Journal of Imaging Science and Technology® 64(6): 060405-1–060405-13, 2020.

Combining Visual Analytics and Machine Learning for Reverse Engineering in Assembly Quality Control

Patrick Ruediger, Felix Claus, Bernd Hamann, Hans Hagen, and Heike Leitte
Technische Universität Kaiserslautern
E-mail: ruediger@cs.uni-kl.de

Abstract. *The authors introduce an integrative approach for the analysis of the high-dimensional parameter space relevant for decision-making in the context of quality control. Typically, a large number of parameters influence the quality of a manufactured part in an assembly process, and our approach supports the visual exploration and comprehension of the correlations among various parameters and their effects on part quality. We combine visualization and machine learning methods to help a user with the identification of important parameter value settings having certain effects on a part. The goal to understand the influence of parameter values on part quality is treated from a reverse engineering perspective, driven by the goal to determine what values cause what effects on part quality. The high-dimensional parameter value domain generally cannot be visualized directly, and the authors employ dimension reduction techniques to address this problem. Their prototype system makes possible the identification of regions in a high-dimensional parameter value space that lead to desirable (or non-desirable) parameter value settings for quality assurance. They demonstrate the validity and effectiveness of our methods and prototype by applying them to a sheet metal deformation example.*

[DOI: 10.2352/J.ImagingSci.Technol.2020.64.6.060405]

1. INTRODUCTION

The rapid advancements made in technology supporting an entirely digital integrated approach from digital design and to digital and automated manufacturing lead to the need of a much improved approached to assuring production quality demand. Our work presented here was driven by this demand. Lean production [1] is key to reducing idle times and improving storage capacity in plants. In-sequence production systems ensure that assembly parts are delivered at assembly stations in time and proper sequence. These automated concepts are only viable when a zero-failure (or near-zero-failure) environment can be assured as stopping a production process for human intervention is prohibitively expensive. The necessary quality control (QC) can be divided into *a priori* or *ex ante* QC, with *a priori* QC becoming more important for achieving the zero-failure goal. *Ex-ante* QC is based on collecting a vast amount of data for each produced part in short time; *a priori* QC only considers few available pilot parts.

Solution approaches for the *ex-ante* QC problem, especially those using machine learning (ML), are not well-suited for *a priori* QC. In order to overcome the lack of data, simulations are used, with the goal of synthetically generating the needed amount of data resulting from a high sampling of parameter value ranges to capture the characteristics of a real-world environment. These simulation systems are often called "digital twins," i.e., they are viewed as virtual computer simulations capable of replicating all relevant physical properties appropriately (see [2–9]). Digital-twin data augment available data produced by a pilot phase. A part being assembled and measured can now be classified as acceptable or non-acceptable. Making an acceptance/rejection decision merely on the few data produced by a pilot phase is not sufficient. Rejection in this production phase causes high costs as it interrupts the workflow. The important question that must be answered is this: Can one determine production parameter values such that a part can still be acceptably assembled, or is it possible to propose a viable way for avoiding rejection? To answer this question, the dominant relevant parameters and parameter values that lead to rejection must be determined.

We investigate how this reverse engineering problem can be tackled with an effective combined use of visual ensemble analysis, parameter space exploration, and an ML method. For visual ensemble analysis, considering the case of sheet metal deformation analysis as driving scenario, we analyze displacement and stress field variance and explore data via an exploratory contouring approach. For parameter space exploration, we interactively visualize distributions via violin plots [10]. This visualization can be complemented by scatter plot matrix (SPLOM) plots that permit a more detailed analysis by showing pairwise parameter cross-correlations. Parameter space analysis is further supplemented by performing principal component analysis (PCA [11]) to detect higher-order clusters. We have tested some commonly used classifier methods (i.e., the k-nearest-neighbor classifier [12], random forest classifier [13], and adversarial neural network classifier [14]), for determining parameter space regions defining "accept" and "reject" regions. Our application data set contains symmetries, shows strong cross-correlations between parameters, and has anisotropic value ranges. Our research was motivated by these research questions:

1. *Is a combination of visual analytics and ML techniques effective for detecting the parameter values causing specific effects under given boundary conditions?*

Received July 14, 2020; accepted for publication Nov. 17, 2020; published online Dec. 31, 2020. Associate Editor: Zeev Zalevsky.
1062-3701/2020/64(6)/060405/13/$25.00

2. *Is a high-level similarity analysis and visualization sufficient for our purpose?*

We have devised and implemented an incremental approach in our system. We start with the simulation ensemble analysis, complemented by summarizing distances between measurements and simulation data, producing a single similarity value. We analyze parameter value impact by using distribution characteristics and descriptive statistics. Subsequently, we incorporate correlations of parameter similarity relative to test data, using simple regression techniques. In order to reduce parameter cross-correlation complexity, we use PCA for dimension reduction. Finally, we assess how well an ML approach can predict dominant parameters.

1.1 *Related Work*

Increasing use of digital technology in factories demands increased use of data visualization techniques for data comprehension. The survey of Zhou et al. [15] classifies visualization methods based on the application scenario and industry sector. Our approach was driven by the needs of the automotive industry, more specifically, automobile preproduction. Following Zhou's taxonomy, our method concerns the "phase between" the design and production phases. Xu et al. introduced ViDX [16] for the analysis of assembly line performance. Their tool combines historical and real-time data. The system devised by Ramanujan et al. [17] analyzes similarities in computer-aided design (CAD) repositories, while an approach developed by Wu et al. [18] concentrates on condition monitoring. Leu et al. [19] provided an overview of the steps necessary to acquire reliable assembly simulations, including the one considered by us. A prominent example of the combined use of decision trees and dimension reduction techniques is included in Sun et al. [20].

Projection-based methods, including the method described by Bui-Tahn et al. [21] and Gaggero et al. [22], are appropriate for inverse and optimal design. Combining these methods with ML and neural network techniques has produced promising results for parameter space prediction in inverse design applications [23–25]. ML-based analysis approaches pose new visualization challenges since interactive and visual ML systems are still in their infancy [26]. Following the steps of parameter estimation, we can employ several methods. To quantify similarity for vector field ensemble elements, for example, Jarema et al. [27] proposed a visual analysis method combining statistical analysis and comparative visualization used in 2D space. Considering our application scenario, i.e., sheet metal deformation, we contemplate a deformation field on a surface as a 2D vector field.

Definition and labeling of clusters of ensemble elements can be carried out on the local scale using an entire surface geometry, using the method of Rieck et al. [28] that clusters elements based on persistent homology. Such an approach addresses the problems encountered with unlabeled data on a given surface. When a user labels classes, the level of confidence in decisions made is an important aspect considered in methods like the one of Kumpf et al. [29], helping with the understanding of different outcomes. Analyzing pairwise parameter correlation behavior for a high-dimensional parameter space can lead to visual clutter. Correlation maps [30] and interactive regression lenses [31] are methods tackling this issue. Causal and correlated behaviors often coincide, and direct visualization methods rely on correlation coefficients only. The method of Wang et al. [32] solves this problem by including additional quality measures computed via statistical correlation in a visualization. For example, a visual analysis approach to evaluate the capabilities of epidemic prediction models was discussed by Bryan et al. [33].

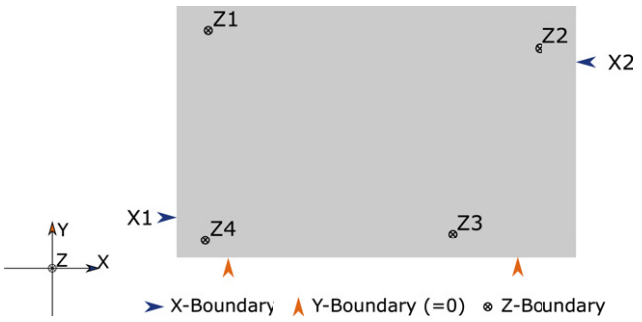

Figure 1. Generic sheet metal example. The boundary conditions for the simulations used in this article are depicted and labeled. The directions (X, Y and Z) for this part are indicated.

2. USE CASE: ASSEMBLY-INDUCED DEFORMATIONS

2.1 *Generic Metal Sheet Deformation*

We present an example based on the deformation of a simple sheet metal car body part to show the performance of the proposed method. The chosen simulation model shows a similar deformation per length ratio like real exterior car components during assembly. Thus, this example is representative for real-world use cases. Figure 1 shows the geometry and used boundaries. The shown part has dimensions of 50 mm × 30 mm and has a thickness of 0.5 mm. The material model is a linear-elastic model with the properties of steel (E-Modulus of 210 GPa and Poisson ratio of 0.3). The boundaries are shown in the picture as well, two boundaries located opposite each other on the edges in X-direction, four boundaries in Z-direction on the part surface, and two boundaries in Y-direction on the same edge. The Y-boundaries are set to zero to hold the part in place. The other directions are modeled as displacement load and varied within the tolerance interval of +/− 0.1 mm, which leads to different shapes the part can take on.

