# ANALYSIS OF SIMULATED AND RECORDED DATA OF CAR FLEETS BASED ON MACHINE LEARNING ALGORITHMS

Marcin Hinz[1], Fabian Hienzsch[2], and Stefan Bracke[3]

[1] *University of Wuppertal, Chair of Reliability Engineering and Risk Analytics, Gaussstrasse 20, Wuppertal, Germany, m.hinz@uni-wuppertal.de*
[2] *University of Wuppertal, Chair of Reliability Engineering and Risk Analytics, Gaussstrasse 20, Wuppertal, Germany, fabian.hienzsch-hk@uni-wuppertal.de*
[3] *University of Wuppertal, Chair of Reliability Engineering and Risk Analytics, Gaussstrasse 20, Wuppertal, Germany, bracke@uni-wuppertal.de*

*Knowledge of failure behavior and failure modes regarding the complete life cycle is fundamental within the early development phases in the automotive industry. The goal is the detection of product risks and the analysis of the product reliability. Hence, the challenge is the analysis of the potential prospective customer usage conditions and their transformation in prototype test procedures during the product development process.*
*Efficient testing of components, products and systems has become a very challenging task for reliability engineers within all technical disciplines. Detection and application of the right boundary conditions in the test plans appears to be the key solution for the increase of the testing efficiency. Therefore, the optimization of the testing plans requires a specific and systematic analysis of the field behavior in form of the field data.*
*Machine learning addresses the issue of recognizing patterns and other regularities of data with the aim to find general rules and to make predictions. Machine learning algorithms are available in a wide range and can be used for statistical analysis and data mining, especially in case of big data problems.*

## I. INTRODUCTION

The aim of this paper is the analysis of a fleet behavior and comparison of claimed and non-claimed vehicles using different machine learning algorithms to identify the main differences in a certain fleet. For this purpose OBD (On Board Diagnostics) signals are recorded in various car types and drives. On the other hand, lots of drives are simulated based on the Discrete Fourier Transformation (DTF) and Monte Carlo simulation (MC) to provide a huge training data for the rule learning algorithms. All signals are analyzed in order to provide information about every single drive (out of the speed signal e.g. max speed, max acceleration, duration of the drive are determined and stored in a data base).

In this paper, data in form of matrixes with a dimension of more than 5.000 of cars and more than 100 of variables is analyzed. The expected result is the rule / pattern for the cause of a damage of a vehicle.

The found patterns (distinctive features), which distinguish the damaged and non-damaged vehicles are the basis information for the optimization of the test procedures of the next generation products. This approach provides a direct link between the analyzed field data out of the usage phase of a product and the test plans out of the development phase of the next generation product (cf. Figure 1).

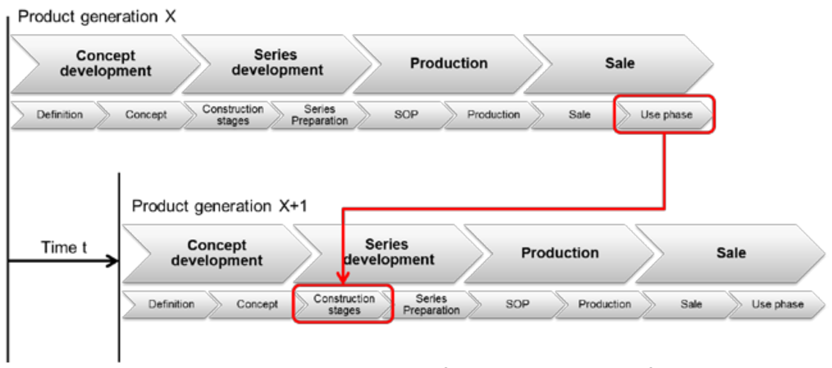

Fig. 1. – Link between development phases of two generations of a technical product

## II. ON BOARD DIAGNOSTICS - OBD II

On-board diagnostics (OBD) refers to a vehicle´s diagnosis and reporting capability and is part of almost all cars and light trucks on the road today. Since the '70s car manufacturers started using electronic means to control engine functions and diagnose technical problems, primarily to meet EPA (Environmental Protection Agency) emission standards. Over time these diagnostic systems have become more sophisticated and powerful. In the early stages, there were few standards and each car manufacturer implemented their own proprietary systems and signals. Over the years the situation improved though and companies agreed on standardized digital communication ports. The latest standard is called OBD-II and was introduced in the mid-'90s. It provides almost complete engine control and also monitors parts of the chassis, body and accessory devices, as well as the diagnostic control network of the car to be sure the car is performing to OEM standards.

