D5.8 – Rapid Quality Diagnosis v2

WP5 – BUILD: Rapid Manufacturing Line Qualification and Reconfiguration

i4Q has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 958205.

Document Information

GRANT AGREEMENT NUMBER	958205	ACRONYM	i4Q
FULL TITLE	Industrial Data Services for Quality Control in Smart Manufacturing		
START DATE	01-01-2021	DURATION	36 months
PROJECT URL	https://www.i4q-project.eu/		
DELIVERABLE	D5.8 – i4Q Rapid Quality Diagnosis		
WORK PACKAGE	WP5 – BUILD: Rapid Manufacturing Line Qualification and Reconfiguration		
DATE OF DELIVERY	CONTRACTUAL 31-Dec-2022	ACTUAL	30-Dec-2022
NATURE	Report	DISSEMINATION LEVEL	Public
LEAD BENEFICIARY	CERTH		
RESPONSIBLE AUTHOR	Spyridon Paraschos, Myrsini Ntemi, Georgia Apostolou, Ilias Gialampoukidis (CERTH)		
CONTRIBUTIONS FROM	7-IKER		
TARGET AUDIENCE	1) i4Q Project partners; 2) industrial community; 3) other H2020 funded projects; 4) scientific community		
DELIVERABLE CONTEXT/ DEPENDENCIES	This document extends the implementation status of the Rapid Quality Diagnosis solution (i4Q^{QD}) as presented in *D5.2 i4Q Rapid Quality Diagnosis*.		
EXTERNAL ANNEXES/ SUPPORTING DOCUMENTS	None		
READING NOTES	None		
ABSTRACT	The deliverable D5.8 *Rapid Quality Diagnosis* v2 is a Technical Specification document which provides a thorough description of the updates made during the second implementation phase of the i4Q Rapid Quality Diagnosis solution. A thorough explanation of the strategic planning used to address the rest of the pilot requirements is provided in this deliverable. The description of the problem-solving procedure is presented in detail, starting with the close examination of the available data, the analysis of the methodologies employed for the resolution of the task and finally the evaluation results deriving from the process. The outcomes of the experimentations conducted on the provided industrial pilot datasets support the effectiveness of the Rapid Quality Diagnosis framework.		

Document History

VERSION	ISSUE DATE	STAGE	DESCRIPTION	CONTRIBUTOR
0.1	07-Nov-2022	ToC	First Version of Table of Contents	CERTH
0.2	15-Nov-2022	ToC	Second Version of Table of Contents send to the consortium	CERTH
0.3	26-Nov-2022	Draft	First Draft of the Deliverable to be sent for internal review	CERTH, IKER
0.4	02-Dec-2022	Internal Review	Internal Review	TUB, CESI
0.5	08-Dec-2022	Draft	Addressing comments from internal reviewers	CERTH, IKER
1.0	13/12/2022	Final Document	Final quality check and issue of final document	CERTH

Disclaimer

Copyright message

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS/ACRONYMS

AI	Artificial Intelligence
DAS	Data Acquisition Software
DIT	Data Integration and Transformation
DR	Data Repository
GAN	Generative Adversarial Network
i4Q_QD	i4Q Rapid Quality Diagnosis
ID	Identification
LightGBM	Light Gradient Boosting Machine
ML	Machine Learning
MRMR	minimum Redundancy – Maximum Relevance
NN	Neural Network
QD	Quality Diagnosis
RF	Random Forest
ROC	Receiver Operating Characteristic
SMOTE	Synthetic Minority Over-sampling Technique
UI	User Interface

Executive summary

Deliverable **D5.8** *Rapid Quality Diagnosis v2* is an updated version of **D5.2** which presents the technical advancements of the i4Q Rapid Quality Diagnosis solution (i4Q^{QD}). Specifically, it is a Technical Specification document which showcases an in-depth description and analysis of the i4Q^{QD} algorithms that have been developed during the second implementation phase of the solution. Firstly, a summarization of the pilot requirements that have been addressed is provided, followed by the development breakdown of the delivered algorithms. The structure of the implementation presentations consists of:

- **A description of the task at hand and the related pilot data.**
- **A thorough explanation of the methodologies employed to guide the development of the algorithm.**
- **Quantitative results that present a summarization of the conducted experiments and evaluation metrics of the proposed method.**
- **Qualitative results that showcase insightful visualizations regarding the algorithm's performance.**

Additionally, an introduction to user interface that will host the solutions developed by CERTH is presented, describing the technologies that were employed and offering a glimpse into the data visualization and user interaction capabilities of the platform.

This document **i4Q** D5.8 v2 is an update of v1 of D5.2, for this reason it contains information of the 1st version together with the updates developed in this 2nd version.

Document structure

Section 1: Contains the technical specifications of the **i4Q Rapid Quality Diagnosis**, including an overview and an architecture diagram. It is geared for software developers.

Section 2: Details the updates made in the implementation status of the **i4Q Rapid Quality Diagnosis**, describing the development process of the algorithms and the design of a user interface.

Section 3: Provides the conclusions.

APPENDIX I: Provides the PDF version of the **i4Q Rapid Quality Diagnosis** web documentation, which can be accessed online at: **http://i4q.upv.es/18_i4Q_QD/index.html**

1. Technical Specifications

1.1 Overview

The Rapid Quality Diagnosis (i4Q^{QD}) is assigned to the Platform Tier's Monitor and Diagnostics sub-component. This solution offers primarily rapid product quality diagnosis and smart alerts when defective products are identified. It is a micro-service that provides very precise monitoring of the product quality throughout the production process. In addition, it offers configurations of crucial parameters that may cause product faults, as well as suggestions for their rapid reconfiguration. It detects product quality defects and generates alerts to notify human operators or other i4Q analytical solutions so that the necessary corrective actions can be conducted prior to the occurrence of permanent product failures.