2.2 *Automotive Car Body Part Variation Simulation*

For a more application-driven view, a real-world example from the automotive industry was chosen. The geometry is an engine hood. This part has two hinges, two locks, and two buffers as mechanical boundaries attaching the hood to the chassis, see Figure 2. A finite element (FE) simulation predicts deflections during the assembly process

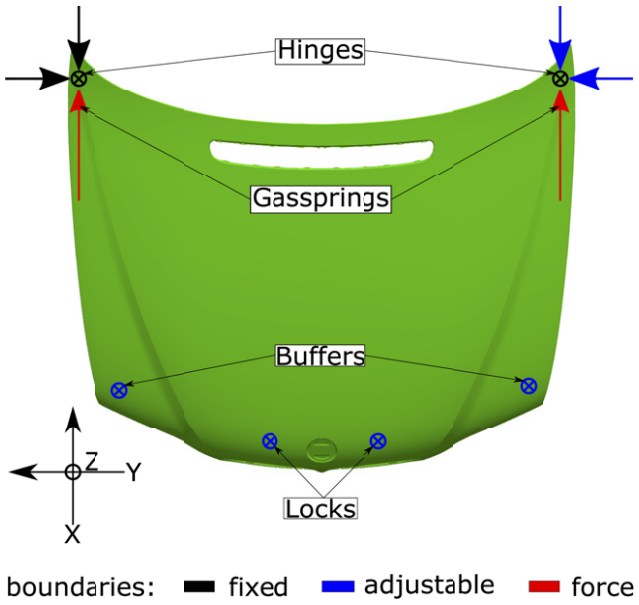

Figure 2. Sheet metal car body part used in automotive industry. The used boundaries for the simulation are highlighted, with external forces used via gas springs. The coordinate system for this part is shown.

of the part. However, the final shape of a material part can vary due to production uncertainties. To deal with uncertainties and tolerances during the production, the engine hood's boundaries, i.e., hinges, locks, and buffers, are adjustable. Adjusting these boundaries properly to obtain an acceptable gap and flushness is a challenging task [34]. A post-assembly measurement induces necessary corrections. The goal of the proposed method is to find the best set of changes from measured deviations, which forms the optimal set of adjustments. The method uses as input an assemble of statistical distributed simulations that cover the solution space spanned by the available adjustment possibility of each boundary. The used car hood is an assembly containing seven individual sheet metal parts, connected by spot welds and different types of adhesives. Based on the CAD files, a simulation model was created by meshing the geometry with 3D-shell elements and connecting the assembly considering spot weld, adhesive positions, and thicknesses of components. The material model is linear-elastic with an e-modulus of 210 Gpa and a Poisson rate of 0.3. Two fixed, external loads, modeling the gas springs near the hinges with the magnitude of 580 N each, complete the model.

2.3 *Use Case Classification*

We can characterize the presented application scenario as follows:

- Parameters generally correlate with each other.
- Correlation (strength and direction) depends on value range.
- Parameters are anisotropic (direction dependent).
- The distance metric is anisotropic.
- Even on a full point-wise scale, the mapping between parameter and resulting displacement is, in general, not bijective.
- The effect of parameters is neutralized under some conditions. (Some combinations of parameters and value ranges can lead to the same displacement field.)

In general, parameter estimation leads to multiple "acceptable" candidates and may include the "neutral parameter sets."

3. METHOD

In each step of the analytics pipeline, numerous methods are available. Figure 3 depicts our general workflow. Selection of methods is guided by three principles:

- Solving the problem of parameter estimation/Finding the dominant parameters.
- Arriving at a minimal set of parameters (for the method itself).
- Requiring a small integration effort (in terms of software system design).

3.1 *Data Analysis and Preprocessing*

The input data is a collection of simulation results on a predefined 2-dimensional grid. The grid is generally unstructured and consists mostly of quads and triangles. The displacement field is stored as a point-wise vector value, indicating the direction of the deformation. The stress tensor is transformed into a scalar field using the von Mises yield criterion and is stored for each cell. The results are then transformed into VTK unstructured grid files and arranged in a simple database format to match the varied parameters with the result files. We further enrich this database with the calculated distances to a test data set (see Table I). For the distance between two parts, we use the normal difference of each cell's displacement vector and summarize it for the whole geometry using root-mean-square (RMS). Based on these summarized distance values, we can define two classes for acceptance and withdrawal. In general, there is no restriction on using regression rather than classification. We favored the classification approach as it is more common among the application and more natural to visualize and interpret. Once a suitable classification method is identified, switching to regression is possible.

3.2 *Visual Ensemble Analysis*

We start our evaluation with a rough analysis of the simulation ensemble. Therefore, we calculated the statistical properties for both fields (Displacement "U," Mises-Stress "S") for each cell. The resulting field still preserved the original topology of the part and allowed us to investigate the whole geometry distribution. In Figure 4, the standard deviation of both fields is visualized. In addition, we used contours to highlight further the regions with a similar response to the induced loads.

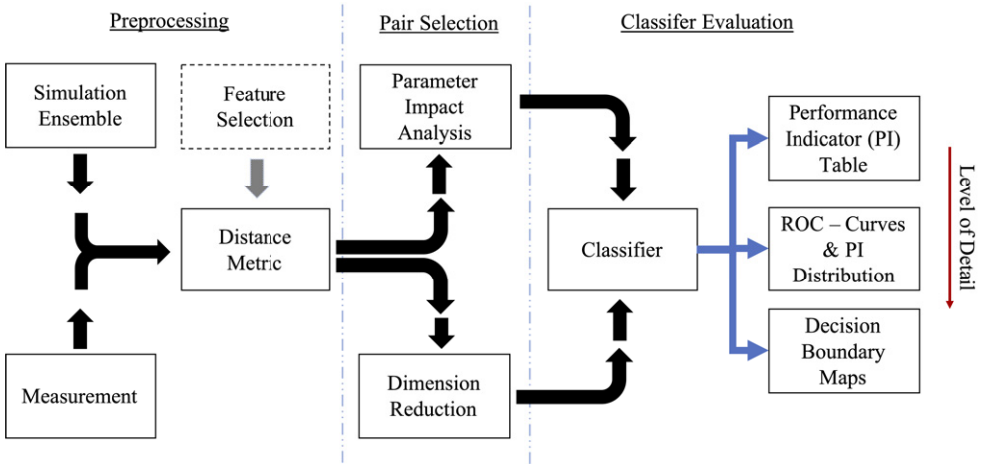

Figure 3. Workflow for classifier comparison. The workflow steps are preprocessing, parameter pair selection, and classifier evaluation. The preprocessing step integrates simulation and measurement data and calculates distance values for the whole domain or features selected *a priori*. In the pair selection step, a combination of visual methods, e.g., scatter plot matrices and statistical methods like principal component analysis (PCA), can be used. Classifier performance is evaluated at multiple levels of detail.

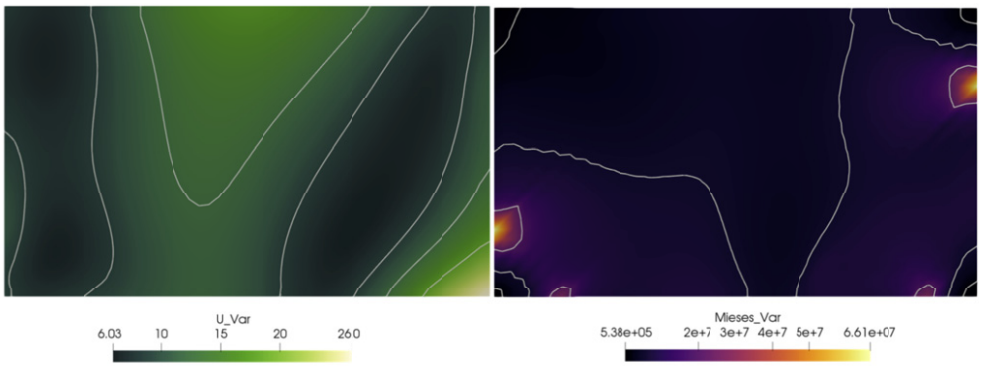

Figure 4. Visual Ensemble Analysis. The standard deviation of the resulting fields (Displacement U and Mises-1 Stress S) are calculated along each cell. This allows a visualization of the resulting deviations along the whole geometry. Contours are used to highlight similar behaving regions. The displacement field shows the linking of the *Z*-parameters, in the dark green areas, while the stress field indicates higher impact for the *X*-parameters (yellow). See Fig. 1 for definitions of the parameters.

