Journal of Machine Engineering, 2018, Vol. 18, No. 4, 60–72
ISSN 1895-7595 (Print) ISSN 2391-8071 (Online)

Received: *19 January 2018* / Accepted: *27 November 2018* / Published online: *20 December 2018*

root cause analysis, machine learning,
decision tree, simulation

Tobias MUELLER[1*]
Jonathan GREIPEL[1]
Tobias WEBER[2]
Robert H. SCHMITT[1]

AUTOMATED ROOT CAUSE ANALYSIS OF NON-CONFORMITIES WITH MACHINE LEARNING ALGORITHMS

To detect root causes of non-conforming parts - parts outside the tolerance limits - in production processes a high level of expert knowledge is necessary. This results in high costs and a low flexibility in the choice of personnel to perform analyses. In modern production a vast amount of process data is available and machine learning algorithms exist which model processes empirically. Aim of this paper is to introduce a procedure for an automated root cause analysis based on machine learning algorithms to reduce the costs and the necessary expert knowledge. Therefore, a decision tree algorithm is chosen. A procedure for its application in an automated root cause analysis is presented and simulations to prove its applicability are conducted. In this paper influences affecting the success of detection are identified and simulated e.g. the necessary amount of data dependent on the amount of variables, the ratio between categories of non-conformities and OK parts as well as detectable root causes. The simulations are based on a regression model to determine the roughness of drilling holes. They prove the applicability of machine learning algorithms for an automated root cause analysis and indicate which influences have to be considered in real scenarios.

1. INTRODUCTION

Despite the increasing availability of automated data science methods, analysis of non-conforming parts (short: non-conformities; defined as violation of tolerance limits) is still dependent on the expertise of employees [1, 2]. This expertise is used to detect root causes in small batch production processes of complex products and is therefore – in connection with the associated work force – a big time and cost factor for companies [3]. Regarding the manual analysis, the result of a root cause analysis (RCA) is affected by operator influences. This dependency is reinforced by an increased complexity due to the vast amount of process data available in modern production environments. While large amounts of data have

[1] Laboratory for Machine Tools and Production Engineering WZL of RWTH Aachen, Chair of Production Metrology and Quality Management, Aachen, Germany
[2] Boeing Research & Technology, Europe
[*] E-mail: T.Mueller@wzl.rwth-aachen.de

https://doi.org/10.5604/01.3001.0012.7633

potential to include a lot information, the complexity may lead to wrong conclusions or causal relations being undiscovered. Thus, an automated solution is required [4].

New developments in the fields of mathematics and computer science offer possibilities to manage and analyse large amounts of (complex) data automatically. In particular, the different Machine Learning (ML) approaches promise fast and reliable results [4]. Its increasing application is assisted by the ability of analysing complex data of different types and sources, finding patterns in unstructured raw data and calculate models for prediction, regression or detection [5, 6].

The overall goal of this paper is to automate the previously manual RCA with ML algorithms, without using predefined root causes as training. Non-conformities are analyzed and attributed to responsible process parameters (root causes) without special knowledge about the production process or the product. This work specifically addresses the use case of small batch production. Due to the low amount of data of such a production and the resulting non-applicability of the algorithms, a production-related simulation model is set up with which the ML algorithm is to be trained. The trained model is then applied to real data and root causes can be analyzed. During the simulation, the main aspect is the identification and quantification of various influencing parameters on the performance of ML. These are to be considered in the simulation and in the transfer to the real production scenario.

After a short description of current RCA, ML is introduced against the background of an automated RCA. The selection of an appropriate ML algorithm, the training of it and the different influencing parameters are described in Chapter 3 and 4.

2. ROOT CAUSE ANALYSIS

The process of RCA consists of the collection of data and its systematic analysis to perform a root cause identification. Eliminating the root cause of a non-conformity means to prevent the occurrence of it [7].

Common tools for RCA are Cause-and-Effect Diagrams, Interrelationship Diagrams and Current Reality Trees. Using the same causal logic, all of them can be used individually or in tandem. The methods process the data, so that the unstructured data can be sorted and root causes for non-conformities can be identified by uncovering input/output relationships [8].

Dogget A.M. shows in [9], that it is not possible to distinguish the three methods in their ability to identify root causes. The methods are characterized by manual processes and thus are dependant on the experts’ background. Furthermore, the processes become tedious with increasing complexity attributing to the fact that different experts will arrive at different conclusions [8].

To counteract these disadvantages, the detection of root causes of non-conformities can be done using automated algorithms. In [12–14] different approaches to an automated RCA can be found. However, these have either a pure reference to software testing or the performance of computer resources and therefore cannot be applied to the context of a production processes. In [14], Pederson H. creates the link of the RCA to a complex product using the comparison of different timeslots – a timeslot of normal operation and one of misbehaviour operation. Although the method recognizes different so called misbehaviour patterns, in the context of the comparison, however, it only provides a list of possible indicators, which were rated one to one hundred in each case.

