The VLDB Journal (2023) 32:1037–1064
https://doi.org/10.1007/s00778-023-00780-6

REGULAR PAPER

Exploiting domain knowledge to address class imbalance and a heterogeneous feature space in multi-class classification

Vitali Hirsch[2] · Peter Reimann[1] · Dennis Treder-Tschechlov[2] · Holger Schwarz[2] · Bernhard Mitschang[2]

Received: 16 February 2022 / Revised: 21 December 2022 / Accepted: 6 January 2023 / Published online: 27 February 2023

Abstract
Real-world data of multi-class classification tasks often show complex data characteristics that lead to a reduced classification performance. Major analytical challenges are a high degree of multi-class imbalance within data and a heterogeneous feature space, which increases the number and complexity of class patterns. Existing solutions to classification or data pre-processing only address one of these two challenges in isolation. We propose a novel classification approach that explicitly addresses both challenges of multi-class imbalance and heterogeneous feature space together. As main contribution, this approach exploits domain knowledge in terms of a taxonomy to systematically prepare the training data. Based on an experimental evaluation on both real-world data and several synthetically generated data sets, we show that our approach outperforms any other classification technique in terms of accuracy. Furthermore, it entails considerable practical benefits in real-world use cases, e.g., it reduces rework required in the area of product quality control.

Keywords Classification · Domain knowledge · Multi-class imbalance · Heterogeneous feature space

1 Introduction

Many real-world use cases consider multi-class classification tasks instead of binary classifications (e.g., see [7,19,25,41,55]). In multi-class classification, a classifier trained on historical data must choose one of more than two class labels for each new observation. The number of possible class labels in real-world use cases typically ranges from ten to even thousands [7,19,25,41].

The data characteristics in real-world multi-class problems often impose several analytical challenges for classification approaches that have a negative effect on the classification performance. In this paper, we consider the two challenges of a multi-class imbalance and a heterogeneous feature space. *Multi-class imbalance* means that the class labels occur in an imbalanced way in the data [13,17,42]. Here, many learning algorithms tend to ignore the patterns of class labels that are underrepresented. The main problem of a *heterogeneous feature space* is that each class label may be associated with multiple class patterns that are represented by different and overlapping value ranges [17,42]. This makes it hard to detect clearly distinguishable class patterns.

These two challenges usually occur together in a broad range of real-world multi-class problems in various application domains. This for instance concerns problems across all stages of an industrial value chain, e.g., manufacturing or e-commerce [19,25,26,41,45]. Examples are a fault diagnosis of complex products or a classification of products into various product types. Moreover, both a multi-class imbalance and a heterogeneous feature space arise in applications

The authors thank the Ministry of Science, Research and Arts of the State of Baden-Wurttemberg for financial support of this work within the sustainability support of the projects of the Excellence Initiative II in the Graduate School of Excellence advanced Manufacturing Engineering (GSaME).

✉ Peter Reimann
Peter.Reimann@gsame.uni-stuttgart.de

Vitali Hirsch
vitali.hirsch@ipvs.uni-stuttgart.de

Dennis Treder-Tschechlov
dennis.treder-tschechlov@ipvs.uni-stuttgart.de

Holger Schwarz
holger.schwarz@ipvs.uni-stuttgart.de

Bernhard Mitschang
bernhard.mitschang@ipvs.uni-stuttgart.de

[1] Graduate School of Excellence advanced Manufacturing Engineering (GSaME), University of Stuttgart, Stuttgart, Germany

[2] Institute of Parallel and Distributed Systems (IPVS), University of Stuttgart, Stuttgart, Germany

of data-driven medical diagnoses. For instance, Chan et al. summarize corresponding problems from 70 articles related to data-driven detection of various types of skin cancer [7]. Throughout all these application domains, the two analytical challenges may be found in different kinds of data, e.g., sensor data, text data, and image data.

Various research communities work on the design and optimization of algorithms for machine learning and data engineering, e.g. for sampling, feature selection, and classification. However, most of these algorithms are only suitable to mitigate the negative effects of one of the two abovementioned analytical challenges [19]. In turn, they worsen the effects of the respective other challenge. Finally, the application of these algorithms that are tailored to single challenges even reduces prediction performance [19]. Thus, we argue that a classification approach has to consider both analytical challenges and their mutual influences.

In this paper, we propose a novel classification approach that addresses both a multi-class imbalance and a heterogeneous feature space. In contrast to most related work, our approach makes explicit use of available domain knowledge in terms of a taxonomy to systematically prepare the training data. The taxonomy allows for segmenting the training data into several sample subsets. This way, the feature space within each subset is much more homogeneous. Furthermore, our approach uses metrics characterizing the class imbalance to come up with informed decisions how to further partition the subsets. Thereby, we address multi-class imbalance.

This paper covers comprehensive extensions to an article previously published at PVLDB [21]. This prior article mainly focuses on the application and evaluation of our approach for a certain use case of a multi-class problem. This is a data-driven fault diagnosis in End-of-Line (EoL) testing of complex products [18,19]. Here, the domain taxonomy is represented by a product hierarchy that organizes different product variants into product groups based on the similarity of the variants. The major outcome of this use-case-specific evaluation is that our approach outperforms any baseline solution in terms of classification accuracy [21].