Table I. A schematic view of the studied cases and their properties. + refers to load in positive direction, − in negative direction, and 0 means no load is induced at this position. We have picked a set of exceptional load cases, to specifically test the performance in extreme conditions.

ID	Property	X1	X2	Z1	Z2	Z3	Z4
2	Ambiguous cases	+	0	0	0	0	0
7		0	−	0	0	0	0
5	Translation in X	+	−	0	0	0	0
365	Pure Translation	−	+	+	+	+	+
730	Rotation	0	0	+	−	−	+
458	Maximum Load	+	−	−	+	−	+

3.3 *Parameter Space Exploration*

3.3.1 *Similarity Calculation*

At first, we analyzed the displacement field and split the domain into clusters in a preprocessing step. On these clusters or the whole domain, we apply a distance metric between the measurement/test data and the simulation/training data. This distance metric is normalized, leading to a general similarity term using the following formula:

$$S_i = 1 - \frac{\Delta(U_i, \hat{U})}{\max(U_i)}. \tag{1}$$

Here, U_i refers to each displacement field of the training data set, and $\hat{U}$ refers to the reference displacement field. Δ is synonymous for the chosen distance metric. In our use cases, we use the RMS error. Hereafter, we will only refer to similarity as our primary dependent attribute. The resolved similarity calculation is the starting point for the parameter estimation task. To better cope with the classification methodology in quality control, we have to separate the value range of the similarity into discrete classes. For simplicity, we chose only two classes: Acceptance and Withdrawal. We only have to define one threshold value, typically referred to as th_{high} in the figures.

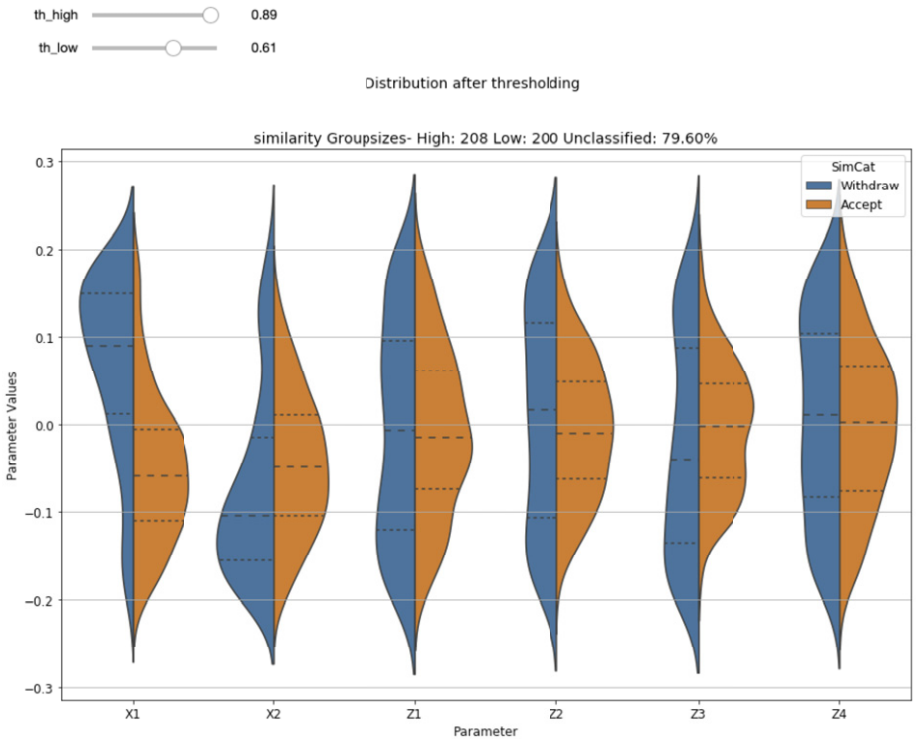

Figure 5. Distribution patterns using violin plots. An example of how violin plots can be used interactively to spot patterns in the distributions for different similarity thresholds. The orange distribution refers to the acceptance class, the blue to the withdrawal class. The distribution is shown for each of the six parameters. Given the example, the values of X1 are more frequently negative for higher similarity values and positive for lower values, which copes with the test sample used here.

3.3.2 *Distribution Analysis*

Our first parameter space visualization uses violin plots [10], see Figure 5. We define a threshold value for similarity to separate our data set in two classes. For each class, we plot the distribution of each parameter. With this visual representation, we can compare parameter impact, in terms of how often it is present in each class. We can relatively compare the mean and quartiles, but, more importantly, spot differences in the shape of the distribution. Of particular interest are distributions with multiple maxima or minima (multi-modal distributions), as these suggest the presence of some symmetry.

3.3.3 *Scatter Plot Matrix (SPLOM)*

The predictive result for separately investigating each parameter is limited. Therefore, we extend the analysis by using scatter plots, kernel-density estimates, and linear regression for each parameter pair and compare the distribution and correlation with similarity in more detail, see Figure 6. The arrangement in a SPLOM allows us to combine three types of visualizations in one view. In the lower diagonal, we plot the distribution of each class based on the respective parameter values using a kernel-density estimator (KDE). It allows us to identify high-density regions for parameter combinations. In the upper diagonal, we use a scatter plot with linear regression for each class to highlight the interaction between the two parameters. The correlation from the upper diagonal plots allows us to determine in what direction to look for cluster separation. On the main diagonal, we plot the distribution density of the classes for each parameter.

3.3.4 *Principal Component Analysis*

At this point, we have analyzed the parameters individually and found correlations between them. In the next step, we try to reduce dimensionality and focus on pairs of parameters or linear recombinations of our initial parameter space. We investigated two techniques: hierarchical linear regression (HLR) and principal component analysis (PCA) [35]. In the HLR approach, we use ordinary-least-square (OLS) as the model with one constant factor. First, we apply the linear regression model to each parameter individually and sort the output by explained variance (R-squared). We use this order to incrementally add each parameter in the model and observe its impact on R-squared. Based on the added value to R-squared in the model, we can determine what parameters to focus on. In general, this method allows us to use any model, and it has the highest potential to incorporate application knowledge into the system. Unfortunately, it restricts us to the original parameter space, and choosing a good model is a complicated task. PCA finds linear recombinations of the original parameter space based on a correlation or covariance matrix. The only parameter we have to choose is the threshold value for similarity. PCA is applied to each class. From the bi-plots, shown in Figure 7 we can determine what original parameters contribute to the principal derived components. The more they align with the

