UNIVERSITY OF TWENTE.

Improving the defect detection performance of Incoming Quality Control

Master Thesis

Joep de Vent

28-Aug-24

Industrial Engineering & Management

Title:
Improving the defect detection performance of Incoming Quality Control

Course:
Master Thesis

Author:
J.M.H. (Joep) de Vent
University of Twente
Faculty of Behavioural, Management and Social Sciences
Master Industrial Engineering & Management
Specialisation in Production and Logistics Management

Supervisors:
Dr. E. (Engin) Topan
First supervisor UT
Faculty of behavioural, Management and Social Sciences
University of Twente
Dr. H. (Hao) Chen
Second supervisor UT
Faculty of behavioural, Management and Social Sciences
University of Twente
Maarten van Tintelen
First supervisor VDL ETG Almelo
Supply Chain Engineer Digitalisation
VDL ETG Almelo
Joe Mullins
Second supervisor VDL ETG Almelo
Supplier Quality Manager
VDL ETG Almelo

Organisation details:
University of Twente
Drienerlolaan 5
7522 NB Enschede
The Netherlands
VDL ETG Almelo
Bornsestraat 345
7601 PB Almelo
The Netherlands

Preface

Dear Reader,

This document is the result of my thesis study, conducted as the final assignment for my Master in Industrial Engineering and Management at the University of Twente. The study aims to improve the Incoming Quality Control process at VDL ETG Almelo, with the goal of enhancing the detection rate by refining the selection process. I am glad that this study gave me the chance to dive into machine learning techniques, a new area for me. It proved to be a very educational and relevant topic for this time.

I want to thank VDL ETG Almelo for giving me the chance to work on this project and for their organisational-wide support throughout the study. Special thanks to my supervisors, Maarten and Joe. Your enthusiasm for this project and its applications was a great motivation to me. I am especially grateful for the freedom I was given in conducting the research. Additionally, I want to thank my colleagues from the supply chain department for making my days at the office a lot more enjoyable.

A big thank you to my first supervisor at the University of Twente, Dr. Engin Topan. Your guidance was essential in completing this research. I would also like to thank my second supervisor Dr. Hao Chen. Your suggestions and feedback helped me improve the predictive models significantly.

This thesis marks the end of my student life. I want to take this final opportunity to thank my family and friends for their unwavering support throughout this period. Your encouragement and belief in me have been invaluable. I am deeply grateful for your constant presence and support during this journey.

I hope you find this study informative and useful.

Enjoy the read!

Joep de Vent
Hengelo, August 2024

Management Summary

This thesis study is conducted at VDL ETGA, which is a tier-one design and manufacturing supplier in the high-tech industry. The activities within VDL ETG Almelo (VDL ETGA) include the assembly of modules for semiconductor machines. A significant part of the items needed for assembly are purchased from suppliers. The quality of these items is not always sufficient. 740 quality complaints were registered on purchased items in 2023. Currently, only 9% of supplier complaints are detected at incoming goods. All other defective items are transported, cleaned, repacked, and stored, before they go to the cleanrooms. 82% of the supplier complaints are detected at assembly in the airlock or cleanroom. The total costs of a supplier complaint depend on where in the supply chain it is detected. It is estimated that the additional costs of a complaint found at assembly are €3,000 per complaint.

Two employees of the Incoming Quality Control (IQC) department perform the inspections on incoming goods. VDL ETGA believes that the current process is not able to effectively select which items to inspect. Additionally, IQC often lacks clear direction on what specific aspects to inspect. The objective of this thesis study is to increase the IQC detection percentage by improving the selection and instructions for IQC inspections. The research question is as follows:

How can defects among incoming items be predicted, and how should VDL ETG Almelo use this information in their inspection policy to improve the IQC detection performance?

A predictive model is developed to predict which purchase order lines contain defective items. The prediction of defects among incoming items is modelled as a binary classification problem with supervised learning. The tested classifiers are Logistic Regression (LR), Decision Tree (DT), Support Vector Machine (SVM), XGBoost, and CatBoost. Backward stepwise feature selection is applied to find the most important predictive features. The final model includes seven features. The three key features are the item code, the item price, and the supplier.

The model is trained on 130,119 purchase order lines from January 2021 to January 2024. The evaluation dataset contains a total of 13,285 deliveries between February and April 2024, of which 144 include a complaint. Table 1 shows the performance metrics with a classification threshold of 80%, meaning that the model classifies a purchase order line as defective if the probability of being defective is greater than 80%. Table 1 demonstrates that the CatBoost model achieves the highest F1 score, which is the most crucial metric as it reflects the balance between sensitivity and precision. The sensitivity is the total percentage of the 144 complaints correctly classified. The precision is the percentage of true complaints among all purchase order lines classified as positive. SVM cannot be included in this table, because it uses a different method to make predictions. Nonetheless, the AUC of 0.76 shows that SVM does not perform well for this project.

Table 1: Performance metrics of the models for a classification threshold of 80%

Classifier	F1 score	Sensitivity	Precision	Accuracy	AUC
CatBoost	**14.9%**	**27.8%**	10.2%	96.1%	**0.84**
Logistic Regression	12.8%	**31.9%**	8.0%	95.1%	0.81
XGBoost	11.6%	13.2%	**10.3%**	**97.8%**	0.82
Decision Tree	8.9%	16.0%	6.2%	96.4%	0.79

On average, IQC performed 3 inspections per day between February and April 2024. The selection was mostly based on manually flagging items for inspection that were defective in recent previous deliveries. The realised performance was a sensitivity of 9.7%, a precision of 7.8% and an F1 score of 8.7%. The classification threshold of the CatBoost model is tuned so that the average of daily

inspections is equal to IQC. This results in a sensitivity of 18.8%, a precision of 14.9%, and an F1 score of 16.6%. These metrics show that our prediction model outperforms the current inspection policy.

After discussions with the engineers at VDL, we take €108 as costs for one inspection, and €3,000 of costs for each missed defect at IQC. The Total Cost of Quality (TCQ) is the sum of the inspection costs and costs of missed defects. The IQC performance from the evaluation set is extrapolated to calculate expected quality costs for 2024 if the old inspection policy was used, which are equal to €1,644,000. If we continue with 3 inspections per day with the inspection policy from the model, the predicted annual costs of quality would be €1,482,000. This results in expected savings of €162,000.

Performing 3 inspections per day is not necessarily optimal. We further investigate the total costs of quality to find the optimal number of daily inspections. Figure 1 shows the annual TCQ for IQC per number of daily inspections. The minimum annual costs, while using the model, are €1,367,000 at 16 inspections per day. Thus, the expected savings for the optimal policy are equal to €277,000 per year.

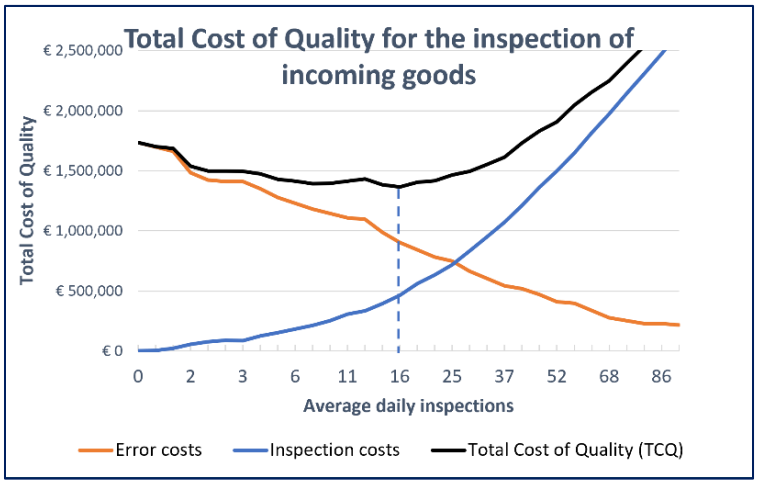


Figure 1: Total cost of Quality for the inspection of incoming goods (inspection cost: €108 and error cost: €3,000)

At an average of 1.5 hours per inspection, 16 inspections would cost 24 hours each day. To achieve this number of inspections, the capacity of IQC would have to be expanded from 2 to 4 employees. Therefore, the best strategy for direct implementation is to let the model select purchase order lines for inspection until the current IQC capacity is filled.

The new inspection model has been implemented at VDL ETGA. The implemented model selects purchase order lines and accompanies them with inspection instructions. The instructions are based on qualitative information from previous complaints for each unique item. This information is used to inform IQC what type of defect they should look for during the inspection. Acceptable Quality Limit tables are used to determine the inspection sample size within the selected purchase order lines.

As of writing this in August 2024, 24 selected purchase order lines have been delivered and inspected. 6 of the inspections resulted in the detection of defective items. The precision of the inspections is 25%. This precision equates to a success rate of 1 in 4 inspections. The calculated real-life performance in the evaluation set was a precision of 7.8%, so approximately a success rate of 1 in 12 inspections. The average daily inspections remained at 3 for this period.

The implemented prediction process is fully automated. A member of the supplier quality department manually flags the selected purchase order lines in the ERP system and adds the instructions from the model. The flagging of purchase order lines in the ERP system can be automated once the new ERP system is in use in 2025. The functional ownership of the model lies with the supplier quality manager, and the technical ownership lies with the supply chain digitalisation engineer.

Contents

List of Figures

List of Tables

Glossary

Term	Description
Baan IV	The name of the ERP system currently in use at VDL ETGA.
CAQ REM	The name of the quality complaint management software currently in use at VDL ETGA.
Defect	An item within a purchase order line that is not conforming to standard.
False Negative (FN)	Incorrect prediction of a non-defect.
False Positive (FP)	Incorrect prediction of a defect.
Feature	A column in the dataset that is used to make predictions
Incoming goods	The collection of all items purchased from suppliers and delivered to VDL ETGA.
Incoming Quality Control	The department at VDL ETGA responsible for inspecting incoming goods
Inspection	Activity performed on incoming goods to detect defective items before they are forwarded to the production
Purchase order line	A purchase order is created for every delivery. Every unique item within the purchase order is registered in a purchase order line.
Supplier complaint	The registration of a defect caused by a supplier.
True Negative (TN)	Correct prediction of a non-defect.
True Positive (TP)	Correct prediction of a defect.

List of acronyms

Acronym	Description
AQL	Acceptable Quality Level
AUC	Area Under the Curve
BAQT	Before Acceptance Quality Test
DT	Decision Trees
FAI	First Article Inspection
FN	False Negative
FP	False Positive
HIP	High Impact Part
IQC	Incoming Quality Control
kNN	k-Nearest Neighbours
LR	Logistic Regression
ML	Machine Learning
QE	Quality Engineer
RF	Random Forest
SQL	Structured Query Language
SVM	Support Vector Machines
TN	True Negative
TP	True Positive
VDL ETGA	Van der Leegte Enabling Technologies Group Almelo
WMS	Warehouse Management System

1 Introduction

This chapter introduces the research problem and the company where the research is performed. Section 1.1 gives background information about VDL relevant to this assignment. It is followed by an analysis of the problem in sections 1.2 through 1.4. Section 1.5 explains the goal and research questions of the project. The last section of this chapter is the plan of approach to solving the research problem.

1.1 Company background

The Van Der Leegte Groep (VDL Groep) was founded in 1953. The company started as a supplier of machined parts for Philips and DAF Trucks. Nowadays the company is active in 19 countries and has over 16.000 employees. The group consists of more than 100 companies, divided into five clusters: Hightech, Mobility, Energy, Infratech and Foodtech. Each company has its own specialism. The combined turnover of the VDL Groep was 6.35 billion euros in 2023 (VDL Groep, 2024).

VDL Enabling Technologies Group (VDL ETG), founded in 1900 as the Philips Machinefabrieken, has been part of the VDL Groep since 2006. It is part of the Hightech cluster and has locations in the Netherlands, Switzerland, Singapore, China, and the United States of America. VDL ETG is a tier-one design and manufacturing supplier. Its customers are original equipment manufacturers in the semiconductor, solar, medical, mechanisation, and analytical markets. The VDL ETG location in Almelo fulfils system integration for mechatronic subsystems and modules with 1,500 employees. This includes the production of parts and modules for semiconductor machines. The activities in Almelo cover design, parts production, assembly, and quality control. The in-house facilities for parts production consist of 5-axis milling, precision grinding, sheet-metal work and laser cutting machines. The mechanical and electrical assembly takes place in cleanrooms (VDL ETG, n.d.).

The activities within VDL ETG Almelo (VDL ETGA) are split into four main workflows, namely Systems-1, Systems-2, Projects, and Parts. Systems-1 and -2 are departments where systems are built in serial production. Systems-1 and -2 both cover several systems for two different customers. The end products are a module of the customers' machines. Currently, the products from Systems-1 account for the largest share of the workload. Temporary, low-volume and one-off products fall under the Projects workflow. The Parts workflow is for all manufactured parts in the machining and sheet-metal departments. The manufacturing departments are an internal supplier to the assembly.

This project takes place at VDL ETGA in the supply chain department. The primary responsibility of the supply chain department is the optimisation of logistics processes. The department has three sub departments. The first sub department is Logistics and Transportation. They manage the flow of materials and information arriving at, moving within, and leaving VDL ETGA. The second sub department is Master Planning. This sub department is responsible for the synchronization of sales forecasting and production planning. The last sub department is Business Engineering & Intelligence. They manage the ERP system Baan IV, the warehouse management system, and the underlying databases. They also perform data analysis and dashboarding. This assignment was initiated by the Digitalisation section within the supply chain department.

1.2 Action problem

VDL ETGA has reported an average of 124,000 defective or sub-quality items (henceforth referred to as 'defects') annually since 2021. 77% of the defective items belong to the Systems-1 workflow. These defects are caused either internally or by suppliers. Suppliers were identified as responsible for 58% of all defective items. A supplier is considered the cause of a defect if a purchased item arrives at VDL

ETGA while it is already defective. Examples of supplier quality defects are incorrect product dimensions, scratches on the surface, and deviations in the threaded connections.

All defects that are found are reported in a quality complaint. Complaints that are caused by suppliers are called supplier complaints. A considerable influence on the total cost of a supplier complaint is where in the supply chain it is detected. Currently, only 9% of supplier complaints are detected during the inspection of incoming goods. If defects are not found here, then the items are transported, cleaned for cleanroom use, repacked to prevent contamination, and stored, until they are unpacked again at the airlocks before the cleanroom assembly areas. All these activities are waste if the item proves to be defective during assembly. Figure 1.1 shows where the supplier complaints are detected. 82% of supplier defects are detected at assembly (this location includes the airlock). In a previous research project, all process steps related to handling quality defects, including the time spent in each step, were mapped out (Dieperink, 2023). From this research, in consultation with VDL ETGA, it is estimated that the additional costs of a complaint found at assembly, instead of during the incoming goods inspection, are €3,000 per complaint.

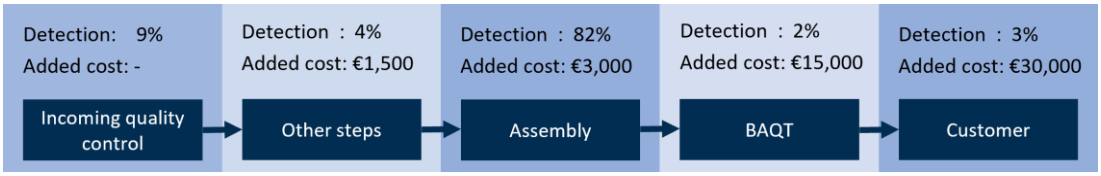


Figure 1.1: Order in process for most common detection locations of product defects

An action problem is a discrepancy between the norm and reality, as perceived by the problem owner (Heerkens & van Winden, 2021). Using this definition and the information above, we define the following action problem for this project:

"The costs of poor quality are increased because supplier defects are detected too late."

1.3 Problem context

The given problem is related to other problems that VDL ETG experiences. Defects in production cause a standstill of machines or assembly lines. Defective parts result in extra material handling and waste if replacement parts must be supplied. The number of items arriving at VDL ETGA is too high to inspect all of them. When an item is selected for inspection, it sometimes happens that existing supplier defects are not detected. Lastly, supplier defects only exist because a supplier shipped a defective item to VDL ETGA. A problem cluster can be used to indicate the connections between problems (Heerkens & van Winden, 2021). Figure 1.2 shows the problem cluster for this situation.

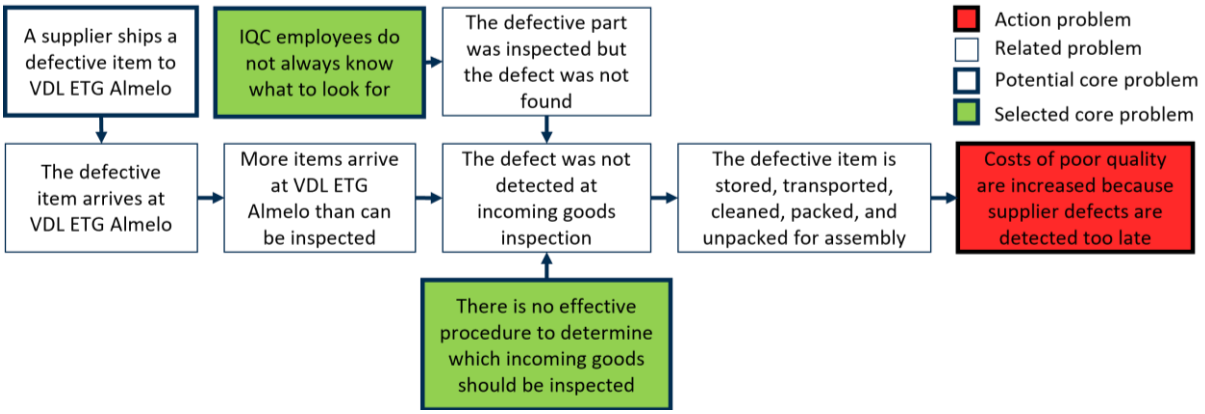


Figure 1.2: Problem Cluster

From the problem cluster a core problem is selected. This problem has no direct cause, and it becomes the core problem to be solved during the project (Heerkens & van Winden, 2021). Three potential core problems were identified in the problem cluster, namely:

1. The supplier ships a defective item to VDL ETG Almelo.
2. IQC employees do not always know what to look for during an inspection.
3. There is no effective procedure to determine which incoming goods should be inspected.

Employees of the supplier quality department collaborate with suppliers to prevent quality issues as much as possible. However, VDL ETGA has limited influence on their suppliers' processes. It is therefore unlikely that supplier defects can be prevented altogether. Core problem 1 is out of scope for this project.

VDL ETGA suspects that sometimes supplier defects are detected in assembly on parts that were inspected by incoming quality control. Currently, it is difficult to trace a purchased item back to a specific purchase line once it enters production. That makes it unclear how often existing supplier defects are not caught during an inspection. That is why core problem 2 is selected as one of the two core problems of this project.

It is known that only 9% of supplier defects are caught by the incoming quality control. It is also known that far more purchased items arrive at VDL ETGA than can be inspected. In 2023 1.3% of Systems 1 purchase lines were flagged for inspection. The current process is not able to determine inspection priorities sufficiently, due to the high number and diversity of products arriving at the warehouse. This is why core problem 3 is selected as second core problem for this project. In conclusion, this project focuses on the selection and prioritisation of incoming goods to inspect. The selected core problems for this project are as follows:

2. IQC employees do not always know what to look for during an inspection.
3. There is no effective procedure to determine which incoming goods should be inspected.

1.4 Problem approach

Ideally, VDL ETGA is able to predict which incoming parts are defective. Therefore, to solve the core problem, a model will be developed that calculates the probability of an incoming item being defective and selects incoming items for inspection based on that probability. Data about historic quality defects and item characteristics will be used to calculate defect probabilities. The solution to this problem results in a reduction of the number of defective parts that are forwarded from the incoming goods to the production departments.

This section describes the strategy for solving the core problem. Stakeholders play a key role in mobilising support for this project. The stakeholders are the departments supply chain, supplier quality, and incoming quality control. Representatives of these departments are kept informed and consulted for important decisions. The incoming quality control departments is mostly interested in the result. The supply chain and supplier quality departments have a direct involvement in the project. The project schedule is shared with them, and progress of the project is discussed in weekly meetings.

Knowledge will be acquired from literature about incoming goods inspections in the manufacturing industry. Data from VDL ETGA is needed to analyse historical trends in supplier quality issues. This data is available, but its quality is uncertain. Therefore, knowledge must be acquired internally about VDL ETGA's ERP systems and the state of the data. Literature will be researched about data cleaning, preparing, and analysing to assess the quality of the data.

This project includes several risks which are taken into consideration. There is a risk that a significant correlation between historic defects and incoming goods cannot be found. In that case, it is not possible to predict future defects with the dataset. This risk is unavoidable and was taken into consideration when it was decided to conduct this research. The choice in software programme to be used for the solution can result in additional costs. It is therefore important to analyse the costs and expected savings to minimize the risks of any required investments.

1.5 Research problem

The scope is demarcated to purchased items for Systems-1. The goal of this research is to improve the number of supplier defects detected at incoming goods inspection relative to the supplier defects detected at assembly. The current percentage of Systems-1 defects detected at incoming quality control is 9%. The goal is achieved by researching how a predictive model can be created and implemented that analyses purchased items, calculates the probability of a specific purchase item being defective, and uses that information to decide which incoming items should be inspected. The objective is translated into the main research question. The research question to be solved during this project is as follows:

How can defects among incoming items be predicted, and how should VDL ETG Almelo use this information in their inspection policy to improve the IQC detection performance?

Each of the following sub research questions covers a specific part of the main research question. Each question is answered chronologically in a separate chapter of the thesis. Together, the sub research questions provide the information required to answer the main research question from section 1.6. The sub research questions are as follows:

1. What do the relevant processes surrounding incoming items inspections at VDL ETGA look like?
    1. What are relevant production and logistics processes?
    2. How does VDL ETGA currently perform the inspection of incoming items?
    3. How is determined which parts to inspect, as well as how to inspect?
    4. What is the current performance of the IQC process?
2. What methods are proposed in literature regarding the selection of incoming items to inspect?
    1. What methods and algorithms are generally used for this purpose?
    2. How should the required data be prepared, cleaned, and merged?
    3. How can the created model be tested and validated?
3. What should a predictive model for the selection of incoming goods inspections look like, and how can it be tested and validated?
    1. What should the model for this problem look like?
    2. How can the model be tested and validated?
    3. Which classifiers should be tested with the model?
4. What is the performance of the selected possible solutions in predicting supplier defects at VDL ETGA?
5. Which solution should be recommended for implementation at VDL ETGA?
6. How should the recommended inspection model be implemented into the Incoming Quality Control processes at VDL ETGA?

1.6 Research approach

During this project a model is developed that predicts which incoming items are most likely defective. The model also provides information about how the items should be inspected. The detection performance of incoming quality control should be improved by using the model. The approach to solving the research problem follows the chronology of the research questions. The 6 sub research questions are divided into phases. The following sub paragraphs explain per project phase what

information is needed to answer the research questions. This includes an explanation of how the information is obtained, analysed, and implemented. The main research question is answered in the project conclusion phase.

