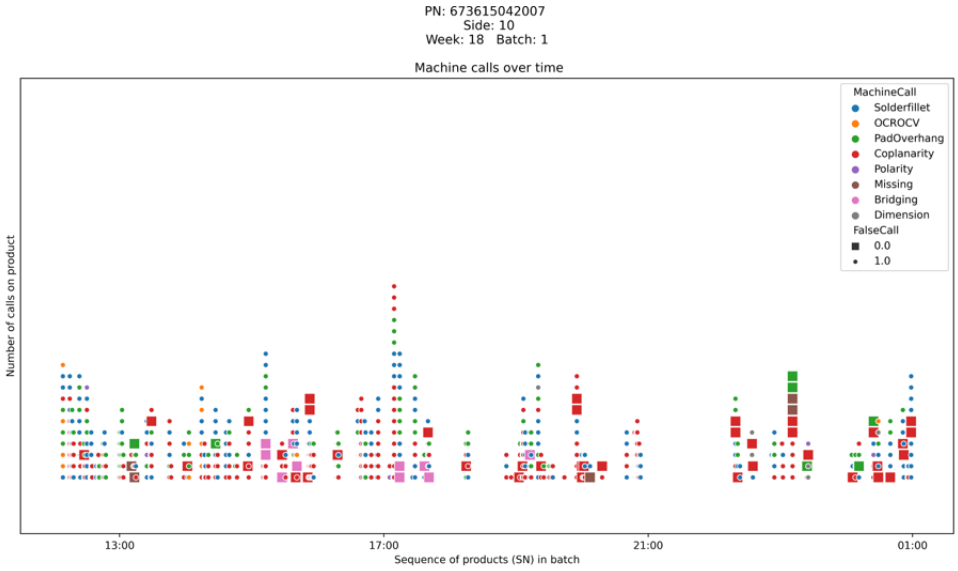
Appendix C Exploratory Data Analysis

C.1 Descriptive statistics

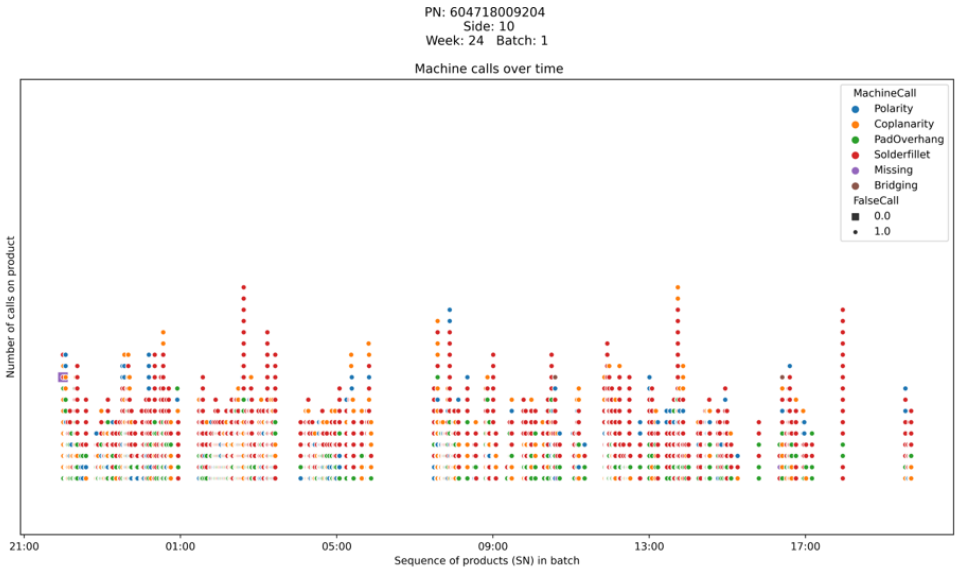
Table 29: False call percentages per product type

PN	Panels with false call	Total panels	Percentage with false call
6023-1600-0605	534	899	59.4%
6047-1800-9204	2453	2799	87.6%
6298-1300-3904	51	281	18.1%
6649-1000-2226	1049	4050	25.9%
6661-1900-0501	586	849	69%
6736-1504-2007	1521	1978	76.9%
6736-1602-9407	1224	2331	52.5%
6761-1200-5901	614	674	91.1%
6782-1700-1809	886	1392	63.6%

C.2 Error flags over time within a batch



(a) Example 1

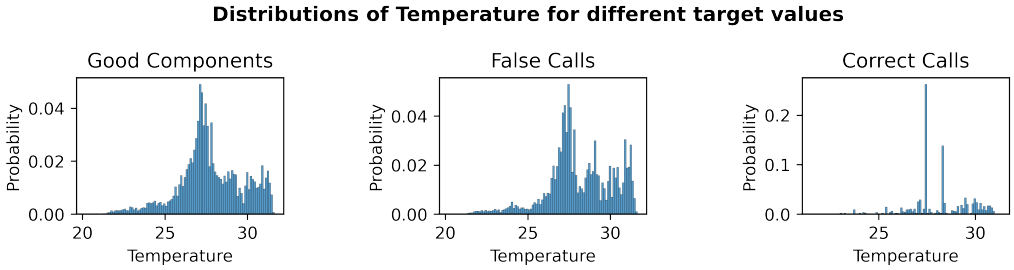


(b) Example 2

Figure 43: Error calls over time in example batches

C.3 Process variables distributions per target category

C.3.1 Screenprinting: machine environment



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	28.096	1.634	21.509	27.459	27.969	29.439	31.289
FALSE CALL	36347.0	28.13	1.976	20.209	27.069	27.819	29.669	31.629
GOOD	7388342.0	27.734	1.979	20.209	26.719	27.499	29.089	31.629