In summary, an algorithm has to be found which automatically analyzes root causes of non-conformities in a (small batch) production process without special expertise or knowledge about the process

3. DECISION TREES FOR AUTOMATED ROOT CAUSE ANALYSIS

The design of many ML algorithms allows analysing a vast amount of data with high dimensionality and is therefore an alternative in cases of highly complex data/problems and vast required expert knowledge [2, 15]. In addition to the possibility of automation, ML algorithms offer the possibility of eliminating human influence on the results of a detection of root causes. It addresses these problems by extracting a model for describing the relationships directly from the observed data without external input and improves the accuracy and/or the efficiency of the detection of root causes by discovering regularities. For non-conformities, responsible process parameters are identified [3, 15, 16].

Common methods of ML are described and it is explained how an algorithm is selected for the automated RCA of non-conformities in a small batch production (Chapter 3.1). This selection then forms the basis for the simulation used to train the model and later for the use in the real production scenario. Chapter 3.2 describes the exemplary RCA with a C5.0 decision tree.

3.1. DESCRIPTION AND SELECTION OF A ML ALGORITHM FOR AN AUTOMATED ROOT CAUSE ANALYSIS

ML methods are classified into four categories depending on the amount and type of supervision:

- *Supervised learning*: In supervised learning, the training data fed to the ML method includes the desired solutions (e.g. classification in OK and Not-OK), called labels [12]. With supervised learning a predictive model can be generated [17].
- *Unsupervised learning*: In unsupervised learning the complete training dataset is unlabelled. Its objective is to benefit from the insight gained by summarizing data in different ways. It is used to discover patterns without any existing knowledge about the dataset [17].
- *Semi-supervised learning*: In semi-supervised learning algorithms can deal with partially labelled and unlabelled data. Most semisupervised learning algorithms are combinations of unsupervised and supervised algorithms [17].
- *Reinforcement Learning*: Reinforcement Learning does not need labels but a goal that needs to be defined. The learning algorithm can observe the environment, select and perform actions, and get rewards in return the closer it gets to the goal. The aim

of the algorithm is to identify what is the best strategy, to get the most reward or success over time [12].

Fig. 1 gives a short overview about ML algorithms based on the type of training.

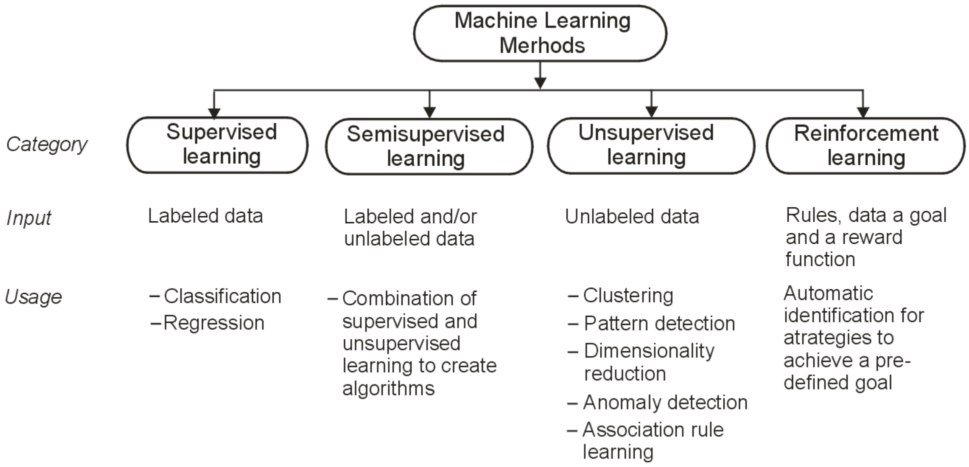

Machine Learning Merhods
Category
Supervised learning
Semisupervised learning
Unsupervised learning
Reinforcement learning
Input
Labeled data
Labeled and/or unlabeled data
Unlabeled data
Rules, data a goal and a reward function
Usage
– Classification
– Regression
– Combination of supervised and unsupervised learning to create algorithms
– Clustering
– Pattern detection
– Dimensionality reduction
– Anomaly detection
– Association rule learning
Automatic identification for atrategies to achieve a pre-defined goal

Fig. 1. Overview about categories of ML methods based on the type of training, inputs and use cases (based on [17])

Considering the use case of an automated RCA in production processes, labelled data (OK and n.OK – conformal, non conformal) can be assumed. Regarding the necessary input for the different categories of ML algorithms, supervised learning algorithms suit the described use case best (compare with Fig. 1). For the identification of the root causes, the algorithm should analyze the process independently. The aim is to identify unknown root causes with the algorithm from measured process parameters without learning the algorithm with the root causes. For this purpose, the structure of the (process) model created by the algorithm has to be analyzed. Only models in which the input/output relationship is transparent are suitable for such an analysis [18]. Accordingly, a white box model is chosen for the automated RCA approach. Black-box models [19] like the Support Vector Machines or Neural Networks algorithm are not suitable. They would need pre-defined root causes as training. Due to its wide distribution and robustness, the Decision Tree is a suitable white box model and supervised learning algorithm for a RCA of non-conformities.