Project plan

In this phase the project is defined, and relevant literature is researched. Research questions 1 and 2 are covered within this phase. The research questions are formulated as follows:

1. *What do the relevant processes surrounding incoming items inspections at VDL ETGA look like?*
2. *What methods are proposed in literature regarding the selection of incoming items to inspect?*

In this phase the current production and logistics processes relevant to the project are analysed. VDL ETGA's current methods to perform incoming parts inspections is explained, and the current performance of the process is measured. The data is collected with two methods. The first method is via conversations with supply chain, supplier quality, and incoming quality control representatives. The second method is an internal document search. Data from the ERP system regarding complaints caused by suppliers is used to measure the current performance. Chapter 2 answers research question 1.

A Literature review is performed to find what selection methods are commonly used regarding incoming goods inspections. Relevant algorithms for this problem are researched. Literature about best practices in data preparing, cleaning, and merging is reviewed as well. The literature review is performed in chapter 3. It includes a more detailed explanation about the literature search approach.

Model design

The activities in this phase of the project include designing the predictive model and selecting variants of the model that are tested with data from VDL ETGA. Research question 3 is addressed during this phase. The question is formulated as follows:

3. *What should a predictive model for the selection of incoming goods inspections look like, and how can it be tested and validated?*

Based on output from literature review a model is created. Also from literature, a selection is made consisting of potentially good performing algorithms for this application. In the model design it is specified what data is required as input, the operations that are performed, and what the model outputs. The decisions about which models and the number of models and algorithms to test are made after the literature review and discussions with stakeholders. This phase also includes the creation of a method to test the selected models and algorithms. Research question 3 is answered in chapter 4 of the thesis.

Quantitative analysis of possible solutions

In this phase the selected models are tested, and their performance is measured. Research 4 is answered during this period. The research question is formulated as follows:

4. *What is the performance of the selected possible solutions in predicting supplier defects at VDL ETGA?*

The required information for this research question in the model creation and testing phases. The model uses a training data set as input and a testing data set to measure the performance. Both data sets are compiled from VDL data. The source(s) for the VDL Data are exports from the ERP system Baan IV, the warehouse management system WMS, or the data warehouse in use (iQBS). Data cleaning techniques researched in the literature study are applied to prepare the data for the model. This research question covers chapter 5 of the thesis.

Comparison of the tested models

In this phase the test results and performance of the models are compared. After this comparison it will become clear which model performs best. Research question 5 is answered in this phase. The research question is formulated as follows:

5. *Which solution should be recommended for implementation at VDL ETGA?*

The information required in this phase are comparison techniques suitable for this purpose, found in the literature review. A method to compare the results is created from this information. The input data is collected from the test results and model performance of the previous phase. The solutions are compared to the current performance established in the project plan phase. The comparison of the tested models will also be answered in chapter 5 of the thesis.

Implementation

An implementation plan is developed and executed for the recommended model to VDL ETGA. This phase covers research question 6. The question is formulated as follows:

6. *How can the recommended model be implemented into the incoming goods inspection process?*

Information on how to implement the possible solution models are researched during the literature review. The implementation plan is a step-by-step guide of the activities that are performed to successfully implement the recommended solution. The implementation is covered in chapter 6 of the thesis.

Project conclusion

The main research question is answered in this phase. The required information to answer this question is collected during all previous phases. No new information is collected during this phase. The answers to the previous questions are analysed to draw conclusions, formulate recommendations, and discuss limitations.

2 Context Analysis

Chapter 2 contains the context analysis of the core problems in this project. The context contains all current processes and performance figures that are relevant to this project. The Incoming Quality Control processes are described in sections 2.1. The systems that provide the information about purchased items and quality defects are described in section 2.2. The current performance of the incoming quality control is measured in section 2.3. Lastly, the chapter is concluded and directions for the literature study in chapter 3 are provided. This chapter answers research question 1:

What do the relevant processes surrounding incoming items inspections at VDL ETGA look like?

2.1 Quality inspection processes

The following subsections explain the quality inspection processes at VDL ETGA that are relevant to this project. In section 2.1.1 the current IQC process is explained. Section 2.1.2 discusses the current method that is used to select which items are inspected. Section 2.1.3 contains a brief overview of the other quality inspection processes that are related to this project.

2.1.1 Incoming Quality Control process

Purchased items are delivered to the warehouse at VDL ETGA. Warehouse employees register all incoming goods in the warehouse management system (WMS) after they arrive. This information is synchronised with the ERP system Baan IV at a near real-time basis. Two different algorithms determine which purchase lines should be inspected. These current algorithms are explained in section 2.1.2. Items that are selected for inspection are transported to a temporary location at the warehouse. A daily updated list from the data warehouse is used showing which purchase lines should undergo Incoming Quality Control (IQC) inspection. This list also includes a brief explanation on how that purchased item should be inspected. IQC notifies the warehouse employees when they are ready to inspect the items.

Three people work full time in the IQC department. Two of them perform the inspections of incoming goods for all workflows. The other IQC employee is responsible for counting stock in the warehouse. A quality engineer can create specific inspection instructions for each item in a software programme called Compact.Net. If the inspection requires measurements of the product dimensions, then the item is often transported upstairs to IQC's workplace. On average IQC must inspect 3 purchasing lines per day. The number of incoming items varies per purchase line. IQC employees must determine themselves how large the sample size for an inspection will be per line. It takes approximately 1.5 hours to inspect a purchase order line. If IQC employees determine that the item is defective they open a quality complaint REM.Net. The process for creating quality complaints is explained in section 2.2.1. In REM.Net the IQC employee defines and describes the problem they found. The complaint is then forwarded to a quality engineer.

A quality engineer receives this complaint. They must then decide whether the complaint is justified or not. Items of unjustified complaints are considered OK and are returned to the warehouse where they are further processed. If the complaint is justified, the quality engineer determines whether a deviation notice suffices or that the item is rejected. A deviation notification states exactly what the deviation is, but the item can still be used for assembly. Deviation notices can be used, for example, if the non-conformity is visual but not functional. The process for rejected items is carried out in consultation with the responsible buyer for that item. The buyer must agree that the supplier caused a rejection. If the buyer agrees, then the items are brought to a designated warehouse location from where they are

transported back to the supplier. The buyer is responsible for the process of returning rejected items. Figure 2.1 shows a flow chart of the incoming quality control and item rejection process.

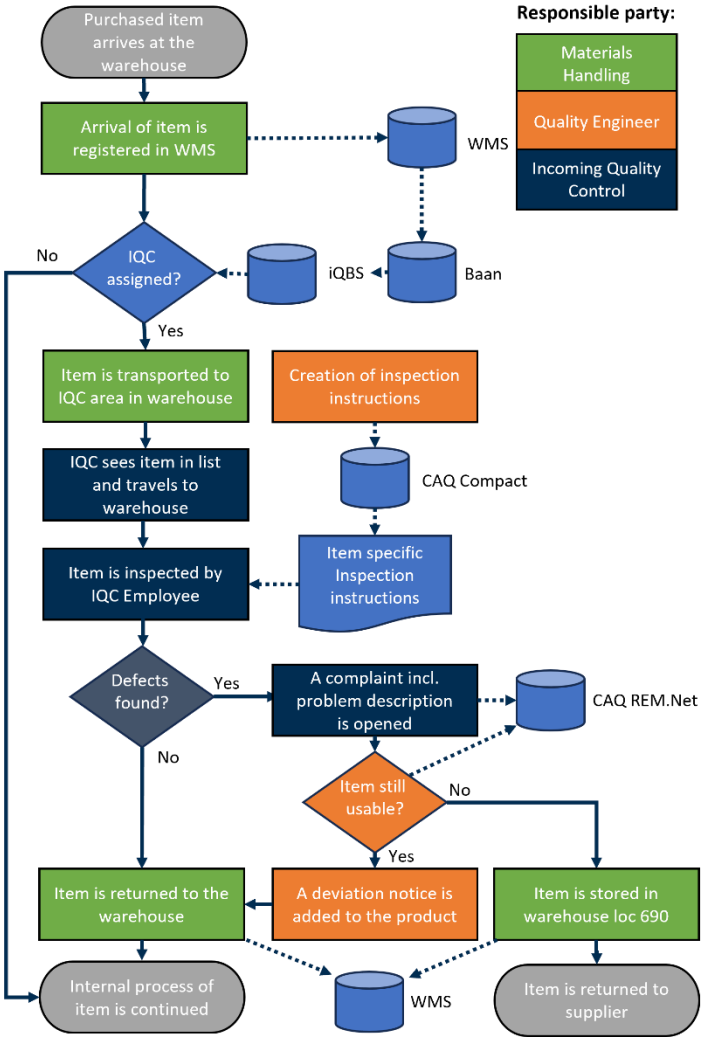


Figure 2.1: Flow chart of the incoming quality control process

2.1.2 Current selection method for IQC

Currently, two separate scripts determine which incoming items should be inspected. The scripts are written in Structured Query Language (SQL) statements and are executed as stored procedures in the Baan database. These scripts iterate over the purchase order lines in Baan and run subsequently overnight. The first script checks if the incoming items are new, have a new supplier, or have not been purchased in the last two years. If any of these checks is true, then the item is submitted to a First Article Inspection (FAI). The scripts modify the FAI field in the purchase order line(s) in Baan directly. In an FAI, the item is thoroughly inspected and measured. The results are included in an FAI report. FAI is a mandatory policy at VDL ETGA and is not subject to change.

The second script is to be replaced during this project with a better performing selection method. This script iterates over purchase lines that are not selected for an FAI and determines which of the remaining purchase lines should be inspected as well. The script checks if the purchased item has an HIP-code. HIP stands for High Impact Part. These are items that have been classified as critical by either the customer or VDL ETG. The HIP-Code is a number between 1 and 4 or blank. The most important items are HIP-code 1. The script also takes the item group into consideration. All item groups are numbered. The script enables a Boolean variable in the column 'Inspection' in Baan if the purchase line

is selected for inspection by IQC. Once a night the purchasing data is extracted from Baan into iQBS, which is used to generate a list of items to be inspected by IQC.

In a previous effort, it was decided that groups below 210 and above 515 are generally not selected for inspection, because these groups contain items that were empirically found to be less problematic. Also, items that merely required an FAI or material certificate were no longer categorised as IQC inspection. As a result of the stricter selection, the inspection capacity is currently not a bottleneck at IQC. IQC is able to inspect all selected purchasing lines on time. However, the current script is not effective in finding defects between incoming parts. This is discussed further in section 2.3.

Outside of the scripts, purchase lines are sometimes manually flagged in Baan for inspection. For example, when a defect has been found on a purchased part. In that case the responsible purchaser manually sets the Boolean inspection column to true in Baan, so that the next delivery of that item is inspected.

2.1.3 Related quality inspection processes

Approximately 91% of the supplier quality defects are not detected at IQC. If IQC does not detect the defective item, it continues in the internal process. The item is cleaned, packed, and stored until it is needed for assembly. From here on there are several moments where the defect can still be detected, either accidentally or in a formal inspection. This subsection describes the most common locations and activities where the supplier defects are detected.

Before items can enter the clean rooms, they must go through an air lock. Small items arrive in batches on shelf trolleys at the air lock. The items are manually unpacked, cleaned if necessary, and put onto a clean trolley inside the air lock. The manual handling of the product allows for the person in the air lock to perform a quick visual inspection. During a visual inspection the airlock employee can detect defect types like general finishing of the surface or damage to the product.

If no concerns are raised, then the items are forwarded to the cleanroom. Here the separate parts and components are assembled into a final product. Problems that lead to the registration of a supplier defect can arise during the assembly steps. For example, when two parts do not fit together, the items are measured. This can lead to the detection of an error in the dimensions of a supplied item. 90% of the supplier defects because of wrong dimensions are detected at assembly. Similarly, 80% of the supplier defects about faulty threaded connections are detected during the assembly process. The assembly process consists of several steps performed by multiple assembly employees. Each employee is supposed to check the quality of their work before the subassembly is forwarded to the next employee. This quality control step provides another opportunity to detect supplier defects.

Lastly, the final product is tested at VDL ETGA. The Before Acceptance Quality Test (BAQT) is the final check before the product is shipped to the customer. Here, the functional performance of the product is tested. If it fails, an investigation is started to find out which part caused the failure. A potential outcome is that a purchased part caused the functional failure. This type of supplier defect is not something IQC can check. A solution to this for the selection of purchase order lines to inspect by IQC is presented in chapter 6. If a defect on the item is detected during any of the process steps, then the complaint registration process from section 2.2.1 is followed.

2.2 Information systems

VDL ETGA uses several software programmes that collect, store, and present valuable information for this project. Section 2.2.1 shows where, and how, all quality complaints are registered. VDL ETGA uses

an ERP system called Baan IV. All purchase order lines are registered and tracked in this system. Section 2.2.2 summarizes the purchasing information that is extracted from Baan IV.

2.2.1 Registration process of quality complaints

All product defects are registered in quality complaints on REM.Net. REM.net is a software programme by CAQ AG for complaint and nonconformity management. The software can be used to analyse mistakes and process violations to detect their causes and triggers (CAQ AG Factory Systems, n.d.). Figure 2.2 shows the REM.Net screen where the complaint information is registered.

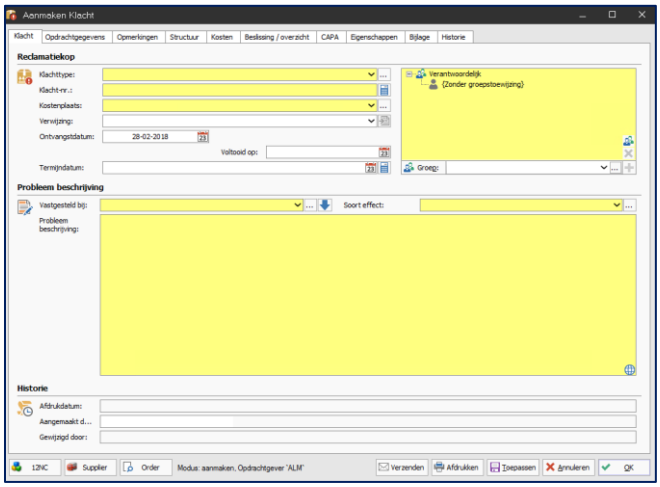


Figure 2.2: Example of registering a complaint in REM.Net

The registration process is set up so that the following information fields must be filled in to REM.net to log a quality complaint:

- Type of complaint
- Complaint number
- Responsible person
- Supplier (if applicable)
- Problem description
- Item code
- Number of defective items
- Encountered location
- Complaint trigger
- Date of detection

There are two types of complaints. Complaints are either internal complaints or supplier complaints, depending on who caused the complaint. A complaint number is then created automatically. This is a unique ID for every complaint. The responsible person for internal complaints is the production leader. The buyer of the defective item is responsible for supplier complaints. The supplier is registered as well. The problem description is a text box that can be used to provide additional information about the complaint. The encountered location is the department where the defective items were first encountered. The locations that can be selected in REM.Net are listed in Table 2.1.

Table 2.1: List of locations that can be selected as where the quality complaint was found

Encountered location	Explanation
Incoming goods	Detected at incoming goods inspection
FAI	Detected during a First Article Inspection
Material Handling	Detected during internal transport of the item
Calibration	Detected while calibrating equipment
Floor stock shopfloor	Detected when picking the floor stock on the production floor
Audit	Detected during an internal audit
External	Detected at other external locations than supplier or customer
Supplier	Detected at in-process supplier step (surface treatment, etc.)
Cleaning	Detected while cleaning the product

Cleaning centre	Detected at the cleaning centre
Outgassing	Detected while outgassing the item
Process control PMD	Detected while checking item with a personal measurement device
Process control next process step	Detected while product is inspected at next process step before start
Assembly	Detected at the airlock before assembly or during assembly
In process QC	Detected by in-process quality check
QCON	Detected by in-process quality check
BAQT	Detected during final product test
Final inspection	Detected during final inspection of the product
Customer	Detected after the product has been shipped to the customer

All complaints are classified under one of 19 defined complaint triggers, regardless of where they were detected. A complaint trigger is the reason why an employee noticed a complaint. The 19 complaint triggers are listed and explained in Table 2.2.

Table 2.2: Complaint triggers and explanation for when they should be used in CAQ-REM

Complaint trigger	Associated non-conformities
General Finish	Surface finish: burrs, roughness, sharp edges, machining marks, visual
Damaged	Deformation, dents, impressions, scratches on non-Critical to Quality (CTQ) surfaces
Coding, Engraving	Unique production number, location, completeness, sticker, readability, ID label
Completeness	Excess or insufficient number of delivered parts, swapped parts, incorrect product
Contamination	Fingerprints or other grease/oil residue, uncontrolled vibration, outgassing, oxidation, dust, particles, grinding residue, shavings
Critical Defect Area	Scratch on or incorrectly finished Critical to Quality surface/vacuum surface
Documents	Missing, incomplete, or incorrect documents directed to the customer
Electrical Connection	Connection, soldering, transition resistance, ESD (Electrostatic Discharge), shielding
Functional	Error codes during testing, component (co)operation, test results
Calibration	Incorrect calibration, out-of-calibration, rejection of measuring instrument during calibration
Welding and Soldering Joints	Density, strength, inclusions, discolouration, sealing
Adhesive and Sealant Joints	Adhesion, excessive or insufficient application, contamination, sealing
Air and Liquid Sealing	Leakage, kinked hoses, damage, vacuum
Dimensional Error	Length, width, height, diameter, flatness, form, and positional tolerances
Surface Treatment	Stains, adhesion, uniformity, colour tolerance, caps, masking
General Assembly	Placement, torque, loose component, incorrectly assembled, wrong part
Threaded Connections	Thread depth, incorrect thread, absence of thread
Raw Material	Composition, pre-treatment, rolling direction, hardness, casting defects
Safety and Environment	Unsafe situations, waste, oil leakage
Packaging	Absent, damaged

2.2.2 Purchasing information

Data about purchased items is stored in purchase order lines in Baan IV. The purchasing data for Systems-1 is analysed to extract information about the items that were delivered to VDL ETGA. The number of items that are supplied to VDL ETGA has been growing significantly for the past years. Figure 2.3 shows the increase in the number of delivered purchased items for systems-1 per year since 2021. The annual growth in this period was 28%. VDL ETGA expects that this growth continues through 2024-2030 due to the increasing demand for microchip machines. The high volume of incoming goods makes it infeasible to inspect all incoming goods at the warehouse before they are forwarded to the

production departments. The increasing number of incoming items makes it even more important to have an effective selection procedure for the inspection.

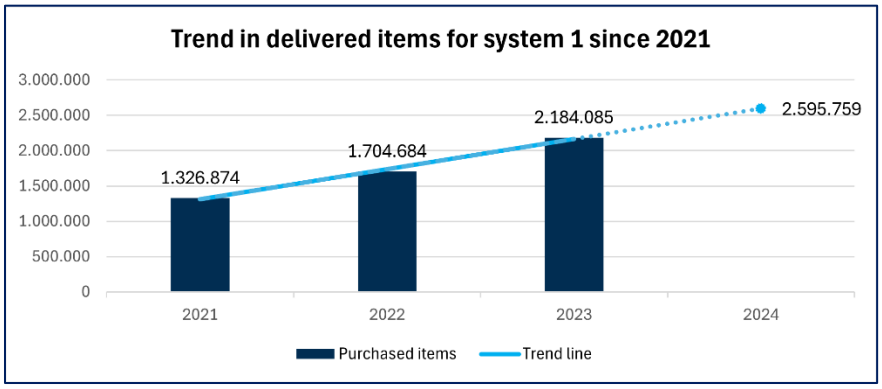


Figure 2.3: The number of purchased items since 2021 for Systems-1, including the expectation for 2024

The purchasing data for Systems-1 is used to extract information about the specific items that were delivered in 2023. All unique items used at VDL ETGA have an identification number and all unique items belong to an item group. 9033 unique items were delivered in 2023, divided into 35 item groups. The item groups range from complete modules assembled by suppliers to sheet metal parts to cables. Figure 2.4 shows an overview of the distribution of incoming items for Systems 1 in 2023.

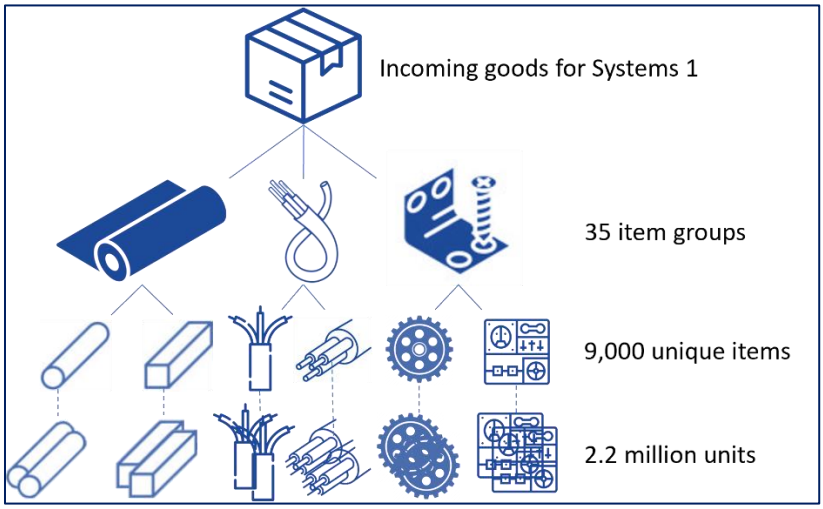


Figure 2.4: Overview of incoming items for Systems 1 in 2023

322 unique suppliers delivered the items. 80% of the total value of purchased items was delivered by 36 suppliers. Sometimes the same item is bought from different suppliers. 93% of the items were bought from only one supplier. The maximum number of suppliers for a single item is 3.

2.2.3 iQBS

The usage of multiple information systems means that the required information to answer a question can be stored in separate databases. To answer the question, the data must first be put together. VDL ETGA uses iQBS to solve this problem. iQBS provides the data warehouse where data from all sources is stored in the same format. Within iQBS the data can be cleaned and manipulated to improve the data quality of the reports. Figure 2.5 shows the flow of information for the current reports that use purchasing and quality data.

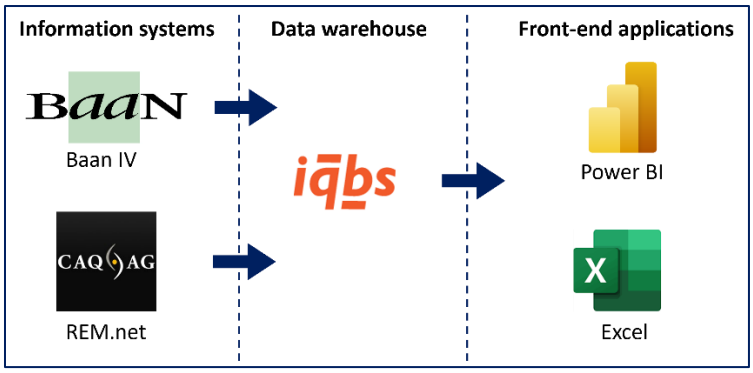


Figure 2.5: Flow of data for purchasing and quality Business Intelligence solutions at VDL ETGA

The prepared datasets for a specific purpose are called views in iQBS. One of the views is the Quality Data Cube. This view combines the purchasing data from Baan IV and quality data from REM.net into one dataset. The Quality Data Cube provides crucial information for the model that is designed in chapter 4.

