DOI: 10.1111/cgf.14452

COMPUTER GRAPHICS *forum*
Volume 0 (2022), number 0 pp. 1–14

RfX: A Design Study for the Interactive Exploration of a Random Forest to Enhance Testing Procedures for Electrical Engines

J. Eirich,[1,2] M. Münch,[2,3] D. Jäckle,[2] M. Sedlmair,[4] J. Bonart[2,5] and T. Schreck[6]

[1]University of Bamberg, Bamberg, Germany
[2]BMW Group, Munich, Germany
[3]Universtiy of Ulm, Ulm, Germany
[4]University of Stuttgart, Stuttgart, Germany
[5]Fraunhofer IWU, Dresden, Germany
[6]Graz University of Technology, Graz, Austria

Abstract

Random Forests (RFs) are a machine learning (ML) technique widely used across industries. The interpretation of a given RF usually relies on the analysis of statistical values and is often only possible for data analytics experts. To make RFs accessible to experts with no data analytics background, we present RfX, a Visual Analytics (VA) system for the analysis of a RF's decision-making process. RfX allows to interactively analyse the properties of a forest and to explore and compare multiple trees in a RF. Thus, its users can identify relationships within a RF's feature subspace and detect hidden patterns in the model's underlying data. We contribute a design study in collaboration with an automotive company. A formative evaluation of RFX was carried out with two domain experts and a summative evaluation in the form of a field study with five domain experts. In this context, new hidden patterns such as increased eccentricities in an engine's rotor by observing secondary excitations of its bearings were detected using analyses made with RfX. Rules derived from analyses with the system led to a change in the company's testing procedures for electrical engines, which resulted in 80% reduced testing time for over 30% of all components.

Keywords: human–computer interfaces, interaction, visual analytics, visualization

CCS Concepts: • Human-centred computing → Visual analytics; Systems and tools for interaction design; Graphical user interfaces

1. Introduction

In this paper, we contribute a design study on the use of *Random Forest* (RF) visualization for analysing data from test stations in a manufacturing process of the automotive industry. We address the problem of detecting faulty electrical vehicle parts that do not meet the quality requirements and thus need to be excluded from the manufacturing process. With the fast up-ramping of new assembly lines for electric vehicles across the globe [HKB*19], this problem has gained much attention across industries recently. To detect faulty parts, engineers typically rely on recording and analysing signal measurements, as widely used in many other engineering domains [SMF*20, EJS*20].

Traditionally, engineers analyse one measurement at a time and compare it to a ground truth baseline measurement. However, with the advent of new technologies, this one-at-a-time approach does not scale anymore. For testing modern electrical engines, for instance, engineers in our design study need to analyse more than 20 partially dependent signals within a single test procedure. Apart from the large burden of increasing manual work, it is also not possible to understand more complex patterns in the measurement data. At the moment, interaction effects between multiple measurements are completely left out. Without detecting issues stemming from such more holistic problems, however, faulty parts might not be detected (false negatives) or a high number of correct parts might be incorrectly labelled as faulty (false positives).

An apparent solution to deal with such situations is to leverage machine learning (ML)-based classification approaches. With these models, one can automatically explore many different combinations of measurements and see which combinations led to errors. This approach works specifically well if the objectives that define an

error are well-defined and when the user is knowledgeable about ML [SMM12]. Both assumptions do neither apply to the users of our design study nor other design studies outside our application domain, for example in clinical research [BBJ*17] or music classification [RAZ*18]. Past examples in our application domain [SIB*11] have already shown that error detection is inherently ill-defined and needs the tacit knowledge of domain experts to be spotted. Similarly, this target audience usually comprises a high degree of domain knowledge, while dedicated expertise in data analysis might not be available [ACKK14, Via13].

Our design study focuses on a specific ML technique, RF [Bre01], which are among the more popular ML methods [HH15]. RF is an ensemble learning method that usually includes hundreds of independent *Decision Trees* (DTs). RFs are relatively easy to implement, less prone to overfitting than similar approaches such as DTs, and well-suited to detect dependencies in their feature subspace. They also serve well to find combinations of features from measurements that improve part testing, making them a good choice for our target domain [EJS*20, PBLG19]. However, without proper visual interfaces, RFs can be hard to interpret for users, specifically, for those who have little or no ML background. Efforts were made that sought to make RFs more interpretable through visualization [HH15, HWWH19, BBM*15, YXZC12]. As none of the existing approaches have been tested within a real-world scenario yet, our design study is meant to add ecological validity to this design space.

As a result, we present *RfX* (*Random Forest Explorer*), a visual analytics (VA) system that resulted from a design study project [SMM12] with automotive engineers at BMW. *RfX* aids domain experts with no data advanced analytics background, such as ML model building, in the interactive analysis of RF models. The system allows identifying DTs that contain the most promising measurement combinations for new test procedures, while users are able to interactively explore, compare and enhance DTs in light of their domain knowledge. In this context, we aim to find a solution to the following problem *'How can optimal rules be derived from an RF by domain experts with no background in advanced data analytics?'* In summary, our contributions are as follows: (1) the problem characterization and abstraction of our use case; (2) *RfX*, a VA system for the interactive visualization of an RFs decision-making process and (3) a field evaluation of our approach with five automotive engineers.

2. Related Work

In this section, we present approaches to visualize RFs and DTs, and present completed design studies in the automotive industry.

2.1. Random forest visualization

In a recent study, Hänsch *et al*. [HWWH19] distinguish between four groups of classical RF visualizations. *Abstract visualizations* show the model itself. *Result-driven* visualizations focus on the result of an RF [HH15] through the provision of graphical interpretations of the systems classifications. *Data-driven* visualizations map model outputs to their input data [SJC08]. *Parameter-driven* visualizations show internal parameters, such as specific features or split criteria [HH15]. Even though various works have aimed at visualizing RFs within or by combining the above-mentioned categories [SJC08, KvdWVW01, BBM*15], barely any tool allows the interactive exploration of several RF's properties, as well as the effects of parameter changes on the model's structure and output [HWWH19]. Exceptions are provided by Hänsch and Hellwich [HH15] and Yanh *et al*. [YXZC12], using a 3D representation to create a botanically inspired forest. However, one drawback of 3D visualizations are views that contain occlusion and perspective distortion effects [CCF97]. To tackle this issue, Hänsch *et al*. [HWWH19] render individual trees in a 2D space in a follow-up research paper.

The above-mentioned studies can result in very complex visualizations, especially when many DTs are used. Thus, other researchers suggest identifying most representative regions within an RF [WL19, BB17] and group similar DTs. This is important, because this means that it is not necessary to analyse ever tree in detail. Instead, only trees that represent a group of trees should be considered. Bakirli *et al*. [BB17] summarize recent studies on estimating tree similarities into the two categories: *semantic* and *structural* similarities. *Semantic similarity* [NKT08, ZJ12] measures internal tree properties, such as common feature subspaces between DTs for particular decision classes. In turn, *structural similarity* (also referred to as *syntactic similarity* [WL19, MS04]) compares structural tree attributes such as nodes, branches or number of leaves [Per11, Per13]. In this regard, Bremm *et al*. [BLH*11] develop a similarity score by combining leaf-, element- and edge-based measures. Dogra and Kobti[DK13] provide a method for finding similarities between knowledge represented by different DTs. Miglio and Soffritti [MS04] created an algorithm for tree similarity that combines tree structure information with agreement percentages on the testing set. Weinberg and Last [WL19] select the most representative DT from ensembles of DTs in big data environments. All of the above-mentioned approaches do either focus on the representation of the entire forest or the grouping of multiple DTs. While the former often results in complicated visualizations when models are very complex, the latter does not consider any kind of visual support at all, which makes an interpretation for domain experts with no analytics background cumbersome or even impossible.

2.2. Decision tree visualization

As an integral part of an RF's visualization, DTs must be visualized in a thoroughly adequate way. In this regard, von Landesberger *et al*. [LKS*11] give an overview of techniques that are appropriate for displaying DTs (*Node-link diagrams, treemaps* and *icicle plots)*. *Node-link* techniques are probably the most well-known and often used tree visualizations [WFH*01, XHC*07, EW11]. In these approaches, nodes are represented as glyphs and relationships as links from parent to child. These visualizations can lead to scalability problems, especially when applied to large DTs. Thus, they are often combined with other visualization techniques. For instance, Bremm *et al*. [BLH*11] combine node-link diagrams with distance matrices, Behrisch *et al*. [BKSS15] with scatterplot views and Elzen *et al*. [EW11] with streamgraphs. As introduced by Schneiderman [Shn92], *Treemaps* are rectangular shapes that recursively subdivide rectangular spaces according to an underlying hierarchy. Even

though they make good use of the available space, their nodes often overlap, which can lead to difficult distinctions of hierarchical structures [LKS*11].