A decision tree starts with a root node where a first decision is required. The decision either passes through decision nodes or ends in terminal nodes. A decision node as well as the root node requires a decision for one of the alternatives. Even the number of alternative decisions is not limited, the number of possible choices in one decision node is restricted to one alternative. Ending in a terminal node means, a final decision can be made. The decision tree is complete when all decision nodes end in terminal nodes (see Fig. 2) [12].

Decision trees need labelled data. The data can be attributive or variable. One of the advantages of decision trees is that they require very little data preparation. They aim to help with making decisions by providing simple choices. The choices are easily understood without statistical knowledge [12]. The decision tree produces a model with minimal user input [11] and uses a method called recursive partitioning. This method splits the data in subsets, which are repeatedly split into smaller subsets until a stopping criterion, for example the entropy of data, is met [12].

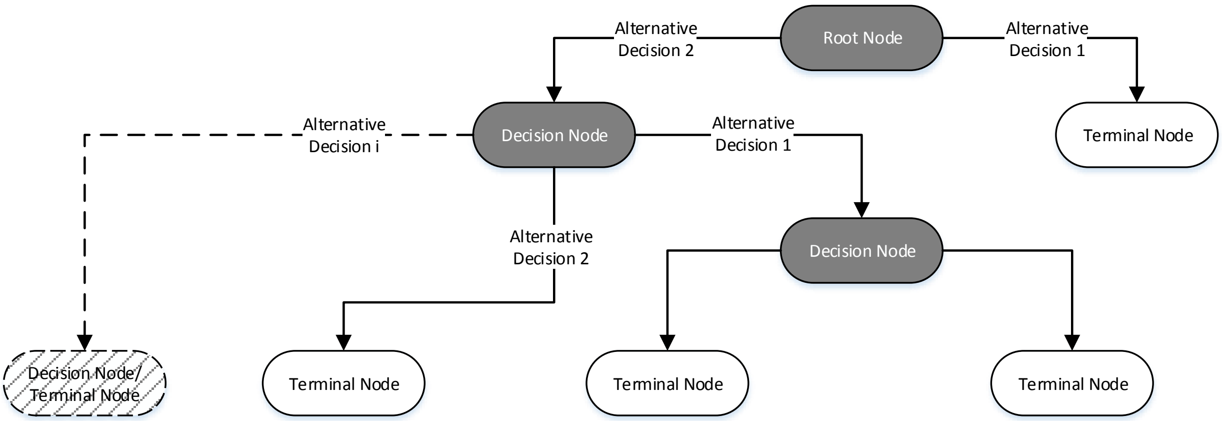


Fig. 2. Components of decision trees according to [12]

As a result, decision trees can give rules that lead to certain non-conformities. The rules are mostly designed according to ‘if conditions’. For the identification of root causes, the tree is decomposed. Using the characteristics of white box models, the tree is analysed stepwise. Each decision node represents a process parameter with its value range and is therefore a possible root cause.

For the selection of the decision tree algorithm to be implemented in the scenario of the small batch production process, it must be taken into account that both attributive and continuous data (together) must be able to be evaluated via the model. Neither continuous nor attributive data can be categorically excluded in the use case. There are two types of decision trees: binary trees and non-binary. Binary ones can only have two nodes as a output of previous node – for non-binaries no restrictions are set. [20] Two of the most common algorithms (both able to handle attributive and continuous data) are the CART algorithm (binary) [21] and the C5.0 algorithm (non-binary). Since in general binary decisions trees lead to larger and therefore more complex trees, the decision was made for the C5.0 algorithm. Additionally, the C5.0 algorithm has become the industry standard for decision trees and for many problems it delivers results directly out of the box. For a better performance of the C5.0 algorithm the input data is discretized [12].