1.2 Architecture Diagram

The i4Q^{QD} solution comprises of processes, ML and AI models that map to the Platform and Edge Tiers of the i4Q Reference Architecture (Deliverable 2.7). The following are the strengths of the i4Q^{QD} solution in relation to the aforementioned mapping:

Strengths:

- **Platform Tier:** The i4Q^{QD} mapping to the "Data Brokering and Storage" sub-component guarantees that this solution has access to a vast array of manufacturing data. The abundant data sources allow for the optimization, fine-tuning, and generalization of predictive models, hence boosting the reliability of the product quality diagnosis system. If necessary, the analysis results from the i4Q^{QD} solution can be stored in the i4Q^{DR} solution. The mapping of i4Q^{QD} to the "Data Analytics and Services" subcomponent enables the solution to offer product quality diagnosis reports to other analytic i4Q solutions. These reports provide crucial information on parameter configurations associated with product failures, and they may be utilized to implement corrective machine reconfigurations or to promptly halt a machine in order to prevent the production of additional defective products. The i4Q^{QD} mapping to "Digital Twin Services" sub-component enables the ML and AI models, developed for product quality diagnostics, to be trained on simulated data in the absence of actual manufacturing data. Moreover, this data may also be used to improve the training performance of the ML models when sensor data alone are insufficient to achieve acceptable detection accuracy.
- **Edge Tier:** The i4Q^{QD} mapping to the "Data Management" sub-component allows more efficient data pre-processing and feature engineering from several sources (e.g., i4Q^{DR}, i4Q^{DIT}) to provide accurate product quality predictions. Feature engineering in combination with feature importance give crucial information into the parameter settings pertaining to the existence of product defects.

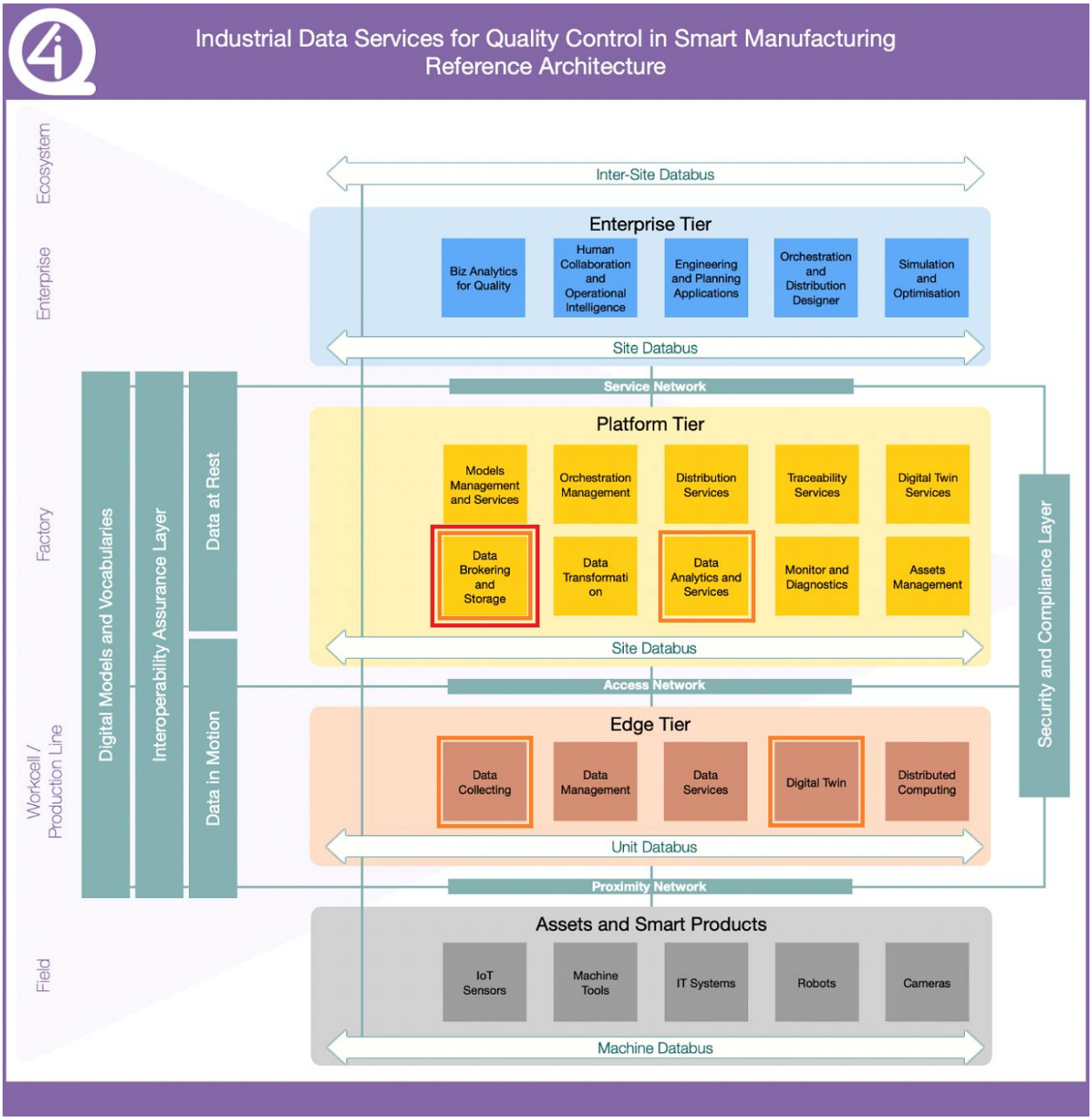

Figure 1. i4Q Reference Architecture mapping with i4Q^{QD}

2. Implementation Status

2.1 Current implementation

Following the development status of the i4Q^{QD} as described in D5.2, the implementation phase of the solution continues to complete the rest of its mapped pilot requirements. Specifically, these requirements are the following:

- **FARPLAS:** The requirement to *stop the machine [BP1_PC4r3.4.3]* which is connected to the deployment of machine learning algorithms that could infer the finish quality of the manufactured pieces based on the predicted lifespan of the tool. In case a failure is detected, an alert will be sent to the appropriate i4Q solution to stop the machine. In addition, whenever the machine is stopped, the solution should consider the reason for the stop when calculating the real-time KPI 412 and the time the machine is halted when calculating the remaining KPIs. To solve this issue, the i4Q^{QD} solution utilizes primarily AI model training, estimates of important machine parameter settings, and parameter optimization.
- **FACTOR:** The *capability to mapping part-label data correctly [BP1_PC6r9]* which requires the development of a machine learning algorithm to monitor the accurate label assignment for all inspected parts produced by the injection molding machines, owing to the fact that this data will influence parameter optimization. To address this problem, the i4Q^{QD} solution features several class imbalance techniques to combat the lack of training data as well as AI models to reliably detect faulty product quality.

2.1.1 FACTOR - Stop the machine

2.1.1.1 *Data & Task Description*

The FACTOR pilot dataset is comprised of a datasheet that has been generated through the information recorded in the Nakamura WT-150II machine placed in FACTOR facilities through a new Data Acquisition Software (DAS), the FANUC MT-Link. This software is configured to obtain the data from the sensors and the actuators of the machine in cycles of 100ms, 1 minute or 1 hour. Besides this information, the quality department from FACTOR has generated a datasheet to stablish the quality of the manufactured piece. For example, in a cylindrical piece, they measure its diameter to compare this result against the constructive parameters from the draft. Attending to the result obtained, they have divided these measures into three stages:

- **Green (0):** Below the tolerance limit. In the example, the measure obtained differ from the specifications in less than a 5%.
- **Yellow (1):** In the limit of the tolerances. In the example, the manufactured piece differs from the manufacturer datasheet in a range higher than 5%, but lower than 6%.
- **Red (2):** Outside the tolerances. In the example, the diameter of the piece differs from the draft information in more than 6%.

2.1.1.2 *Methodology*

The study has generated a solution to estimate the quality of a piece mechanized through milling operations in a Nakamura WT-150II. This solution is based on Generative Adversarial Networks (GANs), a machine learning algorithm divided in two Neural Networks (NNs):

- **Generator:** The main NN trained to estimate the quality of the manufactured piece

- **Discriminator:** The secondary NN trained to distinguish fake from real data

The methodology proposed sought to train a model to estimate the quality of the manufactured piece through the generator. To obtain this estimation, during the training phase, the generator will be feed with data from the real procedure to establish the relation between these inputs and the desired output, in this case, the quality of the manufactured piece. Thanks to this procedure, the generator would create a synthetic sample representing the behavior of the piece quality. This batch of synthetic samples would be mixed with samples obtained from the machine during previous procedures. The mix of original and generated samples will be fed to the discriminator to classify them between real or fake data. If the discriminator fails to distinguish between the generated and the original samples, the algorithm is trained, as the generator produces samples practically identical to the ones obtained from the milling operations. For example, in the use-case proposed, the generator would be trained with information recorded from previous milling operations, while the discriminator would be feed with samples that represent the piece quality obtained for such previous milling operations.

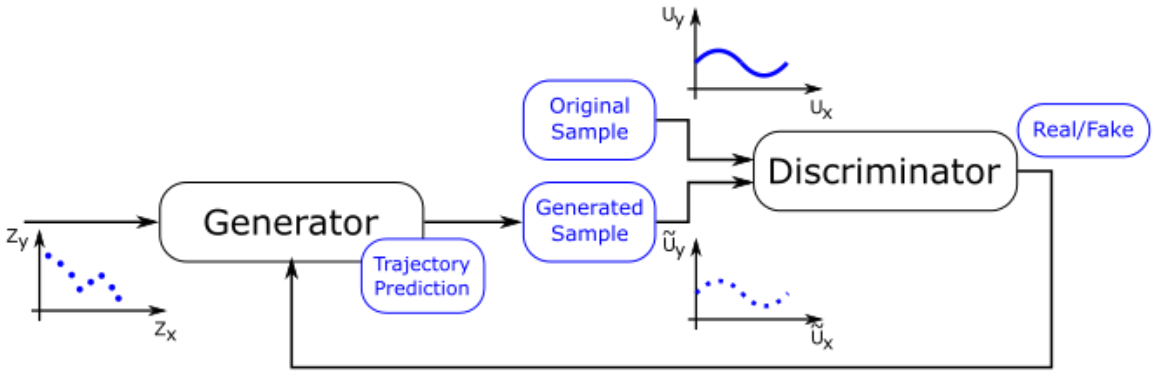

Figure 2. General description of the GAN fields

After the system is trained, the generator could be exported as a standalone algorithm. This data-driven solution would represent the model of the system, in this case, it would be able to estimate the quality of the manufactured piece by the Nakamura WT-150II. The main benefit of this approach is its replicability, as the system could be retrained. For example, when a new milling operation is introduced, the user would be able to retrain the GAN to obtain a new model that represents this new behavior. In addition, this methodology brings the flexibility of picking other data-driven approaches in the generator, allowing to particularize the solution to the use case. Despite its benefits, the methodology also presents several difficulties, as the models heavily rely on the information recorded from the real procedure. For example, in the current use-case, the model must predict the quality of the piece. To make this estimation, the generator is trained with real data from the industrial process to establish the relation between the machine signals and how they modify the final quality of the piece. In addition, the discriminator has to be feed with real samples obtained from the machine to compare these signals against the generated samples to stablish if they are real or fake.