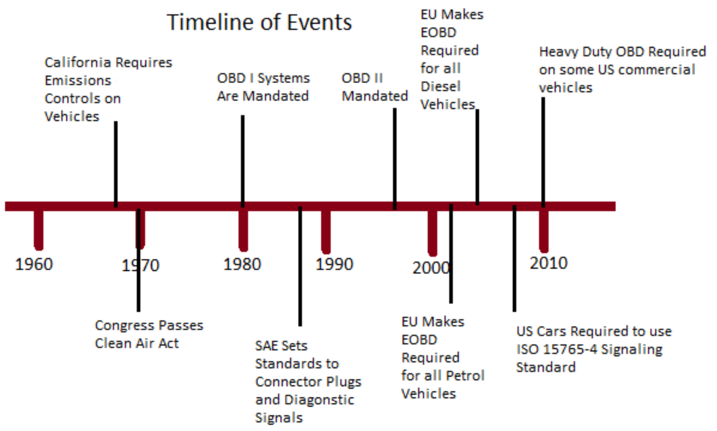
Timeline of Events
California Requires Emissions Controls on Vehicles
OBD I Systems Are Mandated
OBD II Mandated
EU Makes EOBD Required for all Diesel Vehicles
Heavy Duty OBD Required on some US commercial vehicles
1960
1970
1980
1990
2000
2010
Congress Passes Clean Air Act
SAE Sets Standards to Connector Plugs and Diagonstic Signals
EU Makes EOBD Required for all Petrol Vehicles
US Cars Required to use ISO 15765-4 Signaling Standard

Fig. 2. – Historical development of the OBD II standard (cf. [F1])

The OBD-II standard specifies the type of diagnostic connector and its pinout, the electrical signaling protocols available, and the messaging format including an extensible list of DTCs (“Diagnostic Trouble Codes”). As a result of this standardization, a single device can query the on-board computer(s) in any vehicle.

By issuing specific diagnostic codes, OBD II pinpoints the problem, allowing garage technicians to make quicker and more effective repairs. This saves both time and money. OBD II is an early warning system that alerts vehicle owners about various problems in the car at a stage where repairs might be less costly or perhaps even done during the warranty period. OBD II inspections are also quicker and more efficient than previous tests [F20].

The development and the functionality of OBD II systems were explained in detail many international standards (e.g. [F20] and [F21]). For the purpose of this paper, the data was collected with a standard OBD II data logger with 2 MB memory connected to the serial CAN-Bus interface. Following cars have been recorded:

- BMW E83 [*1] (2.0d, R4 engine type, production date 2007) – 84 rides with 8 signals each
- Peugeot 208 [*2] (HDi FAP 68, production date 2014) – 42 rides with 17 signals each
- Seat Leon [*3] (1.2 TSI, production date 2013) – 30 rides with 28 signals each

All in all, 156 car rides with various numbers of signals and various lengths (kilometrage and time) have been recorded. Due to the comparability of the car rides, only eight signals have been chosen for further analysis (cf. Table I). It shall be noticed that the air flow rate is not measured and recorded in case of Seat Leon [F1].