3.2. EXEMPLARY ROOT CAUSE ANALYSIS WITH A C5.0 DECISION TREE

The application of the decision tree as the basis for the automated RCA is divided into two steps: The training and the RCA. For the training, simulation data is used to build the tree structure. Discretized process parameters and inspection parameters (OK/n.OK-labels) for every part are the input for the decision tree algorithm. 80% of the given data was used for the training and 20% for the test of the model. Such a test prevents the so called overfitting – “fitting to individual data points rather than the trend” [22]. The test adopts the developed model and verifies the results determined with it by applying the model to the remaining 20% of the data. The general procedure and the resulting decision tree using the given data is shown in Fig. 3 and Fig. 4:

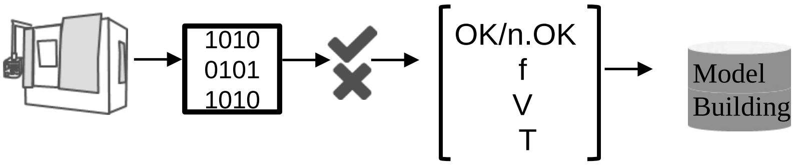
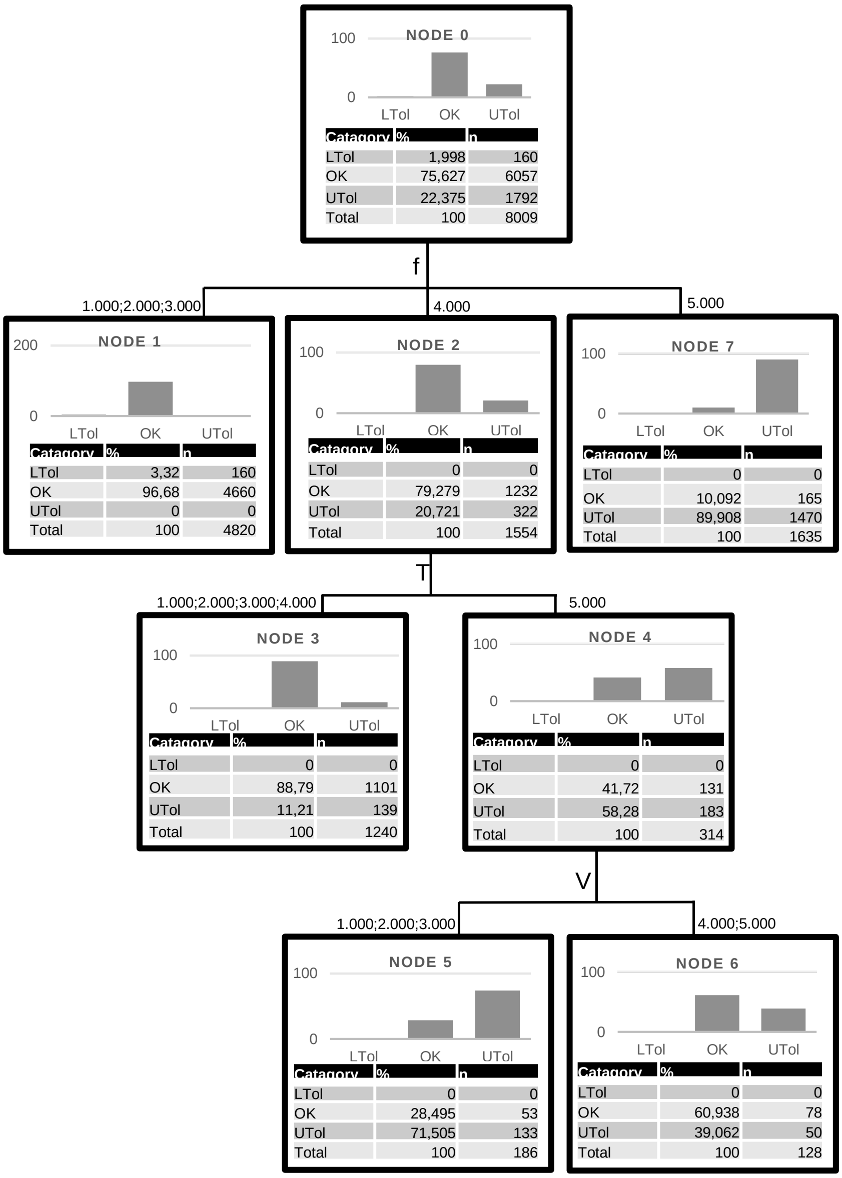


Fig. 3. Procedure of Model Building: Generated process data is labelled and, together with other parameters, serves as input for modelling