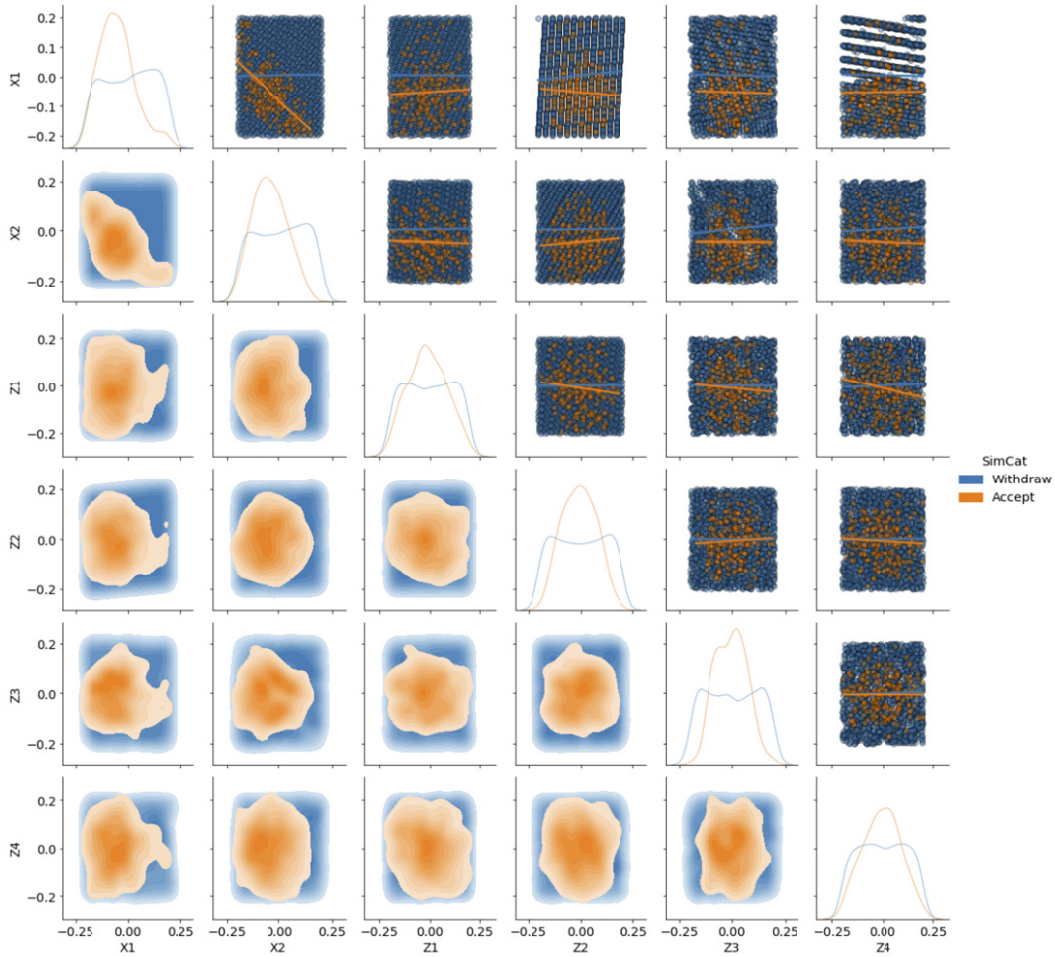

Figure 6. Scatter plot matrix (SPLOM) with kernel-density estimator (KDE). For a complete pairwise correlation analysis we use the concept of a SPLOM. In the upper diagonal we use a simple scatter plot which is colored based on the set class labels and enhanced with linear regression curves for each class. On the lower diagonal we used a KDE to better visualize the shape of the distribution of the two classes. This high-level view allows us to spot relevant correlations in addition to the ones already found in the ensemble analysis, e.g. row three (Z1) and column six (Z4) indicates a negative correlation.

axes of a principal component, the better we can separate our dimensions. The lengths of the arrows depict the relative impact of each parameter on the derived components. If multiple principal components have a similarly explained variance, we investigate each combination separately to determine good splits.

3.4 *Classification via Machine Learning*

In the final step of parameter estimation, we use a set of supervised learning classification methods. All implementations used in this evaluation are from the "scikit-learn" [36]. The classifiers are chosen in a way to represent a vast range of methodologies, taking into account the special properties of the application scenario. Analyzing the commonalities and differences of them helps us to select the appropriate ones for further optimization.

3.4.1 *Nearest Neighbor Classifications*

This is considered as a type of instance-based learning, i.e., it is not constructing or fitting any internal model but simply classifies the training data based on a majority vote [12]. The only parameters to choose is the number of neighbors for the query and a weight function. With increasing k, noise is reduced in favor of distinctness of classification boundaries. In general, this method suffers from the "curse of dimensionality" [37, 38]. In our evaluation, $k = 3$ turned out to be a good trade-off value, as we expect to have only a small degree of noise in the data. The weights are uniform, as our input space is uniform.

3.4.2 *Decision Tree and Random Forest Methods*

These methods follow the principle of learning a set of simple decision rules (if-then-else) from training data [13]. The method of decision trees resembles the manual progress of an experienced person familiar with an assembly [20]. Decision trees are the only classifier in this evaluation that can be easily visualized even for higher dimensions. The decision tree has the highest number of parameters. First, the quality criterion for the split is chosen. We picked Gini impurity, as we did not observe any significant differences

for *depth* > 4 in comparison to entropy (information gain). For the split strategy, we chose best over random, as our training data is nearly uniform. For all other tree parameters, we used default values, either minimum or maximum values. The simple decision tree tends to over-fit with these settings. To address this shortcoming, we extended our approach by using the random forest classifier, which fits several decision trees to various sub-samples and averages the results. We found out that $n = 10$ subdivisions are a good trade-off between computational time and increased predictive quality.

3.4.3 *Neural Network*

We used a fundamental neural network definition [14] as starting point for further investigations. A multi-layer perceptron classifier was chosen with a rectified linear unit function for the activation, one hidden layer, and a total number of 40 perceptrons ($6 \times 16 + 16 + 16 \times 2 = 144$ weights). The default L2 penalty (0.0001) was chosen with a constant learning rate. A small test series revealed that a maximum of 4000 iterations is a good choice as termination criterion. With two layers, we are still in the class of continuous functions and can keep the method comparable to the others, as proven by Brutzkus et al. [39]. In general, a two-layer network can learn polynomial functions of degree r over d-dimensional inputs [40].

3.4.4 *Confusion Matrix*

To objectively compare the quality of the parameter estimation, we use the concept of confusion matrix [41–43]. Each method itself, and the combination of methods, leads to a prediction range in parameter space. The input test parameter set and a predefined acceptable deviation are defining the true conditions for the confusion matrix. The acceptable deviation depends strongly on the specific application's accuracy goal. In the case of the generic example, we chose 5% of the total parameter value range. For the real model, we chose 10%. In the confusion matrix we counted the number of true positives *tp*, true negatives *tn*, false positives *fp*, and false negatives *fn* for our total size of the training data set *n*. For the evaluation and interpretation, we focus on these three ratios derived from those numbers:

$$\text{Accuracy (ACC)} = \frac{tp + tn}{n} \tag{2}$$

$$\text{Sensitivity (TPR)} = \frac{tp}{tp + fn} \tag{3}$$

$$\text{Specificity (TNR)} = \frac{tn}{tn + fp}. \tag{4}$$

Accuracy describes the ratio of all correct predictions concerning the total population. Accuracy does not provide information about the type of correct/incorrect predictions. Therefore we additionally use the sensitivity (true positive rate), which describes the method's ability to detect the true parameter values correctly. A high value of sensitivity reliably rules out negative predictions, while a positive result in high sensitivity prediction is not necessarily useful for ruling in the parameter set. Specificity deals with this shortcoming. A method with high specificity value reliably handles positive parameter set predictions. Considering our application, one or the other rate is more important. In ML, high accuracy is generally preferred over sensitivity or specificity. We consider two cases:

1. The preproduction phase and quality assurance benefit from a high sensitivity over specificity to detect anomalies and failures reliably.
2. Mass production and online steering applications favor a high specificity to minimize spoilage.

Once fitting the training data is accomplished, each classifier and its underlying data model can be used to predict the target class, given a set of parameter values. Typically, a classifier does not just return the best class label; it also generates a probability for each class in the model for specified values. More formally,

$$\text{Predict}(PS)_{CL} = [p(\text{class}_0), p(\text{class}_1), \ldots, p(\text{class}_i)] \tag{5}$$

$$p(\text{class}_k) = 1.0 - \sum p(\text{class}_j), \quad \text{for } j \neq k, \tag{6}$$

where *PS* determines the tested input parameter set, *CL* describes the used classifier, and $p(\text{class}_i)$ is the probability for predicting class_i.

In our experimental setting, the exact parameter values are known. We evaluate the predict function for values in a certain range, with a step size h used for visualization resolution. We plot results from the predict function as a heat map with value ranges between zero and one, see Figure 8. The red square is marking the acceptance window. We can link the three performance values in each row in Figure 9 with our derived decision map (see Fig. 8). The sensitivity is the summary result of the Acceptance Window and specificity of the rest; accuracy summarizes both. Based on these effective visualizations, we were able to deduce a general workflow for classifier comparison, see Fig. 3. For each classifier, we start with the summary performance indicators, extend the analysis using ROC curves, see Figs. 9, 10, and finally evaluate the shape of the decision boundary map (see Fig. 8).

4. RESULTS

In our evaluation, we have depicted three exceptional load cases that are challenging to classify without context; see Table I. The first group covers an ambiguous symmetric case for the parameters X1 and X2. The second group consists of pure translation or rotation. Finally, we consider the maximum load case to observe the classifiers' performance on the boundaries. We will focus on the ambiguous load and the maximum load case for the demonstration of the workflow. We first analyze the generic sheet metal case in more detail and then adapt the results for the automotive car body part.