TABLE I. List of the data logger-readable signals for each car type

| Signal | *1 | *2 | *3 |
|---|---|---|---|
| Vehicle speed | x | x | x |
| Engine RPM | x | x | x |
| Engine coolant temperature | x | x | x |
| Engine load | x | x | x |
| Air flow rate | x | x | |
| Intake air temperature | x | x | x |
| Intake manifold absolute pressure | x | x | x |
| Battery voltage | x | x | x |

## III. SIGNAL SIMULATION

Different signals of OBD-parameter of various vehicles were recorded with an OBD data logger. Acquired results of analyzed OBD II (On Board Diagnostics) specific data of different signals are used as input for the simulation models. OBD II signals can be divided in signals with and without dynamic behavior (see Figure 3) [F1]. As a consequence two different models for signal simulation with different approaches were developed (adapted to the respective signal type). The first model (DFT/MCS Method) is based on transformed data by Discrete Fourier Transformation (DFT) [F2] and is used for simulating signals with dynamic characteristics. The second model uses fitted Poisson Distributions [F3] for simulating signals without dynamic characteristics. Both simulation methods are based on the Monte Carlo simulation method [F4].

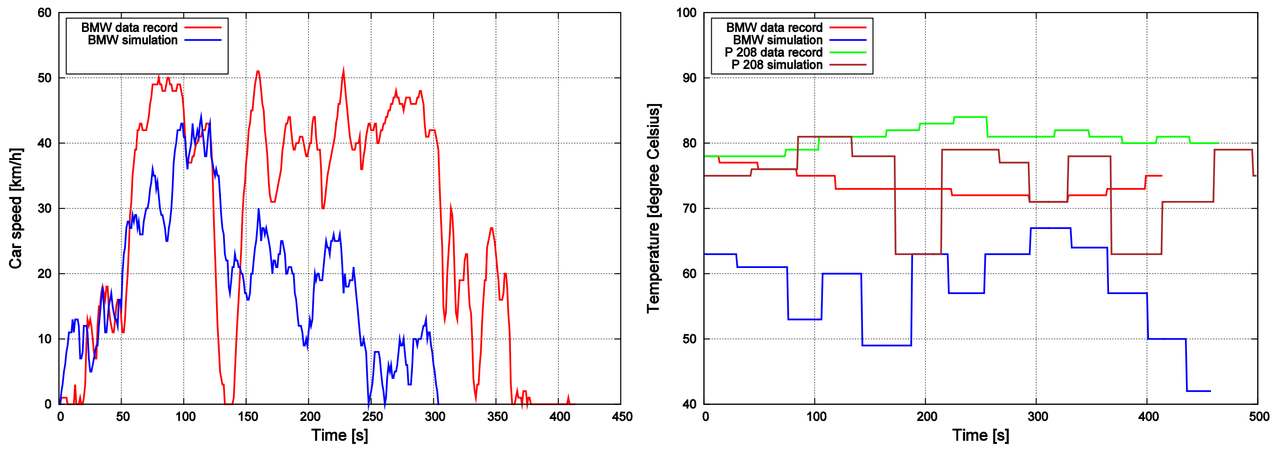

Fig. 3. Signals with dynamic (left chart) and without dynamic behavior (right chart)

### III.A. Modeling

The DFT/MCS Method [F5] uses data of a previously performed analysis but can also be built up on empirical knowledge. The DFT was applied on a sample of OBD II signals where the amplitudes and phases of the signals were calculated. Furthermore, the contained frequencies and number of data elements of each signal were determined. According to Figure 4 the DFT/MCS Simulation Model is performed as follows.