2.1.1.3 Quantitative results

FACTOR has recorded information of thirty-seven variables from the Nakamura WT-150II machine (**Table 1**). These variables represent the behavior of the machine during milling and cutting operations. They have been stored inside a datasheet separated through columns. Besides this information, FACTOR has also provided a datasheet with the quality of the manufactured piece during these operations.

Both datasheets have been combined to represent the quality of the piece attending to the mechanical operations realized in the piece during the milling procedure. The datasheet has

the thirty-seven variables distributed in columns, obtaining a file that contains the behavior of the machine during the milling operation. This datasheet has been associated with a piece quality.
Despite these thirty-seven variables describe the behavior of the machine during the operation, they affect to the final quality of the piece in various degrees, as there are variables that have a higher impact. In order to determine how these variables, impact on the quality of the piece, each parameter has been analyzed through a minimum Redundancy – Maximum Relevance (mRMR) algorithm. This method sought to identify the set of features that have the maximum impact.

Order	Column Number	Variable	Percentage [%]
1	16	NAKA2_TheNumberOfMachinedPartsNakamura2P1_ValueMax	**0,9418**
2	20	NAKA2_CuttingFeedSignalNakamura2P1_Value	0,8671
3	21	NAKA2_CuttingFeedSignalNakamura2P2_Value	0,8375
4	25	NAKA2_Temp_APCNakamura2P1A0_ValueMax	0,4185
5	33	NAKA2_Temp_APCNakamura2P2A1_ValueMid	0,3873
6	9	temperatura_motor_5	0,3828
7	14	NAKA2_TheNumberOfMachinedPartsNakamura2P1_ValueMin	0,3469
8	37	NAKA2_Temp_APCNakamura2P1A3_ValueMax	0,333
9	27	NAKA2_Temp_APCNakamura2P2A0_ValueMid	0,3326
10	15	NAKA2_TheNumberOfMachinedPartsNakamura2P1_ValueMid	0,3326
11	32	NAKA2_Temp_APCNakamura2P2A1_ValueMin	0,3116
12	36	NAKA2_Temp_APCNakamura2P1A3_ValueMid	0,3116
13	18	NAKA2_TheNumberOfMachinedPartsNakamura2P2_ValueMid	0,3074
14	26	NAKA2_Temp_APCNakamura2P2A0_ValueMin	0,2945
15	28	NAKA2_Temp_APCNakamura2P2A0_ValueMax	0,2945
16	23	NAKA2_Temp_APCNakamura2P1A0_ValueMin	0,2945
17	7	temperatura_motor_2	0,2897
18	24	NAKA2_Temp_APCNakamura2P1A0_ValueMid	0,2818
19	6	temperatura_amb	0,2667
20	10	temperatura_motor_7	0,2543
21	11	temperatura_motor_8	0,2539
22	13	temperatura_taladrina	0,2454
23	8	temperatura_motor_3	0,2434
24	34	NAKA2_Temp_APCNakamura2P2A1_ValueMax	0,2401
25	5	humedad_amb	0,2343
26	35	NAKA2_Temp_APCNakamura2P1A3_ValueMin	0,2324
27	17	NAKA2_TheNumberOfMachinedPartsNakamura2P2_ValueMin	0,2219
28	2	corriente_x_torreta_inferior	0,2052
29	12	temperatura_motor_9	0,2021
30	3	corriente_z_torreta_inferior	0,1673
31	22	NAKA2_OPERATE_Value	0,1595
32	19	NAKA2_TheNumberOfMachinedPartsNakamura2P2_ValueMax	0,1562
33	4	corriente_z_torreta_superior	0,1455
34	1	consumo_total	0,0211

35	29	NAKA2_Temp_APCNakamura2P1A1_ValueMin	0
36	30	NAKA2_Temp_APCNakamura2P1A1_ValueMid	0
37	31	NAKA2_Temp_APCNakamura2P1A1_ValueMax	0

Table 1. mRMR for the variables studied in the Nakamura WT-150II

The GAN algorithm has been trained with the datasheet previously studied, feeding the generator with the most promising variable was the column sixteen or the "NAKA2_TheNumberOfMachinedPartsNakamura2P1_ValueMax". This variable has been selected as it has the best performance to determine the quality of the piece attending to the mRMR study previously done. Due to the characteristics of the GAN algorithm, in addition to train the generator, the algorithm must also train the performance of the discriminator.
For the experiments (**Table 2**), the parameters selected to optimize the GAN training are the learning rate and the number of epochs. In the first batch of experiments, when the learning rate is at 0.001, the GAN algorithm enters in a recursive loop, where despite the number of epochs increments, the generator and the discriminator are unable to obtain a higher accuracy. Due to this regard, in the second batch of experiments, the learning rate has been decreased to 0.0005, obtaining a better response. In the final step, the learning rate has been reduced to 0.0001 to study the new responses. Despite the accuracy when the number of epochs is low is better with a learning rate of 0.0005, when the number of epochs increase, the accuracy also rises. Due to this fact, it has been done a final study increasing drastically the number of epochs with a learning rate of 0.0001.

Learning Rate	Number of Epochs	Generator Accuracy [%]	Discriminator Accuracy [%]
0.001	100	62,73	75,81
0.001	200	69,19	69,11
0.001	300	69,32	69,32
0.001	500	69,31	69,00
0.0005	100	48,44	78,88
0.0005	200	56,26	80,22
0.0005	300	72,56	82,36
0.0005	500	78,61	88,97
0.0001	100	39,28	9,71
0.0001	200	68,64	37,47
0.0001	300	81,21	38,89
0.0001	500	82,34	62,46
0.0001	**2000**	**92,26**	**88,75**

Table 2. GAN study, accuracy of the generator and the discriminator

2.1.1.4 Qualitative results

After the 2000 epochs training, the GAN algorithm is able to predict the quality of the manufactured piece for other inputs. Through the study, it has been determined that the generator offers better responses when it gives the response in a spectrum instead of the three levels previously mentioned. As an example, the **Figure 3** below presents the response of the generator for two of the trainings (the blue and the green dots).