2.3 Current performance of IQC

On average 159 purchase lines with Systems-1 items were delivered to VDL ETGA every workday in 2023. The daily average of purchase lines that were selected for inspection by IQC was 2. The year 2023 counted a total of 752 supplier complaints. The distribution of where in the process these complaints were found is visualized in Figure 2.6. The 'other' banner consists of process steps in between IQC and assembly, for example the cleaning department and material handling. 9% of supplier defects were detected by IQC. A large majority of the defects were not discovered until the airlock before cleanroom assembly or while assembling the items. Both locations are classified as location Assembly in REM.Net.

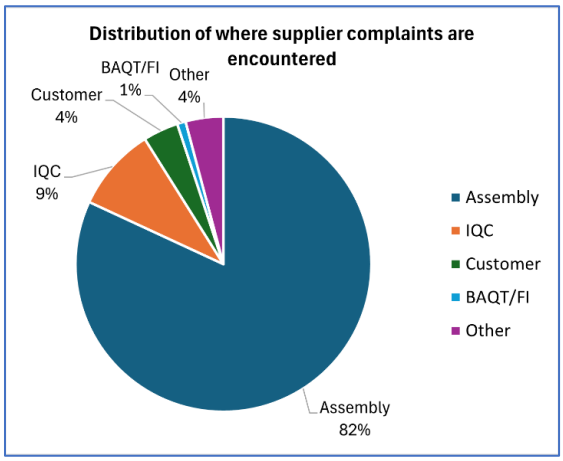


Figure 2.6: Distribution of where Systems-1 supplier complaints were encountered in 2023

All supplier defects arrive at VDL ETGA while they are already defective. That means that in theory 100% of supplier defects could be detected at IQC. Therefore, the two most important KPIs during this project are the IQC detection percentage and inspection precision. The IQC detection performance is measured through these two metrics. The KPIs are calculated as follows:

$$IQC\ detection\ \% = \frac{Count\ of\ supplier\ complaints\ detected\ at\ IQC}{Total\ count\ of\ supplier\ complaints} * 100\%$$

$$IQC\ precision\ \% = \frac{Count\ of\ inspections\ resulting\ in\ a\ supplier\ complaint}{Total\ count\ of\ inspections} * 100\%$$

Figure 2.7 shows a 3-month moving average of the IQC detection percentage and precision KPIs since 2022. The IQC detection percentage has been increasing since 2022. This is because of previous efforts in filtering out items deemed irrelevant for IQC. For example, items that only required a material certificate check were no longer flagged for IQC inspection. This reduced the noise in the data and allowed IQC to focus on actual quality inspections. However, as Figure 2.7 shows, the average inspection precision has remained at approximately 10%. Meaning that 1 in every 10 inspections results in the detection of supplier defects. The objective of this project is to further increase the detection percentage and precision, by creating a more effective selection policy.

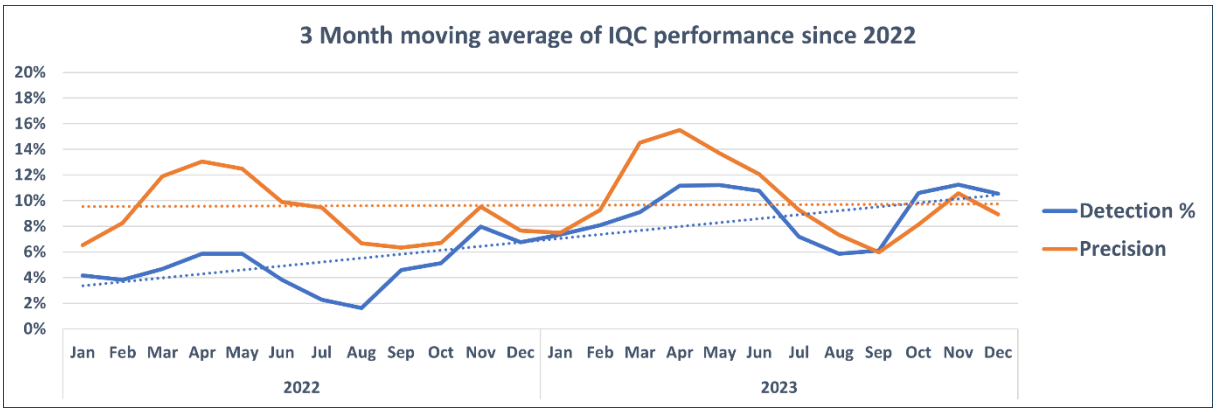


Figure 2.7: 3 months moving average of supplier defects that were detected at IQC

2.4 Summary

The purchasing data shows that millions of items are delivered to VDL ETGA every year. There is no effective procedure that determines which of these items should be inspected by IQC. The information about quality complaints is not explicitly used in the scripts that determine which items should be inspected. The IQC detection performance is measured by two metrics, namely detection percentage and inspection precision. The expectation is that further improvements can be made to the IQC detection performance if information about defects was used in the inspection selection process.

To make defect predictions, information can be looked at specifically for each item, or broader by looking at the item groups and suppliers. IQC does not register dimension measurements or similar data. Instead, this project focuses on purchasing data and quality data from earlier defects. This restriction on data is taken into consideration in the search for solution approaches in the literature study.

3 Literature Study

This chapter covers the literature study of this project. Section 3.1 describes how the literature was searched. Section 3.2 presents common models used for defect predictions. Section 3.3 describes theory regarding data collection and processing. Section 3.4 explains how predictive models are trained and scored. Lastly, section 3.5 explains how predictive models should be implemented. The theory found in literature is used to design the model in chapter 4. This chapter answers research question 2:

What methods are proposed in literature regarding the selection of incoming items to inspect?

3.1 Literature search approach

The objective of the literature study is to find relevant theory and common approaches to predicting quality defects. Along with theory, earlier applications of quality predictions in incoming quality control will be researched. First, to make the search process more accurate, common terminology in the field of quality is defined. Then, the search strategy is explained. This includes an explanation of the searched databases, the search terms, and the selection criteria.

3.1.1 Quality terminology

First, preliminary research is performed on quality management to determine relevant search terms. One definition of quality is consistent conformance to customers' expectations. A quality problem is a gap between the quality specification and the actual quality of a product. Quality Control is a subfield of Quality Management, and its focus lies on detecting and treating quality problems (Slack et al., 2016). Supplier quality represents the ability to meet or exceed customer expectations within critical performance areas consistently. The customers in this case are both the buying company and the end customer (Monczka et al., 2009). Incoming Quality Control then is the detection and treatment of quality problems caused by suppliers.

Inspection is the examination of a product. It compares the delivered results with the set standards. Inspections can be performed before production starts, during production, and/or after production has finished (Suresh et al., 2022).

Total Quality Management (TQM) is a management philosophy to enhance quality and productivity in organisations. Quality control is incorporated in TQM. In his book about TQM, Kiran uses the phrase 'inward inspection' to describe the process of inspecting incoming materials from suppliers. There are four types of inspection methods: dimensional measurements, go-no-go checking, functional checking, and visual inspection. The quantity that is inspected can be divided into two main categories. There is 100% inspection, where every incoming item is inspected, and there is sampling inspection, where only a selection is inspected. Sampling can be done at random or with statistical techniques (Kiran, 2017). Literature is searched for the use of statistical techniques in predicting item quality and performing product inspections.

3.1.2 Search strategy

The terminology from 3.1.1 is used to formulate search strings. These are used to search in online databases for relevant scientific papers. Table 3.1 shows the search strings and the number of documents that were found in Scopus.

Table 3.1: Search strings used in the literature study

Search string	Items found	Deemed relevant
Incoming goods quality inspection optimization	5	3
"Incoming quality control"	44	0

“Predicting product quality”	46	4
“predict- AND supplier AND defect-”	25	3
"Predictive model" AND "quality inspection"	21	3

Depending on the number of scientific papers found in the search, an iterative approach is used to refine the search string. Then the titles of the articles are scanned, and irrelevant articles are filtered out. The abstracts of the remaining articles are read and based on selection criteria it is decided whether the articles are interesting to read for this project. The inclusion criteria are as follows:

- The scientific paper describes sampling methods for incoming quality control.
- The scientific paper uses historic quality data to predict (supplier) product defects.

In chapter 2 it was established what types of data are available at VDL ETGA. Based on this information the following exclusion criteria are used:

- The scientific paper is aimed at the detection of defects in software applications.
- The scientific paper uses (live) condition-based data to predict product defects.

The remaining articles are read and summarized in the following sections. The references of the read articles are checked for potentially relevant articles that were not found during the searches in the databases.

3.2 Defect prediction models

A way of achieving the highest level of quality is to analyse past defects and predict future events (Verna et al., 2021). The main goal of quality predictions is to predict the behaviour of a product by applying a learning function that derives knowledge from prior information (Li et al., 2020). One of the most relevant technologies for predictive analytics in manufacturing processes is Machine Learning (ML) (Schmitt et al., 2020). ML differs from optimisation in its capability for generalization. Optimisation algorithms can minimize the error on a training set, but ML is aimed at minimizing the error on unseen samples (Suvrit Sra et al., 2011). Future incoming items are unseen samples in this project. ML is therefore better suitable than regular optimisation algorithms.

ML is a subset of Artificial Intelligence (AI). It is a technique that improves system performance by learning from experience. The experience is input as data to a model. The objective of ML is to develop a learning algorithm that builds a model from the data. The model is then used to make predictions on data it has not processed before. If the output is discrete, it is called a classification problem. A binary classification problem only has two classes. For example, an item is either defective or not defective. Problems with continuous output are called regression problems (Zhou, 2021).

Statistical ML, or Statistical Learning (SL), is the application of ML in the field of Statistics. In this project statistics are used to analyse and predict item defects. So, in the context of this project, ML and SL are synonymous and literature on both topics will be used. Most SL problems can be categorised as either supervised or unsupervised. The categories differ in their learning methods. Supervised learning means that for each observation of the predictor(s) there is an associated result (called response). With unsupervised learning there are measurements of the predictors for each observation, but the response is unknown. Unsupervised learning can be used to understand the relationship between observations (James et al., 2023).

We relate defective incoming items to a selection of predictors. And, after a certain period, it is known whether a delivered item was defective. So, the prediction of defects among future incoming items will be modelled as a binary classification problem with supervised learning. The learning data are

deliveries and quality complaints from the past years. (Schmitt et al., 2020) designed a predictive model framework specifically for quality inspections. The remainder of this literature study follows the layout as displayed in Figure 3.1.

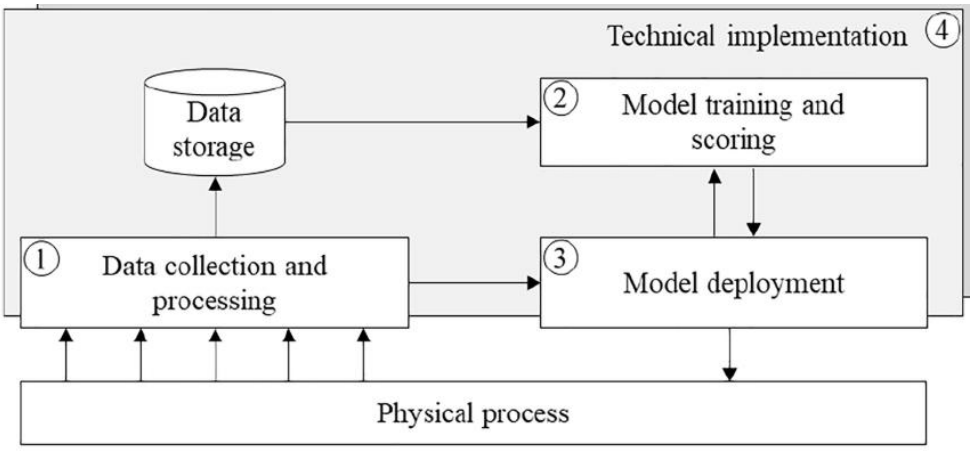


Figure 3.1: Layout of the predictive model-based quality inspection framework (Schmitt et al., 2020)

3.3 Data collection and processing

Data collection for prediction models can be divided into two steps, horizontal and vertical collection. Horizontal data collection is the identification and selection of relevant data sets/sources. This process is covered in chapter 2 of this thesis. Vertical data collection is the selection of a representative sample of historic data. The quality of the data must be evaluated before it can be used in a ML model (Schmitt et al., 2020). Vertical data collection and data quality evaluation can be performed by data preprocessing techniques. Data preprocessing techniques verify the quality of the data. Analysing data that has not been correctly screened for problems can produce misleading results. Data preprocessing is split into data preparation and data reduction (García et al., 2015).

In the data preparation phase the required measures are identified and implemented to treat missing values, redundancies, or inconsistencies in data. The preparation steps include data cleaning, transformation, and integration. Data integration means bringing together data from multiple sources into a single dataset (García et al., 2015).

Data cleaning consists of operations to correct bad data. An application of data cleaning is the imputation of missing data. The aim is to fill in missing values in a column with a reasonable estimated value. In most cases this is better than leaving it blank (García et al., 2015). Inappropriate handling of missing values may introduce bias and can result in misleading conclusions. A fundamental advantage of data imputation is that the missing value treatment is independent of the learning algorithm used. Data imputation can be performed with k-nearest neighbours. The attributes of a dataset often are not independent of each other. Thus, by identifying relationships among columns, estimates for missing values can be determined through the closest known values (Luengo et al., 2012).

After the data is cleaned it can be transformed. Data transformation involves converting and consolidating the data so that the predictive model can use it. This includes the construction of features (García et al., 2015). Features are the input variables to the model. Features are also referred to as predictors or as variables and are categorised as either numerical or categorical. Categorical features must be encoded before they can be used in a ML model. If features are encoded by index numbers, then the ML model assumes there is an order in importance of the values. A solution for this is one-hot encoding. In this process dummy variables are created for all unique values of the original variable. The variables are binary, the dummy variable that matches with the value of the original variable is

assigned 1, and the other dummy variables are assigned 0. A downside to this method is that it results in highly correlated dummy variables. For example, if all other dummy variables are known, then the value of the last dummy variable can be predicted with certainty. A solution to solve this problem is to arbitrarily remove one dummy variable from the dataset. Numerical and categorical features can be used simultaneously as input after this transformation (James et al., 2023). Figure 3.2 visualises the process of one-hot encoding, with one dummy variable removed.

Original data

id	Feature (colour)
1	Red
2	Blue
3	Green
4	Blue

One-hot encoded data

id	colour_red	colour_blue
1	1	0
2	0	1
3	0	0
4	0	1

Figure 3.2: An example of one-hot encoding for a feature with 3 unique values

Data reduction means reducing the size of the dataset by filtering out irrelevant or redundant features and instances. Figure 3.3 shows the difference between these two methods. Instance selection is used to remove rows from the data that hold no valuable information for the predictions. These rows add only noise to the model (García et al., 2015). Feature selection can also be performed within ML models, so that the model decides which features are relevant. This topic is discussed in section 0.

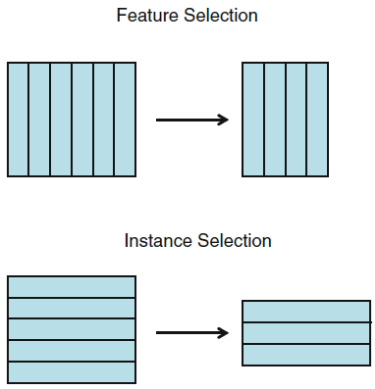


Figure 3.3: Forms of data reduction (García et al., 2015)

3.4 Model training and scoring

The ML model is developed in the training phase. The way a model is trained varies per classifier. The best performing algorithm cannot be determined upfront. Therefore, different algorithms must be tested and evaluated for each individual application. The pre-selection should be based on complexity, interpretability, and computation speed (Schmitt et al., 2020). A selection of supervised learning classification techniques, or classifiers, used for defect or quality predictions are discussed.

3.4.1 Classifiers

Common classifiers include k-Nearest Neighbours (kNN), Naïve Bayes classifiers (NB), Decision Trees (DT), Logistic Regression (LR), Support Vector Machines (SVM), Random Forests (RF), and Artificial Neural Networks (ANN) (Schmitt et al., 2020). (Canciglierie et al., 2021) tested the algorithms KNN, RF, Gradient Boosting Machine (GBM), and eXtreme Gradient Boosting (XGBoost). These were selected because of earlier applications in similar studies and because of their interpretability. (Yorulmuş et al., 2022) used LR, SVM, RF, GBM, XGBoost, and CatBoost. In all cases, the model learns to estimate an output based on one or more inputs. This can be modelled as

$$Y = \{0, 1\}\,(output)$$

$$X_j = input\ variable\ (predictor)\ j$$

In SL the assumption is made that there is some relationship between Y and $X = (X_1, X_2, \dots, X_p)$. This relationship can be written in the general form:

$$Y = f(X) + \epsilon$$

$$\epsilon = Random\ error\ term$$

f represents the systematic information that the predictors provide about the response. The inputs X are available, but the true f is unknown. By using ML models, we want to estimate f. With an estimation of f we can make predictions for the response variable Y using

$$\hat{Y} = \hat{f}(X)$$

Since the random error averages to 0, it can be left out of the equation. The goal of ML techniques is to minimize the error of the predictions. There are linear and non-linear methods to estimate f. Observations that are used to train the model to estimate f are called the training data. In the case of a linear model in a parametric approach, only the parameters $\beta_0, \beta_0, \dots, \beta_p$ must be estimated. We want to find their values such that

$$Y \approx \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

Non-parametric approaches do not make explicit assumptions about the functional form of f. This avoids the risk that the chosen functional form to estimate f is vastly different from the true f. This makes non-parametric approaches more flexible. In general, fitting a more flexible model requires estimating a greater number of parameters. These more complex models can lead to overfititng the data, which means that the model follows the errors too closely. Because of this, linear models can sometimes result in more accurate predictions than non-linear models (James et al., 2023).

KNN is a non-parametric approach. It assigns the class to a new observation that most of the k-nearest observations also belong to. The k observations with the most similar predictor values are considered the nearest neighbours. The value k can be tuned (James et al., 2023). Logistic regression and Tree-Based Methods are explained in more detail in the following two sections.

3.4.1.1 Logistic Regression

LR is a model that is suitable for binary qualitative responses. It models the probability that a certain observation belongs to a class. The response value is always in the range between 0 and 1, and therefore provides a meaningful estimate of $Pr(Y|X)$. The probabilities are used to classify the observations. For example, all $P(X) > 0.5$ are classified as defective item. This threshold can be tuned as desired (James et al., 2023).

It is called Multiple Logistic Regression if two or more predictors are used to classify a binary response. Multiple Logistic Regression uses the following logistic function:

$$p(X) = \frac{e^{\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p}}{1 + e^{\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p}}$$

This is a rewritten equation of the logit function and the parameters and predictors:

$$log\left(\frac{p(X)}{1 - p(X)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

$e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}$ returns a value between 0 and infinity. So, for large values p(X) goes to 1 and for small values p(X) goes to 0 (James et al., 2023).

A maximum likelihood method is used to find estimates for the parameters $\beta_0, \beta_0, \ldots, \beta_p$. This method searches for the parameter values that make predictions closest to 1 for observations labelled as 1, and predictions closest to 0 for observations labelled as 0 (James et al., 2023).

3.4.1.2 Support Vector Machines

Support Vector Machines (SVMs) are an extension to the maximal margin classifier so that they can be applied to classes with non-linear separation. SVMs are used for classification tasks. They attempt to find the best hyperplane that splits data points of different classes with a maximum margin between the lines and data points. By finding the maximum margin it becomes easier to classify new data points after training. Figure 3.4 shows an example of a linear SVM separating binary classes on a plane with two features (James et al., 2023).

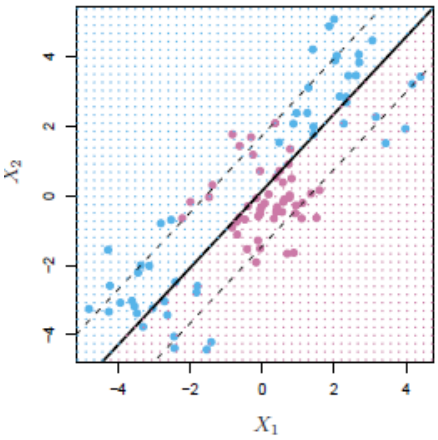


Figure 3.4: Example of a linear SVM that seeks a boundary to separate binary classes on a plane (James et al., 2023)

3.4.1.3 Tree-Based Methods

Tree-based Methods use decision trees to split the predictor values into a number of simpler regions. A new observation is then classified into the most common response in the region it belongs to. (Quinlan, 1986) visualized a simple decision tree for a positive (P) or negative (N) prediction based on the weather outlook, as seen in Figure 3.5.

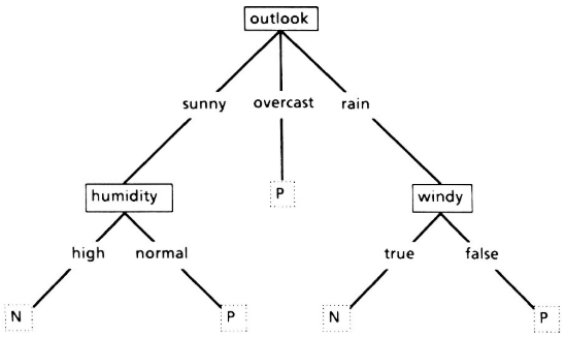


Figure 3.5: Simple decision tree for binary classification (Quinlan, 1986)

One decision tree is not particularly good at making predictions for large datasets. Ensemble methods improve the classification performance by combining many trees into a single model. Random Forest and Gradient Boosting Machine are examples of ensemble methods. RF builds, or grows, a large number of decision trees on training samples and then takes the average of the trees. Each time a split is considered, a random sample of the predictors is chosen a split candidate. GBM also gives predictions

by averages over trees. Gradient Boosting is a sequential method, each of the tree that is added is added to improve the performance of the previous collection of trees. The boosting approach learns slowly but prevents overfitting. By fitting small trees, the model fit improves slowly in areas where it is not performing well (James et al., 2023).

There are many variants of GBM. Two widely used variants are XGBoost and CatBoost. XGBoost is developed to handle sparse and big datasets efficiently. It has been the best performing model in many machine learning challenges (Chen & Guestrin, 2016). The CatBoost model is another variant of GBM and is specifically developed for categorical variables with many possible values. It uses ordered target encoding to encode categorical features, without the risk of data leakage (Prokhorenkova et al., 2017). According to their specificity and negative predictive value, the GBM and CatBoost algorithms perform the best at correctly classifying rare events (Yorulmuş et al., 2022).