Welch's Test:
GOOD & FALSE CALL
t-value = -38.121
p-value = 0.0

2 sample K-S test:
Statistic: 0.099, p-value: 0.0

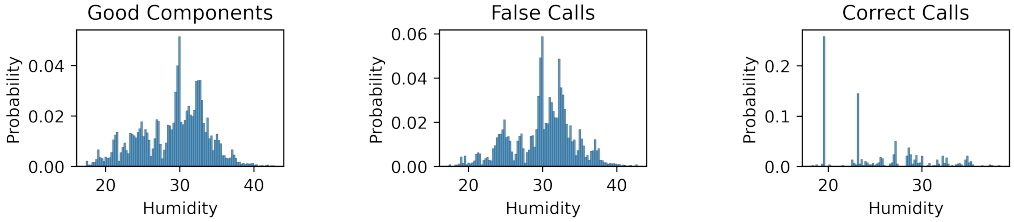
Welch's Test:
GOOD & CORRECT CALL
t-value = -6.064
p-value = 0.0

2 sample K-S test:
Statistic: 0.265, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = 0.565
p-value = 0.572

2 sample K-S test:
Statistic: 0.168, p-value: 0.0

Figure 44: Screenprinting machine temperature



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	25.526	5.085	18.242	19.584	24.486	29.209	38.321
FALSE CALL	36347.0	30.062	4.256	17.324	27.208	30.468	32.696	42.766
GOOD	7388342.0	29.21	4.554	17.324	25.645	29.864	32.477	42.766

Welch's Test:
GOOD & FALSE CALL
t-value = -38.028
p-value = 0.0

2 sample K-S test:
Statistic: 0.088, p-value: 0.0

Welch's Test:
GOOD & CORRECT CALL
t-value = 19.839
p-value = 0.0

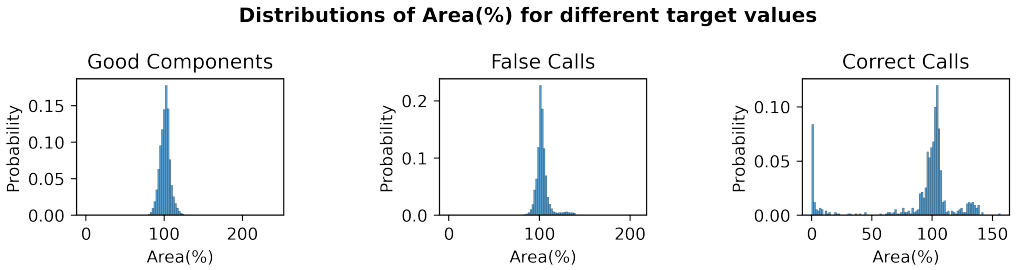
2 sample K-S test:
Statistic: 0.352, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = 24.25
p-value = 0.0

2 sample K-S test:
Statistic: 0.436, p-value: 0.0

Figure 45: Screenprinting machine humidity

C.3.2 Screenprinting: paste features



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	89.528	36.205	0.0	93.522	101.249	104.963	156.453
FALSE CALL	36347.0	103.251	9.978	0.0	99.582	101.92	105.052	207.981
GOOD	7388342.0	101.227	7.359	0.0	96.86	101.336	104.806	240.976

Welch's Test:
GOOD & FALSE CALL
t-value = -38.621
p-value = 0.0

2 sample K-S test:
Statistic: 0.148, p-value: 0.0

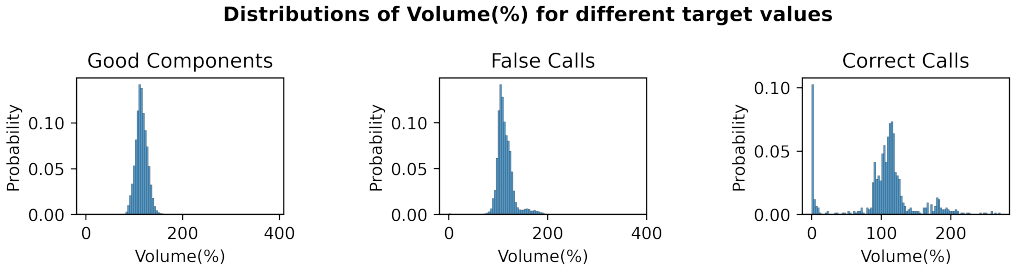
Welch's Test:
GOOD & CORRECT CALL
t-value = 8.85
p-value = 0.0

2 sample K-S test:
Statistic: 0.18, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = 10.373
p-value = 0.0

2 sample K-S test:
Statistic: 0.217, p-value: 0.0

Figure 46: Screenprinting area (%)



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	103.575	49.985	0.0	93.284	109.731	119.895	270.726
FALSE CALL	36347.0	113.91	18.637	0.0	103.311	110.348	120.757	381.142
GOOD	7388342.0	114.796	12.388	0.0	106.889	114.044	122.42	389.534

Welch's Test:
GOOD & FALSE CALL
t-value = 9.057
p-value = 0.0

2 sample K-S test:
Statistic: 0.141, p-value: 0.0

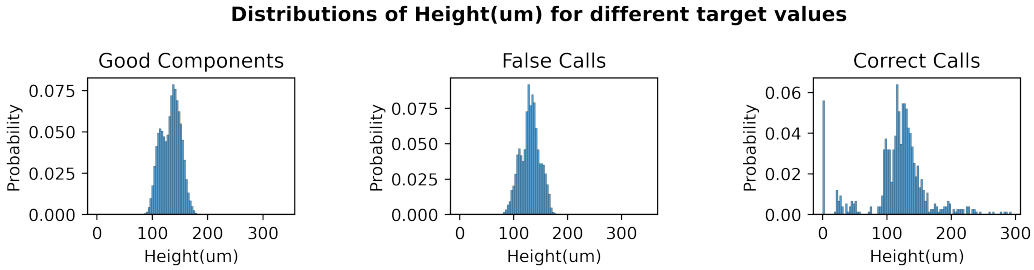
Welch's Test:
GOOD & CORRECT CALL
t-value = 6.148
p-value = 0.0