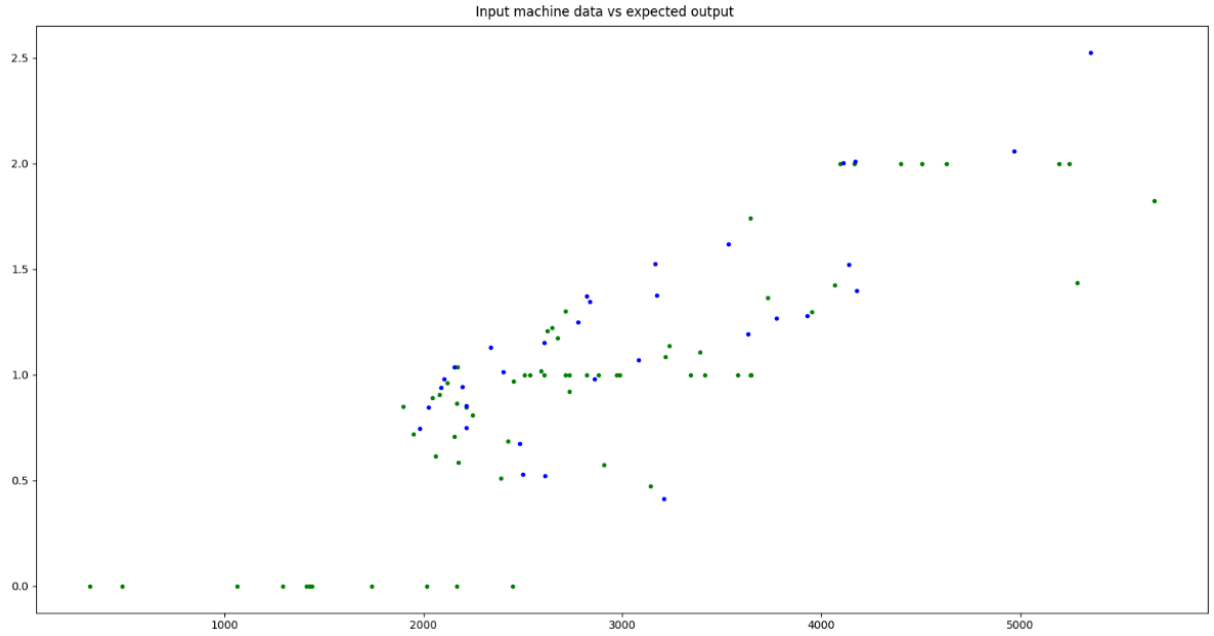

Figure 3. Response of the GAN algorithm after 2000 epochs

To provide a solution following the three quality levels, green (0), yellow (1) and red (2), the green estimation produced by the GAN has been normalized. In addition, the response has been compared with the original information recorded from the Nakamura WT-150II.

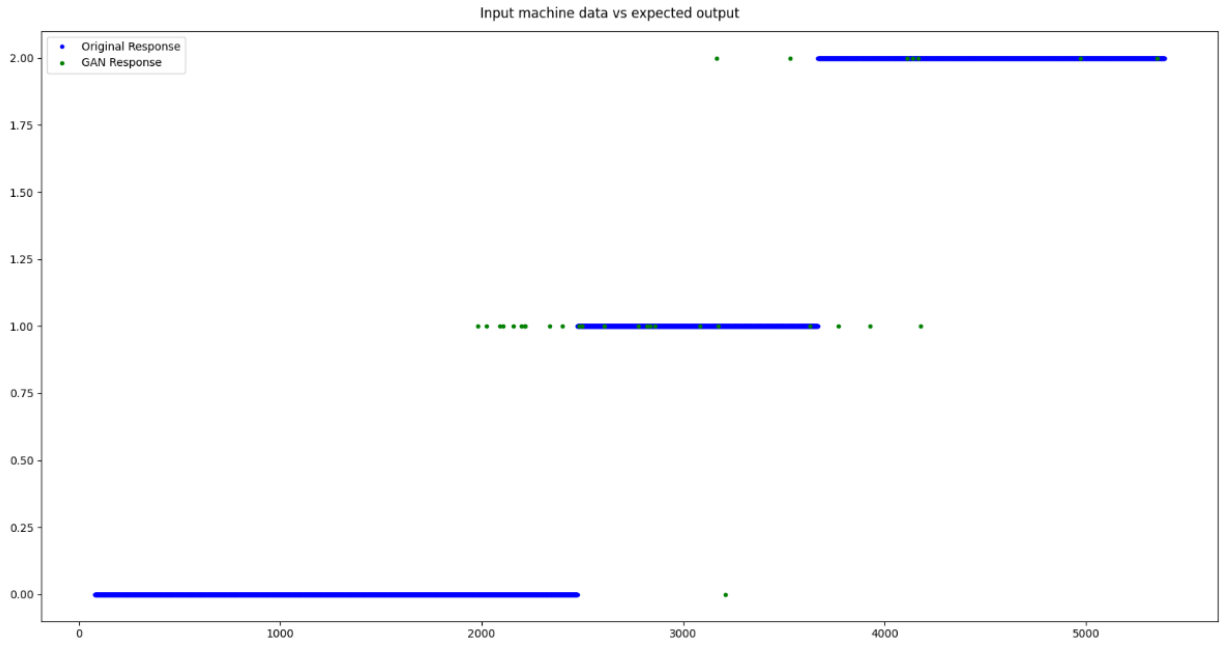

Figure 4. Normalization of the GAN algorithm after 2000 epochs

As the estimation shows in **Figure 4**, there are some dispersions between the original data and the one predicted by the generator. These discrepancies are due to the fact that despite the fact that it has been carried out studies to detect the most relevant variables, there are some missing information that has not been registered by the DAS. It is advised to do further studies to obtain more information of the real procedure to understand which are the main variables that modify the response of the machine attending to the fault.