3.4.2 Feature selection

Feature selection is used to find a minimum set of features, while the resulting predictions remain as close as possible to the predictions made if all features were used (García et al., 2015). Critical feature selection decreases the risk of overfitting and has a positive effect on the computation time (James et al., 2023). Feature selection itself can be computationally intensive if all feature combinations are considered. Two efficient methods are forwards and backwards stepwise selection.

Forward stepwise selection starts with one predictor and calculates which one results in the best predictions. It then keeps the best and adds another predictor. This process is repeated until p-1 predictors are added. Approximately p^2 models are considered. It then tests the best solution of every iteration and picks the overall best model. A drawback is that there is no guarantee of finding the optimal model because the optimum combination might change when adding another predictor due to correlations between predictors (James et al., 2023).

Backward stepwise selection starts with all predictors, then subtracts one at a time. The results between the two methods may vary because of correlations between predictors. Backward selection can only be used if the number of samples n is greater than the number of features p. Otherwise the full model cannot be fitted. In contrast, forward stepwise selection can be used even when n < p, and so is the only viable subset method when p is very large (James et al., 2023).

Feature selection can also be performed within the ML model. Least Absolute Shrinkage and Selection Operator (LASSO) can be used to tune the parameters of all predictors. It can also set the value of parameters to 0. It thereby excludes these predictors from the model (James et al., 2023).

3.4.3 Model validation

(Schmitt et al., 2020) and (Canciglierie et al., 2021) mention that they validated their algorithms with cross validation. Cross validation prevents overfitting. It divides the dataset into k parts (often 5 or 10). It fits the model on the first k-1 sets and then estimates the performance of the classifier on the kth set. This is repeated until all sets have been used as validation set. The average error is then calculated. The test error estimate is called cross validation error (James et al., 2023). Figure 3.6 visualises cross validation with k=5.

Figure 3.6: Example of cross validation with k=5 (Patro, 2021)

Class imbalance means that there are a lot more instances labelled as one class than the other class in a dataset. An insufficient number of samples of a class hurts the accuracy of the ML model. Imbalance is often high in applications for quality predictions. There are a lot more non-defects than defects. A solution used in one case study for quality inspection was to select only items marked as defective for inspection, thereby ignoring all negative predictions (Schmitt et al., 2020). Another method to handle class imbalance is through the application of preprocessing techniques for imbalanced datasets. There are two methods for this, undersampling and oversampling. Undersampling methods create a subset of the original data by eliminating majority class instances. Oversampling methods create a superset of the original data by replicating instances or creating new instances of the minority class (López et al., 2013). Both methods can improve the classification performance of the model. (Al-Rahman, 2021) visualised the results after applying under- and oversampling to a dataset. This visualisation is shown in Figure 3.7.

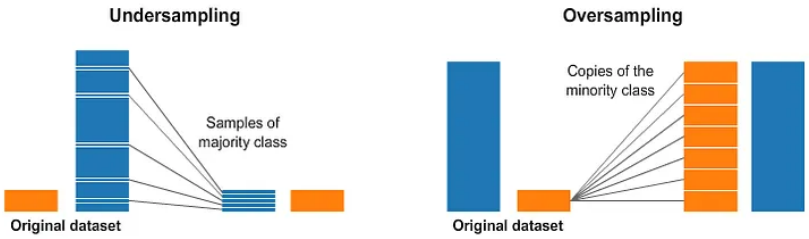


Figure 3.7: Results of under- and oversampling (Al-Rahman, 2021)

3.4.4 Model scoring

When classifying the quality of a product as ok or not ok, four metrics are relevant: accuracy, precision (TPR), recall (sensitivity), and F1 score (Sarkar et al., 2018). The sensitivity as used in literature is the same as the IQC detection percentage KPI as defined in Section 2.3. These metrics can be calculated from the values in a confusion matrix (James et al., 2023). Figure 3.8 shows a confusion matrix.

		Test result	
		0	1
True situation	0	True negative (TN)	False positive (FP)
	1	False negative (FN)	True positive (TP)

Figure 3.8: Confusion Matrix for Binary Classification (Kohl, 2012)

The values from the confusion matrix are used to calculate performance metrics. For example, the accuracy is the percentage of correct predictions. The formula to calculate accuracy is as follows (Schmitt et al., 2020):

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

Accuracy alone is not suitable for applications with unbalanced representation of the classes. Imbalance is often the case for quality-related industrial applications. It is not suitable, because in imbalanced datasets assignment of all instances to the dominant class would result in a high accuracy and no incentive to differentiate between the classes. A solution to this is to use the trade-off between the true-positive-rate (TPR) and false-positive-rate (FPR) (Schmitt et al., 2020).

$$TPR = \frac{TP}{TP + FN}$$
$$FPR = \frac{TN}{TN + FP}$$

Two other metrics for binary classification are precision and negative predictive value. These measure the percentage of correctly predicted positive and negative cases, respectively. The formulas to calculate sensitivity and specificity are as follows (Kohl, 2012):

$$Precision = \frac{TP}{TP + FP}$$

$$Negative\ predictive\ value = \frac{TN}{TN + FP}$$

The F1-score is the harmonic mean between the precision and TPR scores. It is calculated as follows:

$$F1\ Score = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

The graphical representation of the trade-off between sensitivity and specificity is captured by receiver operating characteristics (ROC) curves. The model whose ROC curve is closest to the top left corner is deemed to have the best trade-off (Schmitt et al., 2020). Figure 3.9 shows an example of an ROC curve.

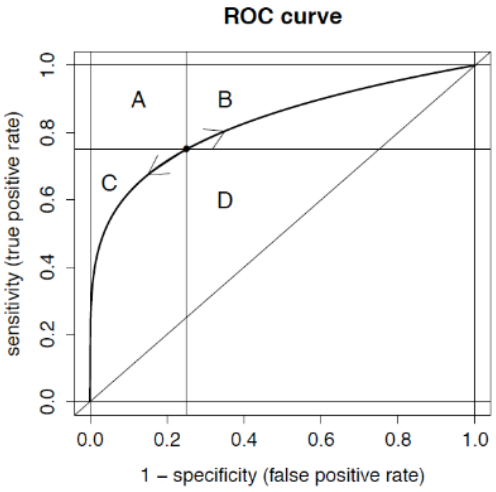


Figure 3.9: Example of an ROC curve (Kohl, 2012)

3.5 Implementation

This section describes the steps to implement a ML model for quality inspections. In the following subsections the practical and theoretical considerations regarding implementation are discussed, respectively.

3.5.1 Inspection model

The ML model must be integrated into the inspection process for a successful deployment. Earlier applications of ML models for quality prediction provide a guideline for the activities to be carried out.

Inspection exclusively based on the prediction model requires high model accuracy to reach the level of conventional inspection. Otherwise, hybrid approaches are a promising alternative, where the prediction model supplements the conventional inspection approach (Schmitt et al., 2020). Generally, the defect probabilities are obtained by prediction models and combined with inspection variables. From this combination a pair of indicators are defined for developing an inspection strategy (Verna et al., 2021). The model can be analysed after deployment by comparing the predictions to the logged data. This information can be displayed together in dashboards (Heymann et al., 2022).

(Schmitt et al., 2020) have created a model with supervised ML algorithms to predict final product quality using recorded process parameters. The predictions are used to filter out the items classified as defect-free. Then, only the items that were classified as not-OK are inspected. (Yorulmuş et al., 2022) created a model that searches for defects among defect-free approved products. So, products that are declared good, but in reality contain defects that were overlooked by quality inspection. It is modelled as a classification problem. The inspection policy is to inspect products that are classified as falsely declared defect-free. The inspections must be conducted before the items are sent to the customer. (Canciglierie et al., 2021) used a supervised machine learning model between assembly workstations to predict part failures in real-time. 5 attributes were used as input, which they considered a small number. Among the attributes was process measurement data, which was collected through sensors. The inspection policy is to stop and inspect the process if the ML model predicts the dimension deviations are going out of their control limits. The model is applied in real-time by storing the data in text files and sending it to a backup server. The machine learning model extracts the information from the backup server.

(Filz et al., 2020) model and analyse the effects of several inspection strategies by simulation. The effects of false positives and negatives are included in this analysis. The paper describes three possible inspection strategies. The first is to perform no inspections. The second is a 100% inspection strategy. The last strategy is called sample inspections, where only a selection of all products is inspected. The selection is made through a sampling method. The cost of each strategy consists of the error costs and the cost of inspections. If more inspections are conducted, then more errors are avoided. So, a trade-off exists between inspection costs and error costs. If the expected detection increase per inspection is known, then the total Cost of Quality (TCQ) can be calculated as the sum of the inspection cost and the error cost. The total costs are plotted over the error prevention percentage. This way the percentage that minimises the TCQ can be identified.

Inclusion and support of stakeholders is crucial for implementation of any ML model. Both can be improved through explainability techniques. If stakeholders understand why an ML model assigned a quality label to a product, it can increase their trust in the prediction and allows for better understanding of the process (Bhatt et al., 2020).

The quality predictions can be used in quality assurance as well. When integrated upstream the predictions could be used before the products reach the conventional inspection process (Schmitt et al., 2020). For example, within the suppliers' processes, and thus avoiding supplier defects reaching VDL ETGA altogether.

3.5.2 Technical implementation

The technical implementation of a ML model largely depends on the specific situation and is difficult to generalize. There are four main challenges to overcome, namely: limited processing capabilities, data dimensionality and volume, memory constraints, and time constraints for execution times (Schmitt et al., 2020).

From the reviewed papers, only (Schmitt et al., 2020) and (Krauß et al., 2020) were found to explicitly mentions the used programming language. They used Python for the technical implementation. Python is a free and open-source programming language (Python Software Foundation, 2024). Python has become an increasingly popular language for data science. ISLP and SkLearn are packages that support the development of ML models in Python (James et al., 2023).

After deployment, an ML model must be retrained to maintain a high performance level (Kavikondala et al., 2019). Therefore, a retraining strategy must be chosen. A model can be retrained periodically based on a fixed schedule. A disadvantage to this is that retraining does not necessarily lead to a different model, and so it is inefficient in the use of resources. Another method is to retrain the model on demand after certain criteria are met. For predictive quality use cases, notifications about performance measures falling below a limit can facilitate targeted actions. Through analysis of the monitoring data, the extent of the retraining is determined (Heymann et al., 2022).

A challenge is that ML projects generally require expertise from manufacturing and data science. A possible solution is to automate the processes of data preparation, integration, modelling, and deployment. (Krauß et al., 2020) research methods to automate the usage of ML models for predictive quality in production. The automation of ML is also known as AutoML. Auto-SkLearn is an application of AutoML. It can automate data preparation steps and the selection of a classifier. However, Auto-SkLearn does not support data integration. Other AutoML applications require paid subscription plans. Although AutoML can simplify the process of deploying ML models, it is often outperformed by manual implementation. AutoML still requires programming knowledge, and overfitting is a reported issue (Krauß et al., 2020). The concept of AutoML is still interesting and will be used in this project. Automating the repetitive and uncreative tasks increases the available time for creative tasks. For example, automated preparation enables the user to spend more time on tasks that improve future performance.

3.6 Conclusion

Table 3.2 on page 26 summarizes the methods used in similar studies compared to this project. For this project, a binary classification model with supervised learning is used to predict defects among the incoming items at VDL ETGA. The development of the model in chapter 4 follows the steps of data preprocessing, model training, scoring, and deployment. The scoring of the different algorithms is used to choose the best performing algorithm in chapter 5. 5-fold cross validation with time series splits is used to validate the model. The algorithms to be tested are LR, SVM, DT, CatBoost, and XGBoost.

The best performing model is implemented at VDL ETGA. Section 3.5 forms the theoretical basis for the implementation in chapter 6. The implementation consists of a functional and technical part. The technical implementation concerns the deployment of the model and automation of the process steps. The predictions are used as input for the inspection policy in the functional implementation. The performance of the model is used in a cost calculation to determine the best inspection strategy. The explainability of the models is used to maximize the support from stakeholders.

The comparable studies that were found during the literature search focussed on a limited number of internal items, utilizing in-process product measurement data and primarily numerical features. As explained in chapter 2, this data is not available for incoming goods at VDL ETGA. This makes predicting supplier quality a different challenge, due to the wider variety of items and a more limited availability of data. This thesis develops a predictive model for supplier defects, directly tackling this challenge and contributing to the theoretical understanding of quality control in supply chain management.

Table 3.2: Comparison of this project to similar studies found during the literature search

Article	Goal	Problem similarity	Classifier	Target value and metrics	Inspection model
(Bhatt et al., 2020)	Explainable machine learning in deployment.	Use of interpretable models to improve support for implementation.	General.	General.	N/A
(Canciglierie et al., 2021)	Predicting defects at assembly workstations.	Predicting defects.	KNN, RF, GBM, XGBoost.	Measurement error of dimensions. Objective is percentage rework reduction.	N/A
(Heymann et al., 2022)	Deployment of predictive quality models in production.	Usage of data to predict quality problems in production.	General.	Binary classification. Metrics are Accuracy, Precision, Recall, and F1 score.	N/A
(Krauß et al., 2020)	Automating machine learning models to predict quality in production.	Automating the data preparation, integration, and model deployment.	XGBoost and RF.	Three classes to also predict failure location. The metric used is the F1 score.	N/A
(Kavikondala et al., 2019)	Automated retraining of machine learning models.	Retraining of ML models after implementation.	General.	General.	N/A
(Tong et al., 2018)	Applying Data Mining in semiconductor Quality Control.	Predicting defective items in a manufacturing environment.	SVM, RF.	Binary classification with imbalance. Accuracy, Precision, and Recall used as metrics.	N/A
(Schmitt et al., 2020)	Predictive model-based quality inspections.	Inspections based on defect predictions with classification models.	DT, NB, LR, SVM, GBT.	Binary classification with imbalance (<1% positive class). Accuracy, Precision, and Recall used as metrics.	ML used to filter out most defect-free items. Only inspect items classified as not-OK.
(Verna et al., 2021)	Planning inspections by defect predictions.	Inspections based on defect predictions.	Power-law regression.	Number of defects per product unit as target value. Missed defects and inspection cost are used as metrics. The objective is to minimize the total cost of quality.	Use defect probabilities to perform inspections that minimize trade-off between false negatives and false positives.
(Yorulmuş et al., 2022)	Find products that were declared defect-free after inspection but in reality contain defects.	Usage of complaints data to predict future defects.	LR, SVM, RF, GBM, XGBoost, and CatBoost.	Binary classification.	Inspect products classified as falsely declared defect-free, before they are sent to the customer.
(Filz et al., 2020)	Choosing a quality inspection strategy through simulation.	Determining what quality inspection policy should be used.	Simulating predetermined strategies.	Minimizing the total cost of quality.	A random sampling inspection strategy.
This project	Predicting which incoming goods likely contain defective items and selecting those for inspection.		LR, DT, SVM, XGBoost, CatBoost.	Binary classification (1% belongs to positive class). Objective is to optimize F1 score.	Prioritise inspections based on calculated probability.

4 Model Design

This chapter provides the design of the model that is used to solve the prediction part of the core problem. First, the outline of the model is explained. The subsequent sections of this chapter follow the outline of the model. The corresponding research question to this chapter is question 3:

What should a predictive model for the selection of incoming goods inspections look like, and how can it be tested and validated?

The problem of predicting defects among incoming items is modelled as a binary classification problem with supervised learning. The solution is divided into three parts, so that each part can be analysed and optimised separately. Figure 4.1 shows the outline of the ML model for this problem. Sections 4.1 shows the data preparation steps that must be taken before the models can be trained. Section 4.2 explains how the models are trained. Section 4.3 explains how the models are optimised for each classifier. The final models are trained with the selected features and best hyperparameter settings. Section 4.4 shows how the models are evaluated. The results for the model optimisation and evaluation are presented in chapter 5.

Figure 4.1: Outline of the Machine Learning model

4.1 Data preparation

The data preparation covers the first part of the model. The section is split into data preprocessing, data integration, and data encoding. Data preprocessing takes raw data as input and prepares it for integration. During integration, the data is merged into one dataset. Data encoding ensures that the ML model correctly interprets the features and response data.

4.1.1 Data preprocessing

Two sources provide the data that is used as input for the ML model. The main source is the quality data cube. The cube pulls purchasing data from Baan and quality complaints from REM. Then, the data is cleaned and improved within the cube. These activities ensure that the item characteristics are correct, and the date column is filled. The data is stored in one table in an Excel file and is refreshed through an SQL-query. Currently the purchasing lines and quality complaints are appended, but the purchasing lines and corresponding supplier complaints can be combined into a single record if a match is found. The data must be pre-processed before these matches can be found.

There are other reports that use the quality data cube for different purposes, and it therefore contains several rows and columns that are irrelevant to this project. These are filtered out to reduce the data size and improve the computation times. The first preprocessing step is splitting the quality data cube into separate purchasing and quality datasets, both filtered in Systems-1 data only. It becomes clear that some values are missing, as shown in Figure 4.2.

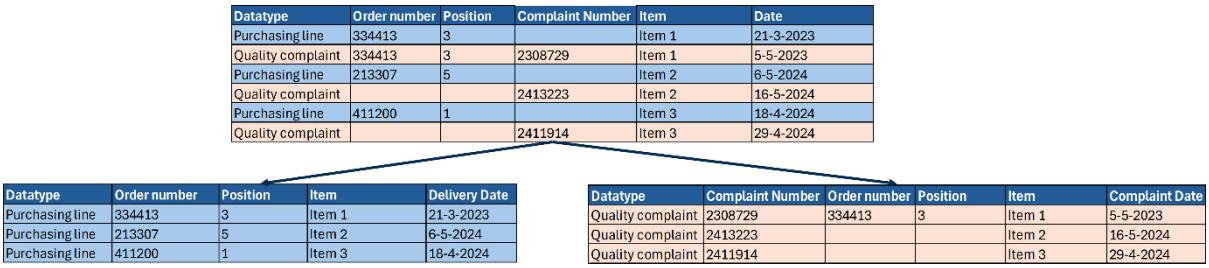


Figure 4.2: Visualisation of the data preprocessing

Each row in the purchasing dataset is called a purchase line. A purchase line is a combination of a purchase order and the position within the order. Generally, every unique item in the order gets a position number. All purchase lines that have been delivered since January 2021 for Systems-1 are in the dataset. As seen in Table 4.1 there are only few missing values in this dataset. This is due to previous efforts to improve the data quality.

Table 4.1: Data quality of the purchasing data

Column name	Data type	# unique values	Missing values
Order number	Numerical	157,424	0%
Position	Numerical		0%
Item code	String	11,775	0%
Item group	String	36	0%
Price	Numerical	N/A	0%
Supplier	String	311	0%
Supplier country	String	15	0%
Delivery date	Date	N/A	0%
Commodity	String	48	0%
Inspection	Boolean	2	0%
Product recognition	String	43	0%
Product type	String	52	1.1%
Cost component	String	19	0%
Ordered quantity	Numerical	N/A	0%
HIP code	Categorical	3	90.6%

The HIP code has many empty rows, because a code is only assigned to the most important items. The codes range from 1 to 3. An empty field means that the item has not been identified as a high impact part. For this project the empty rows are put into a separate category labelled '4'. The inspection column in the purchasing dataset shows whether a purchase order line was selected for inspection by IQC. This column is important, because in chapter 5 it will be used to compare the expected performance of a trained ML model to the actual performance. The inspection and commodity columns are not included in the quality data cube but are imported from a file with raw purchasing data.

The LR and SVM model work better if numerical features are normalized. Normalization ensures that each features contributes equally, and that regularization is applied uniformly. The numerical features for this project are the ordered quantity and the item price. The values of feature 'j' are normalized by replacing the values with a standard score z_{ij} for each sample 'i'. The following formula is used, where u_j and s_j are the mean and standard deviation of the training samples:

$$z_{ij} = \frac{x_{ij} - u_j}{s_j}, \forall i, j$$

Tree-based models are not sensitive to the scale of the features. So, CatBoost, XGBoost, and the decision tree do not require normalization of numerical features.

The quality dataset contains every complaint registered at VDL ETGA since 2021. This includes complaints related to other workflows, internal mistakes and complaints that were later retracted. Among all registered quality complaints, 20% were classified as unjustified complaints. The unjustified complaints are removed from the dataset, along with complaints related to other workflows than Systems-1. As explained in chapter two, justified supplier complaints can be resolved by either returning the item to the supplier or adjusting the internal process. For the prediction of defects in this project, no distinction is made between these two flows, because the process at IQC is the same for both types of complaint. The order number column is imported from a file that contains raw quality complaints data. Table 4.2 shows the quality of the supplier complaints dataset.

Table 4.2: Data quality of the supplier complaints

Column	Data type	# unique values	Completeness
Complaint number	String	2,228	100%
Item code	String	1,104	100%
Encountered in	Categorical	16	100%
Authorization	Categorical	3	86%
Registration date	Date	N/A	100%
Workflow	Categorical	1	100%
Order number	Numerical	N/A	55%
Position	Numerical	N/A	47%

The column authorization is blank for 14% of the remaining rows, because for these complaints a decision has not yet been made. In consultation with the quality department, it was decided to include the blank rows in the dataset, because they generally contain the most recently available information regarding item defects. The order number and position are blank for 45% and 53% of the rows, respectively. The operators say that it is not easy to retrace the exact purchase order in Baan if the complaint is logged after IQC. Furthermore, by studying the filled in rows, it becomes clear that 'order number' is broadly interpreted while logging the quality complaint. The filled in order number is sometimes a purchase order, but often it is a production order or sales order number. It is important to know the related purchase order because it is used to integrate the purchasing and supplier complaints datasets. A solution to the problem of unknown purchase orders is shown in section 4.1.2.

4.1.2 Data integration

The goal of the ML model is to predict which incoming items are defective. The incoming items are represented by purchase lines. The supplier quality complaints indicate which purchase lines contained defective items. To train the ML model, the purchasing lines and supplier complaints datasets must first be integrated. The data integration step takes the prepared purchasing and quality datasets from section 4.1.1 as input, and adds the supplier complaints to the corresponding purchasing line. It does this based on the order number and position as registered in the complaint. It is therefore necessary to solve the problem of missing values in these columns.

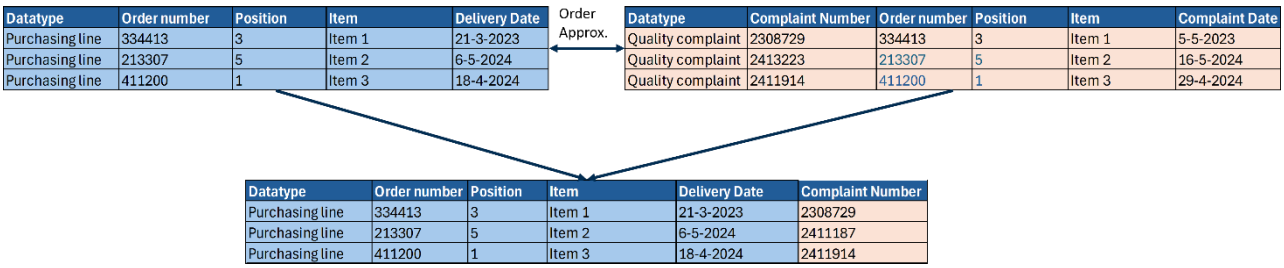
An exact match in purchase lines was found for 7% of the supplier complaints by using the registered order numbers. For the remaining complaints, an approximation algorithm was used to determine a purchase line. Algorithm 1 below shows the steps taken to approximate the purchase lines.