By contrast, *icicle plots* place child nodes next to their parent and have been widely used to visualize DTs. For example, Ankerst *et al*. [AEK00] and Liu and Salvendy [LS07] use icicle plots to visualize DTs with continuous data. They also show class distributions by colouring the bars according to class values. Even though icicle plots do not make use of the entire available space, their advantage is that the parent nodes do not overlap with their children. Von Landesberger *et al*. [LKS*11] point out that the combination of *node-link diagrams* with *icicle plots* or *treemaps* increases the interpretability of a DT and thus allows for flexible analysis. All of the above-mentioned studies describe DTs outside the context of an RF. In addition, none of them has been validated within a large industrial setting or in our application domain.

2.3. Design studies in the automotive sector

Design studies in the automotive sector are mostly carried out for engineering design and anomaly detection. In the case of engineering design, efforts have been made to visualize in-car communication networks [SIB*11, SFMB12] or the exploration of multi-criteria alternatives for rotor designs [CMMK20]. Recent studies have been carried out making use of anomaly detection to detect error-prone produced parts in test stations of large-scale manufacturing processes [SMF*20, EJS*20]. While some of the mentioned studies acknowledge the need for interpretability of applied ML models [EJS*20], considerably less work has been dedicated to the development of systems that enable the analysis of such models. Grounded on previous findings, we did build such a system, that visualizes the decision-making process of an RF classifier.

3. Methodology

In this study, we primarily followed the nine-stage framework for conducting design studies of Sedlmair *et al*. [SMM12]. Taking the system design into consideration, we additionally used the *Design Triangle* by Miksch and Aigner [MA14] and the *Nested Model* for visualization design and validation by Munzner [Mun10]. As means of gaining access to experts with domain knowledge and to better comprehend the problem, we developed the system in close cooperation with two domain experts at BMW, who focus on the development of testing procedures for *electrical engines*. Both have a background in mechanical engineering and reported not being familiar with data advanced analytics methods, such as the training of an RF. We carried out a formative evaluation, in which we discussed iterative prototypes with the experts, who guided us during the development of *RfX* and provided us with feedback about and relevant tasks. A summative evaluation of the system was carried out with five domain experts from BMW. Methodological details for this downstream evaluation are provided in Section 8.

4. Problem Characterization

We now analyse the domain problem, the underlying data characteristics and relevant tasks.

4.1. Domain problem

The main problem we focus on is to support engineers in detecting measurement errors in part testing. To ensure high-quality standards, each part is tested through test stations in serial manufacturing processes, before it is allowed to be assembled into a car. Domain experts currently develop test procedures for automated testing of produced parts during different product development stages. In the early development stages, tests are developed in lab experiments by evaluating sensors for measuring product behaviour in specific circumstances. An example is torque behaviour during speed ramps of an engine to test its durability. Recorded data mainly comprises time series data that are analysed manually. Here, a manual analysis is feasible since part numbers are very low.

When a test is capable of detecting a possible error, for example a faulty gear of an engine, the test procedure is finished and deployed to its respective test station. New errors may appear that were not anticipated during product development phases, however, especially in the early stages of a test stations installation. In this stage, it is hard to adjust the already assembled test stations (e.g. installing new sensors is very costly), resulting in the need to use existing sensor equipment to detect new errors. To address this problem, engineers seek to detect new errors through analysing combinations of existing signal measurements. Currently, this is done manually by sequentially checking different combinations of signals, based on hypotheses that the engineers derive from their domain knowledge. Due to the high number of produced parts in this stage, however, these analyses result in lengthy and tedious processes. For example, one domain expert is responsible for the development of an end of line test station for *electrical engines*. However, this station is testing hundreds of engines per day, recording 1024 measurements where each measurement is capable to detect an error.

The main idea behind our approach is to leverage ML to detect more complex errors. ML naturally lends itself to this problem, since common ML techniques, such as RFs, are well-suited to map multiple features: from measurements to detected errors. In the course of discussions with our two lead users, however, it became clear that they have little to no ML expertise at the moment. As one user commented when we showed him classifier results *'Can you please explain to me why the model predicted this kind of error and say exactly which were the most important factors for this decision'.* While training the users in ML might be a natural solution, this is not always possible or realistic; we explore how far visual interfaces can go towards making DTs more accessible to non-ML experts. The goal lies in enhancing existing test procedures for the detection of produced parts that do not meet quality requirements and go beyond simple baseline tests. Here, we aim to enable domain experts to derive rules from an RF that are capable of enriching current test procedures. A simplified rule can be: '*If measurement A is greater than a given threshold X, then evaluate measurement B as a next step'.* Here, an analysis of an RF does not necessarily need to result in an optimized classifier that fits well to predict certain errors, but '*good enough*' to enhance existing test procedures.

4.2. Data abstraction and sample datasets

As input data for our visualization we use a binary RF classifier, according to Breiman [Bre01]. This model is appropriate because it is

well-suited for the handling of high dimensional linear dependent data and is comparatively less complex than other ML approaches. Each RF consists of multiple DTs, whereby their number can be specified manually when setting the hyperparameters for the forest. Each DT consists of nodes and edges, where in our case, each node has always two children. An internal node consists of a feature from a measurement and a split threshold. Generally, it is hard to find out the exact number of DTs of each RF or the maximum depth of a DT in an RF since they depend on the properties of use case-specific data. We thus adapted these parameters in collaboration with our domain experts for the data sets we used in this design study. In addition, our classifier was constructed with the following hyperparameters:

First, we apply *bootstrapping* to overcome the common issue of the *bias-variance-tradeoff* [MRS08], often resulting in misclassifications. The maximum number of features in a tree are limited to $\sqrt{\#features}$ of all available features. As a split criteria, we use the *Gini importance* [LWSG13]. These hyperparameters are important for the abstraction of an RF since they give a general structure of tree behaviour. When using this hyperparameter set, each DT in the RF chooses its features randomly, splits the classes according to the *Gini importance* and at minimum five samples in a leaf. For an initial prototype of our system, we trained an RF with the *wine data set*, a widely used data set for ML use cases [ZF07, TD04]. We used this prototype for first discussions with our two lead users to gather first feedback on how to improve the visualization design. After our prototype was completed, we applied it to two datasets from our manufacturing setting. For each dataset, we analysed sensor data from two different test stations. Each sensor records acoustic measurements and comprises of multivariate time-series data. Detailed information about the feature extraction process can be found in Eirich *et al*. [EJS*20]. The two datasets are briefly described in detail.

1. The first dataset contained an RF trained with 219 features from electrical engines. An electrical engine is the final engine, which is assembled into a car and contains an electrical machine, a gearbox and an inverter. Features were extracted acoustic measurements of 900 randomly selected electrical engines produced between February 2020 and July 2020. Measurements were recorded at an end of line test station and served to analyse eccentricities of the rotor of an electrical engine. As an error class, 400 engines contained high eccentricities. This dataset was used in the evaluation with our domain experts to evaluate the usability of our system and the validity of our approach.

2. The second dataset contained an RF trained with 66 features from electrical machines. An electrical machine contains a rotor and a stator and is assembled into an electrical engine. Features were extracted from acoustic measurements of 200 randomly selected electrical machines produced between September 2020 and December 2020. Measurements were recorded at an inline test station and served to analyse the behaviour of the bearing inside an engine. As an error class, 100 electrical machines contained anomalies in the A-Bearing. This dataset was used above all to add further ecological validity to our approach with a second use case. We will also use this dataset as a running example in this paper, where 60 test samples from the initial dataset are visualized in Section 6.

4.3. Tasks

We began with the task characterization by observing our domain experts first using the prototypes and later the finished system. We focus on tasks to derive rules from promising feature combinations of multiple DTs. To derive relevant tasks, we used the taxonomies of Brehmer and Munzner [BM13] and Sedlmair *et al*. [SHB*14]. The former is appropriate because it forms the bridge between high- and low-level tasks, which is particularly helpful to embed them into the daily routines of the domain experts. The latter is well-suited because it particularly considers tasks that are relevant for ensemble modelling approaches such as RFs. Following the experts, we identified the following tasks **(T)**:

T1 *Partition RF*: To derive a rule from an RF, each analysis starts with the partitioning to find one single cluster of similar DTs or to find and compare multiple clusters of similar DTs within the RF. It includes an inspection of clustered DTs and their properties to answer questions like *'Are there visible groups of similar DTs in the RF visualization?'*. The task is also about gaining a first overview of the data space to answer questions such as *'How many groups of DTs exist?'* or *'How many DTs are within each group?'*

T2 *Identify individual representative DT*: Next, the decision to select a DT that is representative for a cluster within the RF is made. This task includes two sub-tasks. First, browsing (**T2.1**) through alternative similar DTs within a selected cluster and inspecting their high-level attributes, such as accuracy or number of nodes, allows identifying a subset of trees of special interest. For example, the accuracy of a DT can lead to the decision to select or skip a DT. This is followed by a comparison phase **T2.2**, where users primarily judge the superiority of one DT over other identified trees. Of course, the execution of (**T2**) involves going back and forth between the subtasks.