2 sample K-S test:
Statistic: 0.231, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = 5.654
p-value = 0.0

2 sample K-S test:
Statistic: 0.213, p-value: 0.0

Figure 47: Screenprinting volume (%)



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	117.321	47.312	0.0	102.795	122.155	137.536	293.299
FALSE CALL	36347.0	130.461	19.376	0.0	118.019	130.753	142.011	349.52
GOOD	7388342.0	133.044	17.755	0.0	118.608	134.853	146.238	340.36

Welch's Test:
GOOD & FALSE CALL
t-value = 25.369
p-value = 0.0

2 sample K-S test:
Statistic: 0.1, p-value: 0.0

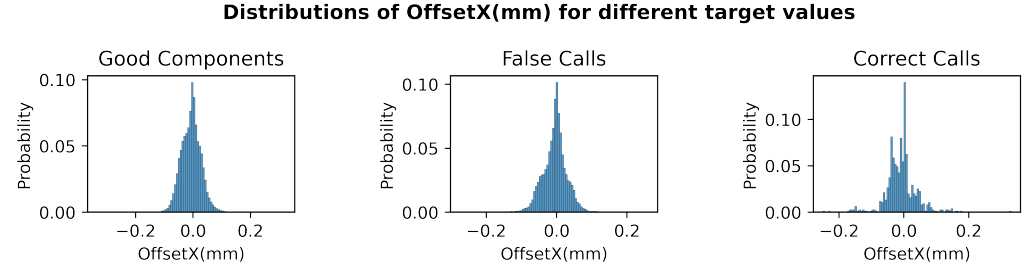
Welch's Test:
GOOD & CORRECT CALL
t-value = 9.101
p-value = 0.0

2 sample K-S test:
Statistic: 0.245, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = 7.592
p-value = 0.0

2 sample K-S test:
Statistic: 0.207, p-value: 0.0

Figure 48: Screenprinting height (um)



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	-0.009	0.048	-0.25	-0.034	-0.008	0.009	0.331
FALSE CALL	36347.0	-0.006	0.037	-0.275	-0.024	-0.003	0.013	0.26
GOOD	7388342.0	-0.007	0.035	-0.335	-0.03	-0.006	0.014	0.322

Welch's Test:
GOOD & FALSE CALL
t-value = -7.531
p-value = 0.0

2 sample K-S test:
Statistic: 0.06, p-value: 0.0

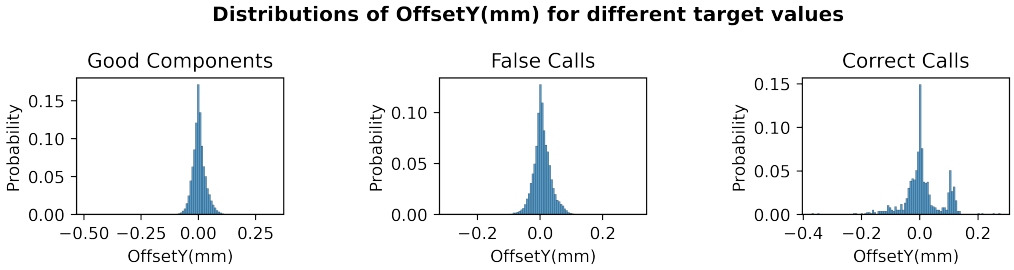
Welch's Test:
GOOD & CORRECT CALL
t-value = 0.938
p-value = 0.348

2 sample K-S test:
Statistic: 0.067, p-value: 0.002

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = 1.75
p-value = 0.08

2 sample K-S test:
Statistic: 0.098, p-value: 0.0

Figure 49: Screenprinting offset X



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	0.007	0.068	-0.371	-0.022	0.0	0.03	0.276
FALSE CALL	36347.0	0.007	0.031	-0.292	-0.009	0.004	0.022	0.31
GOOD	7388342.0	0.003	0.029	-0.491	-0.014	0.001	0.018	0.332

Welch's Test:
GOOD & FALSE CALL
t-value = -22.486
p-value = 0.0

2 sample K-S test:
Statistic: 0.071, p-value: 0.0

Welch's Test:
GOOD & CORRECT CALL
t-value = -1.712
p-value = 0.087

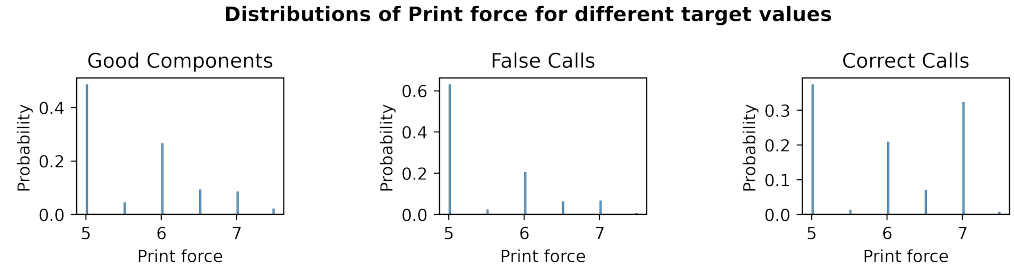
2 sample K-S test:
Statistic: 0.171, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = -0.247
p-value = 0.805