2.1.2 FARPLAS - Capability to mapping part-label data correctly

2.1.2.1 Data & Task Description

The dataset provided by Farplas consists of 110,045 records of parts produced by injection molding machines. Each record of the dataset contains information regarding the configuration settings of the molding machine along with the quality conformity of the produced part. The dataset covers the quality control records of parts of various mold ids. As instructed by those responsible for the provision of the dataset the different mold IDs should be treated separately, as they correspond to different lines of products. The goal of i4Q^{QD} is to reliably classify the quality status of the produced parts between tolerable and defective, effectively reducing the problem at hand to a binary classification.

2.1.2.2 Methodology

Before proceeding with the development of the classification algorithm, it was essential to properly study the available data and assess how to approach the problem's resolution in an appropriate manner. As advised, the algorithm should be applicable on parts with common mold ID, in order to achieve more accurate and customized predictions. Therefore, the dataset was initially split into subset based on the injection mold id of the parts, ending up with a total of 7 subdivisions. From these subsets, the one with the most samples available was chosen to develop the predictive algorithm and to evaluate its efficacy. The chosen subset corresponds to the parts with injection mold id '*2297*' with a number of 38,419 records. Moreover, the value that characterizes the quality of the parts is not binary. The faulty parts have several quality codes depending on the nature of the defect. Therefore, in order to transform the problem into a binary classification, all the faulty parts were combined under a single label that corresponds to the defective products.

A negative characteristic underlying the data, regardless of the subset, is the severity of imbalance present between the sample number of good and faulty products. In the case of the subset that was selected, the percentage of parts that are labeled as faulty in relation to the total number of instances is only 2.4%. As a conscience, the necessity for the utilization of class imbalance techniques is apparent. These methods were integrated and employed during the training procedure of the classification algorithm to ensure more reliable results. However, in order to explore the utilization of class imbalance methods in the training pipeline of the classifier, it is essential to also describe the classifier itself.

Initially, three different machine learning classifiers have been tested on the selected data subset, to study their performance and identify which one is more suitable for the task. Specifically, the algorithms tested were a Decision Tree, a Random Forest and a Light Gradient Boosting Machine classifier. At first, the data were split into training and test sets. The training set is the biggest portion of the available data (80%) and is utilized to train the model and ultimately learn the patterns underlining the data. On the other hand, the test set is comprised from the rest of the data (20%) and is utilized to provide an unbiased evaluation of the performance of the model after the training process has been completed.

For the training process, the classifier along with data-preprocessing and class imbalance techniques were wrapped into a single pipeline. The training pipeline includes the following components:

- **Class imbalance methods**

  The first technique employed to combat the imbalance present was Tomek Links. This is a form of under-sampling as it discards a certain number of samples from the class with the most instances. In simple terms, the technique eliminates samples from the majority class that are close to the minority class, thus effectively assisting the separation of the two classes whilst slightly minimizing the sample difference.

  The second technique used to tackle the problem was SMOTE. This method constructs synthetic data for the minority class to balance the two classes. This method was integrated into the training pipeline in consultation with the i4Q^{DIT} solution. Further information about the application of SMOTE can be found in the related deliverable D4.9 of the i4Q^{DIT} solution.

- **Data scaling**

  The dataset contains several records and each one is described by a set of attributes (variables). If the attributes are of numerical nature, then it is possible that different attributes are represented by values of different scale. Utilizing scaling allows the classification model to consider each variable on equal footing, thus preventing as bias toward a certain group of variables.

- **Classification model**

  As stated previously, three classifiers were initially tested to figure out the most suitable one. To properly examine the performance of the classifiers class imbalance specific metrics were selected, such as the balanced accuracy score, the weighted F1 score and the ROC (Receiver Operating Characteristic) accuracy score. Additionally, the on every classifier a cost sensitive strategy was utilized to penalize more the misclassification of the minority class samples.

In the following subsection, the result of the initial training procedure to determine the appropriate algorithm as well as the fine-tuning process to optimize the selected model are described in detail.

2.1.2.3 Quantitative results

The training process for the three classifiers was conducted within a 10-fold cross-validation. The cross-validation allow to achieve generalized model training as it avoids the introduction of bias towards a specific data portion. The performance comparison of the classifiers in the test set is summarized in the following table.

Classifier	Balanced Accuracy
Decision Tree	53.2%
Random Forest	55.9%
LightGBM	**74.2%**

Table 3. Performance comparison between Decision Tree, Random Forests and LightGBM

It is clear that the LightGBM outperforms the other ML approaches. The results may not appear to be particularly impressive, but it should be noted that this is a case of extreme imbalanced data with low a low count of faulty part instances.

That kind of performance was not the product of a classifier using stock parameters. Multiple combinations of hyper-parameter values have been evaluated in order to fine-tune the model's performance. This was accomplished through the utilization of grid-search that facilitates the model parameter testing. The parameters chosen for optimization along with the values tested are the following:

- **Learning rate = [0.01, 0.05, 0.1],** determining the step size of each iteration in to the loss function.
- **Number of estimators = [80, 100, 160],** defining the number of boosted trees.
- **Number of leaves = [32, 64, 128],** determining the max number of tree leaves.
- **Max depth = [16, 24, 32]**, specifying the max depth of each boosted tree.
- **Class weight = [{0:100, 1:1}, {0:80, 1:1}, {0:120, 1:1}]**, defining the weight assigned to each class.