T3 *Explore individual DT*: When an interesting DT is selected, its low-level attributes, such as split thresholds or feature names are analysed in detail. For example, the name of a node and its child together with the split threshold can be directly mapped to a rule expressed as the statement: *'If measurement A > X, then evaluate measurement B'*. The resulting rule can be verified by reviewing the distribution of the classes in each node. This task is related to *Uncertainty* [SHB*14] since it involves the evaluation of the reliability of the model output mapped to the real data instances.

T4 *Optimize DT*: The split thresholds can be adapted in the light of previous domain knowledge. Thus, a suggested rule from a DT can be iteratively refined until all split thresholds are properly adapted. We are aware of the fact that individually optimized DTs might result in overfitted models. Thus, each rule that contains adapted thresholds has to be tested with new training data.

5. Data Aggregation for the Visualization

Our approach is closely related to *ensemble pruning*[TPV09, GRF00, DSM16, ZZY17]. In this method, an ensemble size of classifiers is reduced to increase model efficiency and predictive performance [BBSH14]. While efficiency and performance are

undeniably important aspects of model creation, the interactive exploration of individual ensembles in the context of an RF remains the focus of this work. Our approach does not aim to exclude any DT from an RF. Instead, we provide an approach for clustering similar DTs within an RF, allowing the user to explore multiple DTs. Many previous studies showed that a combination of semantic and structural similarities resulted in an efficient grouping of similar DTs [WL17, WL19, BB17, MS04]. We build on this existing knowledge and combine semantic and structural tree similarity scores. The commensurate score of the DTs (T_i and T_j) is computed via a convex combination in Equation (1). To group similar trees, we cluster the resulting matrix. As input data for *RfX*, the matrix with the highest cluster accuracies, that lead to an optimal parameter λ^* in Equation (1), is selected, as outlined in Section 5.3. Instead of other metrics mentioned in the literature (Section 2.1), our metric specifically considers the context of an RF. This is important since ensemble-based DTs differ from DTs in an RF. For example, DTs from an ensemble are generally deeper and thus more susceptible to overfitting.

$$Sim\big(T_i, T_j\big) := (1-\lambda)\, Sim_{\text{stru}}\big(T_i, T_j\big) + \lambda\, Sim_{\text{sem}}\big(T_i, T_j\big), \qquad (1)$$

5.1. Semantic dissimilarity

As one possible score to compute semantic similarity, Weinberg and Last [WL19] measure how often different DTs agree on the test data and assign a similarity score based on the result. Instead of other semantic similarity measurements to compare DTs, such as the *Jaccard Index* [FI18], Weinberg and Last [WL17, WL19] showed in various studies that agreement of DTs is a valid similarity score. Hence, we consider their approach as appropriate for our use case. When evaluating this approach, however, our users reported that they were confused to find that high scores represent DTs that are close to each other. Thus, we inverted the similarity score of Weinberg and Last to a *dissimilarity score* so that the disagreements are counted in Equation (2), where a low score is assigned to a pair of DTs if they often agree and therefore close to each other. For Equation (2), D_{Test} is the set of test samples and $\mathbb{1}_{\{T_i(\mathbf{x})\}}(\cdot)$ is the indicator function of the set $\{T_i(\mathbf{x})\}$.

$$\sum_{\mathbf{x}\in D_{\text{Test}}} \Big(1 - \mathbb{1}_{\{T_i(\mathbf{x})\}}\big(T_j(\mathbf{x})\big)\Big) \qquad (2)$$

To compare both structural and semantic scores, we include a scaling interval. Thus, dissimilarities are scaled to the interval [0,1]. To do this, the semantic score of each pair of trees is divided by the maximum score of two trees within the RF. These values are stored in a dissimilarity matrix $Sim_{\text{sem}} := (Sim_{\text{sem}}^{(i,j)})_{i,j=1}^{N}$, with

$$\begin{aligned} Sim_{\text{sem}}^{(i,j)} &:= Sim_{\text{sem}}\big(T_i, T_j\big) \\ &:= \frac{\sum_{\mathbf{x}\in D_{\text{Test}}} \big(1 - \mathbb{1}_{\{T_i(\mathbf{x})\}}\big(T_j(\mathbf{x})\big)\big)}{\max_{p,q\in\{1,2,\ldots,N\}} \sum_{x\in D_{\text{Test}}} \Big(1 - \mathbb{1}_{\{T_p(\mathbf{x})\}}\big(T_q(\mathbf{x})\big)\Big)}. \end{aligned} \qquad (3)$$

5.2. Structural dissimilarity

Furthermore, we build on the work of Bakirli *et al*. [BB17] to group paths in a tree by their output labels and transform these groups into respective sequences. To be consistent with Equation (2), we refer to this score as structural dissimilarity instead of similarity. We use the Levenshtein distance [Lev66] to measure the dissimilarity between each sequence, which is a popular character-based metric and measures the minimum number of single-character edits between two words. This fits in well with our approach because branches of DTs can be reduced to bitstrings, which is a perfect basis for interpreting the differences as dissimilarities.

For each tree T_i, we extract its *branches* and group them according to the *output-labels*. This assigns each tree T_i a representation as two *tree sequences* (*TS*)s. Each tree sequence $TS_{i,l}$ consists of $m_{i,l} \in \mathbb{N}$, the number of paths in the DT T_i that end with the label l, *branch sequences* (*BS*)s. The branch sequences are representations of the paths within the tree and written as $(s_k + 2)$-tuples, where $s_k \in \mathbb{N}$ denotes the length of the kth branch, excluding root and leaf-nodes. The root of tree T_i is denoted as r_i and $f_{i,k,n}$ is the feature of the splitting node n of the kth branch of the respective tree sequence $TS_{i,l}$.

The root and the features are coded as elements of the alphabet, which we extend by *AA*, *AB*, *AC* and so on if needed, corresponding to their feature names in the training data. A *BS* terminates with the corresponding label $l \in \{0, 1\}$ that gives the output of this branch in the DT. We order the *BS*s of the same label within a DT from the left to the right. For example, a branch can have the sequence A-C-E-1, where 1 represents the class and A-C-E the features splitting the nodes. Next, we concatenate all of the branches for every tree with the same label along our ordering from the left to the right. This gives us two sequences, each of which is equivalent to its corresponding *TS*. These new sequences have the form

$$TS_{i,l} := \Big(r_i, f_{i,1,1}, \ldots, l, r_i, f_{i,m_{i,j},1}, \ldots, l, f_{i,m_{i,j},s_{m_{i,j}}}, l\Big). \qquad (4)$$

Analogously to Section 5.1, two trees have a small dissimilarity score, if they are similar and a high score if they are not similar. We also scale the structural dissimilarity matrix, by maximum scaling, to the interval [0, 1]. This implies that the final dissimilarity matrix (see Equation 1), as a convex combination of the two specialized matrices, is also scaled in the interval [0, 1]. Since the semantic and structural matrices are symmetric, the resulting dissimilarity matrix as input for our visualization is also symmetric.

5.3. Clustering and two-dimensional representation

With both dissimilarity matrices available, we cluster similar DTs. Since these matrices depend on the parameter λ, the content of this section is always dependent on this parameter. However, we suppress the dependency in the following notations. The optimal parameter λ^* for Equation (1) is computed at the end of this section. We choose a clustering approach, where the number of clusters is automatically determined. A widely used method for this is a combination of *k-means* [Mac67] for clustering and an evaluation of the optimal number of clusters by *Silhouette Coefficients* [KLKR90]. This is an appropriate choice over other clustering approaches, such as clustering with DBSCAN, which needs an extensive analysis of the parameters to achieve a good cluster structure. We name the resulting clusters $C_1, \ldots, C_k$. Each DT T_i is represented as a vector

$t_i \in \mathbb{R}^N$, where t_i is the ith row of the dissimilarity matrix and N is the number of DTs in the RF. With that, each DT in the RF is assigned a cluster-label. Next, we project our dissimilarity matrix into a 2D space via *Multidimensional Scaling (MDS)* [Pea01]. Since the accuracy (acc_i) of a tree is an important high-level attribute, we want to take this measurement into account when analysing the clustering. Thus, we expand the 2D representation of each t_i by its accuracy to be $\tilde{t}_i := (x_i, y_i, acc_i)^T$ and compute the arithmetic means $\bar{t}_u$ of every cluster C_u such that

$$\bar{t}_u := (\bar{x}_u, \bar{y}_u, \overline{acc}_u)^T := \frac{1}{|C_u|} \sum_{\tilde{t} \in C_u} \tilde{t}, \tag{5}$$

with $|C_u|$ being the cardinality of cluster C_u. The two-dimensional visual representations of the cluster centres consider the accuracy by weighting the vectors $(\bar{x}_u, \bar{y}_u)^T$ by $\overline{acc}_u$. This gives us the centre c_u of cluster C_u in Equation (6).