2 sample K-S test:
Statistic: 0.168, p-value: 0.0

Figure 50: Screenprinting offset Y

C.3.3 Screenprinting: process parameters



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	5.99	0.859	5.0	5.0	6.0	7.0	7.5
FALSE CALL	36347.0	5.467	0.673	5.0	5.0	5.0	6.0	7.5
GOOD	7388342.0	5.656	0.734	5.0	5.0	5.5	6.0	7.5

Welch's Test:
GOOD & FALSE CALL
t-value = 53.502
p-value = 0.0

2 sample K-S test:
Statistic: 0.144, p-value: 0.0

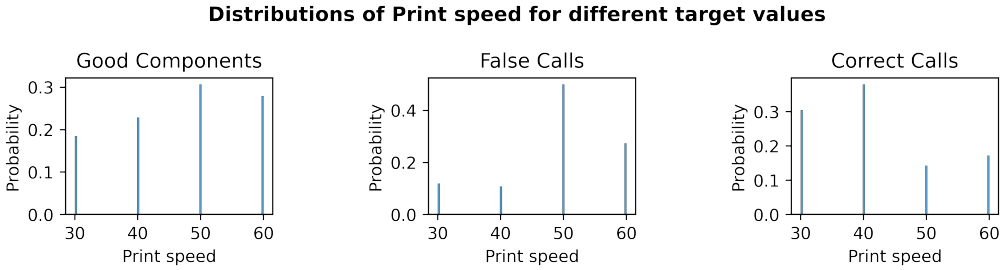
Welch's Test:
GOOD & CORRECT CALL
t-value = -10.637
p-value = 0.0

2 sample K-S test:
Statistic: 0.224, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = -16.569
p-value = 0.0

2 sample K-S test:
Statistic: 0.268, p-value: 0.0

Figure 51: Screenprinting print force



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	41.813	10.51	30.0	30.0	40.0	50.0	60.0
FALSE CALL	36347.0	49.287	9.223	30.0	50.0	50.0	60.0	60.0
GOOD	7388342.0	46.799	10.707	30.0	40.0	50.0	60.0	60.0

Welch's Test:
GOOD & FALSE CALL
t-value = -51.248
p-value = 0.0

2 sample K-S test:
Statistic: 0.188, p-value: 0.0

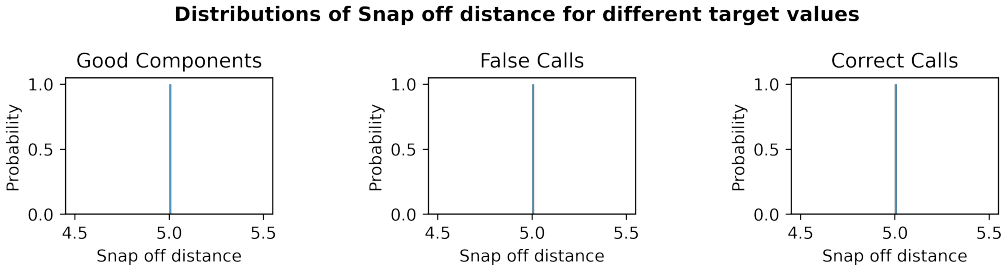
Welch's Test:
GOOD & CORRECT CALL
t-value = 12.991
p-value = 0.0

2 sample K-S test:
Statistic: 0.271, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = 19.32
p-value = 0.0

2 sample K-S test:
Statistic: 0.459, p-value: 0.0

Figure 52: Screenprinting print speed



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	5.0	0.0	5.0	5.0	5.0	5.0	5.0
FALSE CALL	36347.0	5.0	0.0	5.0	5.0	5.0	5.0	5.0
GOOD	7388342.0	5.0	0.0	5.0	5.0	5.0	5.0	5.0

Welch's Test:
GOOD & FALSE CALL
t-value = nan
p-value = nan

2 sample K-S test:
Statistic: 0.0, p-value: 1.0

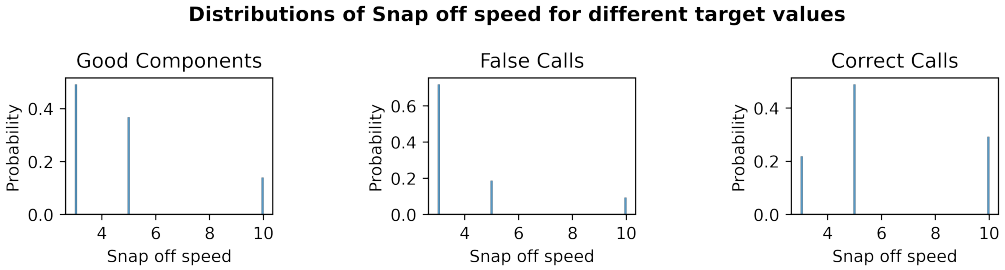
Welch's Test:
GOOD & CORRECT CALL
t-value = nan
p-value = nan

2 sample K-S test:
Statistic: 0.0, p-value: 1.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = nan
p-value = nan

2 sample K-S test:
Statistic: 0.0, p-value: 1.0

Figure 53: Screenprinting snap off distance



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	6.023	2.672	3.0	5.0	5.0	10.0	10.0
FALSE CALL	36347.0	4.032	2.071	3.0	3.0	3.0	5.0	10.0
GOOD	7388342.0	4.715	2.32	3.0	3.0	5.0	5.0	10.0