NODE 0

Catagory	%	n
LTol	1,998	160
OK	75,627	6057
UTol	22,375	1792
Total	100	8009

NODE 1

Catagory	%	n
LTol	3,32	160
OK	96,68	4660
UTol	0	0
Total	100	4820

NODE 2

Catagory	%	n
LTol	0	0
OK	79,279	1232
UTol	20,721	322
Total	100	1554

NODE 7

Catagory	%	n
LTol	0	0
OK	10,092	165
UTol	89,908	1470
Total	100	1635

NODE 3

Catagory	%	n
LTol	0	0
OK	88,79	1101
UTol	11,21	139
Total	100	1240

NODE 4

Catagory	%	n
LTol	0	0
OK	41,72	131
UTol	58,28	183
Total	100	314

NODE 5

Catagory	%	n
LTol	0	0
OK	28,495	53
UTol	71,505	133
Total	100	186

NODE 6

Catagory	%	n
LTol	0	0
OK	60,938	78
UTol	39,062	50
Total	100	128

Fig. 4. Resulting Decision Tree: Left bar – number of classified violations of the lower tolerance limit (n.OK-part); middle bar – number of classified OK parts; right bar – number of classified violations of the upper tolerance limit (n.OK-part)

In the resulting decision tree the impact of the different parameters on the output can be observed without any special expertise or knowledge about the product or the production process. It is correlated to the order the tree is split up. Possible root causes can be detected by interpreting the different bars of the terminal nodes (see Fig. 2) stepwise according to the procedure described in Chapter 3.1. Within the example, a high bar in the middle means of a box that this parameter setting leads to parts within the tolerance. High bars on the left (violation of the lower tolerance) or on the right (violation of the upper tolerance) mean that the parameter setting leads to parts, which are out-of-tolerance (n.OK). For users this means that certain parameter settings respectively combination of parameter settings lead to non-conformities. These settings can be seen as root causes for non-conformities and should be avoided in further production. By using process parameters as model input, the root causes are identified by decomposing the decision tree. It is not necessary to know root causes in advance. They are explicitly not integrated into the training of the algorithm. The use of the root causes for training leads to the fact that unknown root causes cannot be uncovered by algorithms and attempts are made to explain all non-conformities with existing root causes. In Fig. 4, the order of the splits indicates that the parameter f has the biggest impact on the result, after that T, then V. The analysis of the nodes determines that the parameter settings leading to node 1 and 3 lead to less non-conformities (OK parts – 96.68% for node 1, 88.79% for node 3; n.OK parts – 3.32% for node 1 and 11.21% for node 3). The parameter settings leading to the nodes 5 and 7 produce predominantly non-conformities (n.OK. parts – 71.5% for node 5, 89.9% for node 7). The stepwise analysis shows that primary root causes are to be found in node 7 and node 5. This shows that the root causes are a too high f (node 5; $f = 5$) or a too low V (node 7; $V = 1–3$) at $f = 4$ and $T = 5$.

4. SIMULATION-BASED PROOF OF THE APPLICABILITY OF A DECISION TREE FOR THE AUTOMATED ROOT CAUSE ANALYSIS

Within the small batch production scenario, simulations are used to provide a basis for the application of the decision tree. It provides users with knowledge about the algorithm and trains the model, which enables the RCA. The methodology will be described briefly to subsequently identify influencing parameters on the performance of the decision tree, to quantify them with a simulation and to show the applicability of the decision tree to real production scenarios with given characteristics (e. g. complexity or dimensions).

4.1. METHODOLOGY

Due to the low amount of available data in small batch production and the resulting non-applicabilty of ML algorithms, a production-related simulation model is set up to generate data with specific properties. The gained knowledge and the trained model enable the application to the data of the small batch production. Aim of the simulation is to quantify the influence of different scenarios (e.g. sample size or distribution) on the output of the decision tree in a small batch production. Various (performance) influencing parameters have been identified (e. g. sample size or number of parameters). By applying a Monte Carlo simulation to a process model, these parameters can be quantified. It also shows the general applicability of a decision tree to the use case, which will serve as the basis for the application to the real production scenario.

4.2. SIMULATION SCENARIOS

To prove the applicability of the decision tree algorithm to an automated RCA influences affecting the success of the detection are identified and a simulation to quantify the influences is conducted. The Monte Carlo simulation (using R) creates an input with specific properties, which trains a decision tree (created with IBM SPSS Modeler) as part of the automated RCA so that it can later be applied to real data from equivalent production processes. The properties can be taken from the real production scenario - e.g. limits, distributions, special features of the process. The simulation is based on a linear regression model by Davim J.P. [11] to determine the roughness of drilling holes and is described by the cutting speed, the feed rate and the cutting time (see equation (1)). It was chosen for simulation because it is a simple production-related model that describes the effects of process parameters on a product feature (which can be analysed afterwards). As a model, it simplifies the reality of the drilling process and does not represent all influences on the process. For the investigation of the applicability of ML algorithms, the model is sufficient.

$$R_a = 0.587 + 2.588 f - 1.911 \times 10^{-3} V + 8.130 \times 10^{-3} T \tag{1}$$

R_a – roughness [mm], f – feed [mm/rev], V – cutting speed [m/min], T – cutting time [min].