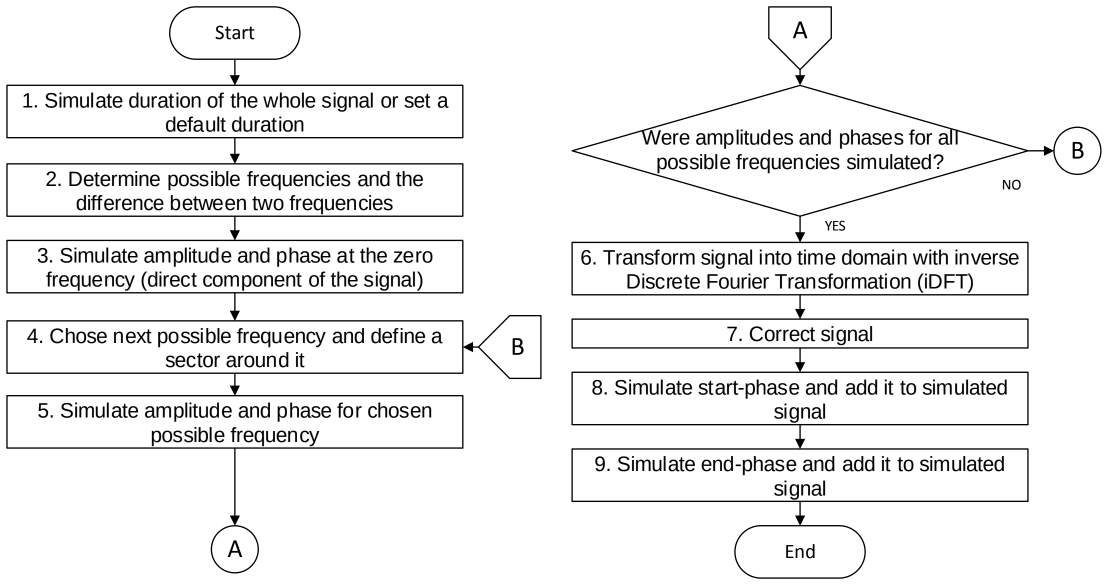
Start
1. Simulate duration of the whole signal or set a default duration
2. Determine possible frequencies and the difference between two frequencies
3. Simulate amplitude and phase at the zero frequency (direct component of the signal)
4. Chose next possible frequency and define a sector around it
B
5. Simulate amplitude and phase for chosen possible frequency
A
A
Were amplitudes and phases for all possible frequencies simulated?
NO
B
YES
6. Transform signal into time domain with inverse Discrete Fourier Transformation (iDFT)
7. Correct signal
8. Simulate start-phase and add it to simulated signal
9. Simulate end-phase and add it to simulated signal
End

Fig. 4. Flow-chart of the DFT/MCS Simulation Model

Step 1: In this step, the duration of the whole signal is MC simulated. In a previously performed data analysis, the durations of all OBD II signals of the control sample were determined. A Weibull distribution function [F6] is fitted to these data and a random number generator is used to create a value which corresponds to the fitted distribution. Alternatively a default value can be set instead of simulating.

Step 2: Based on the previously simulated signal duration or the default duration, all values of included frequencies of the signal are determined. Furthermore, the difference between two successive frequencies is calculated. Another important parameter is the maximum frequency of 0.5Hz. It is given by the sampling rate [F7] of the data logger which has recorded the data of the control samples.

Step 3: Every discrete signal includes amplitudes at the frequency f=0 which indicates the direct component of a signal [F8]. The value of the phase angle is always zero, whereas the amplitudes differ from one signal to another. Therefore, only the amplitude needs to be simulated in this case. This is done by a random number generator based on a fitted Weibull distribution to the data of the amplitudes which are gathered out of the data analysis (MC Simulation).

Step 4 and 5: After the direct component is determined, the amplitudes and phases for all remaining frequencies are simulated. A sector around every frequency is defined with the upper and lower bound of $\Delta f/2$. In each sector a subset of amplitude and phase values is formed. The amplitude is MC simulated in the same way as described before. The phase values though differ from sector to sector and don't follow a distribution model. Therefore, instead of MC Simulation, a phase value is drawn out of the belonging subset for each remaining frequency.

It should be noted that the simulation is performed only for frequencies less or equal to the maximum frequency of 0.5 Hz. This is caused by the shape of the amplitude and phase spectrum of signals. At the maximum frequency, values of amplitudes are axisymmetric (vertical); values of phases are point symmetric. Thereby, at frequencies greater than 0.5 Hz, amplitudes and phases are derived from the previously simulated ones. The first amplitude at f=0 (direct component) however is contained only once in a signal and is not part of the stated symmetries.

Step 6: After all values for amplitudes and phases and their belonging reflections were simulated, the inverse Discrete Fourier Transformation (iDFT) is used to transform the signal back into the time domain.