Welch's Test:
GOOD & FALSE CALL
t-value = 62.641
p-value = 0.0

2 sample K-S test:
Statistic: 0.227, p-value: 0.0

Welch's Test:
GOOD & CORRECT CALL
t-value = -13.406
p-value = 0.0

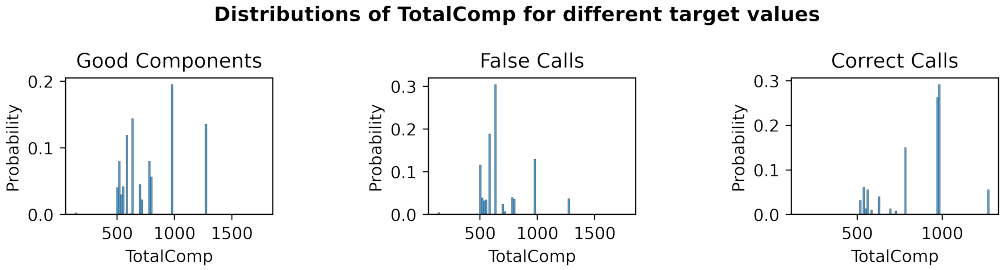
2 sample K-S test:
Statistic: 0.273, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = -20.278
p-value = 0.0

2 sample K-S test:
Statistic: 0.5, p-value: 0.0

Figure 54: Screenprinting snap off speed

C.3.4 Pick & place: components on panel



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	866.763	197.336	168.0	778.0	972.0	980.0	1276.0
FALSE CALL	36347.0	678.462	191.331	132.0	589.0	630.0	722.0	1776.0
GOOD	7388342.0	786.185	253.117	132.0	589.0	692.0	980.0	1776.0

Welch's Test:
GOOD & FALSE CALL
t-value = 106.88
p-value = 0.0

2 sample K-S test:
Statistic: 0.26, p-value: 0.0

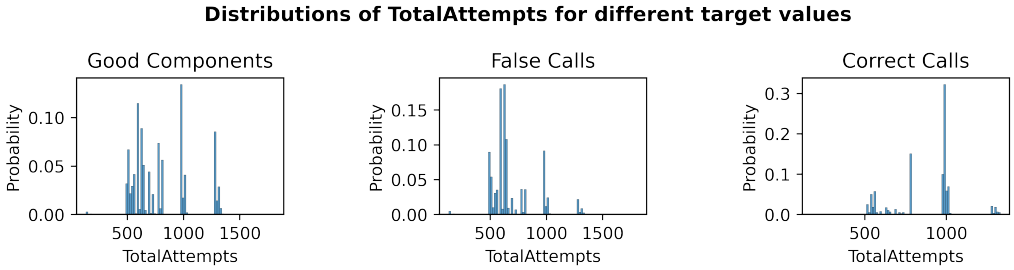
Welch's Test:
GOOD & CORRECT CALL
t-value = -11.182
p-value = 0.0

2 sample K-S test:
Statistic: 0.293, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = -25.882
p-value = 0.0

2 sample K-S test:
Statistic: 0.518, p-value: 0.0

Figure 55: Pick and place total components



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	876.895	202.208	174.0	778.0	984.0	985.0	1330.0
FALSE CALL	36347.0	684.069	194.126	132.0	590.0	631.0	724.0	1807.0
GOOD	7388342.0	792.561	257.34	132.0	591.0	695.0	981.0	1807.0

Welch's Test:
GOOD & FALSE CALL
t-value = 106.09
p-value = 0.0

2 sample K-S test:
Statistic: 0.26, p-value: 0.0

Welch's Test:
GOOD & CORRECT CALL
t-value = -11.421
p-value = 0.0

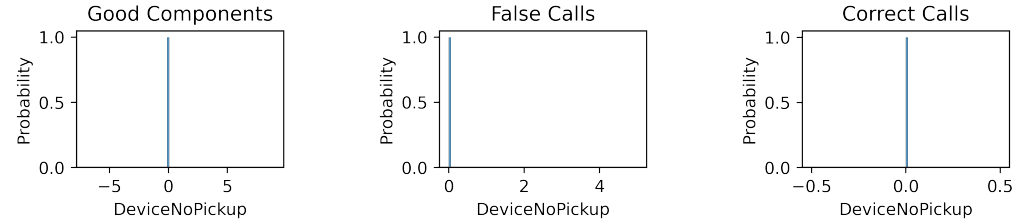
2 sample K-S test:
Statistic: 0.321, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = -25.871
p-value = 0.0

2 sample K-S test:
Statistic: 0.522, p-value: 0.0

Figure 56: Pick and place total attempts

C.3.5 Pick & place: error messages per component type



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
FALSE CALL	36347.0	0.005	0.105	0.0	0.0	0.0	0.0	5.0
GOOD	7388342.0	0.004	0.116	-7.0	0.0	0.0	0.0	9.0

Welch's Test:
GOOD & FALSE CALL
t-value = -1.293
p-value = 0.196

2 sample K-S test:
Statistic: 0.001, p-value: 1.0

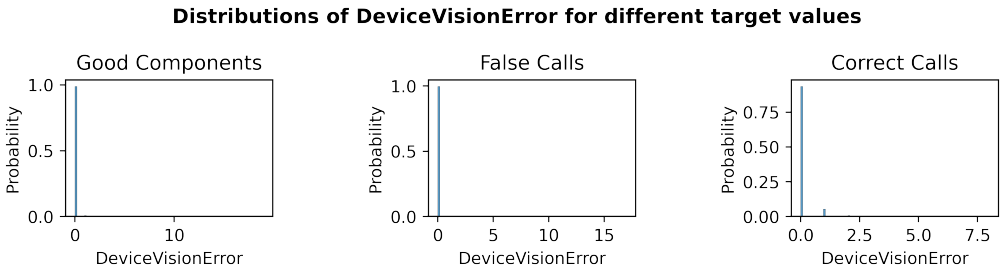
Welch's Test:
GOOD & CORRECT CALL
t-value = 91.496
p-value = 0.0