$$c_u := \left(c_u^x, c_u^y\right)^T := (\bar{x}_u \cdot \overline{acc}_u, \bar{y}_u \cdot \overline{acc}_u)^T \tag{6}$$

With the 2D projections of the DTs, we then find for each cluster C_u the DT that is closest to the theoretical centre c_u as

$$t_u^* := \arg\min_{\tilde{t} \in C_u} \sqrt{\left(x - c_u^x\right)^2 + \left(y - c_u^y\right)^2}. \tag{7}$$

In order to obtain more detailed information, we repeat these clustering steps on every individual cluster to get sub-clusters with their representative trees. Here, the number of sub-clusters of each C_u is also automatically determined by evaluating Silhouette Coefficients. Thus, the RF is divided into clusters and each cluster into sub-clusters. When we define S as a set of trees, then its classification output $l_{S,\mathbf{x}} \in \{0, 1\}$ for a test sample $\mathbf{x}$ is computed in Equation (8). With this, the *accuracy* acc_C of a cluster C is defined in Equation (9), where $\mathbb{1}_{\{l_\mathbf{x}\}}(\cdot)$ is the indicator function of the set that consists only of the sample $\mathbf{x}$'s label $l_\mathbf{x}$. Thus, each cluster can be seen as little RF within the RF.

$$l_{S,\mathbf{x}} := \arg\max_{l \in \{0,1\}} |\{T \in S | T(\mathbf{x}) = l\}|. \tag{8}$$

$$acc_C := \frac{1}{|D_{Test}|} \sum_{\mathbf{x} \in D_{Test}} \mathbb{1}_{\{l_\mathbf{x}\}}\left(l_{C,\mathbf{x}}\right), \tag{9}$$

We use Equation 9 to determine the parameter-value λ^* whose resulting dissimilarity matrix is input to *RfX*. This value is chosen among the values $\{0, 0.1, \ldots, 1\}$ to have the highest *mean-accuracy* of the clusters that is

$$\lambda^* := \arg\max_{\lambda \in \{0, 0.1, \ldots, 1\}} \frac{\sum_{C \subset RF, C \text{ cluster}} acc_C}{|\{C | C \subset RF, C \text{ cluster}\}|}. \tag{10}$$

6. *RfX's* Visualization Components

Our visualization design is shown in Figure 1 and comprises six components labelled from (a) to (f). In general, the workflow of the system is closely related to a *multiattribute choice* as defined by Dimara *et al.* [KLKR90]. In our case, it can be described as finding one or multiple good-performing DTs among a finite number of other DTs and comparing them with each other. This is achieved by inspecting several high- and low-level tree attributes (Views (a)–(d)) of each selected DT. When a DT is considered relevant for the user, one or multiple rules can be derived, verified, and enhanced by the user (Views (e) and (f)_). For the entire demonstration of our visualization, we use the 60 test samples from the second test dataset from Section 4.2 as a running example.

6.1. Random forest view

The scatterplot in (a) helps to partition an RF (**T1**) by providing a big-picture of all available DTs and represents an entry point into an analysis. We use a planar projection, which is a common analysis start to get an initial overview over high-level relationships [SZS*16, JSM*17]. Each DT is represented by a single dot, while the size is increased for highly accurate trees and decreased accordingly. The cluster id and the accuracy are displayed in additional rectangles for each cluster. By hovering over a tree, its low-level attributes are displayed (e.g. accuracy).

6.2. Icicle plot and table view

To facilitate the identification of individual representative trees within clusters (**T2**), we encode them as small multiples [Tuf90]. Representative trees are identified as outlined in Section 5.3 and shown as icicle plot in (b). This view allows the inspection of individual high-level tree attributes (e.g. decision paths), where additional attributes can be inspected via hovering over each icicle plot (e.g. observations in classes 1 and 2). Each row is mapped to the parent cluster via rectangles with the same colours as the clusters in panel (a). Rows are sorted according to the accuracy of all clusters and columns to the accuracy of DTs within a cluster. The class distribution in each rectangle is shown on a divergent colour scale between red (Class 1) and blue (Class 2).

Some users preferred tables over icicle plots. Hence, we added table views for all clusters in (d) and all trees in a cluster in (c). As well as (b), the tables have the purpose to identify a relevant DT (**T2**). Tables are sorted according to their accuracies. The selection of a cluster in (d) and a DT in (c) results in their highlighting in (b) (see the background of clusters for a selected cluster and red dot for selected DT).

6.3. Decision tree view

Each tree is displayed as a node-link diagram in (e), these diagrams are widely used to visualize DTs [EW11, LKS*11, BLH*11]. As high-level tree attributes the tree id, its accuracy, the number of interior (non-leaf) nodes and its confusion matrix are displayed in the upper left of (e). Each cell of a confusion matrix is represented with two triangles, where the matrix rows represent the actual values and the columns the predicted values. To put focus on matrix cells that contain non-zero values, we increase the transparency of empty cells. This further facilitates the decision to select or skip a DT out of multiple DTs. Each node of the node-link diagram is shown as a confusion matrix, which represents—as another small

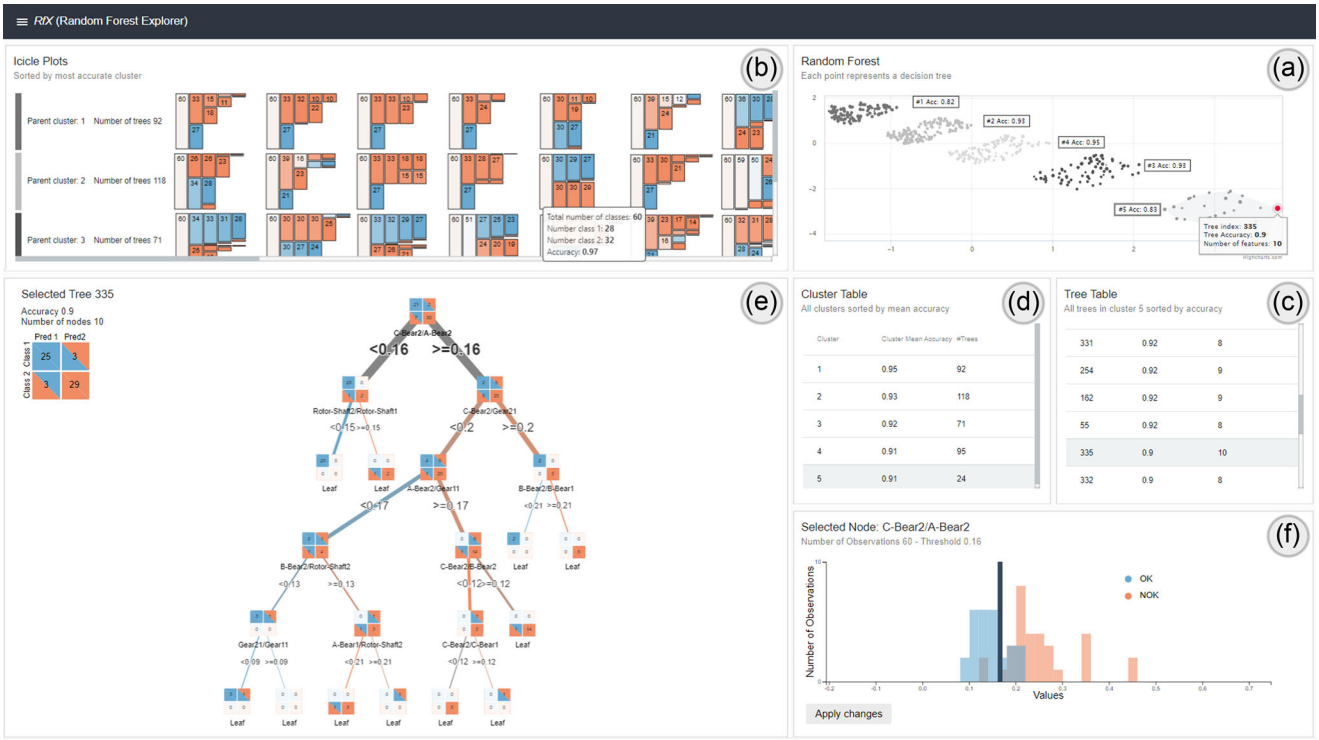

Figure 1: *Screenshot of* RfX *with 400 DTs from our second use case data-set as described in Section 4.2. (a) Visualizes the projection of our combined dissimilarity matrices. (b) Shows most representative tress of each cluster, represented as an icicle plot. Each cluster of similar trees is additionally visualized with a table representation, where (d) shows each cluster and (c) each tree within a cluster. A DT and its properties are shown as a node-link diagram in (e) and the class distribution of a feature as a histogram in (f).*

multiple [Tuf90]—the quality of the split criteria for the distribution inside the node. In addition to the icicle plots, edges are coloured in accordance with the same colour scheme, while split thresholds are displayed on the edges. The name of a feature is displayed under a node. Where previous views help to find appropriate DTs in the RF, the node-link diagram serves as the heart of the system to derive a rule from the DT (**T3**).