Step 7: The transformed signal has to be corrected, because it may contents negative values. Therefore, the signal is shifted in order to set the minimal included value on a specified level (e.g. zero level).

Step 8 and 9: So far simulated velocity signals doesn't start and end at 0 km/h. Therefore the start- and end-phase of the signal need to be simulated. Herewith, local extrema of the signal are determined and difference values are calculated. Subsequently, Regression lines [F9] of the relationship between time and velocity differences are created. They are used to determine the duration for reaching the first velocity value of the simulated signal (duration of start phase) and to determine the duration from the last simulated velocity value to standstill (duration of stop phase). Then, the interim velocity values of start and end phase are approximated with a straight line.

On the contrary, if a default signal duration is set (see Step 1) the start and end phase will be calculated in a different way. A fraction of 5% of the default signal duration is reserved for start and end phase and thus no regression lines are needed.

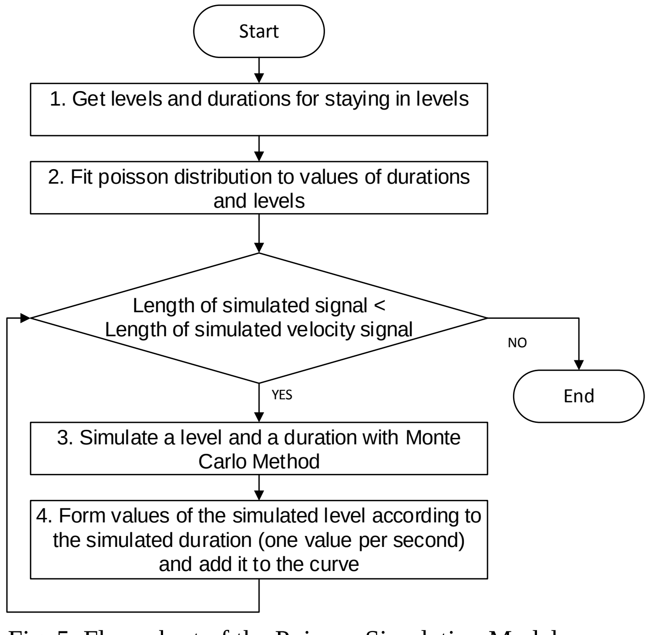
Start
1. Get levels and durations for staying in levels
2. Fit poisson distribution to values of durations and levels
Length of simulated signal < Length of simulated velocity signal
NO
End
YES
3. Simulate a level and a duration with Monte Carlo Method
4. Form values of the simulated level according to the simulated duration (one value per second) and add it to the curve

Fig. 5. Flow-chart of the Poisson Simulation Model

It should be noted that if the duration of the simulated signal (step 1) is much higher than durations of the control sample, the simulation can't be performed because the data in the defined frequency sectors (step 4) are too less. In this case, a number of simulated signals with duration of the 90%-quantile [F10] of the control sample and a remaining rest will be combined. This yields a signal with the original simulated duration.

Signals without dynamic characteristics can be simulated with the Poisson-Method. The Poisson-Method is performed as follows (see Figure 5).

Step 1: In the first step the control samples of the recorded signals are analyzed. All included value levels and their absolute frequencies are detected and stored. Furthermore, signals without dynamic characteristics include different time durations for staying in a value level. These durations are calculated and stored.

Step 2: In this step two Poisson distributions [F3] are fitted to the data (one to the detected levels, the other to the durations of staying in a level) with the maximum likelihood method [F3].

Step 3 and 4 are performed in a loop which is executed as long as the condition (length of simulated signal < default length) is fulfilled. The condition of the loop is set because the total length of the simulated signal shall correspond to a set default length.

Step 3: Levels and durations will be MC simulated in every loop run. A random number generator is used to create a value which follows the respective fitted distribution.

Step 4: After the simulation of a level and duration the data is used to form a value series. The number of values of the series corresponds to the simulated duration. In every loop run new simulated value series are added to the previously simulated.

After the loop is terminated a fully simulated signal without dynamic behavior is existent. However, the signal length can be greater than the default length (caused by the last simulated duration) and must therefore be cut.