ALGORITHM 1: APPROXIMATE A PURCHASE LINE FOR SUPPLIER COMPLAINTS

Input: *Supplier complaints with unknown purchase orders and purchasing data*
Output: *An approximated purchase line for each supplier complaint*

```
1   for each complaint in supplier complaint
2       Potential lines = Purchase lines with delivery date <= complaint registration date
3       Potential lines = Potential lines that match the item in the complaint
4       Expected delay ← Exp. days after delivery depends on encountered location of complaint
5       Expected delivery date = complaint registration date – Expected delay
6       for each line in potential lines
7           Difference = | delivery date – expected delivery date |
8       Select purchase line with smallest difference
9   End
```

Another 68% of the supplier complaints were linked to a purchase order through the approximation algorithm. So, a total of 75% of the supplier complaints in the dataset are ready to be integrated. Most of the remaining complaints are related to purchase order lines delivered before 2021. The last step in the data integration is merging the purchasing data and supplier complaints into one dataset. This step is visualised in Figure 4.3.



Datatype	Order number	Position	Item	Delivery Date
Purchasing line	334413	3	Item 1	21-3-2023
Purchasing line	213307	5	Item 2	6-5-2024
Purchasing line	411200	1	Item 3	18-4-2024

Datatype	Complaint Number	Order number	Position	Item	Complaint Date
Quality complaint	2308729	334413	3	Item 1	5-5-2023
Quality complaint	2413223	213307	5	Item 2	16-5-2024
Quality complaint	2411914	411200	1	Item 3	29-4-2024

Datatype	Order number	Position	Item	Delivery Date	Complaint Number
Purchasing line	334413	3	Item 1	21-3-2023	2308729
Purchasing line	213307	5	Item 2	6-5-2024	2411187
Purchasing line	411200	1	Item 3	18-4-2024	2411914

Figure 4.3: Visualisation of the data integration

4.1.3 Data encoding

The integrated dataset that is created in section 4.1.2 is transformed to improve the performance of the ML model. The complaint number is transformed into the binary response variable, or 'y'. The response variable 'complaint' is 1 if a purchase line contains defective items. Otherwise, it is 0. The response variable makes it possible to use supervised learning for this problem. For each line the prediction can be verified by checking whether a complaint was actually registered to the purchasing line. The delivery date column is encoded to a numerical value that represents the year and week, so that the ML model can process it.

The LR, DT, and SVM models cannot process categorical values as feature variables naturally. These are the variables that are used to make the predictions. Most feature columns in the purchasing lines are qualitative with no natural order. This type of data is known as nominal data (Theisens, 2021). The nominal data must be one-hot encoded to ensure the ML model does not assume any order in the values. The process of one-hot encoding is explained in section 3.3. Figure 4.4 shows an example of what it looks like when encoding is applied to the complains and item groups in the purchasing lines.

Order number	Position	Item Group	Complaint Number
984200	1	Componenten Pneu/Hydr. Anoniem	2113602
426842	15	Componenten Elec. Anoniem	0
426835	8	Cat.3 Mechanisch [allocatie]	2119054

Data encoding

Order number	Position	Item Group [Componenten Pneu/Hydr. Anoniem]	Item Group [Componenten Elec. Anoniem]	Item Group [Cat.3 Mechanisch [allocatie]]	Complaint
984200	1	1	0	0	1
426842	15	0	1	0	0
426835	8	0	0	1	1

Figure 4.4: An example of original to encoded purchasing lines data

4.1.4 Mathematical notation

The mathematical formulation of the problem helps with understanding the operations that are performed within the model during the training stage.

$$n = datapoints(orrows)$$

$$p = predictors(orcolumns)$$

$$x_{ij} = value\ of\ jth\ predictor\ for\ ith\ data\ point$$

$$i = 1,2, \dots, n$$

$$j = 1,2, \dots, p$$

$$\boldsymbol{X} = \begin{pmatrix} x_{11} & \cdots & x_{1p} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{np} \end{pmatrix}$$

$$x_i = \begin{pmatrix} x_{i1} \\ x_{i2} \\ x_{ip} \end{pmatrix}$$

$$x_j = \begin{pmatrix} x_{1j} \\ x_{2j} \\ x_{nj} \end{pmatrix}$$

$$y_i = 0,1 = non - defective, defective$$

$$y_i = \begin{pmatrix} y_1 \\ y_2 \\ y_n \end{pmatrix}$$

4.2 Model training

A ML model must be trained before it can be deployed in practice. In section 4.1 it was described how the purchasing lines are prepared. This data is now used as input to train the ML models. Because of the approximation algorithm and subsequent encoding of the response variable, supervised learning methods can be applied. For each historic purchasing line, it is known whether it contained defective items. The supervised ML models use this information to train a classifier that provides the best predictions on new data. The following subsections explain how the different models are trained.

An important not here is that the model should not be trained on purchase order lines that were delivered less than two months ago. The labels for this data are not representative, because it is likely that most defective items have not been detected yet.

4.2.1 Logistic Regression

The LR model assumes a linear relationship between the predictors and the log odds of the response variable. It assigns a coefficient to every predictor. The model uses a maximum likelihood method to find estimates for the coefficients $\beta_0, \beta_0, \dots, \beta_p$. Only data points from the training dataset are used to find coefficient estimates. While training, the LR model measures the average change in the log odds of the response variable by a one-unit change in a predictor. This becomes the predictor's coefficient. Because it is a linear model, these values can be found independently for every predictor. The maximum-likelihood method results in coefficient values that return, after the log odds transformation, probabilities closest to the actual outcome in the training data. So, probabilities close to 1 for defective items and probabilities close to 0 for non-defective items. The following formulas show an example of the equations if the item group and supplier are included as predictors:

$$log\left(\frac{p_i(defect)}{1 - p_i(defect)}\right) = \beta_0 + \beta_{i1} * ItemGroup_i + \beta_{i2} * Supplier_i + \cdots + \beta_{ip} X_{ip}$$

This equation is rewritten to return a probability for each observation:

$$p_i(defect) = \frac{e^{\beta_0+\beta_{i1}X_{i1}+\cdots+\beta_{ip}X_{ip}}}{1+e^{\beta_0+\beta_{i1}X_{i1}+\cdots+\beta_{ip}X_{ip}}}$$

The LR model is trained after it has completed estimating the coefficients on the training dataset. The equations show that the importance of each feature is reflected by the magnitude of its coefficient. Features with larger coefficients are considered more important. This evaluation of the LR model is explained in section 4.4.

4.2.2 Support Vector Machine

SVM is the second classifier that is fitted to the data. The SVM model uses one-hot encoding to process categorical features. The features are selected in section 5.1.1. The objective function of the linear SVM model for classification problems is as follows:

$$\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} w_{y_i} \zeta_i$$
$$subject\ to$$
$$y_i(w^T \phi(x_i) + b) \geq 1 - \zeta_i$$
$$\zeta_i \geq 0, i = 1, \ldots, n$$

The objective is to minimise the sum of two terms on the training dataset. The first term ($\frac{1}{2} w^T w$) helps to find the classification decision boundary with the largest distance to the closest training samples. This improves the generalization ability of the model, which means that it is more likely to correctly classify new purchase order lines. The second term ($w_{y_i}\zeta_i$) in the objective penalizes misclassifications. Here, ζ_i (zeta) is a slack variable that allows some data points to be misclassified. To account for the class imbalance the class weight variable w_{y_i} is used. 'C' is a hyperparameter that controls the penalty for the misclassifications. The constraints ensure that each sample is either correctly classified or the slack variable is used.

The hyperparameters of the SVM model are tuned in section 5.1.2.5. The penalty strength 'C' is one of the tuned hyperparameters. The other hyperparameters are the number of iterations and tolerance. The number of iterations specify the maximum number of steps the model can take to find the best objective function. The tolerance is a threshold that can stop the optimisation process if the improvement in the objective function is too small in a new iteration.

4.2.3 Tree-based methods

Another type of classifier that is trained are tree-based methods. Both a single decision tree and gradient boosting machine (GBM) are developed. Two different GBMs are trained and tested, namely CatBoost and XGBoost. Both are tree-based gradient boosting algorithms. The CatBoost name is derived from Categorical Boosting. CatBoost and XGBoost are similar. But, unlike XGBoost, CatBoost is specifically designed to handle categorical data effectively.

It is not necessary to encode categorical features before they can be used in a CatBoost model. One-hot encoding creates a sparse matrix which may lead to overfitting. Instead, CatBoost uses ordered target encoding. Ordered target encoding is calculated row by row to prevent leakage that occurs with regular target encoding. Leakage means that the model uses data that it is not supposed to know while training. The following formula is used to encode categorical predictors:

$$X_{ij} = \frac{OptionCount_i + 0.05}{n_i + 1}, \forall j$$

OptionCount is the number of rows with the same categorical value and positive response variable that have occurred in rows before i. n_i is the number of rows it has already seen with the same categorical value. Table 4.3 shows an example of how ordered target encoding works on the item groups.

Table 4.3: Ordered target encoding example

Item Group	Target encoded	Complaint
300	0.05	1
515	0.05	0
300	0.525	1
045	0.05	0
515	0.025	1

Once the categorical features are transformed, the CatBoost model no longer differentiates between the feature variable types. The XGBoost model accepts one-hot encoded or regular mean encoded features. The GBM algorithms then proceed in roughly the same method. The algorithms start searching for the features and corresponding feature values that perform best at splitting the training data. The data is split in such a way that samples with the same class label are grouped together. The maximum number of splits is called the tree depth. The depth is a hyperparameter that can be tuned. This is discussed in section 4.3.2. The nodes at the bottom of the tree are called leaf nodes. Each sample belongs to one leaf node. The leaf values represent the probability of a purchase line being defective. The objective of the CatBoost classifier is to minimize the logarithmic loss (logloss) function. Logloss measures the total difference between the predicted probabilities and the actual class labels of the training samples. The formula for this function is as follows:

$$N = total\ number\ of\ training\ samples$$

$$w_i = Weight\ of\ sample\ i$$

$$y_i = actual\ class\ of\ sample\ i$$

$$p_i = predicted\ defect\ probability\ of\ sample\ i$$

$$Logloss = \frac{-\sum_{i=1}^{N} w_i\left(y_i log(p_i) + (1-y_i) log(1-p_i)\right)}{\sum_{i=1}^{N} w_i}$$

The weights in the logloss function allow for a distinction to be made between the label classes. By default, all weights are equal. However, in the dataset only 1% of the incoming items is defective. If weights were equal, the model would not be incentivised to correctly classify defective items. As a solution, the used weights are based on the occurrence of the classes in the training dataset. The weights are calculated automatically in the model. Now, the model is penalized more severely for misclassifying a defective item.

The boosting part of GBM is that it does not just consider one decision tree, but it creates many trees. The trees are created sequentially, and every iteration adds a tree to improve the performance of the previous collection of trees. So, the model keeps adding trees to reduce the logloss function. The number of iterations is another hyperparameter that is tuned in section 4.3.2. Focusing too much on reducing logloss during training can result in overfitting. CatBoost has a built-in overfitting detector. The detector stops the training procedure earlier than the parameters dictate if overfitting occurs. Then, it retraces its steps back to the model that resulted in the best performance on the evaluation dataset.

The model is done training after it has constructed all the decision tree in each iteration. When the trained model is applied to the test set, it traverses the trees for each sample until a leaf node is reached. The final prediction of a sample is the sum of the leaf values of all trees that were created during the boosting iterations.

4.3 Classifier optimisation

The performance of the models is likely not optimal with all potential features included and with default hyperparameter values. So, during the training stage, the performance of the models is optimised through feature selection and hyperparameter tuning. These two optimisation topics are discussed in subsections 4.3.1 and 4.3.2, respectively.

4.3.1 Feature selection

The number of potential features is significantly smaller for classifiers that do not use one-hot encoding. The ordered target encoding of CatBoost makes it easier to evaluate the added value of each predictor without leakage issues. Both the performance and the computation time of the models can be improved by including only a selection of the potential features. Feature selection improves the performance by reducing noise in the model, and it improves the computation time because fewer models are evaluated.

The Lasso method cannot be applied to a CatBoost model, so a different feature selection method must be used. The data preparation has resulted in 13 potential predictors. Best subset selection would have to evaluate $2^p = 2^{13} = 8192$ models. With an average of 10 seconds computation time per model it would have to run for approximately 23 hours to test all combinations. So, best subset selection is not feasible. Therefore, backward stepwise selection is used. Now the number of models to evaluate is $1 + \frac{p(p+1)}{2} = 1 + \frac{13 * 14}{2} = 92$ models. For 10 seconds per model this is only 15 minutes of total computation time. Algorithm 2 shows the actions that are performed during backward stepwise selection. Feature selection can also be performed in forward steps and this approach is tested as well. The forward stepwise algorithm and its results are shown in Appendix A.

ALGORITHM 2: BACKWARD STEPWISE FEATURE SELECTION WITH CROSS VALIDATION

Input: *Purchasing data with all potentially relevant predictors.*
Output: *Dataframe with all predictors sorted in order of performance improvement.*

1 *Let M_p denote the CatBoost model that contains all predictors.*
2 *Split purchasing data for k fold cross validation*
3 *predictors = list of all potential predictors*
4 *remaining_predictors = predictors*
5 ***For*** *j =0,1,…,predictors-1:*
6 ***For*** *each predictor in remaining_predictors:*
7 $M_{temp} = M_{p-j} - predictor$
8 ***For*** *k =1,..,k:*
9 *Train M_{temp} on folds 1 to k*
10 *Test CatBoost model on fold k+1*
11 *Store average performance of M_{temp}*
12 *Find M_{temp} with highest F1-score and call it M_{p-j-1}.*
13 *Store M_{p-j-1} and its F1-score in a list*
14 *Delete removed predictor in M_{p-j-1} from remaining_predictors*
15 *Select a single best model from $M_1, M_2, \dots, M_p$*
16 ***End***

The inclusion of cross-validation in the feature selection algorithm adds robustness to the model. it reduces the risk that the seemingly best model only performs well on the selected test set. However, regular k-fold cross validation is not suitable for the purchasing data because of leakage. In this context, leakage means that the model would classify item deliveries with training data from future deliveries. Instead, predictions should be made with models that are trained with earlier data only. Splitting the data on delivery date has the closest resemblance to how the model would be used if implemented at VDL ETGA. Because then the ML model is trained on historic data to predict defects among purchasing lines that have not yet been delivered. This is achieved with time series cross-validation. The cross-validation splits are not shuffled, but chronologically expanded for each fold. This makes it possible to estimate test errors on multiple sets, while preventing leakage. Figure 4.5 visualises the time series cross-validation. The most recent split is removed from the training and testing processes, so that it can be used to evaluate the performance of the best model on the most recently available labelled data. The model evaluation is discussed in section 4.4.

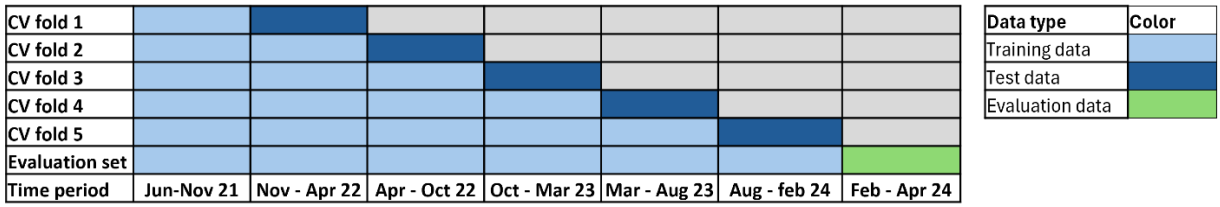


Figure 4.5: Time series cross-validation and evaluation splits visualised

4.3.2 Hyperparameter tuning

One of the parameters that can be tuned is the weight for each class. The class weights are used to account for the imbalance in the dataset. The weights are calculated with the following formula:

$$CW_k = \frac{\max_{c=1\ to\ 2}\left(\sum_{t_i=c} w_i\right)}{\left(\sum_{t_i=k} w_i\right)}$$

The result for the highest occurring class (non-defects) is 1. The other class (defects) is a number larger than one. In this case the value is around 100. The class weights are calculated within each model. The weights are used in the objective function to reward models that are good at predicting the rarer class. Other important hyperparameters and their settings are different for each classifier. Therefore, the hyperparameter tuning is performed per classifier using a cross validated grid search method. Algorithm 3 shows steps that are performed during the hyperparameter search. The specific hyperparameters and ranges tested are explained per classifier in section 5.1.2.

ALGORITHM 3: HYPERPARAMETER TUNING WITH CROSS VALIDATED GRID SEARCH

Input: *Training dataset, classifier M with the selected features, and hyperparameter values*
Output: *Performance metrics for every hyperparameter combination in the input*
for *each combination in hyperparameter values*
    $M_{temp} = M_{combination}$
    For *k =1,..,k:*
        Train M_{temp} *on folds 1 to k*
        Test M_{temp} *on fold k+1*
    Store average F1 score of M_{temp}
    If $M_{temp} > M_{best}$
        $M_{best} = M_{temp}$
End

4.4 Model evaluation

As shown in Figure 4.5, a separate dataset is reserved to evaluate the performance of the model. The evaluation dataset contains purchasing lines that were delivered in February, April, and May 2024. This is the most recently available labelled data because most complaints are only detected and registered weeks after delivery. The model has not seen any of the samples in this dataset during feature selection, hyperparameter tuning, and model training.

The model predicts defect probabilities on the evaluation set. The predictions must be turned into a classification. The classification is based on the predicted probability. Every purchase order line with a probability greater than a certain threshold is classified as defective. The threshold is a parameter that can be tuned after the predictions are made. The effect of this threshold on the performance metrics is analysed in chapter 5. Once the samples are classified, the predictions can be compared to the true labels. There are four possible outcomes of that comparison, which are shown in the confusion matrix in Figure 4.6. In the context of this project, a True Positive (TP) is a defective item marked as defective. False Positive (FP) is a good item marked as defective. True Negative (TN) is a good item marked as good, and False Negative (FN) is a defective item marked as good. So, TP and TN are correct predictions, and FP and FN are incorrect predictions. The values in the confusion matrix are used to calculate performance metrics.

Truth Predicted	0	1
0	TN	FN
1	FP	TP

Figure 4.6: The prediction categories in a confusion matrix

Within the dataset, approximately 1% of the purchase lines contain a complaint. As a result of this imbalance, the accuracy is not a good metric to measure the performance. If all purchase lines were classified as non-defective the accuracy would still be 99%:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} = \frac{0 + 99}{0 + 0 + 99 + 1} = 99\%$$

The goal of this project is to predict defective items, so the positive class in Figure 4.6. Performance metrics that focus on the predictive capability of the positive class therefore better reflect the true performance of the model.

$$Sensitivity = \frac{TP}{TP + FN}$$
$$Precision = \frac{TP}{TP + FP}$$

There is often a trade-off between sensitivity and precision. More inspections means that more defects are found, but it also results in more false positives. This results in a higher sensitivity, but a lower precision. The reverse is also true. A metric called F1 score is used to account for this trade-off. This allows for the model to maximise a single metric. The F1-score is the harmonic mean of the precision and the sensitivity:

$$F1\ score = \frac{2 * Sensitivity * Precision}{Sensitivity + Precision}$$

Another used performance metric is the area under the ROC curve (AUC). The ROC curve shows how well the model can distinguish between the two classes at different threshold settings. It is used to

analyse the trade-off between the true positive rate and the false positive rate. The AUC summarizes the overall performance of the classifier. The interpretation of AUC is as follows:

- AUC = 0.5: The predictive capability of the classifier is no better than a random classifier.
- 0.7 < AUC < 0.8: The performance of the classifier is acceptable.
- AUC ≥ 0.8: The predictive capability of the classifier is excellent.
- AUC = 1: The predictive capability of the classifier is perfect.

AUC gives an insight into the predictive capability of the model in general. However, as explained in section 2.3, the current inspection policy at VDL ETGA is not random. To determine whether the predictive model can improve the percentage of defects detected at IQC, its performance should be compared to the actual performance achieved at VDL ETGA. The evaluation dataset contains both the true labels and the selection of purchase lines that were selected for inspection in real life. This information is used to calculate the actual sensitivity and precision. Via these metrics the performance of the model can be compared to the actual performance at VDL ETGA. The results of this analysis are shown in section 5.2.3.

4.5 Summary

The purchase order lines and supplier complaints are merged into one dataset with an approximation algorithm. That creates the opportunity to use supervised machine learning models for binary classification. LR, DT, SVM, XGBoost, and CatBoost classifiers are fitted to the data. A mix of categorical and numerical features are used as input to predict which purchase order lines contain defective items. 5-fold cross validation is performed to make the feature selection and hyperparameter tuning more robust. Time series splits are used to create the folds, so that leakage is prevented.

It is important that the model should not be trained on purchase order lines that were delivered less than two months ago. The labels for this data are not representative, because it is likely that most defective items have not been detected yet. The most recent split in the data is separated from the training stage and is used to evaluate the performance of the ML models and compare it to the actual performance. This is done in chapter 5.

5 Results

This chapter presents the results of the models that were trained and tested for this project. Section 5.1 discusses the feature selection and how each classifier is optimised. In section 5.2 the performance of the models is compared to find the best performing ML model. The performance scores of the models are also compared to the current performance at VDL ETGA. Lastly, in section 5.3, an inspection policy is derived for the best performing model. The corresponding research questions to this chapter are questions 4 and 5:

What is the performance of the selected possible solutions in predicting supplier defects at VDL ETGA?

Which solution should be recommended for implementation at VDL ETGA?

5.1 Classifier optimisation

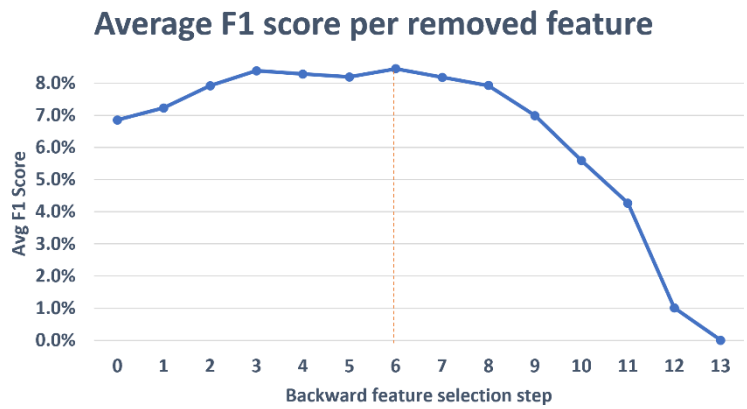
The models for each classifier are optimised for this application through feature selection and hyperparameter tuning. Section 5.1.1 presents the results of the feature selection. Section 0 shows the best hyperparameter settings that were found through the grid search method. Both optimisation methods use the training dataset with purchase order lines delivered from January 2021 up to January 2024. The dataset counts a total of 128,500 purchase order lines and 1,300 supplier complaints.