In the following table are presented only a portion of the grid-search experiments. Specifically, these are the 5 worst and best performing cases:

Parameter Settings	Balanced Accuracy
'class_weight': {0:80, 1:1}, 'learning_rate': 0.01, 'max_depth': 24, 'n_estimators': 80, 'num_leaves': 32	51.07%
'class_weight': {0:100, 1:1}, 'learning_rate': 0.01, 'max_depth': 32, 'n_estimators': 80, 'num_leaves': 32	51.17%
'class_weight': {0:100, 1:1}, 'learning_rate': 0.01, 'max_depth': 24, 'n_estimators': 80, 'num_leaves': 32	51.17%
'class_weight': {0:100, 1:1}, 'learning_rate': 0.01, 'max_depth': 16, 'n_estimators': 80, 'num_leaves': 32	51.18%
'class_weight': {0:120, 1:1}, 'learning_rate': 0.01, 'max_depth': 32, 'n_estimators': 80, 'num_leaves': 32	51.23%
'class_weight': {0:80, 1:1}, 'learning_rate': 0.05, 'max_depth': 16, 'n_estimators': 100, 'num_leaves': 64	69.21%
'class_weight': {0:120, 1:1}, 'learning_rate': 0.05, 'max_depth': 16, 'n_estimators': 100, 'num_leaves': 64	69.24%
'class_weight': {0:80, 1:1}, 'learning_rate': 0.05, 'max_depth': 24, 'n_estimators': 100, 'num_leaves': 64	69.33%
'class_weight': {0:100, 1:1}, 'learning_rate': 0.05, 'max_depth': 32, 'n_estimators': 100, 'num_leaves': 64	69.39%
'class_weight': {0:100, 1:1}, 'learning_rate': 0.05, 'max_depth': 24, 'n_estimators': 100, 'num_leaves': 64	**69.67%**

Table 4. LightGBM fine-tuning

2.1.2.4 Qualitative results

In this section, some visual representations of the qualitative results of the quality conformity detection methodology are shown. **Figure 5** presents the most influential data features that as contribute the most towards the predictions of the LightGBM classifier. The value assigned to each feature determines its importance in the model's predictions. The greater the contribution of the traits to the outcome of the prediction, the greater their value.

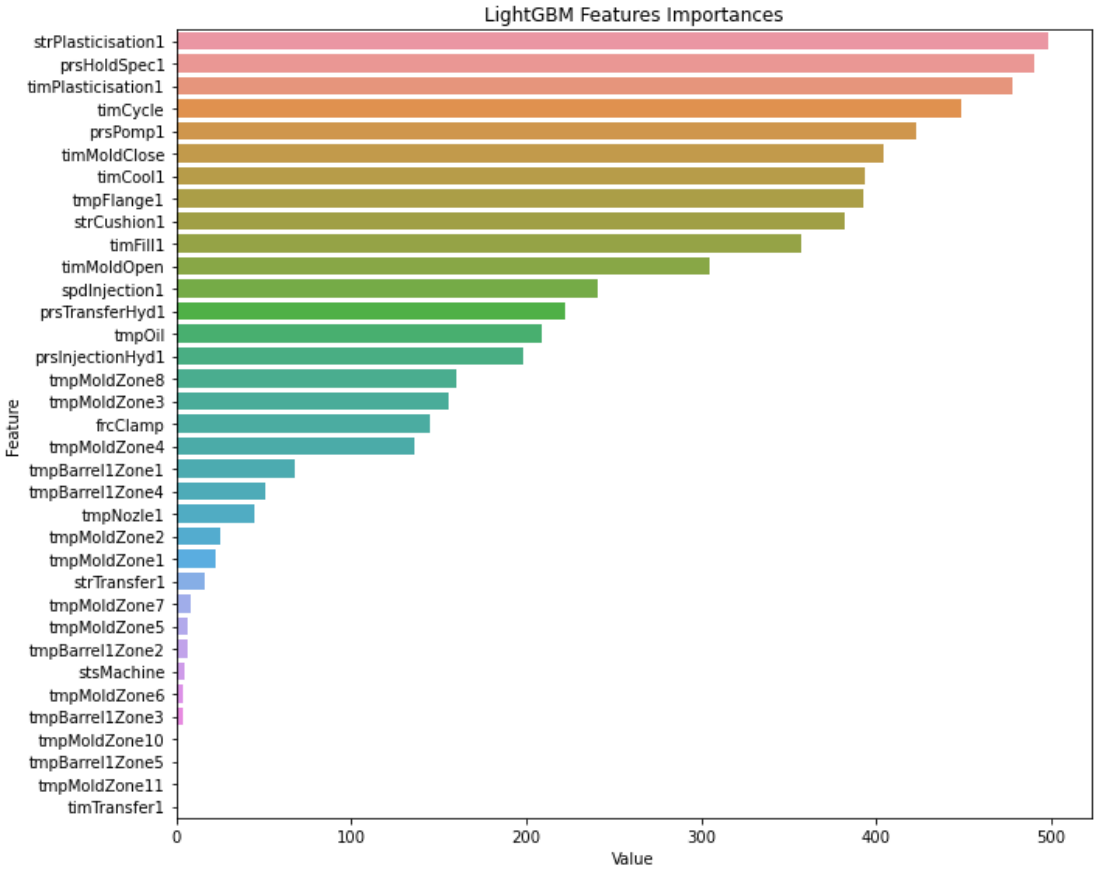

Figure 5. LightGBM feature importance visualization.

Additionally, a visual depiction of the LightGBM prediction performance in the test set is presented in **Figure 6**.