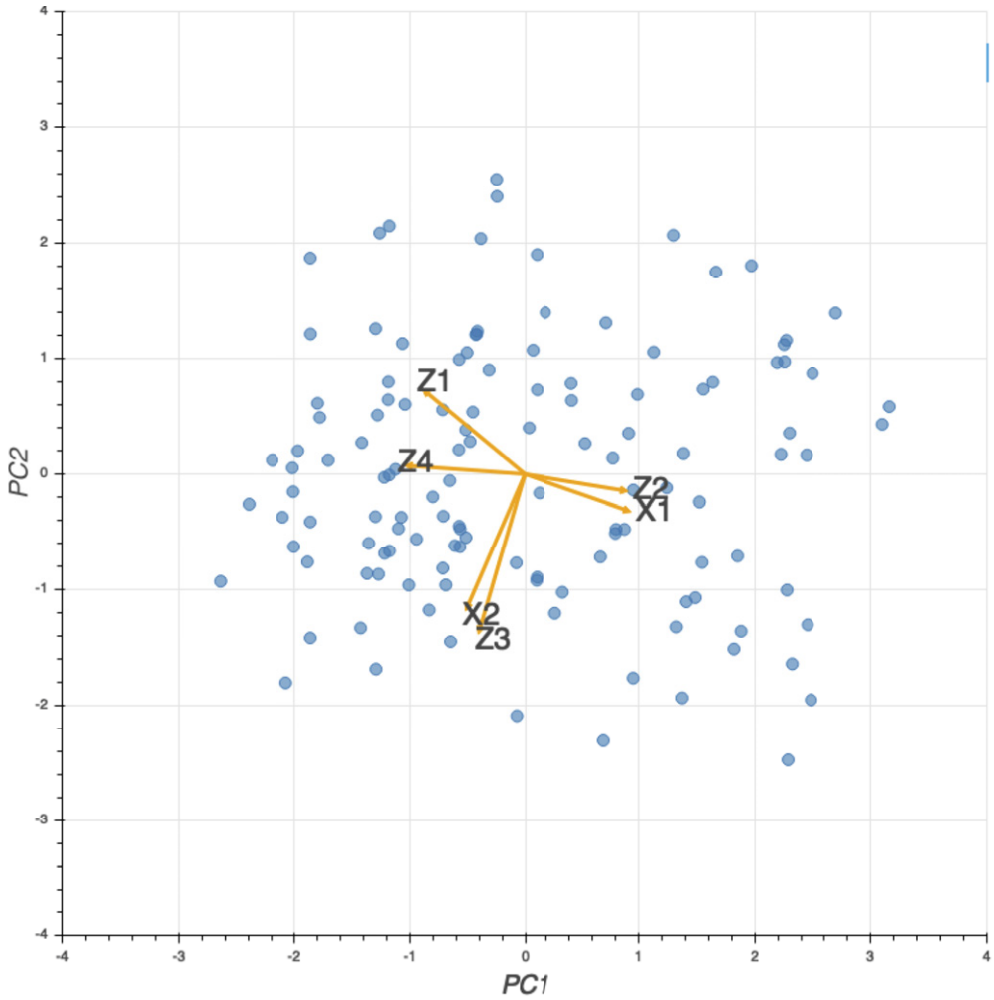

Figure 7. Bi-plot for PCA. With the help of a bi-plot we are able to project the initial parameters with respect to their variance portion on the derived first and second principal component (yellow arrows). The length of the arrows refer to the eigenvalues of the decomposition. In the given example the first principal component mainly consists of X1, Z2 and Z4. The arrows opposite direction indicates a negative correlation between Z4 and X1, Z2. The second principal component consists mainly of X2 and Z3. Z1 has en equal influence on both components.

4.1 *Simulation Ensemble Sensitivity Analysis*

We can observe that the regions vary, depending on which field we use as a basis. The stress variation field shows two dominant peaks left and right, which refer to the parameters X1 and X2 and two smaller peaks at the bottom, which refer to the Y boundary condition, which are fixed in our simulation. Interestingly is that the varied Z-parameters did not show such how variance peaks. From this, we deduce that X1 and X2 are, in general, more dominant than the Z-parameters.

The displacement variation field indirectly shows the correlations to the initial boundary conditions. We can roughly separate four regions, from left to right, that reacts differently. We have two regions spanning from top to bottom with nearly the same width with low variance. The first region connects the parameters Z1 and Z4, while the second region connects Z2 and Z3, which results in an overall higher correlation between those pairs explicitly. In summary, the analysis of the ensemble reveals three major parameter pairs to consider: (X1, X2); (Z1, Z4); (Z2, Z3). In addition, if a single number summary for each part is insufficient to find the dominating parameters, one should consider segmenting the geometry concerning two of these three pairs.

Considering the more sophisticated car body part, we can observe a more complex distribution.

4.2 *Parameter Space Exploration*

At first, we analyze the distribution of similarity while setting the threshold value for acceptance and withdrawal. Fig. 5 shows the results for one of the ambiguous cases (ID7). We summarized our results in Table II concerning the shifting direction of each class observed. For the ambiguous cases, we found a shift in X1 and X2, but not in Z1–4. A distinction between X1 and X2 is not possible. In the translation cases (5,365), the shift in the distribution for X1 is ambiguous (both the acceptance and withdrawal classes are shifted in the same direction). The rotation case also indicates an influence of X1 and X2, which is not correct. For the maximum load case, this method is not suited at all. Now further analyzing the ambiguity, we make use of the SPLOM. We can observe that the acceptance parameter values are clustered in the positive

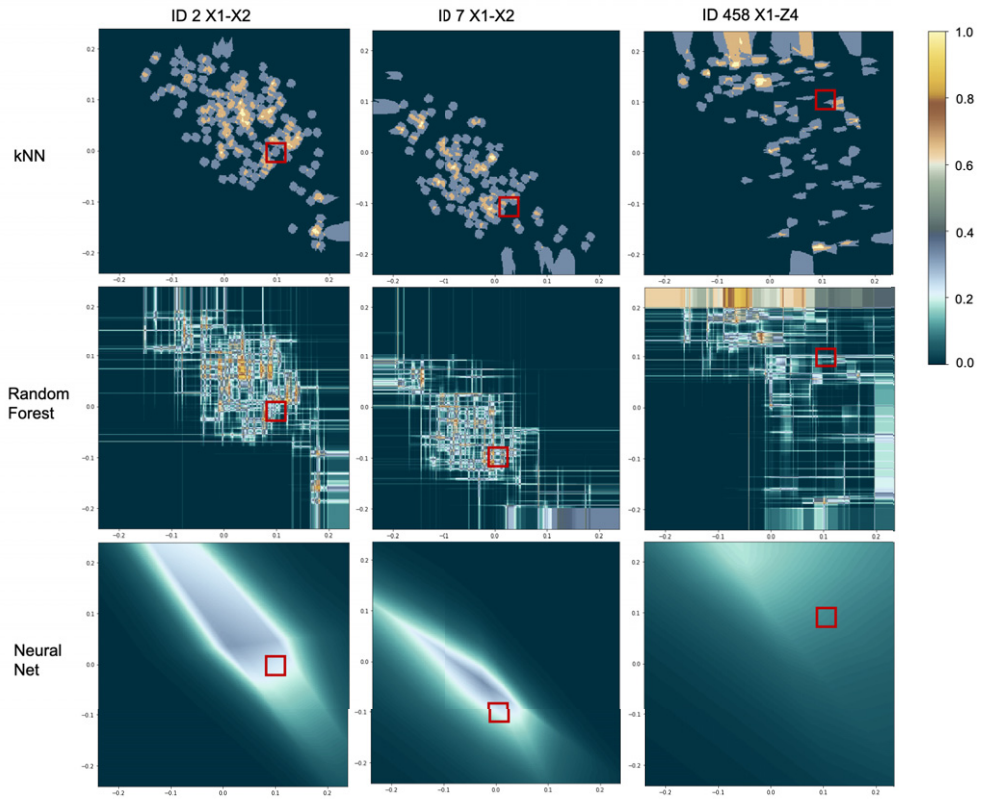

Figure 8. Decision Boundaries for three selected use cases. The heat maps are visualizing the prediction probability of each classifier. A five-step color map is used to point out sharp prediction boundaries with higher values favoring the acceptance class and lower values the withdrawal class. The red squares are indicating the correct value ranges of the test data set. Instead of comparing the raw numbers, e.g., in ROC curves, this visualization allows us to compare the shape of the predictive model between the classifiers. For the ambiguous cases (ID2,7) the predictive quality is better than for the maximum load case (ID458). Also the shape implies a more stable model compared to the maximum load case.

Table II. Results from the distribution shape analysis using violin plots for the generic metal sheet. "A" refers to the acceptance class and "W" for the withdrawal class. + (−) indicates a shift in positive (negative) direction. 0 indicates no significant shift. Double markings indicate a strong shift in one direction. Some cases are captured better than others. In most cases, the dominant parameters are spotted, but the results can be ambiguous (e.g., ID 5,365 X1).