6.4. Histogram view

In the histogram in (f), users can review the actual class distribution where classes are marked with the same colours as in (e) and (b). The number of bins depends on the length of the input data vector. Since in our case, the test data contains 60 samples, we also use 60 bins in the histogram. A histogram is displayed by selecting a node in the node-link diagram. If tree splits are not optimal, users can adapt thresholds individually by dragging the blue rectangle in (f) (**T4**). Thus, a promising rule discovered in (f) can be iteratively refined, while threshold changes are always mapped to all confusion matrices in the node-link diagram. Furthermore, adapting thresholds supports users in the interactive analysis of the decision-making process of the RF, which can result in new insights and knowledge for domain experts [SSS*14].

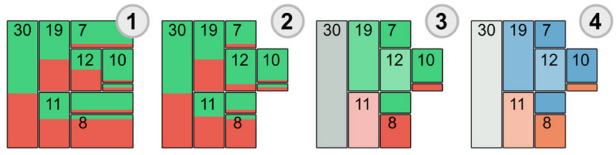

Figure 2: *Evolution of the icicle plots to represent a DT. In a first attempt in (1), we used a treemap, which we changed in (2) to an icicle plot with a qualitative colour scale. In (3), we used a divergent colour scale, which we changed in (4) to account for colour deficiency.*

7. Iterative Design Process

The design of *RfX* was carried out in four main iterations in collaboration with the domain experts. After each iteration, we gathered feedback and adapted our visualization design.

Figure 2 shows the evolution of the icicle plot representation. First, we experimented with treemaps in (1). However, domain experts reported that it was hard to follow single branches of a tree since nodes overlapped on the horizontal axis. Therefore in (2), we used icicle plots, because these allow users to better identify and analyze single branches of DTs [LKS*11]. In a first attempt, we used a categorical colour scale to distinguish between two classes. Here,

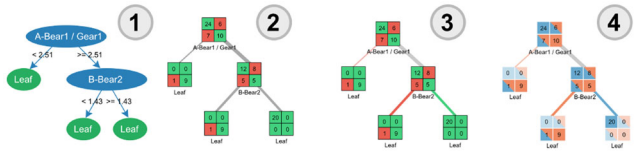

Figure 3: *Evolution of the node-link diagram to represent DTs. In (1), we use a simple node-link diagram. (2) Contains confusion matrices for each node and (3) colours for the edges to account for dominant classes in each split. In (4), we changed the colour to account for colour-blindness and represent each matrix cell as with two triangles (column for prediction and row for real value).*

domain experts reported that they are more interested in an efficient way to analyse the performance of a DT. They also found it difficult to evaluate the split quality of individual nodes in a DT. It was not always clear what the dominant class in a node was, especially in small rectangles with few samples. We thus used a divergent colour scale in (3). With such a colour scale, the dominant class in each node stands out. To ease the evaluation of DTs, we scaled the height of each icicle plot according to its accuracy. Since the colours in (3) did not account for colour deficiency, we adapted the colour scale to tackle this issue in (4).

Figure 3 shows the evolution of the tree views. In (1), we represented all nodes as ellipses. Although this is a common approach to represent DTs [SS18, LKS*11, BLH*11], experts found it hard to evaluate the distribution of samples inside each node. Thus, in (2), we introduced confusion matrices for each node where only false positives and negatives were coloured red. To show the distribution of class splits, we coloured the edges in (3) with the same divergent colour scale as we outlined above. To facilitate the analysis of each confusion matrix, we encoded each cell with two triangles as demonstrated in Section 6.3. Furthermore, we changed to a colour scale that accounts for colour-blindness. These changes allow experts to easily identify false positives, and negatives in a DT and analyse their distributions.

8. Evaluation

In this section, we introduce our evaluation methodology and demonstrate the usability of our system together with the application of found rules to a real-world scenario.

8.1. Methodology

Our evaluation aims at validating the usefulness of the proposed technical considerations (Section 5) and the resulting visualization (Section 6) in terms of effectiveness and problem-solving characteristics for domain experts within their daily routines. We present the results of an expert study, embodying qualitative coding of user feedback in combination with a quantitative usability scale.

Participants: We conducted the study with five domain experts from BMW, responsible for the development of testing procedures for the serial manufacturing of electrical vehicles. They were all male between 23 and 33 years old and had a mean working experience of 2 years in the problem domain. Every expert had a background in mechanical engineering, electrical engineering or physics and hence a higher education. Each of the five also reported being unfamiliar with advanced data analytics methods.

Data: For the evaluation, we used the first dataset as described in Section 4.2. The measurements of the data were recorded to detect engines with increased eccentricities of the rotor. The testing method is split into a general noise vibration part and a dedicated eccentricity measurement. To detect engines with high eccentricities within the first phase, labels from the second testing phase were used to train the model. Due to non-disclosure agreements with BMW, we cannot mention the absolute error rates or the cycle times for each testing procedure.

Task: The following task was given to each user: *'Derive a rule from a feature combination that you found in a DT'*. During the development with our lead users, we observed that tasks generally were carried out according to the workflow *'overview first—then details on demand'* as described in Section 6. Thereby, an example of an execution can be as follows: first, select a cluster either with the scatterplot view or from the cluster list that seems relevant. Next, select a tree from the cluster either via the icicle plots, the scatterplot, or the tree list. Then, explore the DT, by clicking on its nodes to review the class distributions. Finally, interactively change the thresholds if class splits are not optimal. The expected outcome is a rule from a combination of features and thresholds that can improve a current test procedure.

Procedure: The study was conducted in the form of a think-aloud study [SBS94] with one observer taking notes. Each session took on average 90 min and involved a detailed and prescribed walk-through of the system, open-ended questions [Sau20] about its usage and a usability questionnaire. Since the task of the domain experts and the visualization design are both relatively complex, we made sure that both the concept of our combined dissimilarity matrix and each view of *RfX* was understood well before starting the think-aloud study. All interviews were performed online, where each participant executed the same predefined task. The notes taken during the think-aloud study were analysed using a qualitative coding methodology [Cha06]. To quantitatively assess the usability of our system, we applied the *System Usability Scale (SUS)* [Sau20]. This scale is composed of ten statements rated on a Likert scale. The qualitative coding scheme and the quantitative (SUS) provide a comprehensive picture of our tool's deployment readiness level.

8.2. Findings from the think-aloud study

After coding and sorting the participants' comments and our observations, we were able to derive insights about the usability of the system. Four of five users first used the scatter plot to get an overview of the distribution of the clusters and the closeness of trees within each cluster. The remaining user reported that sufficient information was visualized in the list views. Next, users selected a cluster because it either contained an overall high accuracy (three users) or a comparable high accuracy but fewer DTs (two users). After each user selected a cluster, we identified two workflows that were then carried out. First, three users used the table view instead of the icicle plots to select trees with a low number of nodes and

Table 1: *Results of the System Usability Scale [Sau20] with five domain experts responsible for electrical engines.*

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Total
Expert 1	7.5	10	10	10	10	10	10	10	7.5	10	95
Expert 2	10	7.5	10	7.5	7.5	10	10	10	10	7.5	90
Expert 3	10	7.5	7.5	10	10	10	5	7.5	7.5	10	85
Expert 4	7.5	10	7.5	5	7.5	10	7.5	7.5	7.5	10	80
Expert 5	7.5	7.5	7.5	10	7.5	10	7.5	7.5	7.5	7.5	80
Avg.	8.5	8.5	8.5	8.5	8.5	10	8.0	8.5	8.0	9.0	86

Table 2: *Comparison of confusion matrices and model accuracies of a RF, a DT, and a derived ruleset. All modelling approaches were built with new sensor data from new electrical engines, which were produced after the ones we used in our study. 5% of the data contained an error.*

		RF pred.		DT pred.		Rule pred.	
		OK	NOK	OK	NOK	OK	NOK
Real value	Ok	0.93	0.07	0.91	0.09	0.91	0.13
	NOK	0.00	1.00	0.09	0.91	0.00	0.94
Accuracy		0.956		0.911		0.922	

a high accuracy and compared them to trees inside and outside the cluster. They then used the meta information of each tree, such as the confusion matrix to find trees with low false negative rates. The remaining two users used the icicle plots to select a tree. Here, the first user selected the top left DT because it was the most accurate one, while the second user discovered a rule that he thought was interesting in the left DT of the second cluster and thus decided to analyse this DT in detail. For both workflows, a tree was reported as relevant, if it either contained a new rule that had not been noticed by the users before or had a low number of false negatives in the confusion matrix. When asked why some users considered icicle plots and others the tree lists as relevant for their choice to select a DT, the users reported that they either generally preferred lists over visual abstractions or vice versa.