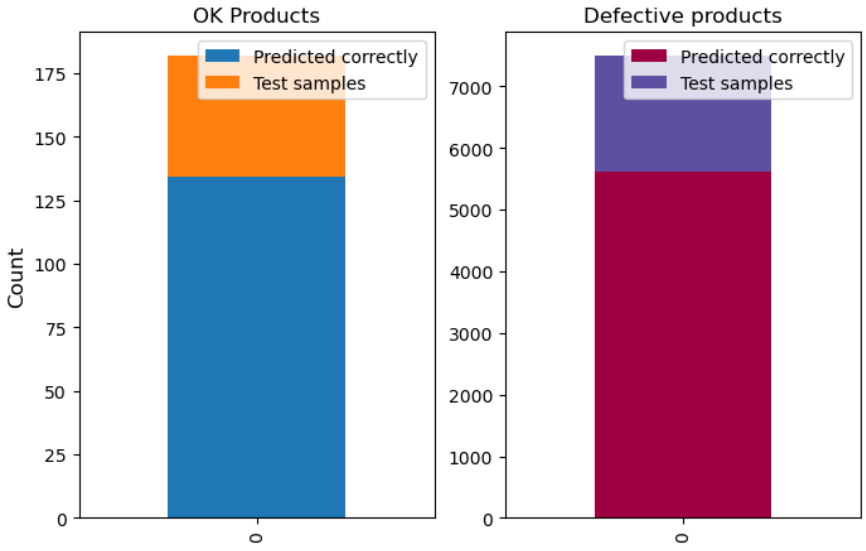

Figure 6. LightGBM prediction performance in the test set.

2.2 User Interface

A front-end mockup of the User Interface (UI) has been prepared that will host the i4Q solutions developed by CERTH.

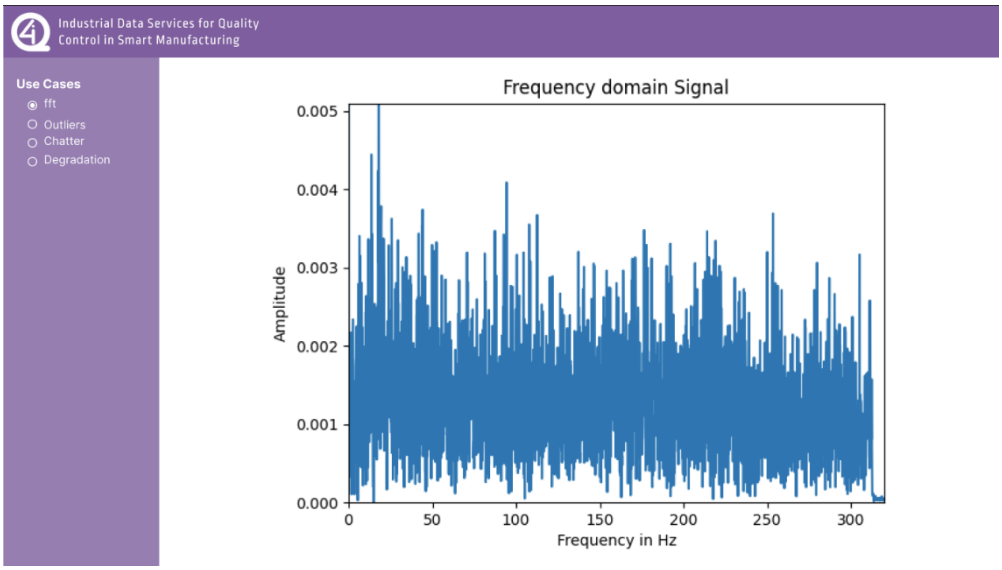

Figure 7. User Interface mock-up

The created UI comprises of three components: **a)** the header, **b)** the side menu, **c)** and the chart-containing main section.

The header contains the logo and title of the i4q project ("Industrial Data Services for Quality Control in Smart Manufacturing"). The menu on the left side of the user interface allows the user to select one of the available use cases: FFT, Outliers, Chatter, or Degradation. When a user chooses an option, the associated graph is presented in the interface's main section. It is also possible to zoom in and out of the chart to focus on a specific section.

The data used for the construction of the graphs are fixed and stored in a static CSV. When a use case is selected from the menu, the server reads the associated CSV file and sends the data to the front-end, where they are properly formatted and provided to the chart in order to be displayed.

The technologies that have been used for the development of the UI are HTML5, CSS3 and JavaScript. A JavaScript library named Chart.js is utilized to generate charts. Finally, the Chart.js plugin chartjs-plugin-zoom implements the zoom functionality.

2.3 History

Version	Release date	New features
V0.7.0	15/10/2022	ML algorithms implementation and evaluation to cover the remaining pilot requirements
V0.7.5	20/11/2022	Front-end mock-up for the User Interface
V1.0.0	30/12/2022	Final version

Table 5. History

3. Conclusions

Deliverable D5.8 *Rapid Quality Diagnosis* is a technical specification document, providing a detailed description of updates made in the implementation stage of the i4Q^{QD} solution. It explains thoroughly the design process of the analytical algorithms aiming to cover the remaining pilot requirements, by providing comprehensive information about the steps of their development.

The LightGBM was discovered to be the most effective classifier amongst other ML classifiers in detecting non-conformity situations regarding the quality of parts produced by injection moulding machines. The introduction of class imbalance techniques in the training pipeline of the ML model, such as SMOTE, Tomek links and cost-sensitive class weighting, played a major role to the efficacy of the model. Through fine-tuning the model achieved a balanced accuracy as of 75% in faulty product detection, given limited data. However, in the event that additional data emerging, the model's performance can definitely be improved.

Additionally, the utilization the GAN algorithm showcased exceptional performance in predicting the quality of the manufactured piece finish from cutting tool machines. Using the sensors and the actuators of the machine acquired through a DAS, the model could determine the quality tolerance of the piece with an accuracy of 92.26 %.

Finally, an introduction to the user interface was provided, outlining the technologies employed and offering a peek of its data visualization and user interaction possibilities.

Appendix I

i4Q Rapid Quality Diagnosis (i4Q^{QD}) web documentation can be accessed online at: **http://i4q.upv.es/18_i4Q_QD/index.html**