5.1.1 Feature selection

The feature selection is performed with the CatBoost model, because the other models are not able to process the inclusion of every potential predictor. The XGBoost model does not support a combination of categorical and numerical features, and the LR, DT, and SVM models become too large if all potential features are one-hot encoded and included.

First, the initial performance metrics of the model are calculated by training the model with all features and default parameters, and then applying it to the evaluation set. The default values of the CatBoost parameters are 1000 iterations, a learning rate of 0.082, and a depth of 10. The CatBoost model is programmed in Python with the CatBoost library. The resulting AUC is 0.79.

Now, stepwise feature selection is applied on the training dataset to select the final features. Stepwise feature selection can be performed either forward or backward. Backward feature selection is used for this project. Backward is better than forward steps at recognising which information from a feature is already covered by other features in the model. The feature selection algorithm as explained in section 4.3.1 is programmed in Python and executed. The total computation time is 30 minutes. Figure 5.1 shows the average F1 score per removed feature of the cross-validated backward stepwise feature selection.



Step	Removed predictor
0	Full model
1	Item type
2	Product type
3	HIP code
4	Country
5	End product
6	Commodity
7	Supplier
8	Item group
9	Ordered quantity
10	Price
11	Cost component
12	Item ID
13	Delivery week

Figure 5.1: The average F1 score per added feature in backward stepwise selection

Figure 5.1 shows that the model improves by removing the worst features. This is because these features merely add noise to the model. The overall highest F1 score in the stepwise feature selection is 8.5%. This is achieved after the sixth feature, commodity, is removed from the full model. So, the final model includes the following features:

- Item code
- Item group
- Supplier
- Price
- Cost component
- Delivery week number
- Ordered quantity

Applying the model with only the selected features to the evaluation set results in an AUC of 0.80. This is an improvement of 0.01 over the original model.

The influence of a predictor on the output of the model can be explained by its SHAP value. SHAP stands for Shapley Additive Explanations. Appendix B shows the SHAP values for each observation per feature. The SHAP values are used to calculate the average importance per feature, which are shown in Figure 5.2. The price and the item code are the most important features. 25% of the predicted defect probability is determined by earlier results from similarly priced items. 24% of the prediction is based on earlier defects on the same item. The third most important feature is the supplier. This shows that there is a significance difference between suppliers in expected quality. Note that the feature importance in the combined model differs from the feature selection in Figure 5.1. While the week of delivery is the most informative single feature, the combined model depends more on other features.

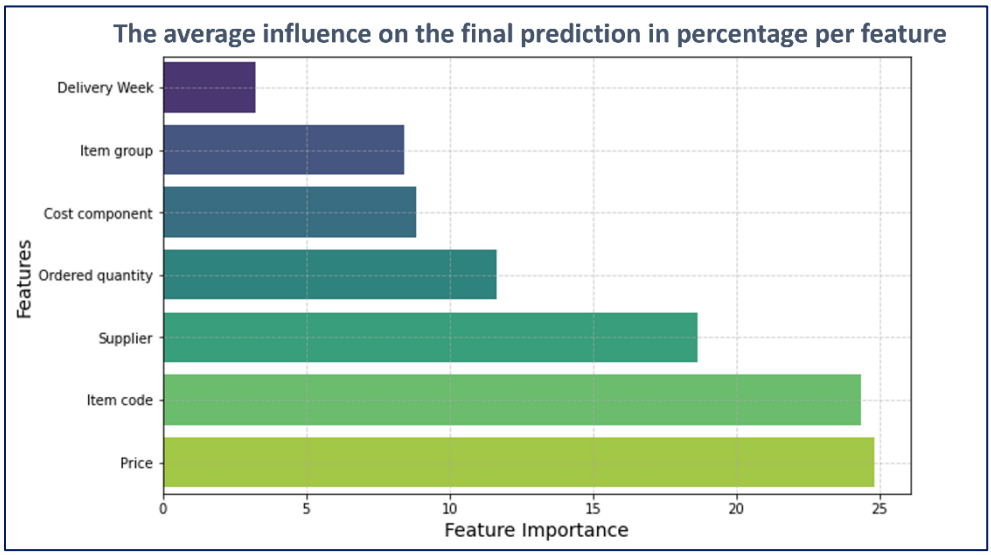


Figure 5.2: The average influence on the final prediction in percentage per feature

The categorical features are one-hot encoded for the LR, SVM, and DT models. There are 6000 unique items in the dataset. The size of the dataset is too large if all item IDs are one-hot encoded. Therefore, only the top 500 item IDs by defect rate, calculated on the training dataset, are included in the model. The top 500 item IDs are then one-hot encoded. Table 5.1 shows the number of features in the dataset after one-hot encoding. The total number is 910 features in the model.

Table 5.1: Total number of columns in the one-hot encoded models

Included feature	Datatype	Feature columns
Top 500 items	Category	500
Supplier	Category	304
Week number	Category	53
Item group	Category	35

Cost component	Category	17
Ordered quantity	Numerical	1
Price	Numerical	1
Total		**910**

The LR, DT, and SVM models are programmed in Python using the Scikit-learn module, and this module supports L1 regularization (Pedregosa et al., 2011). L1 regularization, also known as lasso, is used within the LR, DT, and SVM models. Lasso reduces the number of features with coefficients larger than zero, essentially eliminating unnecessary features from the model. This reduces the risk of overfitting.

5.1.2 Hyperparameter tuning

The following subsections explain per classifier which hyperparameters and respective values were included in the cross validated hyperparameter tuning algorithm. The algorithm from section 4.3.2 is used for the tuning process. The hyperparameters are tuned on the training dataset. Only the selected features from section 5.1.1 are included. The best hyperparameter combination for each classifier is used to compare the performance of the models in section 5.2.1.

5.1.2.1 CatBoost

First, a test is performed with the number of iterations set to 1000 and the overfitting detector turned on. The result is that overfitting occurs after approximately 250 iterations. This number may vary depending on the chosen depth and learning rate. Therefore, a cross validated grid search is used to tune the following hyperparameters:

- Iterations: 100, 250, 300, 500
- Learning rate: 0.05, 0.025, 0.01
- Depth: 10, 11, 12

This means that a total of $4 * 3^2 = 36$ hyperparameter combinations were tested. The total computation time is 3 hours. The top 5 hyperparameter combinations on average F1 score are shown in Table 5.2. The best average F1 score was achieved in the model with 250 iterations, a learning rate of 0.05 per iteration, and a tree depth of 10 levels.

Table 5.2: Top 5 hyperparameter combinations for the CatBoost model by F1 score

Iteration	Learning rate	Iterations	Depth	Avg AUC	Avg F1
28	**0.05**	**250**	**10**	0.780	**8.31%**
18	0.025	500	12	0.795	8.20%
16	0.025	500	10	0.799	8.13%
30	0.05	300	12	0.780	8.10%
17	0.025	500	11	0.797	8.07%

5.1.2.2 XGBoost

The cross validated grid search hyperparameter tuning algorithm is now applied to the XGBoost classifier. Again, only the selected features from section 5.1.1 are included. The regularization strength is tuned through the alpha parameter. The hyperparameters and their values that are tested in the cross validated grid search are as follows:

- Iterations: 200, 250, 300
- Learning rate: 0.025, 0.05, 0.1
- Depth: 4, 5, 6, 7
- Alpha: 0, 1, 10

The XGBoost model trains much faster than the CatBoost model. It only takes 5 seconds to train and evaluate one cross validation split, instead of 2 minutes. The total number of tested combinations is $3^3 * 4 = 108$. The total computation time is 30 minutes. The top 5 hyperparameter combinations on average F1 score are shown in Table 5.3. The best average F1 score was achieved in the model with 200

iterations of tree construction, a learning rate of 0.025 per iteration, a tree depth of 6 levels, and an L1 regularization alpha of 10.

Table 5.3: Top 5 hyperparameter combinations for the XGBoost model by F1 score

Index	Learning rate	Iterations	Depth	Alpha	Avg F1	Avg AUC
80	**0.025**	**200**	**6**	**10**	**10.1%**	0.740
68	0.05	300	6	10	10.0%	0.745
56	0.05	250	6	10	10.0%	0.746
32	0.1	300	6	10	10.0%	0.737
20	0.1	250	6	10	9.9%	0.738

5.1.2.3 Logistic Regression

The number of iterations, regularization strength, and tolerance can be tuned for the LR classifier. The Iterations parameter sets the maximum number of iterations that the solver will run. The default number of iterations is 100. The model might not converge within 100 iterations. Convergence is the point at which the algorithm stops making significant changes to the coefficients. Too many iterations can result in overfitting by focussing too much on minimizing the training error. So, if the LR model does not converge within 100 iterations, either 200 or 300 iterations may be better suitable for this project.

For regularization, LR supports Lasso regression. This type of regularization makes it possible to finetune the feature selection within the model. It can set coefficients of unimportant (one-hot encoded) features to 0, effectively eliminating them from the model. What the model considers unimportant depends on the regularization strength. The strength of the regularization can be tuned through the parameter 'C'. The default value of C is 1. Several other options around 1 are also explored. A smaller value of C means stronger regularization. The following values are tested: 0.01, 0.1, 1, 10, 100.

The last hyperparameter is the tolerance, which controls the precision of the convergence. Specifically, it sets the threshold for how much the cost function must change to not consider the model finished. The default value for the tolerance is 0.001. Other values that are tested are 0.0001 and 0.01. In summary, the hyperparameter grid is as follows:

- Iterations: 100, 200, 300
- C: 0.01, 0.1, 1, 10, 100
- Tolerance: 0.0001, 0.001, 0.01

A total of 45 hyperparameter combinations are tested. The total computation time is 30 minutes. The top 5 hyperparameter combinations on average F1 score are shown in Table 5.4. The best cross validated F1 score in the model is 7.84%. This is achieved with a hyperparameter combination of 300 iterations, a regularization strength of 1, and a convergence tolerance of 0.001.

Table 5.4: Top 5 hyperparameter combinations for the LR model by F1 score

Iteration	Iterations	C	Tolerance	Avg F1	Avg AUC
37	**300**	**1**	**0.001**	**7.84%**	**0.723**
21	200	1	0.0001	7.83%	0.722
6	100	1	0.0001	7.82%	0.723
36	300	1	0.0001	7.82%	0.723
22	200	1	0.001	7.81%	0.724

5.1.2.4 Decision Tree

Tuning the hyperparameters of a decision tree has a large impact on its flexibility. A flexible model has the potential of returning better predictions, at the risk of overfitting. Overfitting is less likely in a more

restricted model, at the cost of prediction accuracy. There are 3 hyperparameters optimised for the DT classifier. By tuning these three hyperparameters the best balance is sought between a flexible model and a more restricted model.

The first hyperparameter is the minimum number of samples in a leaf node. The default, and smallest possible value, is 1. Two other values are tested that are more restrictive, namely 5 and 10. The second hyperparameter is the minimum number of samples required to split an internal node. The smallest possible value for a split is 2, and this allows maximum flexibility. Values of 10 and 20 minimum number of samples are also tested. The third hyperparameter is the maximum depth of the three. The default value is an unrestricted depth, meaning that the tree can keep growing until other stopping criteria are met. Three other maximum depths are tested, namely 10, 20, and 30. So, the hyperparameter grid for the DT classifier is as follows:

- Min samples leaf: 1, 5, 10
- Min samples split: 2, 10, 20
- Max depth: None, 10, 20, 30

A total of 36 hyperparameter combinations are tested. The computation time is 30 minutes. The top 5 hyperparameter combinations on average F1 score are shown in Table 5.5. The best cross validated F1 score in the model is 7.69%. This is achieved with a hyperparameter combination of 5 minimum samples per leaf, 10 minimum samples per split, and a maximum tree depth level of 10.

Table 5.5: Top 5 hyperparameter combinations for the DT model by F1 score

Iteration	Max depth	Min samples leaf	Min samples split	avg F1 score	avg AUC
12	10	10	2	7.69%	0.740
18	10	10	20	7.67%	0.743
17	10	5	20	7.23%	0.743
13	10	1	10	7.23%	0.740
10	10	1	2	7.23%	0.736

5.1.2.5 Support Vector Machine

The three optimised hyperparameters for SVM are L1 regularization strength, convergence tolerance, and maximum number of iterations. These hyperparameters serve similar purposes to those in the previous models. The regularization can eliminate unnecessary features. The strength is set through parameter 'C'. The default value is 1. 0.1, 10, and 100 are tested as well. The tolerance determines when a model is considered converged. The default value is 0.001. 0.0001, 0.01, and 0.1 are also considered. Last, the maximum number of iterations sets the maximum number of attempts to optimise the model. The default value is 1000 iterations. 100 and 10000 iterations are tested too. This results in the following hyperparameter grid:

- C: 0.1, 1, 10, 100
- Tolerance: 0.0001, 0.001, 0.01, 0.1
- Maximum iterations: 100, 1000, 10000

A total of 48 hyperparameter combinations are tested. The computation time is 40 minutes. The top 5 hyperparameter combinations on average F1 score are shown in Table 5.6. The best cross validated F1 score in the model is 5.20%. This is achieved with a hyperparameter combination of C value 10, a maximum of 10000 iterations, and a tolerance of 0.0001.

Table 5.6: Top 5 hyperparameter combinations for the SVM model by F1 score

Iteration	C	Max iterations	Tolerance	Avg F1 score	Avg AUC
27	**10**	**10000**	**0.0001**	**5.20%**	**0.588**
30	10	10000	0.001	5.19%	0.555
33	10	10000	0.01	5.19%	0.624
24	1	10000	0.1	5.08%	0.596
36	10	10000	0.1	5.08%	0.569

5.2 Model performance analysis

In this section the performances of the tested models are compared. All models are trained on purchase order lines from 2021 to January 2024. The models use the best features and hyperparameter settings from the classifier optimisation in section 5.1. The models are evaluated on purchase order lines from February to April 2024. The models have not seen the purchase order lines in the evaluation dataset before, making it a fair a representation for how the model would be used in real life. The evaluation dataset contains 13,285 purchase order lines. A total of 144 supplier complaints have been registered to those purchase order lines. So, approximately 1% of the data is labelled as positive. The imbalance in the evaluation dataset is the same as in the training dataset.

5.2.1 Performance metrics per model

All models are used to make predictions on the validation dataset. The resulting predicted probabilities are used to plot ROC curves. Figure 5.3 shows the ROC curve of each model. As can be seen, the CatBoost, XGBoost, and LR model perform similarly at Sensitivity rates up to 20%. At higher sensitivity percentages, the CatBoost model consistently outperforms the other classifiers. CatBoost finds a better trade-off between the sensitivity and the false positive rate for most classification thresholds. This is reflected by its AUC value of 0.84. The logistic regression model also performs well at every threshold.

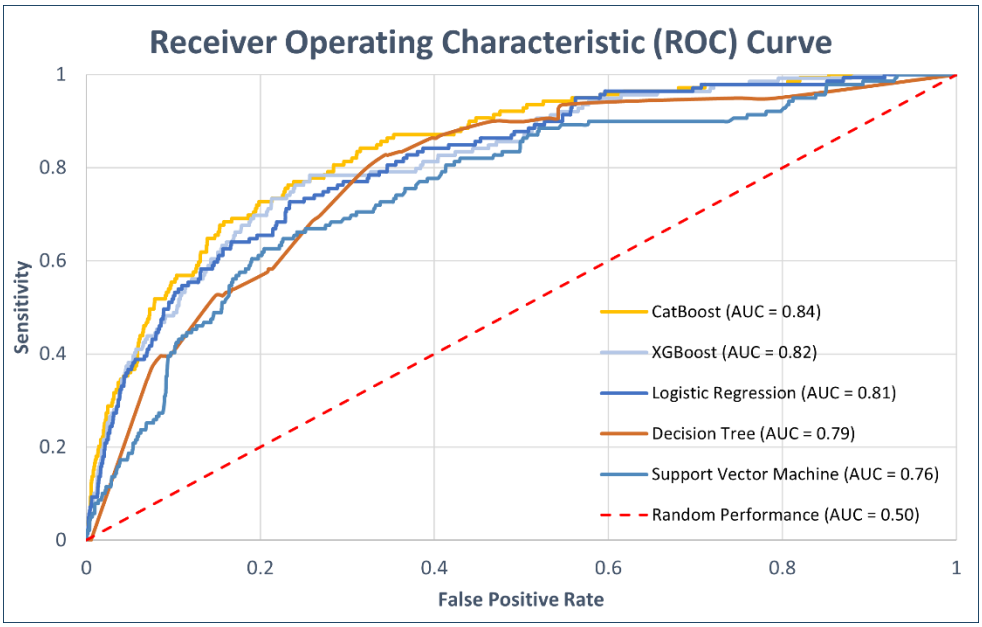


Figure 5.3: ROC curves of the tested models

The ROC curves give an indication about how close the predicted probabilities are to the true labels. Now the F1 score is used to compare how well the models actually perform at classifying the purchase order lines as either non-defective or defective. The predictions are transformed into classification with a fixed threshold. Every purchase order line with a predicted probability larger than the threshold is classified as defective. Otherwise, it is classified as non-defective. The estimated probabilities of

defective items are different for each classifier, so the performance of a classifier varies depending on the selected threshold. Comparing the models on a single classification threshold is therefore not a good method.

Figure 5.4 compares the F1 scores of the models for different classification thresholds between 50% and 95%. None of the models have predicted a probability larger than 95%. The decision tree was not correct in any of its predicted probabilities greater than 83%. Its curve is therefore undefined at thresholds larger than that. Figure 5.4 also shows that the F1 score of the XGBoost model is higher than that of CatBoost for thresholds up to 74%, but CatBoost greatly outperforms all other classifiers at higher thresholds.

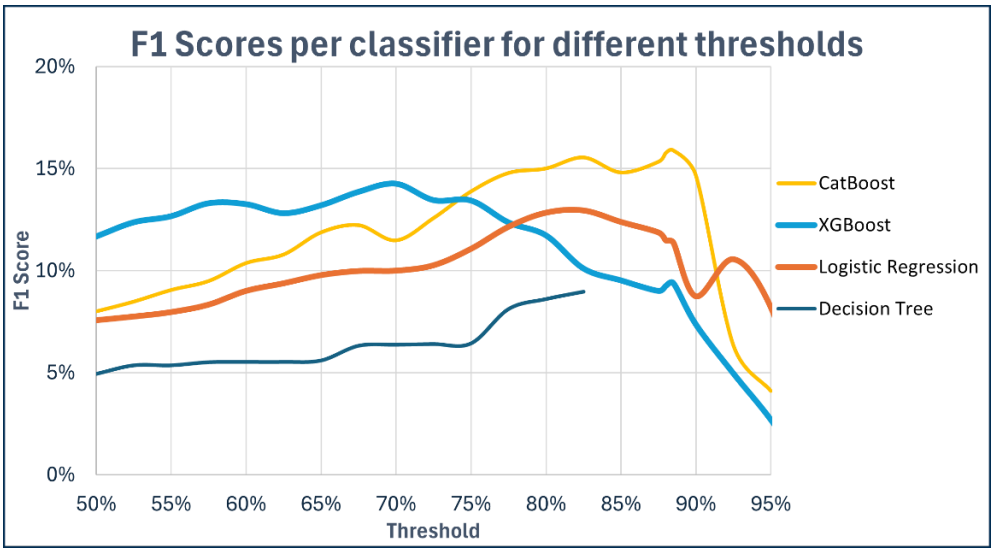


Figure 5.4: Comparison of the models on F1 score for classification thresholds between 50% and 95%

As mentioned, the evaluation dataset contains a total of 13,285 deliveries, of which 144 have a complaint registered to them. The F1 score is the harmonic average between the sensitivity and the precision. Table 5.7 shows what the F1 score means for each model with a classification threshold of 80%. This number is chosen, because it results in a comparable number of daily inspections to real life. The CatBoost model offers the best trade-off between sensitivity and precision. Recall that the sensitivity is the total percentage of the 144 complaints correctly classified. The precision is the percentage of true complaints among all purchase order lines classified as positive. So, for example, at an 80% threshold the CatBoost model has correctly identified 27.8% of the 144 complaints. A precision of 10% means that, of all inspected purchase order lines, 1 in 10 resulted in an actual quality complaint.

Table 5.7: Performance metrics of the models for a classification threshold of 80%

Classifier	F1 score	Sensitivity	Precision	Accuracy	AUC
CatBoost	**14.9%**	**27.8%**	10.2%	96.1%	**0.84**
Logistic Regression	12.8%	**31.9%**	8.0%	95.1%	0.81
XGBoost	11.6%	13.2%	**10.3%**	**97.8%**	0.82
Decision Tree	8.9%	16.0%	6.2%	96.4%	0.79

The dataset, and the relation between the predictors and the response variable, is too complex to be captured in a single decision tree. Therefore, ensemble methods like CatBoost and XGBoost are better suited, as seen in their performance on the evaluation dataset. Lastly, adding the support vector machine to the comparison is less straightforward. Its lack of calculated probabilities renders it difficult to make a one-on-one comparison to the other models. Nonetheless, the ROC curve and AUC show that SVM does not perform as well as CatBoost, XGBoost, and LR for this project.

5.2.2 Recommended classifier

According to their AUC interpretations, SVM, and DT perform at a reasonable level. However, these models are vastly outperformed by CatBoost, XGBoost, and LR. The ensemble methods of CatBoost and XGBoost are a more sophisticated approach than LR. They can make more accurate predictions than LR because they do not assume a linear relationship between predictors and the log-odds of the response variable. This results in better predictive capabilities, at a cost of interpretability. The data preparation of CatBoost is more straightforward as well, since it does not require one-hot encoding of categorical features. For LR, changes to the cardinality of a categorical feature would have to be simultaneously applied to the training and evaluation stage. This is because a change in the order of columns, or their names, would result in errors during evaluation. The CatBoost classifier does not have this problem, because the categorical values are changed into a number within the same column. If a new value is encountered during evaluation, it is assigned 0 by default. Therefore, the model with CatBoost as classifier is recommended to be used as the supplier complaints prediction model at VDL ETGA.

5.2.3 ML compared to realised performance

For every purchase order line in the evaluation dataset, both the true label and whether the items were inspected are known. This means that a confusion matrix can be constructed for the actual performance at VDL ETGA for the same period as used in the evaluation dataset. The confusion matrix for the actual results at VDL ETGA from February to April 2024 are shown in Table 5.8.

Table 5.8: Confusion matrix for actual results in February to April 2024

Complaint Inspected	0	1	Total
0	12,976	130	**13,106**
1	165	14	**179**
Total	**13,141**	**144**	**13,285**

In the period from February to April a total of 179 inspections were conducted at IQC, based on the current inspection policy. In 14 of those inspections the purchase order line contained defective items. It is possible that an existing defect was missed during inspection. For example, it is possible that items from a purchase order line were inspected, declared OK, but later turned out to contain defects caused by the supplier anyway. It is also possible that not every item within the purchase order line was inspected, because only a sample is inspected for purchase order lines with a large quantity of that item. This was the case in 4 defects in this period. The predictive model does not account for this possibility. So, to improve the fairness of the comparison, it does not matter here whether the associated defect was detected at IQC. The inspection model in Section 6.1.1 includes measures to help prevent detection errors in the future. The other 130 defects were detected by other departments at later stages in the process, resulting in additional costs.