After a tree was identified, all users reviewed the names of each node to evaluate if their connection to other nodes made sense for them, where one expert noted *'I expected to find signals that I know, but in fact did find signals, which I did not expect at all. This is good because it makes me think of new relevant relationships within the data'*. Here, the icicle plots, confusion matrices for each node and colours and thickness of each edge were noted as especially helpful (*'It is good to see where the main path inside the tree lies to get a general overview of its behaviour and to evaluate how it separates the classes')*. After selecting relevant nodes, all experts revised the histogram and adapted split thresholds, when they thought that better splits in terms of DT accuracy could be achieved. This resulted in better classification results of the selected DT. All users reported that it was important to explore different types of DTs, which all represented different rule sets. Many of the newly discovered rules were interesting because they showed relationships between signals that users had not thought of previously and which helped them to better understand eccentricities of the rotor.

8.3. Findings from the system usability scale

Taking the quantitative results of the usability survey into account, our system provides good usability according to the adjective equivalent of the achieved SUS score [BKM09]. With a score of 86, our system is well above the average score of 68 [Sau20]. The individual scores are outlined in Table 1. *RfX* scores highest on low inconsistencies (Q6) and lowest on the quick learning of the system (Q7) and confidence in system usage (Q9). A possible explanation for the results from Q7 and Q9 is that interpreting ML models without prior knowledge is by no means a simple task. However, we are not able to draw a final conclusion on this observation, because of the low number of participants involved.

8.4. Anecdotal evidence

As mentioned in Section 8.1, the detection of *electrical engines* with high eccentricities is divided into two phases. The first phase measures general noise vibrations and the second detects dedicated eccentricities. During the think-aloud study, eight promising feature combinations from five DTs were detected by the domain experts with *RfX* and translated into rules. Each rule was tested on new sensor data from completely new electrical engines, which were produced after the ones we used in our study, where 5% contained an error in the rotor system. Due to non-disclosure agreements, we are not able to mention absolute error rates of engines. The low number of faulty engines is because the production quality at our industrial partner is very high and only very few engines are produced that have an error. After testing each rule on this new sensor data, we found that two rules from two different DTs described how increased eccentricities in the rotor system can be detected by observing secondary excitations of its bearings. When combined with the existing testing procedure, over 30% of all engines with high eccentricities were detectable during the first testing phase. Thus, for these engines, the second longer testing phase can be skipped in future, reducing the testing time for engines that do not meet quality criteria by 80%. Furthermore, we wanted to know if these results could also be achieved with a simpler yet intrinsically interpretable model. Thus, we used the same dataset to train a DT as an interpretable model and an RF as a more complex model and compared the results in Table 2. Here, the RF did achieve the best accuracy, the derived rule-set the second best and the DT the lowest. Thus, the derived rule-set outperforms an intrinsically interpretable DT but not a more complex RF.

9. Discussion

In today's highly automated manufacturing processes, the extraction of relevant and meaningful information from high-dimensional data remains a challenging problem [BKSS15, EBJ*21]. In this regard, the cooperation between human experts and ML techniques has often proved to be a promising solution by combining the strengths of both worlds [SMF*20, JFSK15]. We contribute to this challenge with a design study and present the system *RfX*.

RfX provides interactive visual means for identifying relevant DTs within an RF, thus enabling the user to discover relationships and detect hidden patterns. Since identifying relevant DTs is a difficult task for users who do not have an advanced analytics background, the set of visual components we have introduced follows a **dynamic analysis process**. Here, users can interact with our visualization in different ways to explore the decision-making process of an RF and to gain new knowledge. For instance, the user can choose between an *exploratory* and a *target-driven* analysis of the DTs, based on the visual component they choose as an entry point.

The planar projection provides the *exploratory* entry point into the analysis. An overview of this kind enables the user to choose a cluster with high accuracy (e.g. by inspecting the rectangles containing additional information in Figure 1(a)) and trees that are located close to each other. The icicle plots then allow the user to immediately determine decision paths in each tree and to compare them to similar trees within the same cluster.

By contrast, the cluster table supports an efficient, *target-driven* analysis. The cluster table is sorted by the accuracy of each cluster and lets the user select a DT cluster that performs relatively well but contains fewer DTs. The user then selects DTs from the tree table to identify a DT with the lowest possible number of false negatives while minimizing false positives. Here, a DT may contain many false positives but still be relevant, since it contains fewer false negatives. This is especially relevant in manufacturing contexts where further assembly of false negative results in overall faulty parts [EJS*20]. Despite the fact that both analyses have the goal of selecting the most relevant DT, they follow different workflows, in which both address a multivariate choice [DBD17].

The challenge of achieving an optimal trade-off between false positives and negatives is also relevant for the adaptation of split thresholds. While some might argue that this kind of task is best addressed completely automatically, we argue that human domain knowledge is a valuable resource [ACKK14, Via13] that serves well to optimize ML techniques. For example, users might prefer to adapt thresholds, which on the one hand produce a higher number of false positives, but on the other hand, provide a solid threshold that minimized false negatives. In contrast, other users might want to optimize thresholds that both result in minimal false positive and negative rates but are prone to overfitting. In this regard, we also acknowledge that thresholds could be adopted without manual intervention during training by including suitable loss and regularization techniques. Furthermore, we want to stress the fact that an important aspect of our approach is also about newly gained knowledge of domain experts from the interactive exploration of the decision-making process of the RF. New knowledge is difficult to measure [EJS*20], which is why we believe that providing an interactive means to understand complex ML models is important and we are confident that our approach addresses this issue. Even though the main goal of our approach is not to optimize an ML classifier the most efficient way, we showed that human domain knowledge can be an important aspect in better understanding an ML classifier. This is demonstrated by the fact that the derived rule-set from the domain experts outperformed a simple DT in terms of accuracy (see Table 2).

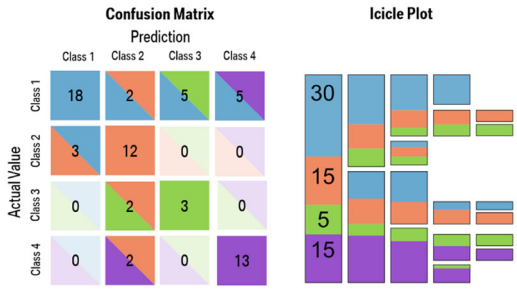

Figure 4: *Examples of a confusion matrix (left) and an icicle plot (right) for a multi-class prediction task using a qualitative colour scale.*

While similar systems of various different types so far have been proposed for exploring the decision-making process of ML classifiers, *RfX* differs from them in some important aspects. For example, Ming *et al.* [MQB19] provide the system (*RuleMatrix*) to derive a list of rules out of a neural network. However, their approach aims at approximating and analysing a single DT, whereas our approach groups similar DTs, leaving the possibility to explore, analyse and compare multiple DTs. Furthermore, *RuleMatrix* uses only flat representations (e.g. list views) and is not evaluated within a real-world context. In turn, *RfX* combines hierarchical (e.g. icicle plots) and flat (e.g. table views) representation and evaluates their interplay within a real-world scenario.

We acknowledge that our study has some limitations, which can be addressed in future research. First, our solution is a specific use case, designed and implemented for a specific problem in a specific company. Furthermore, our evaluation only involves five domain experts. This is a relatively common situation in design studies, where the presented visualizations often tackle very specific problems, which can be addressed by only a few experts [BSKR19, CMMK20, SIB*11, SFMB12, SMF*20]. A second limitation is that we only use a binary classifier. Even though many classification problems can be solved with binary classifiers a multi-class classifier leaves different challenges. To address this issue from a visualization perspective, the icicle plots could be enhanced with a qualitative colour scale to account for more than two classes. Using a qualitative colour scale for multi-class problems would also address the problem of imbalanced data. Here, classes with small sample sizes would still be visible in rectangles in the icicle plots. In the context of the representation of our confusion matrix, more classes could easily be added by also using a qualitative colour scale. An example of how a qualitative colour scale could address a multi-class problem for confusion matrices and icicle plots is provided in Figure 4.