2 sample K-S test:
Statistic: 0.002, p-value: 1.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = 8.429
p-value = 0.0

2 sample K-S test:
Statistic: 0.002, p-value: 1.0

Figure 57: Pick and place no pick up error



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	0.089	0.443	0.0	0.0	0.0	0.0	8.0
FALSE CALL	36347.0	0.006	0.178	0.0	0.0	0.0	0.0	17.0
GOOD	7388342.0	0.019	0.263	0.0	0.0	0.0	0.0	19.0

Welch's Test:
GOOD & FALSE CALL
t-value = 13.764
p-value = 0.0

2 sample K-S test:
Statistic: 0.008, p-value: 0.02

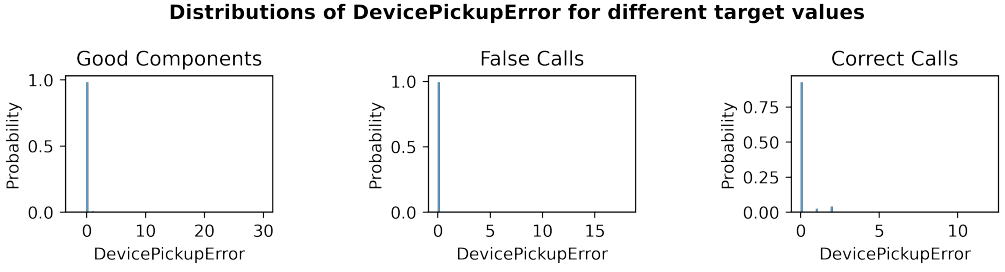
Welch's Test:
GOOD & CORRECT CALL
t-value = -4.333
p-value = 0.0

2 sample K-S test:
Statistic: 0.054, p-value: 0.024

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = -5.122
p-value = 0.0

2 sample K-S test:
Statistic: 0.062, p-value: 0.007

Figure 58: Pick and place vision error



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	0.164	0.798	0.0	0.0	0.0	0.0	12.0
FALSE CALL	36347.0	0.017	0.293	0.0	0.0	0.0	0.0	18.0
GOOD	7388342.0	0.045	0.443	-2.0	0.0	0.0	0.0	30.0

Welch's Test:
GOOD & FALSE CALL
t-value = 17.847
p-value = 0.0

2 sample K-S test:
Statistic: 0.011, p-value: 0.0

Welch's Test:
GOOD & CORRECT CALL
t-value = -4.086
p-value = 0.0

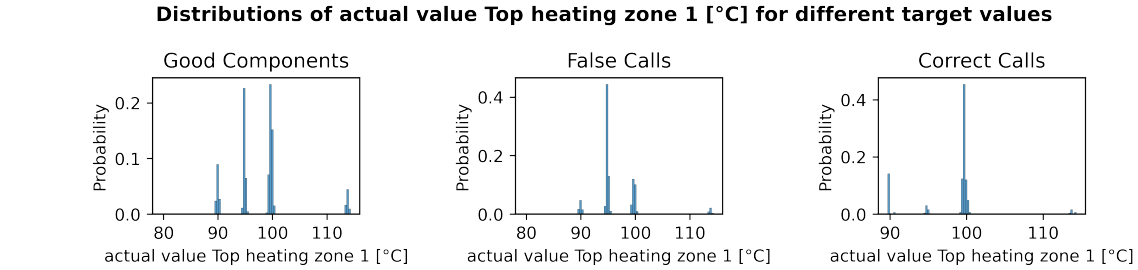
2 sample K-S test:
Statistic: 0.056, p-value: 0.018

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = -5.025
p-value = 0.0

2 sample K-S test:
Statistic: 0.067, p-value: 0.003

Figure 59: Pick and place pick up error

C.3.6 Reflow



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	748.0	98.37	4.504	89.7	99.4	99.7	99.7	114.3
FALSE CALL	36161.0	96.385	4.325	79.7	94.7	94.9	99.5	114.3
GOOD	7310964.0	97.789	5.648	79.7	94.8	99.4	99.7	114.3

Welch's Test:
GOOD & FALSE CALL
t-value = nan
p-value = nan

2 sample K-S test:
Statistic: 0.247, p-value: 0.0

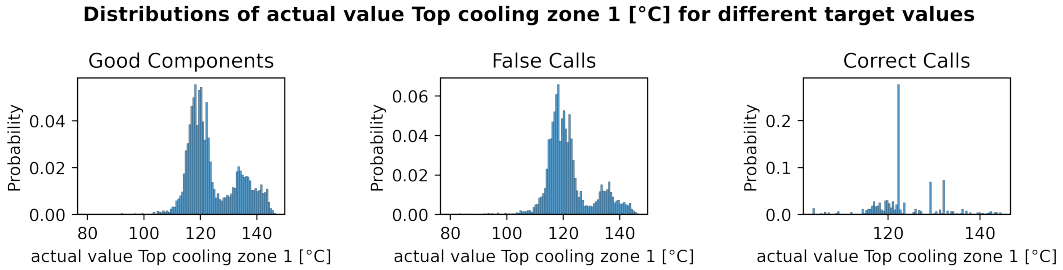
Welch's Test:
GOOD & CORRECT CALL
t-value = nan
p-value = nan

2 sample K-S test:
Statistic: 0.252, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = nan
p-value = nan