	X1		X2		Z1		Z2		Z3		Z4	
ID	A	W	A	W	A	W	A	W	A	W	A	W
2	+	++	−−	+	0	0	0	0	0	0	0	0
7	++	−	−	−−	0	0	0	0	0	0	0	0
5	++	++	++	−−	0	0	0	0	0	0	0	0
365	++	++	++	−−	+	0	+	0	+	0	+	0
730	−	++	+	−−	++	−	−−	0	−−	0	++	0
458	0	−−	−	−−	0	0	0	0	+	−−	++	0

part of X1. For the withdrawal parameter, there is no distinct cluster in one specific region. This way, we can untangle the ambiguity for cases 5 and 365. For the maximum load case, analyzing the SPLOM reveals that every parameter has an influence, which better copes with the expected solution. Nonetheless, it does not favor any direction of the parameters, but we can observe strong correlations between them.

To reduce the complexity, we use the PCA to summarize the parameter correlations into two principal components. Using the bi-plot from Fig. 7, we can group (X1, Z2, Z4) and (X2, Z3), Z1 has an equal influence on both groups. The vectors opposite direction indicates a negative correlation between Z4 and X1, Z2. The length of the vectors for X2 and Z3 further implies a more substantial influence on the components and influencing the first component as well. We will further focus on these derived groups when analyzing the classifier results.

4.3 *Classifier Comparison*

In the predictions discussed next, we will exemplarily show the performance of the classifiers for one of the ambiguous cases (ID 7) and the maximum load case (ID 458). In

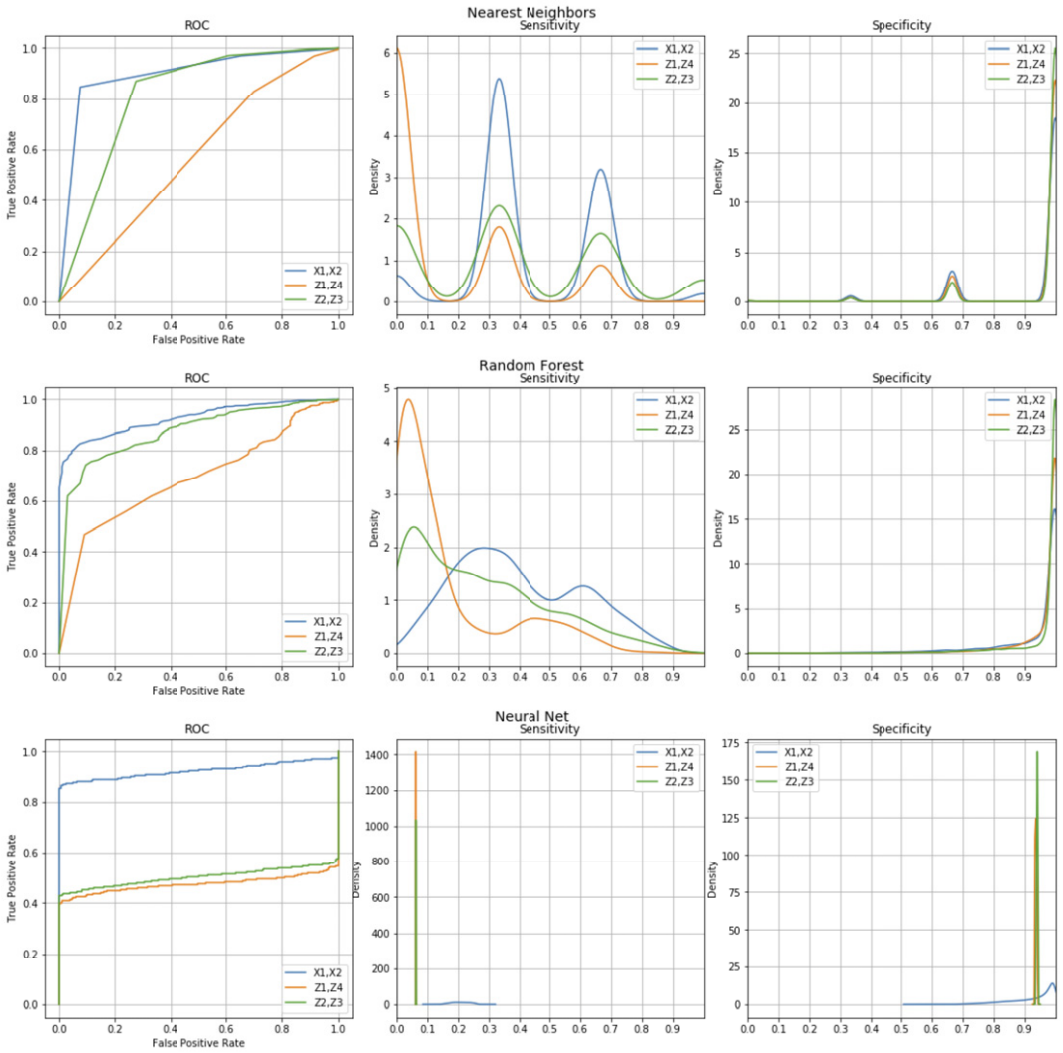

Figure 9. ROC curves and classifier performance indicator distributions for the ambiguous case (ID7). We tested three pairs of parameters here. The ROC curves on the left show the ratio between false positives and true negatives. Throughout all classifiers it indicates the best performance for the parameters X1–X2, which matches our expectations for this case. The specificity (right) is high for all classifiers and parameter pairs. Regarding sensitivity (middle) the random forest approach is the most stable.

general, we observe that there is no overall best way for classification regarding the special cases, which copes with the "No free Lunch" theorem by Wolpert and Macready [44] and specifically holds for ML as well [45]. All classifications have low sensitivity between 0.102 and 0.422 (mean = 0.212, SD = 0.093), while accuracy is always above 0.842 (mean = 0.894, SD = 0.029) and specificity is usually high, between 0.892 and 0.928 (mean = 0.938, SD = 0.02). The classifiers vary from case to case and from one projection to the other. We can only merely explain the differences between the cases as well as along the methodologies or projections just by studying the performance indicators. Therefore, we enhance the analysis with a more detailed view using the decision boundaries as depicted in Fig. 8.

From the analysis of the performance indicators for case ID 7, we can state that all classifiers retrieved the best results for the (X1, X2) pair (see Fig. 9). The specificity is high for all classifiers; only the *k*-nearest-neighbor approach has two low specificity peaks. For the sensitivity, the neural net performs worst. The *k*-nearest-neighbor has the highest sensitivity peak, but also a higher low one. The Random Forest approach is stabler in this regard with an overall higher sensitivity. On the other hand, the KNN and Random Forest approach tend to over-evaluate the importance of the other parameters (Z1–Z4). Now looking at the maximum load case, the difference between the classifiers is shrinking.

4.4 *Comparison among the Classifiers for Selected Cases*

4.4.1 *k-nearest-neighbor Method*

The *k*-nearest-neighbor (KNN) method performs well for cases with few dominant effects like the ambiguous cases 2 and 7 studied here, where only X1 and X2 are varied. In general, we can observe that the classifier performs well in the X1–X2 projection for all cases. The ROC curves for this approach vary significantly. In the sensitivity distribution we can observe the absence of a low sensitivity cluster in case 7

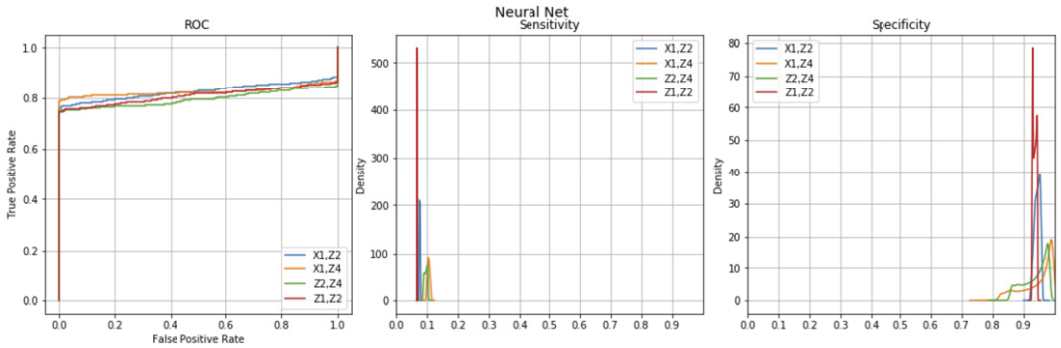

Figure 10. ROC curves and classifier performance indicator distributions for the maximum load case (ID458). For the selected boundary case the classifiers are struggling to fit an adequate model. Only analyzing the ROC curves would imply that the neural net seems to perform well. A closer look at the sensitivity and specificity reveals that this results from very high specificity values shadowing the very low sensitivity.

compared to case 2. Analyzing the decision boundaries (see Fig. 8), we find multiple small clusters with high prediction probabilities. This effect shows that the *KNN* approach is very sensitive to outliers.