Third, the experts also recommended one system improvement. The experts requested the inclusion of help buttons in the system (e.g. a detailed explanation of the icicle plots). Finally, neither the single concepts of our dissimilarity score (e.g. semantic dissimilarity) nor the visualization components (e.g. icicle plots) are novel as such. However, to the best of our knowledge, neither a

combination of dissimilarity scores has been combined with well-established visualization components nor have similar approaches been tested within a real-world scenario. Thus, we regard this work as an exploratory step towards explainable ML in manufacturing settings and plan to further extend and validate the idea of exploring groups of similar DTs in an RF with interactive systems.

10. Conclusion and Future Work

In this paper, we present a design study for the system *RfX*, which allows the deriving of rules from an RF's decision-making process for users with no background in ML. To achieve this goal, we first build on existing work to develop a combined score for *semantic* and *structural* tree similarities that enable the grouping of similar DTs within an RF. Guided by previous work on the visualization of RFs [HH15, HWWH19] and by using state of the art visualization techniques to visualize DTs [LKS*11, EW11], users can partition an RF, identify relevant DTs within an RF, explore each tree individually and adapt tree thresholds in light of their domain knowledge. A validation through a field study with five domain experts from BMW backed our technical considerations and design choices. In addition, as a result of *RfX*'s use, a derived rule improved the overall manufacturing process to detect electrical engines with high eccentricities and resulted in a reduced testing time of 80% for over 30% of engines that did not meet quality criteria from the analysed organization.

Fuelled by the positive feedback from our users and the improvement of the analysed manufacturing processes, future research efforts will build on this foundation. One possible extension of *RfX* will be to add another component to extract and represent features from the data space. Our intention here, is to develop a method that automatically maps important features of a trained model to the existing visualization. One way to achieve this objective could for example, be to use the RF feature importance score or features that often appear in highly accurate trees.

Acknowledgements

We would like to thank all study participants at BMW for their great support while developing *RfX*. This project was founded by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy—EXC 2075-390740016

Open access funding enabled and organized by Projekt DEAL.

References

[ACKK14] Amershi S., Cakmak M., Knox W. B., Kulesza T.: Power to the people: The role of humans in interactive machine learning. *AI Magazine 35*, 4 (Dec. 2014), 105–120.

[AEK00] Ankerst M., Ester M., Kriegel H. p.: Towards an effective cooperation of the computer and the user for classification. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Apr. 2000).

[BB17] Bakirli G., Birant D.: Dtreesim: A new approach to compute decision tree similarity using re-mining. *Turkish Journal of Electrical Engineering & Computer Sciences 25* (Jan. 2017), 108–125.

[BBJ*17] Bannach A., Bernard J., Jung F., Kohlhammer J., May T., Scheckenbach K., Wesarg S.: Visual analytics for radiomics: Combining medical imaging with patient data for clinical research. In *Proceedings of the 2017 IEEE Workshop on Visual Analytics in Healthcare (VAHC)* (2017), pp. 84–91.

[BBM*15] Beck F., Burch M., Munz T., Di Silvestro L., Weiskopf D.: Generalized pythagoras trees: A fractal approach to hierarchy visualization. In *Proceedings of the VISIGRAPP - Computer Vision, Imaging and Computer Graphics - Theory and Applications* (Jan. 2015), pp. 115–135.

[BBSH14] Bhatnagar V., Bhardwaj M., Sharma S., Haroon S.: Accuracy-diversity based pruning of classifier ensembles. *Progress in Artificial Intelligence 2*, 2–3 (2014), 97–111.

[BKM09] Bangor A., Kortum P., Miller J.: Determining what individual sus scores mean: Adding an adjective rating scale. *Journal of Usability Studies 4* (Apr. 2009), 114–123.

[BKSS15] Behrisch M., Korkmaz F., Shao L., Schreck T.: Feedback-driven interactive exploration of large multidimensional data supported by visual classifier. In *Proceedings of the 2014 IEEE Conference on Visual Analytics Science and Technology (VAST)* (Feb. 2015), pp. 43–52.

[BLH*11] Bremm S., Landesberger T., Hess M., Schreck T., Weil P., Hamacher K.: Interactive visual comparison of multiple trees. In *Proceedings of the VAST 2011 - IEEE Conference on Visual Analytics Science and Technology 2011* (Oct. 2011), pp. 31–40.

[BM13] Brehmer M., Munzner T.: A multi-level typology of abstract visualization tasks. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (Dec. 2013), 2376–2385.

[Bre01] Breiman L.: Random forests. *Machine Learning 45*, 1 (Oct. 2001), 5–32.

[BSKR19] Bernard J., Sessler D., Kohlhammer J., Ruddle R. A.: Using dashboard networks to visualize multiple patient histories: A design study on post-operative prostate cancer. *Transactions on Visualization and Computer Graphics 25*, 3 (2019), 1615–1628.

[CCF97] Carpendale M. S. T., Cowperthwaite D. J., Fracchia F. D.: Extending distortion viewing from 2D to 3D. *IEEE Computer Graphics and Applications 17*, 4 (1997), 42–51.

[Cha06] Charmaz K.: *Constructing Grounded Theory: A Practical Guide through Qualitative Analysis*. Sage Publications, London, Thousand Oaks, California, 2006.

[CMMK20] Cibulski L., Mitterhofer H., May T., Kohlhammer J.: Paved: Pareto front visualization for engineering design. *Computer Graphics Forum 39* (June 2020), 405–416.

[DBD17] Dimara E., Bezerianos A., Dragicevic P.: Conceptual and methodological issues in evaluating multidimensional visualizations for decision support. *IEEE Transactions on Visualization and Computer Graphics 24* (Aug. 2017), 1.

[DK13] Dogra I., Kobti Z.: Improving prediction accuracy in agent based modeling systems under dynamic environment. In *CEC 2013: Proceedings of the 2013 IEEE Congress on Evolutionary Computation* (June 2013), pp. 2114–2121.

[DSM16] Dheenadayalan K., Srinivasaraghavan G., Muralidhara V. N.: Pruning a Random Forest by Learning a Learning Algorithm. In *Machine Learning and Data Mining in Pattern Recognition* (vol. 9729). P. Perner (Ed.). Springer International Publishing, Cham (2016), pp. 516–529.

[EBJ*21] Eirich J., Bonart J., Jäckle D., Sedlmair M., Schmid U., Fischbach K., Schreck T., Bernard J.: IRVINE: A design study on analyzing correlation patterns of electrical engines. *Transactions on Visualization and Computer Graphics 28* (July 2021).

[EJS*20] Eirich J., Jäckle D., Schreck T., Bonart J., Posegga O., Fischbach K.: VIMA: Modeling and visualization of high dimensional machine sensor data leveraging multiple sources of domain knowledge. In *Proceedings of the VDS - Symposium on Visualization in Data Science at IEEE VIS* (Nov. 2020).

[EW11] van den Elzen S., van Wijk J.: BaobabView: Interactive construction and analysis of decision trees. In *Proceedings IEEE Symposium on Visual Analytics Science and Technology (VAST)* (2011), pp. 151–160.

[FI18] Fletcher S., Islam M.: Comparing sets of patterns with the jaccard index. *Australasian Journal of Information Systems 22* (Mar. 2018), 1–17.

[GRF00] Giacinto G., Roli F., Fumera G.: Design of effective multiple classifier systems by clustering of classifiers. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000* (2000), vol. 2, pp. 160–163.

[HH15] Hänsch R., Hellwich O.: Performance assessment and interpretation of random forests by three-dimensional visualizations. In *Proceedings of the IVAPP - Conference on Information Visualization Theory and Applications* (Jan. 2015), pp. 149–156.

[HKB*19] Heimes H., Kampker A., Bührer U., Steinberger A., Eirich J., Krotil S.: Scalable data analytics from predevelopment to large scale manufacturing. In *Proceedings of the APCORISE - 2nd Asia Pacific Conference on Research in Industrial and Systems Engineering* (Apr. 2019).

[HWWH19] Hänsch R., Wiesner P., Wendler S., Hellwich O.: Colorful trees: Visualizing random forests for analysis and interpretation. In *Proceedings of the WACV - Winter Conference on Applications of Computer Vision* (Jan. 2019), pp. 294–302.

[JFSK15] Jäckle D., Fischer F., Schreck T., Keim D.: Temporal MDS plots for analysis of multivariate data. *IEEE Transactions on Visualization and Computer Graphics 22* (Nov. 2015), 1.

[JSM*17] Jäckle D., Stoffel F., Mittelstädt S., Keim D., Reiterer H.: Interpretation of dimensionally-reduced crime data: A study with untrained domain experts. In *Proceedings of the IVAPP - Conference on Information Visualization Theory and Applications* (Jan. 2017), pp. 164–175.