The whole algorithm is performed in a loop with a specified number of simulation runs. For getting simulated rides with the corresponding signal lengths to a set of previously simulated velocity signals, the default length equates to the length of a simulated velocity signal and is adjusted in every simulation run.

## III.B. Strategy of the fleet simulation

In this simulation strategy 5000 rides, which consist of signals of selected OBD II parameters, were simulated for the mentioned vehicles which leads to a total amount of 15.000 rides and 115.000 signals in total.

Recorded OBD II rides (real data) of each vehicle were used as input for the algorithms in order to perform the simulations. Table II gives an overview about the selected OBD II parameters of which simulations were performed. Control samples of the parameters are available for the mentioned vehicles except of the air flow rate for the Seat Leon.

As described above, the signals of the parameters can be grouped into signals with and without dynamic behavior (see figure 1). Depending on the behavior, one of the previous described simulation algorithms (DFT/MCS or Poisson) is applied (see table II).

TABLE II. Overview about selected OBD parameter for simulation

| **OBD parameter** | **unit** | **signal behavior** | **Algorithm** |
|---|---|---|---|
| Vehicle speed | [km/h] | dynamic | DFT/MCS |
| Engine RPM | [RPM] | dynamic | DFT/MCS |
| Engine coolant temperature | [°C] | non dynamic | Poisson |
| Engine load | [%] | dynamic | DFT/MCS |
| Air flow rate | [g/s] | non dynamic | Poisson |
| Intake air temperature | [°C] | non dynamic | Poisson |
| Battery voltage | [V] | non dynamic | Poisson |

A simulation of a vehicle starts always with the vehicle speed parameter. Signal simulations of other parameters are performed afterwards and its length corresponds to the signal lengths of the previous simulated vehicle speed signals. In the end, every simulated signal of every parameter of a ride has the same length (number of elements).

## IV. SIGNAL ANALYSIS

Once all signals (measurements and simulations) are available, the analysis (generation of characteristic information as well as additional information) can be performed. For that purpose, an analysis tool based on KNIME software [5] and R programming language has been created. KNIME is a workflow based on open source graphic interface with the possibility of implementation and interpretation of R code. The programmed workflow distinguishes between three different types of signals and provides relevant information.

Characteristic information like maximum, minimum, median, mean, range is calculated in case of all signals. Additional information like e.g. maximum and minimum acceleration, driving pattern (city drive, freeway drive, and highway drive), driven distance (divided additionally into short and long distance) can be derived from the velocity signal. In case of the signals with dynamic characteristics the Discrete Fourier Transformation is performed and a given number (e.g. the first ten) of frequencies and amplitudes are stored. Signals without dynamic characteristics are analyzed also in terms of number of positive and changes of the signals (e.g. the temperature goes up or down).

The whole analysis process provides lots of information about the operating conditions of a product. The analysis of the eight signals (cf. Table II) results in more than 100 characteristics, which leads to matrices (as an input for the rule learning algorithms – training data) with dimensions of over 5.000 rows and 100 columns up to over 15.000 rows and 100 columns.

## V. RULE LEARNING ALGORITHMS

Machine Learning (ML) is a field within artificial intelligence where algorithms are applied on data sets to detect relationships between data and information. It is used in many fields of application and problems in big data analytics, pattern recognition and information evolution. The general task of ML is to find a hypothesis that estimates the target function by generalization of the training experience (training data set). The found hypothesis can be applied for making accurate predictions on new data which are not part of the training process and may content new cases. [F11]

### V.A. Theory

There is a wide range of different Machine Learning algorithms which adopt different methods of learning. Algorithms which are used in this paper follow the concept of Inductive Learning which provides the possibility to recognize patterns and regularities out of training data [F12]. Within Inductive Learning there are several approaches like induction of decision trees, rule induction and instant-based learning [F13]. The following three different types of algorithms which follow different induction approaches are applied for data analysis in this paper:

- C4.5 (J48) [F14] (Approach: Decision Tree Induction)
- RIPPER (JRip) [F15] (Approach: Rule Induction)
- NNge [F16] (Approach: Instance-Based Learning)

**C4.5**: The C4.5 algorithm [F14] or classifier uses training data which consist of classes and a belonging vector of describing data of different attributes as input. The algorithm can be used for predicting the class of new data with previously unseen cases. [F11] The output is a decision tree, which consist of leafs, branches and nodes. A node of the tree represents a feature or an attribute of the training data set and is specified by incoming and outgoing branches which are characterized by values and conditions and used for connecting nodes. On the contrary, leafs of the tree mark the classes (command variables) and their belonging relationships of the attributes can be determined by retracing the way of connected branches and nodes through the tree. [F11]

**RIPPER**: The RIPPER algorithm [F15] also uses training data of the same structure as C4.5. First of all the training data set is sorted by the frequencies of the classes with ascending order. Then rules are learned for each class label with exception of the class with the greatest frequency of instances. If a rule has been found for a class, all covered instances are removed from the training data and the procedure of finding rules is applied again until no more instances are available or if the last found rule has reached a specified level of complexity. Afterwards, the algorithm continues with the next class. The last found rule is always the default rule which expresses, that instances which cannot be classified with the previously found rules belong to the class of the greatest frequency. [F17]

**NNge**: The NNge algorithm [F16] uses training data of the structure described above for the learning process. The algorithm pursues the goal of constructing hyperrectangles out of the training data. A hyperrectangle covers a portion of training instances and its dimension is determined by the range of values in case of a numerical attribute or by the number of items in case of a nominal attribute. Moreover it is orientated to a class label – instances with a different class are treated as conflicts. The process of determining hyperrectangles of NNge is performed for each instance in the following three successive steps:

- Classification: Determine the closest hyperrectangle to the training instance
- Model adjustment: Split the previous determined hyperrectangle if it contains conflicts
- Generalization: Extend the hyperrectangle around the training instance. Generalization proceeds only if no conflicts are covered and if there is no overlapping with other hyperrectangles [F18]

### V.B. Application

Training data in form of a matrix with dimensions of about 5.000 rows and 100 columns is available for every car of a certain fleet. With this amount of data three different analysis cases have been performed:

- Comparison of one car (with a given amount of rides – e.g. 84 for BMW X3) and the simulated data of the same car type. This case is a common analysis of a cause for a damage of one certain car. The dimensions of the trainings data in for of a matrix is over 5.000 rows (simulation + record) and 100 columns.
- Comparison of one car and the simulated data of all car types. This represents a typical analysis of a carry-over part (e.g. a bulb placed in many car types). Here, additionally to the previous analysis, it shall be defined if the damage is specific only in one car type or for all of them. The dimensions of the trainings data is over 15.000 rows (all simulations + one record) and 100 columns.
- To support the analysis of carry-over parts, all simulations and all records are defined as the training data. This analysis shows the differences in the test plans of various car manufacturers for the same parts. The dimensions of the trainings data is over 15.000 rows (all simulations + all records) and 100 columns.

## VI. DISCUSSION OF THE RESULTS

The best results were achieved with the NNge algorithm which gives a complete (and not the shortest) rule as a reason for the damage. Rules realized by this algorithm have almost the same length as the number of characteristic information after the signal analysis process. This means that almost all characteristics are included in the rule, which gives a very precise description of the damage reason.

In contrary, RIPPER and c4.5 provide the smallest possible rule which defines the damage case (in many cases due to measurement errors consisting of only one characteristic). For example all records start with 0 engine RPM (first measure before star) and all simulations start by 800 RPM (due to the specs). This leads to a simple rule, which states that all damaged cars have minimum engine RPM equal to 0. Based on the analysis of the trainings data it is true but still doesn't provide the right answers to the technical question.

This may lead to misinterpretation and wrong understanding of the real cause of damage. For this reason, it is recommended to analyze with several machine learning algorithms and compare the results. The computational time for the application of machine learning algorithms by training data of mentioned dimensions is negligible.