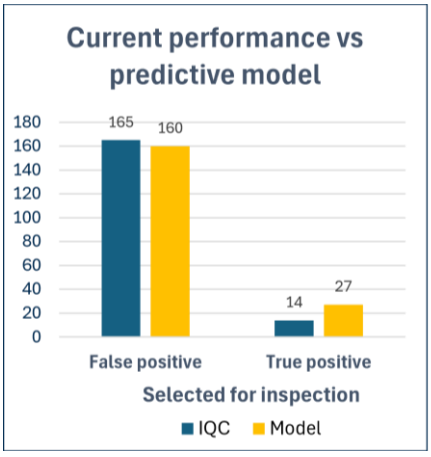
The confusion matrix allows us to calculate performance metrics for the actual results at VDL ETGA. This means that the performance metrics of the predictive model can be compared to the performance metrics currently realised at IQC. There were 64 working days from February to April 2024, holidays excluded. A total of 179 inspections averages to 2.8 inspections per day. For a proper comparison, the performance of the models must be compared to IQC at the same average of daily inspections. If only the purchase order lines classified as positive are inspected, then the total number of inspections is the sum of the true and false positives. This number is used to calculate the resulting average daily inspections for the predictive models.

Table 5.9 compares the actual performance at IQC to the predicted performance of the classifiers, both measured on the validation set. The thresholds of the classifiers are tuned to match the average of 2.8 inspections per day. The DT classifier returns identical probabilities for multiple samples. That is why it does not have the discriminative ability to select fewer than 16 lines for inspection per day. Other than that, the table shows that every classifier can outperform the realised performance. The CatBoost model performs the best.

Table 5.9: Comparison of the classifiers to IQC for the same number of daily inspections

Classifier	Threshold	Inspections	Sensitivity	Precision	F1 score
CatBoost	88%	2.8	18.8%	14.9%	16.6%
XGBoost	80%	2.9	13.7%	10.3%	11.7%
DT	83%	15.9	37.4%	5.1%	9.0%
LR	90%	2.8	10.1%	7.7%	8.8%
IQC	N/A	2.8	9.7%	7.8%	8.7%

Figure 5.5 shows how the results of the comparison between IQC and the CatBoost model should be interpreted. The realised sensitivity at IQC was $\frac{14\ defects\ found}{144\ total\ defects} = 9.7\%$ and the precision was $\frac{14\ succesful\ inspections}{179\ total\ inspections} = 7.8\%$. The F1 score is 8.7%. If, instead, the purchase order lines selected by the predictive model had been inspected, the performance metrics would be better. For the same average number of inspections, the predictive model can detect 18.8% of the defects with a precision of 14.9%. This means that 1 in 7 inspections would be successful, instead of 1 in 13. The F1 score increases from 8.7% to 16.6%.



	IQC	CatBoost
Sensitivity	9.7%	18.8%
Precision	7.8%	14.9%
F1 Score	8.7%	16.6%
Daily inspections	2.8	2.8

Figure 5.5: The actual performance compared to the predicted performance of the model for 2.8 inspections per day

As established in section 2.1, the average time consumption of an inspection at IQC is 1.5 hours. At an hourly rate of €72,- the costs are €108 per inspection. The estimated avoided costs of a defect detected at IQC are €3,000. This information is used to make an estimation of the potential savings if the predictive model is used. Through the model, 13 more defects are detected, while it selects 11 fewer purchase order lines for inspection in the period from February to April 2024. Table 5.10 shows that the estimated avoided costs from the model are €40,238 in those 64 working days. 2024 counts a total of 257 working days. So, anually, this would mean €161,579 of avoided costs.

Table 5.10: Estimation of the avoided costs based on the predicted performance from February to April 2024

Type	Occurrence	Cost per unit	Avoided Costs
Detections increase:	12	€ 3,000	€ 39,000
Inspections avoided:	11	€ 108	€ 1,238
		Total in period	**€ 40,238**
		Annually	**€ 161,579**

5.3 Inspection selection policy

Section 5.2.3 shows that the proposed model outperforms the current inspection method for the same average number of inspections per day. However, in the scripts currently used there is no way to vary the number of purchase order lines selected for inspection. So, this number of performed inspections is not necessarily optimal. The proposed model can be tuned to select any number of purchase order lines per day by changing the classification threshold. An important note here is that the predictive model still suffers from the imbalance in the class distribution, as can be seen in Figure 5.6. The figure shows, for each calculated probability bucket, the number of purchase order lines with and without a registered supplier complaint. For purchase order lines with calculated probabilities greater than 95%, they still only turn out to be defective in 15% of the cases. The figure does show that it is sensible to prioritise purchase order lines for inspections based on the calculated probability of defects. So, for example, if three inspections are to be performed each day, it should be the three purchase order lines with the highest calculated probability.

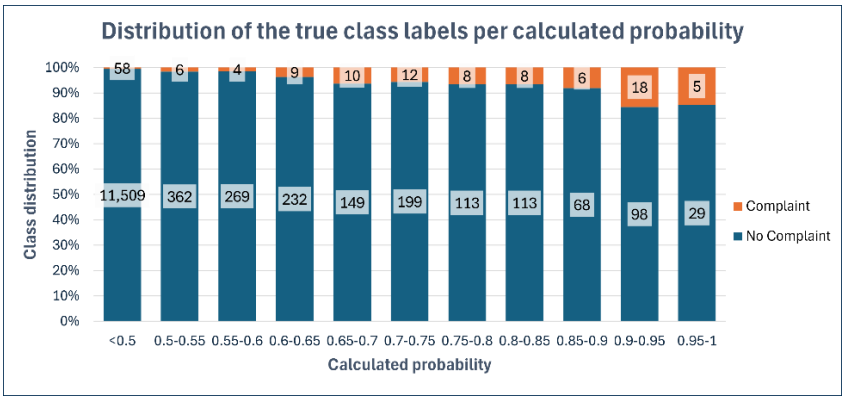


Figure 5.6: Distribution of true class labels per calculated probability by CatBoost on the evaluation dataset

As a result of the class imbalance, there are many false positives between the purchase order lines classified as defective for any chosen classification threshold. The chosen threshold has a large impact on the total number of purchase order lines that are selected for inspection. The trade-off between the total percentage of defects found and the required number of daily inspections is shown in Figure 5.7.

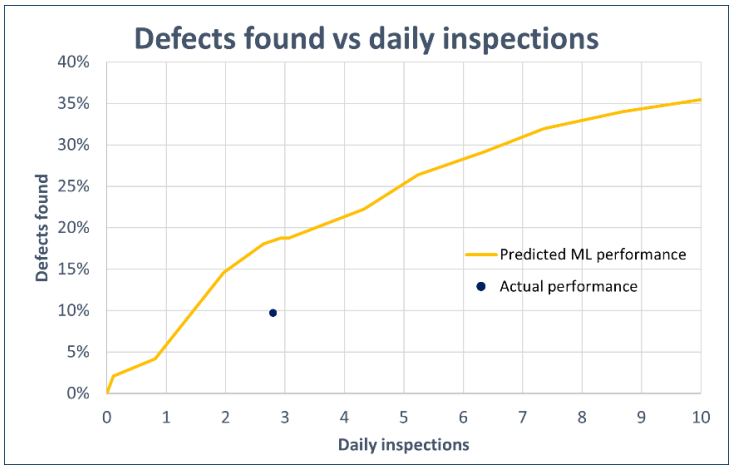


Figure 5.7: Trade-off between the percentage of defects found and the required number of inspections per day

Inspecting a non-defective item costs time but adds no value. This means that every false positive prediction costs €108. The expected savings of a true positive prediction are €3,000, because it results in a defect detection at IQC instead of at a later stage. The sensitivity on the evaluation dataset is used to calculate the error prevention percentage per number of daily inspections. Then, the total cost of

quality is calculated as the sum of the inspection cost and the error costs of defects slipping through IQC. Figure 5.8 shows that the minimum total cost of quality is achieved with 16 purchase order lines inspected per day. This results in annual quality costs of €1,366,567. A 0% inspection policy would result in annual quality costs of €1,735,000. A 100% inspection rate would result in annual quality costs of €6,000,000.

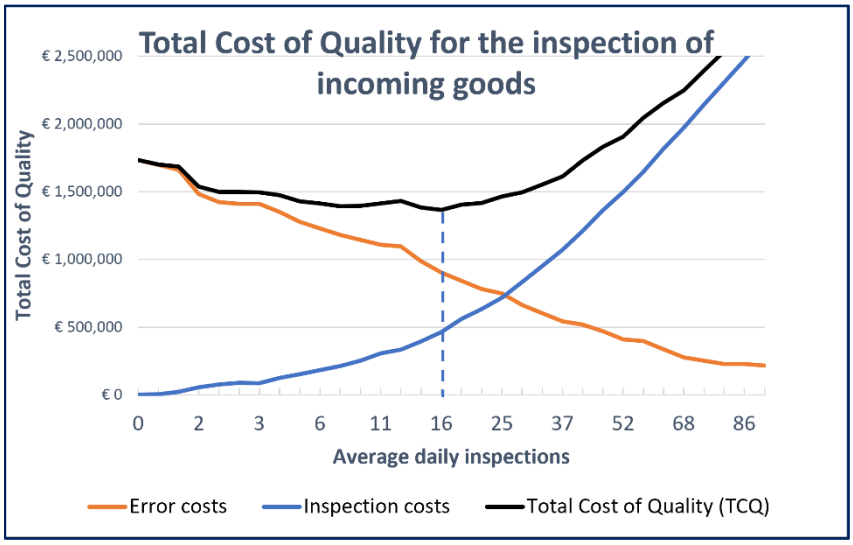


Figure 5.8: Total cost of Quality for the inspection of incoming goods (inspection cost: €108 and error cost: €3,000)

The actual performance from the evaluation set can be extrapolated to calculate the expected quality costs for 2024 if the old inspection strategy was used. There are 257 working days in 2024. 2.8 daily inspections sum up to a total of 720 inspections in 2024. For €108 per inspection this adds to a total of €77,760. The expected number of supplier complaints in 2024 is 578. The actual sensitivity was 9.7%, so 522 are missed by IQC. For €3,000 per missed complaint, this adds to a total of €1,566,000 error costs. So, the expected total costs of quality for IQC are €1,643,760. This means that the expected savings, if the optimal inspection strategy is used, are equal to €277,193 per year. At an average of 1.5 hours per inspection, 15 inspections would cost 22.5 hours each day. This is achievable by adding 2 more IQC employees if each employee inspects for 6 hours per day. This is more than the current capacity at IQC, so the best inspection strategy for direct implementation at VDL ETGA is to fill the IQC capacity with the top N purchase order lines ranked by calculated probability.

5.4 Summary

Cross validation is used to select the features and to tune the hyperparameters of each classifier. The evaluation dataset is used to test the final hyperparameters and features and compare the results to the actual performance. The improvements that have been achieved by optimising the CatBoost model are visualised in Appendix C. Especially the CatBoost and XGBoost models have improved significantly. The predictive models can effectively identify defective incoming items. CatBoost performs better than Logistic Regression, Decision Tree, Support Vector Machine, and XGBoost in this application. The best predictive model, CatBoost, significantly outperforms the current method used at VDL ETGA.

This means that the expected savings, if the optimal inspection strategy is used, can be up to €277,193 per year. To achieve the required number of inspections, the capacity of IQC must be expanded from 2 to 4 employees. The percentage of defective items found comes at a cost of false positives. The ratio between true and false positives largely depends on chosen cut-off probability for classification. For any number of performed inspections, priority should be given to the purchase order lines with the highest predicted probability.

6 Implementation

A predictive model has been developed that analyses incoming parts from suppliers and predicts which items are most likely defective. Now, it is deployed, and an inspection model is created that uses the predictions as input. This chapter explains how the model should be implemented at VDL ETGA. The functional deployment is discussed in section 6.1, and the technical deployment is discussed in section 6.2. The first inspections based on the model have been conducted in real life. The results of the inspections are shown in section 6.3. The corresponding research question to this chapter is as follows:

How should the recommended inspection model be implemented into the Incoming Quality Control processes at VDL ETGA?

6.1 Functional deployment

The functional deployment of the model consists of an explanation of the inspection model, the integration into the existing IQC processes, and the functional maintenance that is required to maintain the performance of the model. The following subsections treat these topics, respectively.

6.1.1 Inspection model

The purchase order lines that are scheduled for arrival in the upcoming period are input into the trained ML model. All purchase order lines have an expected week of delivery, but the exact day varies. The supplier is allowed to deliver on a day in the expected week that suits them best. Therefore, the upcoming period includes purchase order lines that are scheduled for delivery up to two weeks ahead in time. The trained ML model calculates a defect probability for each purchase order line. The purchase order lines are ranked based on this calculated probability. For now, in consultation with the supplier quality manager and incoming quality control department, the top 30 items are selected for inspection. This averages to 3 inspections per day and allows for a representative performance comparison in real life to historical performance.

The qualitative information of previous complaints is analysed for each unique item in the upcoming deliveries. The most common defect type for the supplier of the purchase order line is also added. Additionally, for every selected purchase order line, the feature with the highest influence on the prediction is added. This is particularly useful for purchase order lines with a high calculated probability but no previous complaints for that item. In such cases, the feature explains whether, for example, the supplier or the item price is the main reason the purchase order line is classified as defective. The added information provides IQC with specific instructions on what type of defect to look for during the inspection.

The inclusion of information regarding previous defects and the identification of the most important feature for each selected purchase order line improve the explainability of the model. This allows the supplier quality engineer to compare their experiences and expectations of item and supplier quality to the predictions of the model.

As established in section 2.1.3, the IQC employees cannot check for functional errors in the items. This makes no difference to the defect probability predictions, but it is relevant for the selection of purchase order lines for IQC. If the qualitative info from earlier complaints suggests that the expected failure is a functional failure, the item will not be flagged for an IQC inspection by the supplier quality department.

IQC has started using Acceptable Quality Limit (AQL) sampling for purchase order lines. AQL is the poorest level of quality from a supplier's process that would be considered acceptable (Theisens, 2021).

The AQL table provides an inspection sample size for every purchase order line that is selected. If more than the acceptable number of defects are found in the sample, then the entire batch must be inspected (100% inspection). For example, an AQL of 1% on a sample size of 100 means that at most 1 defective item within the sample is acceptable. In consultation with the supplier quality manager and IQC department, it was decided to include AQL sampling in the inspection model as well. Figure 6.1 shows the flow chart of the recommended inspection model for VDL ETGA.

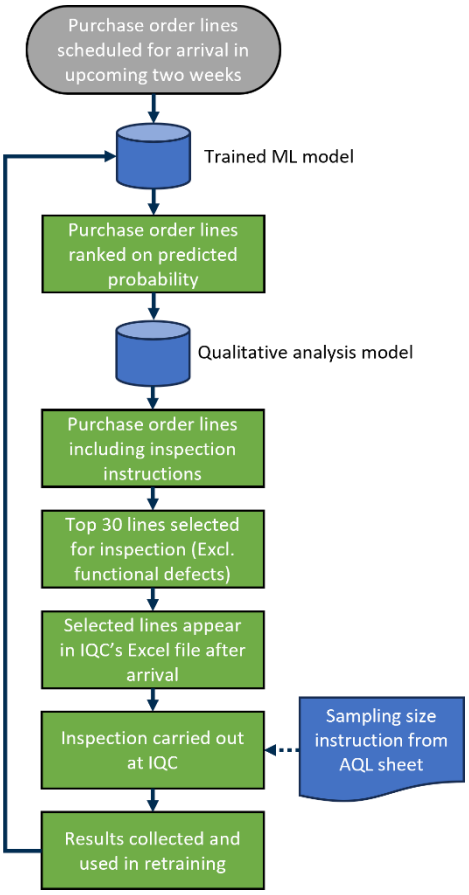


Figure 6.1: Recommended inspection model for VDL ETGA

The purchase order lines that are selected for inspection by the predictive model are manually flagged in Baan IV. Together with VDL ETGA, the decision has been made to keep this process manual for now, because of an ERP transition to Infor LN in 2025. The Infor LN project team expects that this process can easily be automated in Infor LN. A note is added to the purchase order line so that the project team can keep track of the results of inspections that are performed because the model selected it. See Figure 6.2 for what the flagging of purchase order lines and adding the instructions look like in Baan IV.

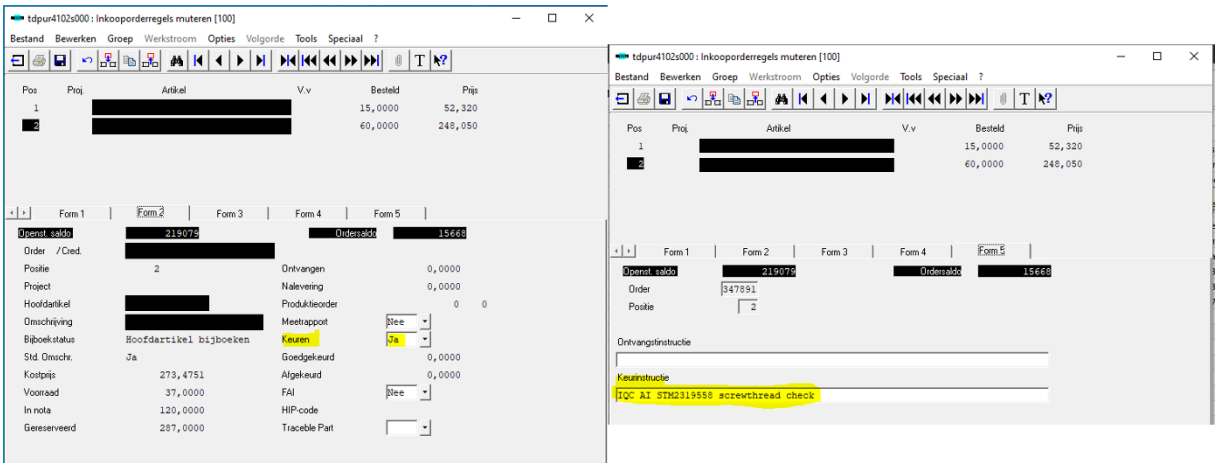


Figure 6.2: Instructions added to purchase order lines selected for inspection

6.1.2 Functional maintenance

The Supplier Quality Manager of Systems-1 is appointed as functional owner of the model. The performance of the model can be monitored in a Power BI dashboard. VDL ETGA already uses Power BI as their dashboarding solution. The relevant performance metrics should be used as KPIs in the dashboard. The dashboard is used to evaluate whether the model is still performing well, or whether the optimisation algorithms should be used again to restore the performance of the predictive model. The KPIs to be tracked in the dashboard are the same as in the performance analysis in chapter 5, namely: sensitivity, precision, accuracy, and F1-score. These should be visualized in current performance indicators and as trend over time. The performance of IQC be tracked in existing quality dashboards at VDL ETGA. Therefore, the development of a dedicated dashboard is out of scope for this project.

6.2 Technical deployment

This section describes the technical aspects of the model deployment and implementation at VDL ETGA. In chapter 5 it was concluded that the model with CatBoost as classifier performs the best at predicting supplier defects. Therefore, the CatBoost model is deployed to be used in the inspection model. First, the data flow is simplified in preparation of the technical deployment. Then, the steps taken to automate the process are outlined. Last, the technical maintenance of the model is discussed.

6.2.1 Automation

The model requires input data to work. Both in the training stage and during the prediction process. For the deployment of the model the data input process is simplified by improvements in the SQL query to iQBS. All the required columns are added to one SQL query so that the number of data sources for the model is reduced from two to one. The data preparation steps are all merged into one Python file. This ensures that a minimal number of Python files must be maintained by VDL ETGA after implementation.

The input files in CSV-format, which the predictive model uses, are replaced with a direct SQL query to the iQBS database. The connection is created with a Python module called pyodbc. Pyodbc is an open-source Python module to access ODBC databases (PyPi, 2024). This change in the input source ensures that the input files do not require a manual refresh, thereby avoiding a tedious and repetitive task for a VDL ETGA employee.

The Python file is then turned into an application. PyInstaller bundles a Python application and its dependencies into a single executable package. This allows any user to run the code without the need to install and open a Python interpreter (Cortesi, 2024). The python code refers to a separate text file

for settings and parameter values. This makes it possible to change settings of the ML model without the need to redo the PyInstaller process. The process of turning the python file into an executable file is run through the command prompt with the command 'pyinstaller ML_Model.py –onefile', as seen in Figure 6.3.

Figure 6.3: The process of turning a Python file into an executable file

The periodic execution of the ML application is set up on a windows server environment. This way the prediction process does not depend on the computer of an employee being powered on. On the server, the ML model exe file is automatically executed via task scheduler. The output of the ML model is an excel file with the purchase order lines that are selected for inspection in the upcoming four weeks period. The items include an instruction based on historical defects on that item.

Figure 6.4: The automatic execution of the ML model on a windows server

The ML model is configured to automatically update the timeline of retraining data. The training data are all purchase order lines from three years ago up to two months ago. An important note to determining the range of dates used for retraining is that most positive labels are only known months after delivery. It is therefore not recommended to train and evaluate performance on too recent data, because these labels are not yet representative of the actual results.

6.2.2 Technical maintenance

The technical ownership of the model lies with the Supply Chain Management Digitalisation department of VDL ETGA. They are responsible for maintaining the files and implementing improvements, if necessary. The Python code is uploaded to Bitbucket. Bitbucket is a Git-based code and continuous integration and delivery (CI/CD) tool optimised for teams using Jira, which is in use at VDL ETGA (Bitbucket, 2024). VDL ETGA also already uses Bitbucket to store coding files and perform coding reviews on code written for the Machining department. Bitbucket supports workflows for safe code development and keeps track of the version history. So, if any changes break the code, there is always the option to return to an earlier working version.

6.3 Preliminary results of the implementation

As of writing this on 30 July 2024, 30 purchase order lines have been selected for inspection by the predictive model. From those, the first 24 purchase order lines have been delivered and inspected. 6 of the inspections resulted in the detection of defective items. The average daily inspections are the same as before at 3 per day. Table 6.1 shows the complaints that were registered after inspecting the selected purchase order lines. The precision of the inspections is 25%. This precision equates to a success rate of 1 in 4 inspections. The observed precision significantly exceeds the anticipated performance of 14%. Given that the dataset consists of only 30 inspections, this increase is likely coincidental. Therefore, an adjustment in precision is expected in the near future. It is too early to calculate a sensitivity, because the total number of supplier defects from this period is not yet known.