With the help of the simulation and a selection of tolerance limits, different scenarios of production processes can be implemented. To simulate the characteristics of a real drilling process, a Monte Carlo simulation randomises the input for the regression (whose output is again the input of the decision tree). Each input variable (*f*, *V*, *T*) underlies a uniform (UD) or normal distribution (depending on the scenario). In the context of the simulations, different parameters (tolerance limits, distributions, etc.) are varied. The parameters and their distribution are described in Table 1.

Table 1 Properties of the Input Parameters of the Simulation Model

Parameter	Mean	Standard Deviation (SD)	Limits of UD	Tolerance Limits
F	0.12	0.027	+/- 3SD	-
V	40	3.33	+/- 3SD	-
T	9	2	+/- 3SD	-
R_a	-	-	-	0.7197; 1.0575

When transferring the algorithms to real production processes, the sensitivity of the output should be considered in relation to the following points identified by a survey of experts in the fields of production and statistics:

1. The effect of the sample size on the amount of correctly identified root causes.
2. The effect of the input distribution on the result.
3. The necessary amount of data depending on the number of variables, higher order terms and interactions.
4. The ratio between categories of non-conformities and OK parts, describing how much non-conformities have to be in the data set to reliably built up a decision tree.
5. The number of intervals for the discretization.

Sample Size

In order to test the sensitivity of ML algorithms to the described points different production scenarios are modelled and simulated. Starting with the influence of the sample size on the amount of correctly classified data points, the simulation is performed with the input limited to a specific number of data points – the sample size – for each parameter (see equation (1)). The results of the algorithm were compared with results from a Monte Carlo simulation and the quotient of these two factors gives a percentage indication of how much data has been correctly classified – the Classification Index (CI). 80% of the data was used to train the model. The middle column in Table and Table indicates how high the CI is for training data. The remaining 20% of the data was used to test the model. The trained model is taken and evaluates how well the data (20%) can be explained by it (right column). As a result, the performance of the algorithm can be determined for different sample sizes and can be seen in Table and Table. Table and Table show that from a sample size of 100 for uniformly distributed data and from 50 for normal distributed data, the CI is within a range of 0.87 percentage points. The high percentage in the tests shows, that the model is not overfitted to the given dataset.

Table 2. Influence of the sample size on uniformly distributed input data ('-' : There is not enough data to create a decision tree. The data can not be split and the tree starts / ends in the root node)

Sample size	Training – Correct Classification in %	Test – Correct Classification in %
10	-	-
50	77.78	71.43
100	88.00	100
1,000	89.86	92.54
10,000	91.02	91.31
100,000	90.84	91.09
1,000,000	90.97	91.10

Table 3. Influence of the sample size on normally distributed input data ('-' : There is not enough data to create a decision tree. The data can not be split and the tree starts / ends in the root node)

Sample size	Training – Correct Classification in %	Test – Correct Classification in %
10	-	-
50	100	100
100	98.67	100
1,000	98.12	99.00
10,000	98.71	98.59
100,000	98.90	98.86
1,000,000	98.15	98.13

Comparison of different input distributions

If the results (Table and Table) between a uniformly distributed and normal distributed input of one sample size are compared, it is noticeable that the CI of the normal distribution is higher for all sample sizes. Applied to reality, the distribution of the input parameters has to be taken into account when choosing an algorithm.

Influence of interdependencies, higher order terms and the amount of variables

To analyse the dependencies between the variables and the number of correctly found root causes, the model is extended by further independent variables. The added variables are not based on any further simulations and serve only for clarification – they do not represent real influencing parameters and relationships. They serve the purpose of analyzing the algorithm and its influences.

A distinction can be made between the addition of linear terms, higher-order terms, and interaction terms. The following formulas were used for the addition of linear terms. The terms (b)-(d) *are continuously added to the basic formula (a) in equation* (2a–d).

$$\begin{aligned} &R_a = (0.587 + 2.588f - 1.911\times10^{-3}V + 8.130\times10^{-3}t) \quad \text{(a)} \\ &+ (0.289\times10^{-3}K) \quad \text{(b)} \\ &- (0.289\times10^{-3}W) \quad \text{(c)} \\ &+ (0.001\ \times Temp) \quad \text{(d)} \end{aligned} \tag{2a–d}$$

K – force, *W* – wear, *Temp* – drilling temperature, b–d – added terms

The results (sample size: 100,000) in Table show that there is no major deterioration of finding root causes, if a linear model can be assumed. For comparison, without added terms the CI randomly varies between 88% and 93% (comparison of (a) in Table 4 and the Test CI of 100,000 in Table 2 with the same model). It can be assumed that the performance of the algorithm is independent of the number of linear terms in the model.