2 sample K-S test:
Statistic: 0.49, p-value: 0.0

Figure 60: Reflow heating zone 1



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	123.521	7.376	103.6	119.5	122.4	129.3	144.6
FALSE CALL	36212.0	121.915	8.421	79.8	116.9	120.0	124.2	146.6
GOOD	7328055.0	124.116	9.168	79.8	117.7	121.1	132.2	146.6

Welch's Test:
GOOD & FALSE CALL
t-value = nan
p-value = nan

2 sample K-S test:
Statistic: 0.113, p-value: 0.0

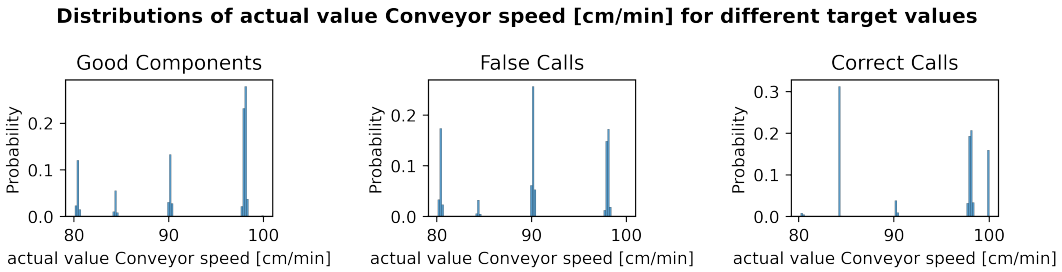
Welch's Test:
GOOD & CORRECT CALL
t-value = nan
p-value = nan

2 sample K-S test:
Statistic: 0.17, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = nan
p-value = nan

2 sample K-S test:
Statistic: 0.257, p-value: 0.0

Figure 61: Reflow cooling zone 1



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	627.0	93.44	6.768	80.2	84.3	98.0	98.2	100.0
FALSE CALL	29673.0	90.435	6.767	80.1	84.3	90.2	98.0	100.0
GOOD	5530371.0	92.697	6.885	80.1	90.0	97.9	98.1	100.0

Welch's Test:
GOOD & FALSE CALL
t-value = nan
p-value = nan

2 sample K-S test:
Statistic: 0.209, p-value: 0.0

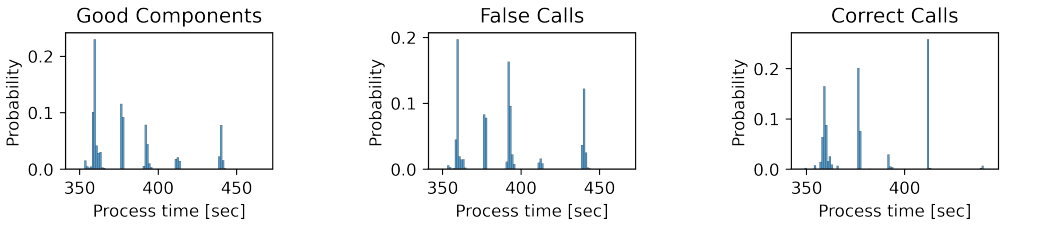
Welch's Test:
GOOD & CORRECT CALL
t-value = nan
p-value = nan

2 sample K-S test:
Statistic: 0.126, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = nan
p-value = nan

2 sample K-S test:
Statistic: 0.214, p-value: 0.0

Figure 62: Reflow conveyor speed



	count	mean	std	min	25%	50%	75%	max
CORRECT CALL	750.0	380.153	22.378	347.0	359.0	376.0	412.0	443.0
FALSE CALL	36347.0	389.509	28.519	347.0	360.0	392.0	394.0	467.0
GOOD	7388342.0	380.419	26.83	347.0	359.0	376.0	392.0	467.0