Table 6.1: True positive purchase order lines that were selected for inspection by the predictive model

Order number and Position	Inspection instruction	Complaint number	Complaint trigger	Problem description
346460-004	Check for contamination	2418902	Contamination	Contamination/Surface treatment not ok.
347891-002	Check the screw threads	2417637	Contamination	Visible stains on EXE top frame and they are not cleanable.
348532-008	Check the screw threads	2417006	Dimensional error	The calibre does not fit in the holes, they are too small.
348538-007	Check for contamination	2418056	Contamination	Discoloration on the surface of the product.
349935-002	Check the screw threads	2418772	Screw thread	The calibre will not go in completely through. 6 out of 32 pieces have the NOK thread.
349935-003	Check the screw threads	2418776	Screw thread	Gauge will not fit completely through. Holes too small.

The experiences from the preliminary results after implementation show that there are certain risks that could compromise future success. Three primary risks have been identified. First, there is a potential risk that at some point no person manually flags purchase lines in Baan anymore. This risk is mitigated by the appointment of a functional owner and by ensuring support from multiple involved departments, namely from the departments supply chain, IQC, quality and purchasing. This risk can be eliminated by automating the flagging process in Infor LN after the transition. Second, there is a possibility that the number of items flagged for inspection may exceed the available capacity. This risk is addressed by including an adjustable classification threshold in the configuration file of the ML model. Increasing the threshold results in fewer inspections and vice versa. Last, unexpected errors or bugs in the code may arise. This risk is mitigated by assigning a technical owner and by providing technical instructions for the model. The technical owner can use this to perform maintenance on the model.

6.4 Summary

The inspection model has been successfully implemented at VDL ETGA. The results prove that the model is capable of predicting defective incoming goods, and it performs better than the old empirical selection method. The prediction process is fully automated. A member of the supplier quality department then manually flags the purchase order lines in Baan IV that are selected for inspection. The flagging of purchase order lines in the ERP system can be automated once Infor LN is in use. The functional ownership of the model lies with the supplier quality manager, and the technical ownership lies with the supply chain digitalisation engineer.

7 Conclusions and Recommendations

This chapter consists of the conclusions, recommendations, and limitations that are derived from the execution of the project. The conclusions and recommendations are formulated point by point. First, Section 7.1 sums up the conclusions of the project. Then, section 7.2 lists the recommendations that are made to VDL ETGA. Last, section 7.3 explains the limitations of this research project.

7.1 Conclusions

Before this project, it was unknown whether supplier defects could be predicted, and whether these predictions were useful for incoming quality control. This led to the main research question, that reads as follows:

How can defects among incoming items be predicted, and how should VDL ETG Almelo use this information in their inspection policy to improve the IQC detection performance?

The findings from all previous chapters have allowed us to answer the main research question of this project. Both the performance on the evaluation set and the implementation have proved that machine learning models based on historical quality data can predict defective items among incoming goods. The preliminary results after implementation show that the new inspection model improves the precision of IQC from 7.8% to 25% It is too early to tell the actual difference in sensitivity and F1 score, but the expectation from the evaluation dataset is that the sensitivity and F1 score increase from 9.7% and 8.7% to 18.8% and 16.0%, respectively. This means that relatively more defects are found at IQC, while performing fewer unsuccessful inspections.

The models can predict supplier defects, but the choice of classifier is non-trivial. There is a large gap in performance between the classifiers. The CatBoost model is the best performing model, due to its specific design for categorical features. In similar use cases, but with fewer categorical features, the XGBoost model might perform better. The CatBoost, XGBoost, and Logistic Regression perform much better than the basic Decision Tree and Linear Support Vector Machine models.

The primary practical contribution of this model is that up to €277,000 of quality costs can be saved annually if the optimal inspection policy from the model is used. The second practical contribution is that, instead of the current laborious empirical selection, the model is dynamic and keeps up with the current trends in supplier quality without human interference. The automation of repetitive and uncreative tasks increases the available time for creative tasks. The responsible VDL ETGA employees now have more time to spend on tasks that improve future performance.

The comparable studies identified during the literature search in chapter 3 focused on a limited number of internal items, utilizing in-process product measurement data and primarily numerical features. As explained in Chapter 2, this data is not available for incoming goods at VDL ETGA. This posed a unique challenge in predicting supplier quality due to the wider variety of items and the more limited availability of data. This thesis successfully developed a predictive model for supplier defects, directly addressing this challenge. The model not only enhances the practical approach to quality control at VDL ETGA but also contributes to the theoretical understanding of quality control in supply chain management.

7.2 Recommendations

This section treats several topics that are beyond the scope of this project but should still be considered in the future. The topics are formulated as recommendations to VDL ETGA in the following paragraphs.

The scope of this project is demarcated to the Systems-1 workflow. This workflow covers the most important products of VDL ETGA and demands the largest capacity utilization from IQC. The cost calculations in chapter 5 show that it would be beneficial to increase the capacity at IQC. VDL ETGA should therefore investigate how they can improve the utilization of the IQC department. Also, there are items from workflows Systems-2, Parts, and Projects that are inspected by IQC. More quality costs can be avoided if the prediction and inspection models are expanded to other workflows.

The implementation could not be fully automated due to limitations of the current ERP system Baan IV. The new ERP system Infor LN works with APIs, that can be used to automatically change parameters within Infor LN. This can be used to automate the flagging of purchase order lines for inspection. Doing this would mean that the implementation is fully automated.

VDL ETGA does not create purchase order lines before delivery for items that are resupplied through Vendor Managed Inventory (VMI). Purchase order lines are created upon delivery for these items. This means that the model is trained to predict defects among VMI items, but it cannot be used to select those items for inspection because their purchase order lines do not yet exist. VDL ETGA should start using some form of (dummy) purchase orders for VMI items including expected delivery dates. It would then become possible to inspect VMI items.

Only 7% of the supplier complaints have a filled in purchase order. That means that for 93% of the supplier complaints a purchase order must be approximated. It is possible that complaints are assigned to the wrong purchase order, and this introduces avoidable uncertainty into the model. VDL ETGA should mandate the registration of purchase order and position in REM.Net for supplier defects.

The quality predictions can be used in quality assurance as well. If integrated upstream, the predictions could be used before the products reach the conventional inspection process (Schmitt et al., 2020). For example, within the suppliers' processes, and thus avoiding supplier defects reaching VDL ETGA altogether.

7.3 Limitations

The performance of a predictive model largely depends on the input data. For the prediction of defects among incoming goods at VDL ETGA, only historic data can be used. Not all factors that contribute to new defects are captured in the information extracted from historical data. For example, changes to a process may affect only new batches, rendering information from old results useless. Furthermore, most supplier defects, and therefore the ML response labels, are only detected months after delivery. This results in an equal delay to the moment when purchase order lines become available for usage in the training of the model. These two points mean that the ML model for IQC will always be reactive. This is a disadvantage to in-process quality control, where process data that can be collected by an organisation internally and in real time. There is no solution to this disadvantage for this project; it is unreasonable and too drastic to request internal process data from all 322 suppliers.

The technical deployment could not be fully automated because of the upcoming transition to the new ERP system. Infor LN should offer much better support for automated processes than the outdated Baan IV is capable of. However, the potential of Infor LN could not be tested during this project. The timing of this project is slightly awkward in that regard.

Bibliography

Al-Rahman, N. (2021). *Undersampling and oversampling: An old and a new approach*. Analytics Vidhya. https://medium.com/analytics-vidhya/undersampling-and-oversampling-an-old-and-a-new-approach-4f984a0e8392

Bhatt, U., Xiang, A., Sharma, S., Weller, A., Taly, A., Jia, Y., Ghosh, J., Puri, R., Moura, J. M. F., & Eckersley, P. (2020). Explainable machine learning in deployment. *FAT* 2020 - Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, 648–657. https://doi.org/10.1145/3351095.3375624

Bitbucket. (2024). https://bitbucket.org/

Canciglierie, A. B., da Rocha, T., Szejka, A. L., dos Santos Coelho, L., & Junior, O. C. (2021). Real-Time Machine Learning Automation Applied to Failure Prediction in Automakers Supplier Manufacturing System. *IFIP Advances in Information and Communication Technology*, *630*, 303–310. https://doi.org/10.1007/978-3-030-85874-2_32

CAQ AG Factory Systems. (n.d.). *REM.Net Complaint Management System*. Https://Www.Caq.de/En/Complaint-Management-System.

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *13-17-August-2016*, 785–794. https://doi.org/10.1145/2939672.2939785

Cortesi, D. (2024). *PyInstaller Manual*. https://pyinstaller.org/en/stable/

Dieperink, A. J. (2023). *Reducing the cycle time for replacements (containment) Systems 1 - VDL Enabling Technologies Group Almelo* [Bachelor Thesis]. Hogeschool Saxion.

Filz, M. A., Herrmann, C., & Thiede, S. (2020). Simulation-based assessment of quality inspection strategies on manufacturing systems. *Procedia CIRP*, *93*, 777–782. https://doi.org/10.1016/J.PROCIR.2020.04.069

García, S., Luengo, J., & Herrera, F. (2015). *Data Preprocessing in Data Mining*. Springer International Publishing Switzerland.

Heerkens, H., & van Winden, A. (2021). *Solving Managerial Problems Systematically*. Noordhoff Uitgevers BV.

Heymann, H., Kies, A. D., Frye, M., Schmitt, R. H., & Boza, A. (2022). Guideline for Deployment of Machine Learning Models for Predictive Quality in Production. *Procedia CIRP*, *107*, 815–820. https://doi.org/10.1016/j.procir.2022.05.068

James, G., Witten, D., Hastie, T., Tibshirani, R., & Taylor, J. (2023). *An Introduction to Statistical Learning*. https://doi.org/10.1007/978-3-031-38747-0

Kavikondala, A., Muppalla, V., Krishna Prakasha, K., & Acharya, V. (2019). Automated retraining of machine learning models. *International Journal of Innovative Technology and Exploring Engineering*, *8*(12), 445–452. https://doi.org/10.35940/ijitee.L3322.1081219

Kiran, D. R. (2017). *Total Quality Management: Key Concepts and Case Studies*. Elsevier Inc.

Kohl, M. (2012). Performance Measures in Binary Classification. *International Journal of Statistics in Medical Research*, *1*(1), 79–81. https://doi.org/10.6000/1929-6029.2012.01.01.08

Krauß, J., Pacheco, B. M., Zang, H. M., & Schmitt, R. H. (2020). Automated machine learning for predictive quality in production. *Procedia CIRP*, *93*, 443–448. https://doi.org/10.1016/j.procir.2020.04.039

Li, J., Zhang, H., Wang, Y., & Cui, H. (2020). A Review of the Applications of Data Mining for Semiconductor Quality Control. *Lecture Notes in Electrical Engineering*, *628 LNEE*, 486–492. https://doi.org/10.1007/978-981-15-4163-6_58/TABLES/2

López, V., Fernández, A., García, S., Palade, V., & Herrera, F. (2013). An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, *250*, 113–141. https://doi.org/10.1016/J.INS.2013.07.007

Luengo, J., García, S., & Herrera, F. (2012). On the choice of the best imputation methods for missing values considering three groups of classification methods. *Knowledge and Information Systems*, *32*(1), 77–108. https://doi.org/10.1007/S10115-011-0424-2/METRICS

Monczka, R. M., Handfield, R. B., Giunipero, L. C., & Patterson, J. L. (2009). *Purchasing & Supply Chain Management*. South-Western Cengage Learning.

Patro, R. (2021). *Cross-Validation: K Fold vs Monte Carlo*. Towards Data Science. https://towardsdatascience.com/cross-validation-k-fold-vs-monte-carlo-e54df2fc179b

Pedregosa, F., Michel, V., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Vanderplas, J., Cournapeau, D., Pedregosa, F., Varoquaux, G., Gramfort, A., Thirion, B., Grisel, O., Dubourg, V., Passos, A., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*(85), 2825–2830. http://jmlr.org/papers/v12/pedregosa11a.html

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2017). CatBoost: unbiased boosting with categorical features. *Advances in Neural Information Processing Systems*, *2018-December*, 6638–6648. https://arxiv.org/abs/1706.09516v5

PyPi. (2024). *pyodbc*. https://pypi.org/project/pyodbc/

Python Software Foundation. (2024). About Python. https://www.python.org/about/

Quinlan, J. R. (1986). Induction of Decision Trees. *Machine Learning*, *1*(1), 81–106. https://doi.org/10.1023/A:1022643204877/METRICS

Sarkar, D., Bali, R., & Sharma, T. (2018). Practical Machine Learning with Python. *Practical Machine Learning with Python*. https://doi.org/10.1007/978-1-4842-3207-1

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. https://doi.org/10.1016/j.aei.2020.101101

Slack, N., Brandon-Jones, A., & Johnston, R. (2016). *Operations Management*. Pearson Education Limited.

Suresh, G., Myilvaganan, R., & Anand, J. (2022). *Operations Management*. Thakur Publication Pvt. Ltd.

Suvrit Sra, Sebastian Nowozin, & Stephen J. Wright. (2011). *Optimization for Machine Learning*. https://mitpress.mit.edu/books/optimization-machine-learning

Theisens, H. C. (2021). *Lean Six Sigma green belt : mindset, skill set and tool set*. Lean Six Sigma Academy.

Tong, P., Lu, J., & Yun, K. (2018). Fault detection for semiconductor quality control based on Spark using data mining technology. *Proceedings of the 30th Chinese Control and Decision Conference, CCDC 2018*, 4372–4377. https://doi.org/10.1109/CCDC.2018.8407886

VDL ETG. (n.d.). *VDL ETG Almelo*. Https://Www.Vdletg.Com/En/Het-Bedrijf/Locations/Almelo.

VDL Groep. (2024, April 5). *About VDL Groep*. Https://Www.Vdlgroep.Com/Nl/Nieuws/2023-Jaar-van-Uitersten-Voor-Vdl-Groep.

Verna, E., Genta, G., Galetto, M., & Franceschini, F. (2021). Inspection planning by defect prediction models and inspection strategy maps. *Production Engineering*, *15*(6), 897–915. https://doi.org/10.1007/s11740-021-01067-x

Yorulmuş, M. H., Bolat, H. B., & Bahadır, Ç. (2022). Predictive Quality Defect Detection Using Machine Learning Algorithms: A Case Study from Automobile Industry. *Lecture Notes in Networks and Systems*, *308*, 263–270. https://doi.org/10.1007/978-3-030-85577-2_31

Zhou, Z. H. (2021). Machine Learning. *Machine Learning*, 1–458. https://doi.org/10.1007/978-981-15-1967-3/COVER

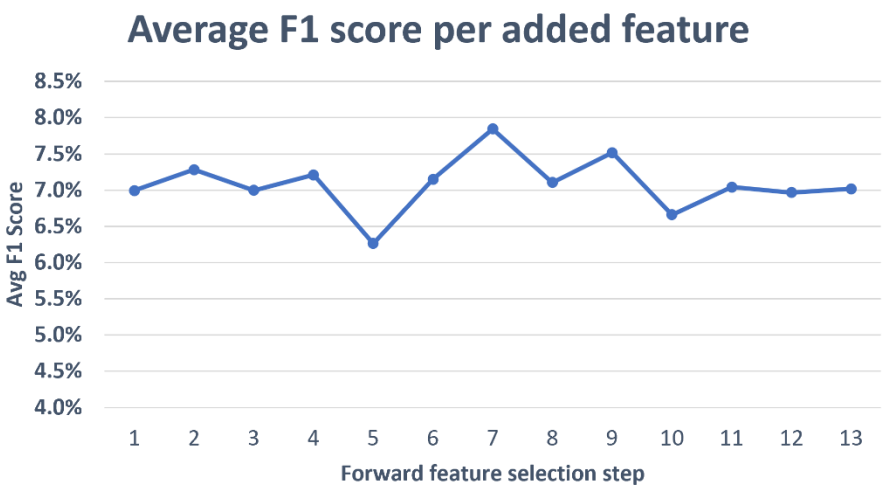
Appendix A: Forward stepwise feature selection algorithm

Forward stepwise feature selection has also been applied to the predictive model. This version is a slight adjustment to the backward stepwise feature selection algorithm. Instead of starting with a full model, it starts with an empty model and adds features one by one. The new algorithm is shown below.

ALGORITHM 2: FORWARD STEPWISE FEATURE SELECTION WITH CROSS VALIDATION

Input: *Purchasing data with all potentially relevant predictors.*
Output: *DataFrame with all predictors sorted in order of performance improvement.*
Let M_0 denote the CatBoost model that contains no predictors.
Split purchasing data for k fold cross validation
predictors = list of all potential predictors
remaining_predictors = predictors
For *j = 1,2,…,predictors:*
    For *each predictor in remaining_predictors:*
        $M_{temp} = M_{j-1} + predictor$
        For *k =1,..,k:*
            Train M_{temp} on folds 1 to k
            Test M_{temp} model on fold k+1
        Store average performance of M_{temp}
    Find M_{temp} with highest F1-score and call it M_j.
    Store M_j and its F1-score in a list
    Remove new predictor in M_j from remaining_predictors
Select a single best model from $M_1, M_2, \ldots, M_p$
End

The results of the average F1 scores of the 5-fold cross validated forward stepwise feature selection are shown in Figure A.1. As seen, the forward stepwise feature selection provides more unstable results than the backward feature selection in section 5.1.1. This is because forward feature selection starts with an empty model and adds one predictor at a time. Early models with few features can be highly sensitive to the addition of each new feature. This leads to significant fluctuations in performance. The initial instability spreads through the entire selection process, making the overall results less stable as well. That is why it was decided to use the selected features from backward stepwise feature selection in the final model.



Step	Added predictor
1	Item number
2	Delivery week
3	Cost component
4	Supplier country
5	Price
6	End product
7	Item type
8	Commodity
9	Product type
10	Ordered quantity
11	HIP code
12	Item group
13	Supplier

Figure A. 1: Average F1 scores of cross validated forward stepwise feature selection

Appendix B: Shap values per predictor

The influence of a predictor on the output of the model can be explained by its SHAP value. SHAP stands for Shapley Additive Explanations. A negative SHAP value means that the feature contributes to a predicted small probability, and a positive SHAP value means that the predictor contributes to a predicted high probability of defects. Figure C.1 shows the SHAP values for the features that resulted in the best F1 score. The plot shows that purchase order lines with a small price have a high spread, but relatively expensive items have a large positive influence on the predicted probability. The supplier feature has a wide variation, indicating that the specific supplier of an item has a substantial impact on the predicted probability. The item code SHAP value shows that specific items result in high predicted probabilities. For other items it has a smaller effect.

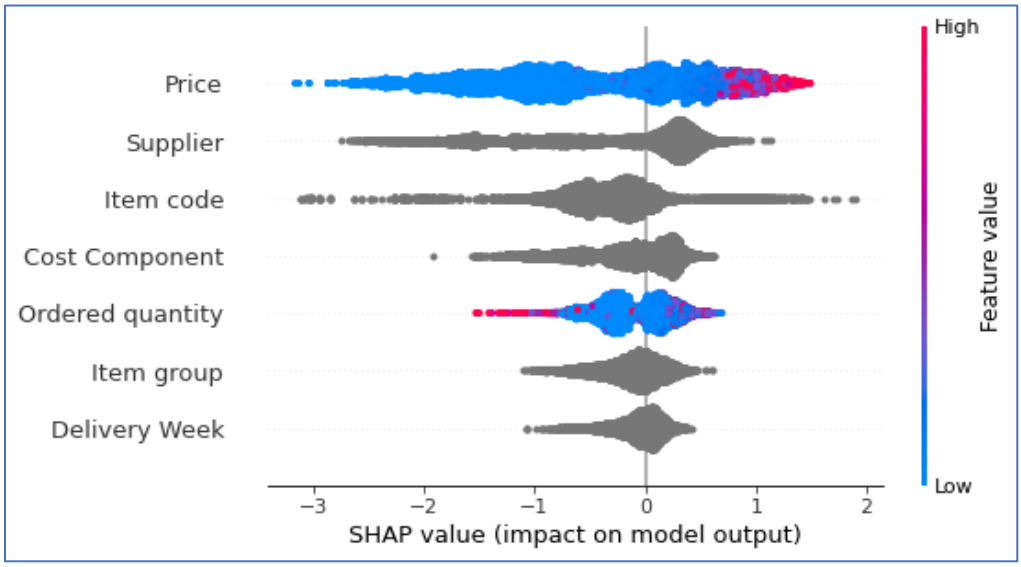


Figure B. 1: SHAP values calculated with the CatBoost model

Appendix C: CatBoost ROC improvements after optimisation

The final CatBoost model resulted in the best performance for this thesis study, but especially the hyperparameter tuning was crucial to achieve this result. Figure C.1 shows how the optimisation techniques of feature selection and hyperparameter tuning improved the performance of the CatBoost model, which increased from an ROC AUC of 0.79 as initial performance to an AUC of 0.84 as final performance.

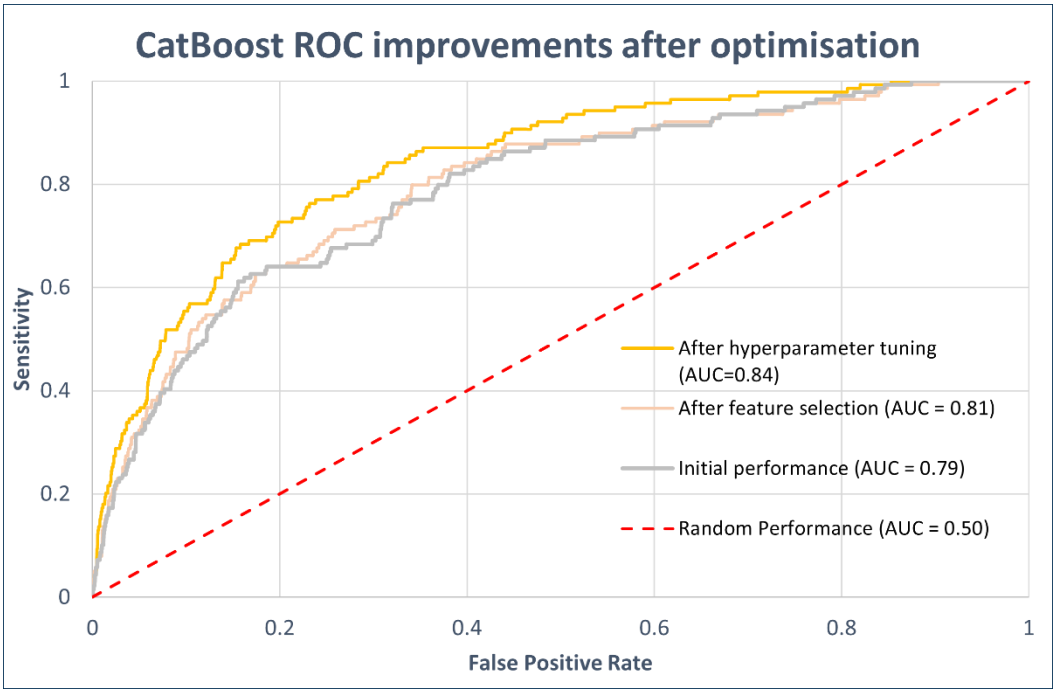


Figure C. 1 ROC improvements of the CatBoost model after optimisation