In order to optimize the test procedures, the achieved results need to be transformed back into signals and reproduced during the test phases of a technical product. This procedure is straightforward and simple to apply.

## REFERENCES

F1. Hinz, M. Temminghoff, P. and Bracke, S.: *Optimization of test procedures based on OBD system specific field data.* RAMS 2016 – 62nd Reliability and Maintainability Symposium, Ticson, Arizona, U.S.A. (2016)

F2. Brigham, E. Oran: The fast Fourier transform and its applications. Englewood Cliffs, N.J.: Prentice Hall, Prentice-Hall signal processing series, 1988

F3. E. Cramer and U. Kamps, *Grundlagen der Wahrscheinlichkeitsrechnung und Statistik: Ein Skript für Studierende der Informatik, der Ingenieur- und Wirtschaftswissenschaften,* 2nd ed. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg, 2008

F4. C. Lemieux, *Monte Carlo and quasi-Monte Carlo sampling*. New York, NY: Springer, 2009

F5. M. Hinz, F. Hienzsch, and S. Bracke, *Development of two methods for the characterisation of an automotive fleet behaviour based on the simulation of single car rides*. UK, Glasgow: European Safety and Reliability Association, 2016.

F6. C.-D. Lai, *Generalized Weibull Distributions*. Berlin: Springer Berlin, 2013.

F7. O. Beucher, *Signale und Systeme: Theorie, Simulation, Anwendung: Eine beispielorientierte Einführung mit MATLAB ; mit 115 Beispielen, 159 Übungsaufgaben und 220 MATLAB-Programmen,* 2nd ed. Berlin: Springer Vieweg, 2015.

F8. M. Werner, *Signale und Systeme: Lehr- und Arbeitsbuch mit MATLAB®-Übungen und Lösungen*, 3rd ed. Wiesbaden: Vieweg+Teubner / GWV Fachverlage GmbH Wiesbaden, 2008.
F9. X. Yan and X. Su, *Linear regression analysis: Theory and computing*. Singapore, Hackensack, NJ: World Scientific, 2009.
F10. M. Feindt and U. Kerzel, *Prognosen bewerten: Statistische Grundlagen und praktische Tipps*. Berlin: Springer Gabler, 2015.
F11. M. Awad and R. Khanna, *Efficient learning machines: Theories, concepts, and applications for engineers and system Designers*. New York: Apress Open, 2015.
F12. A. M.AlMana and M. Aksoy, "An Overview of Inductive Learning Algorithms," (en), *IJCA*, vol. 88, no. 4, pp. 20–28, 2014.
F13. P. Domingos, "Rule Induction and Instance-Based Learning: A Unified Approach," in *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, C. S. Mellish, Ed, San Mateo, Calif.: Morgan Kaufmann, 1995, pp. 1226–1232.
F14. S. L. Salzberg, "C4.5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc, 1993," (en), *Mach Learn*, vol. 16, no. 3, pp. 235–240, 1994.
F15. W. W. Cohen, "Fast Effective Rule Induction," in *Machine Learning: Proceedings of the Twelfth International Conference*, M. Kaufmann, Ed, 1995.
F16. M. Brent, "Instance-Based learning: Nearest Neighbor With Generalisation," University of Waikato, Hamilton, 1995.
F17. J. Hühn and E. Hüllermeier, "FURIA: An algorithm for unordered fuzzy rule induction," (en), *Data Min Knowl Disc*, vol. 19, no. 3, pp. 293–319, 2009.
F18. J. D. Zaharie, L. Perian, and V. Negru, "A view inside the classification with non-nested generalized exemplars," (en), *IADIS European Conference Data Mining 2011*, 2011.
F19. http://www.cvel.clemson.edu/auto/systems/OBD.html
F20. ISO 27145-3:2012: *Road vehicles - Implementation of World-Wide Harmonized On-Board Diagnostics (WWH-OBD) communication requirements.*
F21. ISO 27145-1:2012-08 (E): *Road vehicles - Implementation of World-Wide Harmonized On-Board Diagnostics (WWH-OBD) communication requirements - Part 1: General information and use case definition.*