Welch's Test:
GOOD & FALSE CALL
t-value = -60.63
p-value = 0.0

2 sample K-S test:
Statistic: 0.209, p-value: 0.0

Welch's Test:
GOOD & CORRECT CALL
t-value = 0.325
p-value = 0.745

2 sample K-S test:
Statistic: 0.122, p-value: 0.0

Welch's Test:
FALSE CALL & CORRECT CALL
t-value = 11.262
p-value = 0.0

2 sample K-S test:
Statistic: 0.217, p-value: 0.0

Figure 63: Reflow process time

C.3.7 Component characteristics

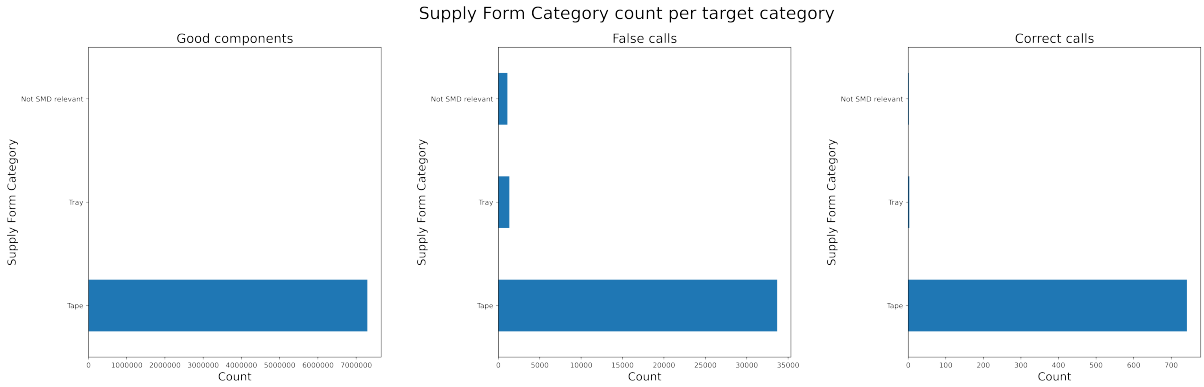


Figure 64: Component supply form

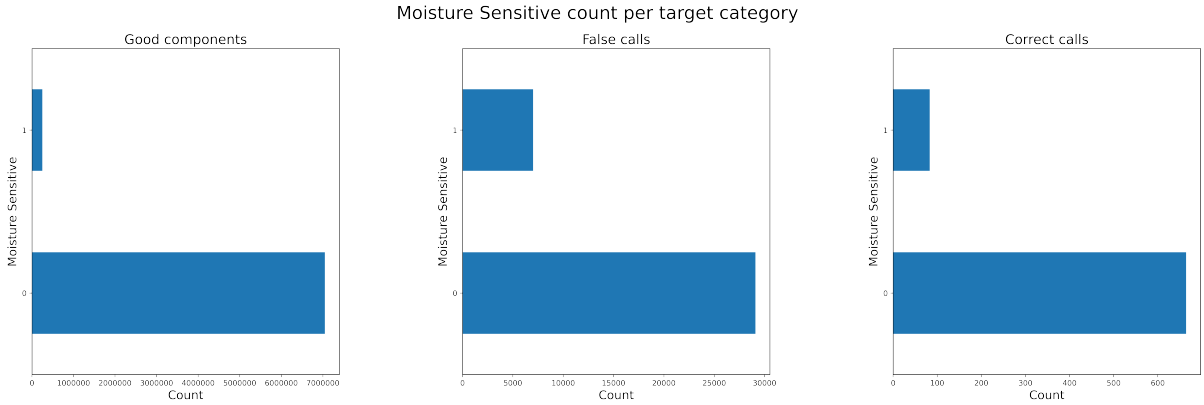


Figure 65: Component moisture sensitivity

Appendix D Modeling

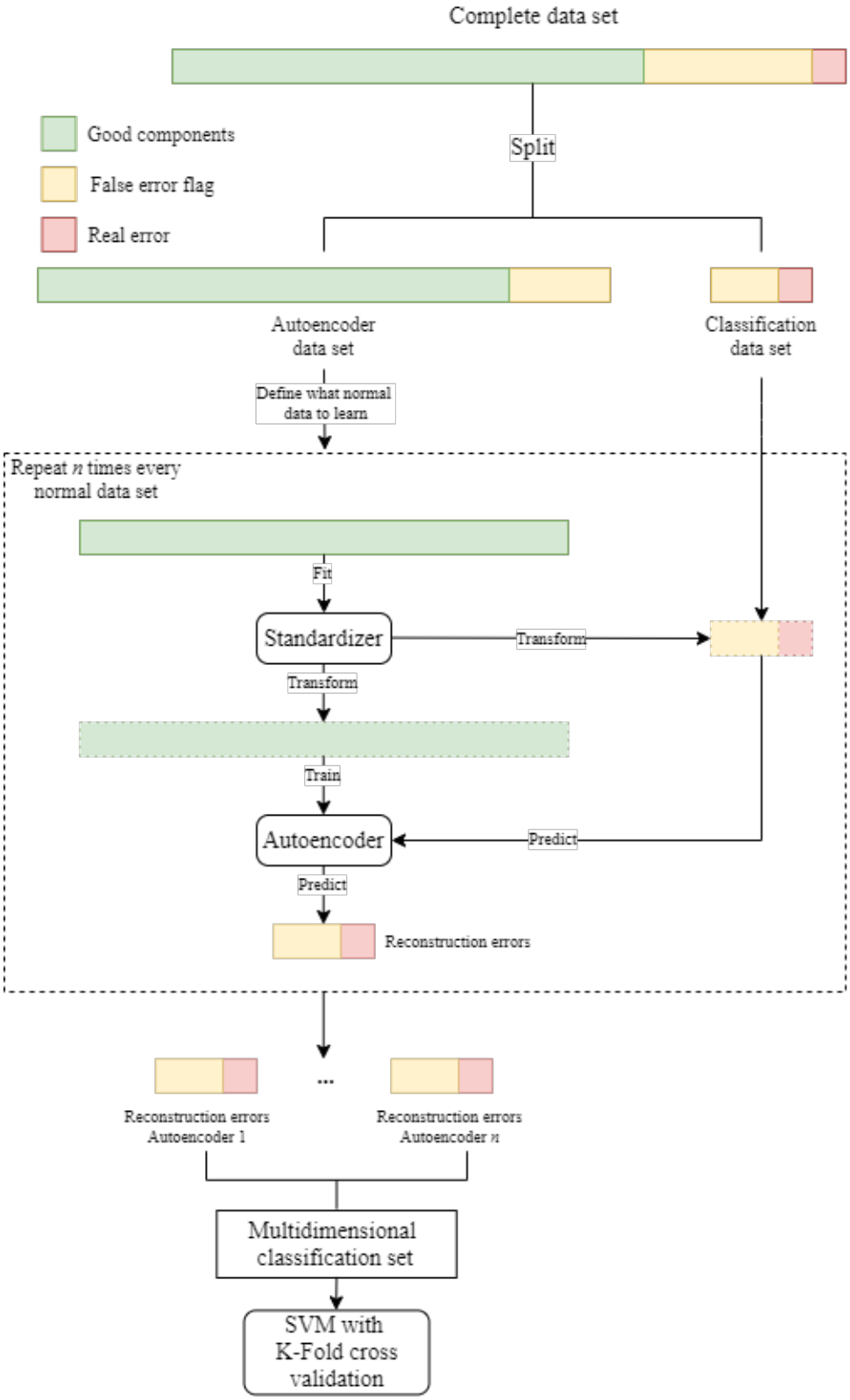


Figure 66: Conceptual representation of hybrid machine learning model