Table 4. Influence of the dimensionality – linear added terms (b,c,d)

Sample size	(a) 3 linear terms CI	(b) 4 linear terms CI	(c) 5 linear terms CI	(d) 6 linear terms CI
100,000	90.9	90.9	90.8	91.2

For the investigation of the influence of higher order terms (reference: equation (1)), the following formulas were used describing terms of the second (equation (3); *T* is quadratic) and third order (equation (4); *V* is quadratic, *T* is cubic). As in equations 2a–d, the extensions of equation 1 are not based on physical relationships. The aim is to quantify the relationship between the performance of the ML and higher-order terms. The physical results for the surface roughness are not comparable with those of equation 1.

$$R_a = 0.587 + 2.588f - 1.911\times10^{-3}V + 8.130\times10^{-3}T^2 \tag{3}$$

$$R_a = 0.587 + 2.588f - 1.911\times10^{-3}V^2 + 8.130\times10^{-3}T^3 \tag{4}$$

Table 5. shows influence of higher order terms. The CI stays within a range of 3.9 percentage points. A slightly positive, proportional effect is suspected.

Table 5. Influence of the dimensionality – adding of higher order terms

Sample size	Linear term (equation 2a) CI	2nd order term CI	3rd order term CI
100,000	90.6	94.5	94

The last simulation includes two examples for additional interaction terms (reference: equation (1)). In equation 5, an interaction term of two variables (*V* and *T*) has been added - in Equation 6 the interaction replaces the individual effects of *V* and *T*. With regard to the physical effects, reference is made to the description of equations 2 and 3.

$$R_a = 0.587 + 2.588f - 1.911\times10^{-3}V + 8.130\times10^{-3}T + 0.289\times10^{-4}VT \tag{5}$$

$$R_a = 0.587 + 2.588f - 1.911\times10^{-3}V \times 8.130\times10^{-3}T \tag{6}$$

As a result (see Table 6) it can be shown that the more influence the interaction term has, the lower is the CI ($\mathrm{CI}_{\mathrm{Example2}}<\mathrm{CI}_{\mathrm{Example1}}$). It also shows that the amount of correctly found root causes is not affected (within this scenario – equation 5), if the linear terms have a higher effect on the dependent variable ($\mathrm{CI}_{\mathrm{Without\ IA}}\approx\mathrm{CI}_{\mathrm{Example1}}$).

Table 6. Influence of the dimensionality – adding interactions (IA)

Sample size	Without IA CI	IA Example 1 CI	IA Example 2 CI
100,000	90.6	91.2	81.5

Effect of n.OK/OK-*ratio*

For the determination of the influence of the ratio between n.OK and OK, the ratio is continuously reduced by enlarging the permitted tolerance and therefore reducing the number of n.OK parts. Table 7 shows that the CI stays acceptable, as long as the ratio of n.OK/OK stays above 7.07%. For ratios lower than that, no decision tree can be created anymore – the algorithm cannot split the data and therefore all data is categorized as OK.

Table 7. Influence of the n.OK/OK – ratio

NOK/OK	17.23%	8.69%	7,41%	7.07%
CI (Training)	91.99	93.12	90.05	91.74
CI (Test)	89.05	93.53	93.24	93.03

Influence of the discretization

The C5.0 algorithm was used for the creation of the decision tree. Therefore, the data was discretized. The following simulation shows the influence of the increasing number of intervals of the CI. Starting with five intervals the amount is increased as long as the tree changes due to the added intervals (for numbers of interval bigger than 40, the tree does not change). As seen in Table 8 the index improves until it reaches a maximum (*) at 15 intervals. A further increase in the number of intervals leads to an overfitted model respectively a model that cannot handle the number of intervals. Therefore, the performance of the decision tree decreases.

Table 8. Influence of the discretization

# intervals	5	10	15	20	25	30	35	40
CI	88.06	94.03	95.02*	88.06	91.04	89.55	91.04	79.6

The simulation demonstrated the applicability of the decision tree for an automated RCA of non-conformities. Through the different simulations it became clear which aspects (sample size, distributions, etc.) in the transfer to real scenarios have to be considered and how these influence the performance of the algorithm. It is shown that when using ML for RCA, the sample size as well as terms of higher order or interactions have to be considered.

5. SUMMARY AND OUTLOOK

With the help of different *simulated* experiments on a simple linear model for the surface roughness it was shown that a decision tree can be applied to the problem of automated RCA of non-conformities without any special knowledge of the process. After identifying critical parameters for the applicability of ML to production scenarios, they were investigated using a Monte Carlo simulation. Additionally, the simulations clarify limits in the application of the C5.0 decision tree algorithm for the detection of root causes of non-conformities contrasting the advantage of an automated, non-knowledge based analysis.