4.4.2 *Random Forest*

The random forest approach is, on average, the best performing classifier with respect to sensitivity (mean = 0.25, SD = 0.11). It performs similarly to the KNN approach, but, due to its random nature, it is more stable in the presence of outliers, producing better results on average. We can observe this as well for the ROC curve which shows slighter changes than for the KNN approach. In the distribution there is only one dominant cluster, which stretches out over the three clusters from the KNN. Due to its weighting of local minima, it allows us to better detect clusters when compared to the KNN approach, see Fig. 8. But the randomness makes it harder to predict the right interval.

4.4.3 *Neural Network*

The rudimentary trained neural network performs poorly in most cases. Average sensitivity is 0.172 and standard deviation is only 0.078, which makes it the most stable method. In the ROC analysis, on the other hand, the neural net achieved overall best results. In the sensitivity distribution there is only one dense cluster, but with low sensitivity. There are only minor variations between the cases, which confirms the low standard deviation. When analyzing the decision boundaries, we observe that the neural net forms one clear cluster around the target parameter set (see Fig. 8), but with low prediction probability. This observation is an example of misled deduction based on merely analyzing methods' higher-level performance characteristics, e.g., ROC curves. In summary the neural net is better deducing a general model and less likely to over-fit.

4.5 *Symmetrical Load Conditions on Real Car Body Part*

In general, the results reassemble those from the generic case, but with more noise. Instead of repeating the results, we want to highlight one anomaly that caught our attention. In the case where only "SB_L" or "SB_R" is varied, which resembles the ambiguous case from the generic example, we spot a significant difference in the predictive quality. For the case "SB_L," the proposed methodology worked as expected, like in the generic use case (see Fig. 11 top). However, for the symmetrical case, "SB_R," none of the studied methods here revealed a consistent and correct dominating parameter. After some further testing, we found out that this effect reoccurs when a significant high load in the positive z-direction at "SB_R" is induced while "S_L" receives nearly zero loads. The decision boundary of the classifiers (see Fig. 11 bottom) clearly shows this. We assume that the current labeling approach does not adequately cover the inherent asymmetric rigidity and distribution of the boundary conditions, requiring a solution at the preprocessing level to capture this property in the distance metric adequately.

5. CONCLUSIONS

We have introduced a visual analytics approach to detect and analyze the dominant parameters in manufacturing assembly processes, supporting *a priori* QC from a reverse engineering perspective. Our approach makes possible a comparison by providing the ability to explore various indicators and use different visualization methods in conjunction to show differences for effective interpretation. Our integrative approach to combining visual analytics and ML-based classification results, with a focus on especially visualizing decision boundaries, allows a user to efficiently explore dominant parameters for a detailed analysis, see Figs. 8, 11. For relatively simpler cases, the analysis of distributions is a viable option. An ensemble analysis provides us with helpful insight for identifying parameter pairs to focus the correlation analysis on. This is an important contribution, as this allows us to reduce the SPLOM to few comparisons. Dimension reduction techniques can be used to identify higher-order correlations, revealing potentially relevant parameter pairs for a specific case. It is possible to use relatively few projections to explore decision boundaries, as demonstrated in our example. We have shown how classifiers can be compared regarding their predictive visual shape in their respective prediction classes. We have used

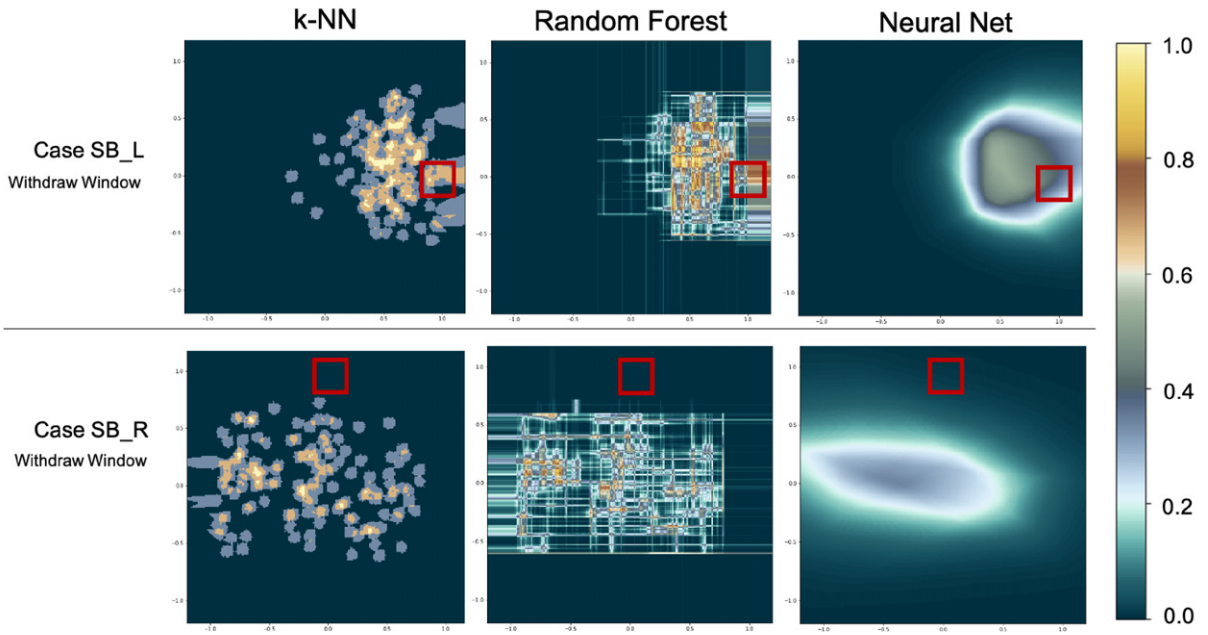

Figure 11. Different classification for symmetric boundary conditions for real car body part (sheet metal). When applying the methods to the sheet metal, we can observe large performance differences for symmetric boundary conditions. The left-side boundary condition (*SB_L*) is better predicted than the right-sided (*SB_R*). The inherent asymmetric rigidity of the part is not captured by this simple approach. For more complex parts an *a priori* feature or cluster definition in the domain space would be needed.

binary classification, but the approach can use any number of classes. The limiting factor is a human's cognitive capacity for visual comparison.

There does not exist a dominating classifier that can handle all exceptional cases better than the others. We can investigate drawbacks of the classifiers for ambiguous load cases. ROC curves used for comparison are ideally part of a methodology, e.g., when comparing different parameters in a random forest approach. For the comparison of methods the shape of decision boundaries leads to better results. Even when employing only a rudimentary approach for labeling and training, classifiers identify trend behavior of dominant parameters, effectively supporting decision-making for ruling out specific parameter sets (high specificity). It should be possible to achieve even better results by improving labeling. This could be realized via feature detection or local clustering instead of considering RMS deformation distance values. Our method allows one to directly compare and interpret changes in the labeling approach for each classifier and classifier parameter sets, which is crucial for comparing neural networks. The high-dimensional data visualization challenge remains, considering the large numbers of parameters involved and the fact that decision boundaries are easily comprehensible only with visualizations in the plane. It is planned to compare different classification parameters and labeling approaches, and extend the framework by adding capabilities for enhanced interactive visualization. It is an open question whether a generally applicable optimal distance metric exists, as we believe that it is the underlying application that determines a best specific metric.

ACKNOWLEDGMENT

This research was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)—252408385—IRTG 2057. The authors thank Q-DAS GmbH for providing CAD models.