[KLKR90] Kaufman L., Leonard Kaufman P., Rousseeuw P.: *Finding Groups in Data: An Introduction to Cluster Analysis*. Hoboken (New Jersey): Wiley-Interscience Publication, 1990. https://books.google.de/books?id=Q5wQAQAAIAAJ.

[KvdWVW01] Kleiberg E., van de Wetering H., Van Wijk J. J.: Botanical visualization of huge hierarchies. In *INFOVIS'01: Proceedings of the INFOVIS - Symposium on Information Visualization* (USA, 2001), IEEE Computer Society, pp. 87.

[Lev66] Levenshtein V. I.: Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady 10* (Feb. 1966), 707.

[LKS*11] Landesberger T., Kuijper A., Schreck T., Kohlhammer J., Wijk J., Fekete J. D., Fellner D.: Visual analysis of large graphs: State-of-the-art and future research challenges. *Computer Graphics Forum 30* (Sep. 2011), 1719–1749.

[LS07] Liu Y., Salvendy G.: Visualization support to better comprehend and improve decision tree classification modelling process: A survey and appraisal. *Theoretical Issues in Ergonomics Science 8* (Jan. 2007), 63–92.

[LWSG13] Louppe G., Wehenkel L., Sutera A., Geurts P.: Understanding variable importances in forests of randomized trees. *Advances in Neural Information Processing Systems 26* (Dec. 2013), 431–439.

[MA14] Miksch S., Aigner W.: A matter of time: Applying a data-users-tasks design triangle to visual analytics of time-oriented data. *Computers & Graphics, Special Section on Visual Analytics 38* (2014), 286–290.

[Mac67] MacQueen J.: Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics* (Berkeley, CA, 1967), University of California Press, pp. 281–297.

[MQB19] Ming Y., Qu H., Bertini E.: Rulematrix: Visualizing and understanding classifiers with rules. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (2019), 342–352.

[MRS08] Manning C. D., Raghavan P., Schütze H.: *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[MS04] Miglio R., Soffritti G.: The comparison between classification trees through proximity measures. *Computational Statistics & Data Analysis 45* (Apr. 2004), 577–593.

[Mun10] Munzner T.: A nested model for visualization design and validation. *Visualization and Computer Graphics, IEEE Transactions on 15* (Jan. 2010), 921–928.

[NKT08] Ntoutsi I., Kalousis A., Theodoridis Y.: A general framework for estimating similarity of datasets and decision trees: Exploring semantic similarity of decision trees. In *Proceedings of the SIAM - International Conference on Data Mining* (Apr. 2008), pp. 810–821.

[PBLG19] Peres R., Barata J., Leitão P., Garcia G.: Multistage quality control using machine learning in the automotive industry. *IEEE Access 7* (June 2019), 1.

[Pea01] Pearson K.: LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2*, 11 (1901), 559–572.

[Per11] Perner P.: How to interpret decision trees? In *Proceedings of the Advances in Data Mining: Applications and Theoretical Aspects* (Aug. 2011), pp. 40–55.

[Per13] Perner P.: How to compare and interpret two learnt decision trees from the same domain? In *Proceedings of the WAINA - International Conference on Advanced Information Networking and Applications Workshops* (Mar. 2013), pp. 318–322.

[RAZ*18] Ritter C., Altenhofen C., Zeppelzauer M., Kuijper A., Schreck T., Bernard J.: Personalized visual-interactive music classification. In *Proceedings of the EuroVis Workshop on Visual Analytics (EuroVA)* (2018), C. Tominski and T. von Landesberger (Eds.), The Eurographics Association.

[Sau20] Sauro J.: Measuring usability with the system usability scale (SUS). http://measuringu.com/sus [Accessed 5th November 2020].

[SBS94] Someren M., Barnard Y., Sandberg J.: *The Think Aloud Method—A Practical Guide to Modelling Cognitive Processes*. Academic Press, 1994.

[SFMB12] Sedlmair M., Frank A., Munzner T., Butz A.: RelEx: Visualization for actively changing overlay network specifications. *IEEE Transactions on Visualization and Computer Graphics 18* (Dec. 2012), 2729–2738.

[SHB*14] Sedlmair M., Heinzl C., Bruckner S., Piringer H., Moller T.: Visual parameter space analysis: A conceptual framework. *IEEE Transactions on Visualization and Computer Graphics 20* (Dec. 2014), 2161–2170.

[Shn92] Shneiderman B.: Tree visualization with tree-maps: 2-d space-filling approach. *ACM Transactions on Graphics 11*, 1 (Jan. 1992), 92–99.

[SIB*11] Sedlmair M., Isenberg P., Baur D., Mauerer M., Pigorsch C., Butz A.: Cardiogram: Visual analytics for automotive engineers. In *Proceedings of the Conference on Human Factors in Computing Systems* (May 2011), pp. 1727–1736.

[SJC08] Shotton J., Johnson M., Cipolla R.: Semantic texton forests for image categorization and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition* (June 2008), pp. 1–8.

[SMF*20] Suschnigg J., Mutlu B., Fuchs A., Sabol V., Thalmann S., Schreck T.: Exploration of anomalies in cyclic multivariate industrial time series data for condition monitoring. In *Proceedings of the EDBT/ICDT - 3rd International Workshop on Big Data Visual Exploration and Analytics* (2020).

[SMM12] Sedlmair M., Meyer M., Munzner T.: Design study methodology: Reflections from the trenches and the stacks. *IEEE Transactions on Visualization and Computer Graphics 18* (Dec. 2012), 2431–2440.

[SS18] Szücs D., Schmidt F.: Decision tree visualization for high-dimensional numerical data. In *Proceedings of the 2018 Fifth International Conference on Social Networks Analysis, Management and Security (SNAMS)* (2018), pp. 190–195.

[SSS*14] Sacha D., Stoffel A., Stoffel F., Kwon B. C., Ellis G., Keim D. A.: Knowledge generation model for visual analytics. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 1604–1613.

[SZS*16] Sacha D., Zhang L., Sedlmair M., Lee J., Peltonen J., Weiskopf D., North S., Keim D.: Visual interaction with dimensionality reduction: A structured literature analysis. *IEEE Transactions on Visualization and Computer Graphics 23* (Jan. 2016), 1–1.

[TD04] Tan P., Dowe D.: MML inferenc of oblique decision trees. In *Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)* (vol. 3339) (2004), pp. 1082–1088.

[TPV09] Tsoumakas G., Partalas I., Vlahavas I.: An ensemble pruning primer. In *Applications of Supervised and Unsupervised Ensemble Methods. Studies in Computational Intelligence* (vol. 245). J. Kacprzyk, O. Okun, G. Valentini (Eds.). Springer, Berlin, Heidelberg (2009), pp. 1–13.

[Tuf90] Tufte E. R.: *Envisioning Information*. Graphics Press, Cheshire, CT, 1990.

[Via13] Viaene S.: Data scientists aren't domain experts. *IT Professional 15* (Nov. 2013), 12–17.

[WFH*01] Ware M., Frank E., Holmes G., Hall M., Witten I.: Interactive machine learning: Letting users build classifiers. *International Journal of Human-Computer Studies 55* (Sep. 2001), 281–292.

[WL17] Weinberg A., Last M.: Interpretable decision-tree induction in a big data parallel framework. *International Journal of Applied Mathematics and Computer Science 27* (Dec. 2017), 737–748.

[WL19] Weinberg A., Last M.: Selecting a representative decision tree from an ensemble of decision-tree models for fast big data classification. *Journal of Big Data 6* (Dec. 2019), 23.

[XHC*07] Xu Y., Hong W., Chen N., Li X., Liu W., Zhang T.: *Parallel Filter: A Visual Classifier Based on Parallel Coordinates and Multivariate Data Analysis* (vol. 4682). Springer, 2007, pp. 1172–1183.

[YXZC12] Yang M., Xu H., Zhu D., Chen H.: *Visualizing the Random Forest by 3D Techniques*. Springer, 2012, pp. 639–645.

[ZF07] Zhong P., Fukushima M.: Regularized nonsmooth newton method for multi-class support vector machines. *Optimization Methods & Software 22* (Feb. 2007), 225–236.

[ZJ12] Zhang X., Jiang S.: A splitting criteria based on similarity in decision tree learning. *Journal of Software 7* (Aug. 2012), 1775–1782.

[ZZY17] Zhang C. X., Zhang J. S., Yin Q. Y.: A ranking-based strategy to prune variable selection ensembles. *Knowledge.-Based Systems 125* (2017), 13–25.

Supporting Information

Additional supporting information may be found online in the Supporting Information section at the end of the article.

Supporting Information