In future research, the insights gained from the simulation as well as the resulting production-related simulation model will be applied to a real small batch production scenario. A real drilling process will be used for the final confirmation of the approach. In addition to the decision tree, further ML algorithms need to be applied to the problem to compare their applicability. In particular, the unsupervised learning method ‘Association Rule Learning’ is considered as interesting. This can be used to objectively discover new relationships without having any knowledge about the data. It should also be examined how the application of a combination of different algorithms (possibly from different categories of ML – described in Chapter 3) to the given question can lead to further results.

REFERENCES

[1] MUNOZ P., DE LA BANDERA I., KATHIB E.J., GÓMEZ-ANDRADES A., SERRANO I. BARCO R., 2017, *Root Cause Analysis Based on Temporal Analysis of Metrics Toward Self-Organizing 5G Networks,* IEEE Transactions on Vehicular Technology, 66, 3.

[2] UHLMANN A. HOHWIELER E., GEISERT C., 2017, *Intelligent Production Systems in the Era of Industrie 4.0 – Changing Mindsets and Business Models,* Journal of Machine Engineering, 17/2, 5.

[3] LANGLEY P., SIMON H. A., 1995, *Applications of Machine Learning and Rule Induction*, Institute for the Study of Learning and Expertise, Palo Alto.

[4] WUEST T., WEIMER D., IRGENS D., THOBEN K.-D., 2016, *Machine Learning in manufacturing: advantages, challenges, and applications,* Journal of Production & Manufacturing Research, 4/1, 23.

[5] SMOLA A., VISHWANATHAN S.V.N., 2008, *Introduction to Machine Learning,* Cambridge.

[6] UHLMANN E., PASTL PONTES R., LAGHMOUCHI A., HOHWIELER E., FEITSCHER R., 2017, *Intelligent Pattern Recognition of SLM Machine Energy Data,* Journal of Machine Engineering, 17/2, 65.

[7] DUPHILY R.J., 2014, *Root Cause Investigation Best Practices Guide,* AEROSPACE, Chantilly.

[8] DOGGET A.M., 2005, *Root Cause Analysis: A Framework for Tool Selection,* The Quality Management Journal, 12/4, 34.

[9] DOGGET A.M., 2004, *A Statistical Comparison of Three Root Cause Analysis Tools,* Journal of Industrial Technology, 20/2.

[10] YANG K., TREWN J., 2004, *Multivariate statistical methods in quality management*, McGraw-Hill, New York.

[11] DAVIM J.P., 2003, *Study of drilling metal-matrix composites based on the Taguchi techniques,* Journal of Materials Processing Technology, 132.

[12] THOMPSON J.M., OLSON M.S., VISWANATHAN G., HAWKS B.A., DOSHI B.A., 2015, *Automated root cause analysis,* US Patent, US 9104572 B1.

[13] PEDERSEN H., 2017, *Automated root cause analysis,* US Patent, US 9606533 B2.

[14] QURESHI W., HASSAN T., ROACH K.B., BALA G.P., 2011, *Automated root cause analysis of problems associated with software application deployments,* US Patent, US 8001527 B1.

[15] PFINGSTEN J.T., 2007, *Machine Learning for Mass Production and Industrial Engineering,* Dissertation, Tübingen.

[16] SHALEV-SHWARTZ S., BEN-DAVID S., 2014, *Understanding Machine Learning – From Theory to Algorithms,* Cambridge.

[17] GÈRON A., 2017, *Hands-on machine learning with Scikit-Learn and TensorFlow. Concepts, tools, and techniques to build intelligent systems*, 2nd release. Beijing [etc.], O'Reilly.

[18] LANTZ B., 2013, *Machine Learning with R,* Packt Publishing, Birmingham.

[19] SEN K., VISWANATHAN M., AGHA G., 2004, *Statistical Model Checking of Black-Box Probabilistic Systems*, CAV 2004, Computer Aided Verification, 202.

[20] RUTKOWSKI L., JAWORSKI M., PIETRUCZUK L., DUDA P., 2014, *The CART decision tree for mining data streams,* Information Science, 266.

[21] ANYANWU M.N., SHIVA S.G., 2009, *Comparative Analysis of Serial Decision Tree Classification Algorithms,* International Journal of Computer Science and Security, Malaysia, 3/3, 230.

[22] CURRAM S.P., MINGERS J., 1994, *Neural Networks, Decision Tree Induction and Discriminant Analysis: An Empirical Comparison*, The Journal of the Operational Research Society, 45/4, 440.