REFERENCES

[1] S. Pavnaskar, J. Gershenson, and A. Jambekar, "Classification scheme for lean manufacturing tools," Int. J. Prod. Res. **41**, 3075–3090 (2003).
[2] C. Zhuang, J. Liu, and H. Xiong, "Digital twin-based smart production management and control framework for the complex product assembly shop-floor," Int. J. Adv. Manuf. Technol. **96**, 1149–1163 (2018).
[3] M. Schluse, M. Priggemeyer, L. Atorf, and J. Rossmann, "Experimentable digital twins—streamlining simulation-based systems engineering for industry 4.0," IEEE Trans. Ind. Informatics **14**, 1722–1731 (2018).
[4] Q. Qi and F. Tao, "Digital twin big data towards smart manufacturing and industry 4.0: 360 degree comparison," IEEE Access **6**, 3585–3593 (2018).
[5] S. Haag and R. Anderl, "Digital twin – proof of concept," Manuf. Lett. **15**, 64–66 (2018).
[6] H. Zhang, Q. Liu, X. Chen, D. Zhang, and J. Leng, "A digital twin-based approach for designing and multi-objective optimization of hollow glass production line," IEEE Access **5**, 26901–26911 (2017).
[7] C. Wagner, J. Grothoff, U. Epple, R. Drath, S. Malakuti, S. Gruner, M. Hoffmeister, and P. Zimermann, "The role of the Industry 4.0 asset administration shell and the digital twin during the life cycle of a plant," *2017 22nd IEEE Int'l. Conf. on Emerging Technologies and Factory Automation (ETFA), Limassol* (IEEE, Piscataway, NJ, 2017), pp. 1–8.
[8] T. H.-J. Uhlemann, C. Lehmann, and R. Steinhilper, "The digital twin: realizing the cyber-physical production system for industry 4.0," Proc. CIRP **61**, 335–340 (2017).
[9] F. Tao and M. Zhang, "Digital twin shop-floor: a new shop-floor paradigm towards smart manufacturing," IEEE Access **5**, 20418–20427 (2017).
[10] J. L. Hintze and R. D. Nelson, "Violin plots: a box plot-density trace synergism," Am. Stat. **52**, 181–184 (1998).

[11] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," Chemometr. Intell. Lab. Syst. **2**, 37–52 (1987).

[12] J. Goldberger, G. E. Hinton, S. T. Roweis, and R. R. Salakhutdinov, "Neighbourhood components analysis," *Advances in Neural Information Processing Systems* (MIT Press, Vancouver, 2004), pp. 513–520.

[13] T. Hastie, R. Tibshirani, and J. Friedman, "Additive models trees, and related methods," *The Elements of Statistical Learning*, Springer Series in Statistics (Springer, New York, NY, 2009), pp. 295–336.

[14] T. Hastie, R. Tibshirani, and J. Friedman, "Neural networks," *The Elements of Statistical Learning*, Springer Series in Statistics (Springer, New York, NY, 2009), pp. 389–416.

[15] F. Zhou, X. Lin, C. Liu, Y. Zhao, P. Xu, L. Ren, T. Xue, and L. Ren, "A survey of visualization for smart manufacturing," J. Vis. **22**, 419–435 (2019).

[16] P. Xu, H. Mei, L. Ren, and W. Chen, "ViDX: visual diagnostics of assembly line performance in smart factories," IEEE Trans. Vis. Comput. Graphics **23**, 291–300 (2017).

[17] D. Ramanujan and W. Z. Bernstein, "VESPER: visual exploration of similarity and performance metrics for computer-aided design repositories," *Volume 3: Manufacturing Equipment and Systems, College Station, Texas, USA* (American Society of Mechanical Engineers, New York, 2018).

[18] W. Wu, Y. Zheng, K. Chen, X. Wang, and N. Cao, "A visual analytics approach for equipment condition monitoring in smart factories of process industry," *2018 IEEE Pacific Visualization Symposium (PacificVis), Kobe* (IEEE, Piscataway, NJ, 2018), pp. 140–149.

[19] M. C. Leu, H. A. ElMaraghy, A. Y. Nee, S. K. Ong, M. Lanzetta, M. Putz, W. Zhu, and A. Bernard, "CAD model based virtual assembly simulation, planning and training," CIRP Ann. **62**, 799–822 (2013).

[20] W. Sun, J. Chen, and J. Li, "Decision tree and PCA-based fault diagnosis of rotating machinery," Mech. Syst. Signal Process. **21**, 1300–1317 (2007).

[21] T. Bui-Thanh, M. Damodaran, and K. Willcox, "Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition," AIAA J. **42**, 1505–1516 (2004).

[22] S. Gaggero, G. Vernengo, D. Villa, and L. Bonfiglio, "A reduced order approach for optimal design of efficient marine propellers," Ships Offshore Struct. **15**, 200–214 (2020).

[23] R. Swischuk, L. Mainini, B. Peherstorfer, and K. Willcox, "Projection-based model reduction: Formulations for physics-based machine learning," Comput. Fluids **179**, 704–717 (2019).

[24] K. Lu, Y. Jin, Y. Chen, Y. Yang, L. Hou, Z. Zhang, Z. Li, and C. Fu, "Review for order reduction based on proper orthogonal decomposition and outlooks of applications in mechanical systems," Mech. Syst. Signal Process. **123**, 264–297 (2019).

[25] S. Freitag, B. Cao, J. Ninić, and G. Meschke, "Recurrent neural networks and proper orthogonal decomposition with interval data for real-time predictions of mechanised tunnelling processes," Comput. Struct. **207**, 258–273 (2018).

[26] L. Jiang, S. Liu, and C. Chen, "Recent research advances on interactive machine learning," J. Vis. **22**, 401–417 (2019).

[27] M. Jarema, I. Demir, J. Kehrer, and R. Westermann, "Comparative visual analysis of vector field ensembles," *2015 IEEE Conf. on Visual Analytics Science and Technology (VAST), Chicago, IL, USA* (IEEE, Piscataway, NJ, 2015), pp. 81–88.

[28] B. Rieck and H. Leitte, "Exploring and comparing clusterings of multivariate data sets using persistent homology," Comput. Graph. Forum **35**, 81–90 (2016).

[29] A. Kumpf, B. Tost, M. Baumgart, M. Riemer, R. Westermann, and M. Rautenhaus, "Visualizing confidence in cluster-based ensemble weather forecast analyses," IEEE Trans. Vis. Comput. Graphics **24**, 109–119 (2018).

[30] Z. Zhang, K. T. McDonnell, E. Zadok, and K. Mueller, "Visual correlation analysis of numerical and categorical data on the correlation map," IEEE Trans. Vis. Comput. Graphics **21**, 289–303 (2015).

[31] L. Shao, A. Mahajan, T. Schreck, and D. J. Lehmann, "Interactive regression lens for exploring scatter plots," Comput. Graph. Forum **36**, 157–166 (2017).

[32] J. Wang and K. Mueller, "The visual causality analyst: an interactive interface for causal reasoning," IEEE Trans. Vis. Comput. Graphics **22**, 230–239 (2016).

[33] C. Bryan, X. Wu, S. Mniszewski, and K.-L. Ma, "Integrating predictive analytics into a spatiotemporal epidemic simulation," *2015 IEEE Conf. on Visual Analytics Science and Technology (VAST), Chicago, IL, USA* (IEEE, Piscataway, NJ, 2015), pp. 17–24.

[34] F. Claus, F.-A. Rupprecht, and H. Hagen, "Online simulation considering production uncertainties to improve assembly quality" (NAFEMS, Québec City, 2019), https://www.nafems.org/publications/resource_center/nwc_19_356/.

[35] D. H. Jeong, C. Ziemkiewicz, B. Fisher, W. Ribarsky, and R. Chang, "iPCA: an interactive system for PCA-based visual analytics," Comput. Graph. Forum **28**, 767–774 (2009).

[36] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: machine learning in Python," J. Machine Learning Res. **12**, 2825–2830 (2011).

[37] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," *Proc. Thirtieth Annual ACM Symposium on Theory of Computing – STOC '98, Dallas, Texas, United States* (ACM Press, 1998), pp. 604–613.

[38] K. L. Clarkson, "An algorithm for approximate closest-point queries," *Proc. Tenth Annual Symposium on Computational Geometry – SCG '94, Stony Brook, New York, United States* (ACM Press, 1994), pp. 160–164.

[39] A. Brutzkus, A. Globerson, E. Malach, and S. Shalev-Shwartz, "SGD learns over-parameterized networks that provably generalize on linearly separable data," (2017), arXiv: 1710.10174.

[40] A. Andoni, R. Panigrahy, G. Valiant, and L. Zhang, "Learning polynomials with neural networks," in *Proc. 31st Int'l. Conf. on Machine Learning, Bejing, China*, edited by E. P. Xing and T. Jebara, Proceedings of Machine Learning Research (PMLR) (ACM, 2014), Vol. 32, pp. 1908–1916.

[41] S. V. Stehman, "Selecting and interpreting measures of thematic classification accuracy," Remote Sens. Environ. **62**, 77–89 (1997).

[42] D. M. W. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," Int. J. Mach. Learn. Technol. **2**, 37–63 (2011).

[43] T. Fawcett, "An introduction to ROC analysis," Pattern Recognit. Lett. **27**, 861–874 (2006).

[44] D. H. Wolpert and W. G. Macready, "No free lunch theorems for search," Technical Report SFI-TR-95-02-010, (1995).

[45] S. Ali and K. A. Smith, "On learning algorithm selection for classification," Applied Soft Computing **6**, 119–138 (2006).