The comprehensive extensions in this paper investigate our approach in more detail with respect to various aspects of its generality. In particular, we show the potential of our approach for other multi-class problems from various application domains and for different kinds of data distributions. To this end, this paper covers the following main contributions:

- We add descriptions and discussions of algorithms for the major steps of our approach for preparing training data. This enhances reproducibility and facilitates the implementation of these major steps as well as their application to data of other use cases.
- We show that the challenges of a multi-class imbalance and a heterogeneous feature space occur in many other real-world classification problems. This includes use cases across the whole industrial value chain and even for data-driven medical diagnoses.
- Furthermore, we discuss the constraints that the available domain knowledge must satisfy in order for our approach to be applicable to the data of these various use cases. This way, researchers and practitioners of other domains see under which circumstances they can apply our approach.
- We provide additional measurements with synthetically generated data sets and different data distributions. This way, we prove that our approach effectively addresses the analytical challenges of a multi-class imbalance and a heterogeneous feature space also for other kinds of data. In fact, it significantly increases classification performance for any of the synthetic data sets. In addition, we come up with a more in-depth discussion of these results. This for instance includes the practical benefits that our approach entails for real-world use cases and that it can address ethnic or gender bias in data.

Section 2 provides insights into real-world data characteristics and the two considered analytical challenges. In Sect. 3, we discuss related work. Section 4 describes our classification approach, including the novel algorithms for its major steps. Section 5 discusses the results of our evaluation for the specific use case of EoL testing. The following two sections prove that our approach is generally applicable to further use cases and that it addresses a multi-class imbalance and a heterogeneous feature space even for other data distributions. Section 6 provides theoretical discussions regarding the generality of our approach, i.e., why it may be applied in other application domains. In Sect. 7, we discuss evaluation results for newly generated synthetic data sets. We conclude and list future work in Sect. 8.

2 Analytical challenges in real-world multi-class classification problems

In this paper, we focus on *multi-class* classification problems. Here, each observation is associated with exactly one of $C > 2$ class labels $c_i \in \mathcal{C} = \{c_1, \ldots, c_C\}$. We train a classifier $\mathcal{M}$ on a historical data set $\mathcal{X}$ with N samples (x_t, y_t). Each x_t characterizes an observation as an element of an F-dimensional *feature space* $\mathcal{F} = \{f_1, \ldots, f_F\}$. y_t represents the target class label c_i associated with x_t. In this section, we exemplify the two analytical challenges of a multi-class imbalance and a heterogeneous feature space by means of the real-world use case for fault diagnosis in EoL testing [19]. Nevertheless, these challenges are prevalent in other application domains as well (see Sect. 6.1).

In our use case, we consider powertrain aggregates, e.g., engines of motor vehicles. EoL testing constitutes the final functional check of such products after assembly. Thereby, product characteristics are tested based on sensor signals from a test bench. If a product does not pass the test, quality engineers try to identify the faulty component that causes the quality issue, e.g., a turbo charger. Operators replace the assumed faulty component and test the product again.

A data-driven classification may help quality engineers by recommending the most likely faulty components [19]. This constitutes a multi-class problem as described above. Each sample in the set $\mathcal{X}$ corresponds to a quality issue, while each class c_i represents one of the possibly faulty components of powertrain aggregates. In our use cases, the sample set $\mathcal{X}$ contains $N = 1050$ samples with $C = 84$ classes and $F = 115$ features [19]. Most of the features are the sensor signals of the test bench, while some of them represent certain product characteristics, e.g., the number of engine cylinders.

Note that it is common in several real industrial multi-class problems that the data sets include such a comparatively small number of samples that are labeled correctly and may thus be used to train classifiers [25,26,41,55]. One of the main reasons is that a correct labeling of samples requires domain expertise and lots of effort [4,37,41]. In the following, we discuss how a multi-class imbalance (C1) and a heterogeneous feature space (C2) even complicate the classification problem.

2.1 C1: Multi-class imbalance

In the sample set $\mathcal{X}$ of the EoL case, the top 5 classes together are contained in 29% of all samples, where each individual class has a comparatively high share of at least 4%. We denote such top classes as *majority classes* c_i^+ and their samples as *majority samples* $\mathcal{X}^+$. Each of the remaining 79 classes individually has a low share of samples, but all together are represented by 71% of the samples. We denote them as *minority classes* c_i^- that are represented by *minority samples* $\mathcal{X}^-$.

This uneven class distribution poses a multi-class imbalance problem [17]. Most classification algorithms tend to ignore minority samples $\mathcal{X}^-$ because they try to maximize accuracy by predicting everything to be one of the majority classes c_i^+. Hence, the resulting classifiers are biased towards majority classes [13]. We however require classifiers with a balanced degree of accuracy for both minority and majority classes. In a worst case, we are otherwise only able to predict five majority classes c_i^+, while 79 minority classes c_i^- and thus 71% of all samples are ignored. Note that this is also an issue in many other real-world multi-class problems [7,25,41].

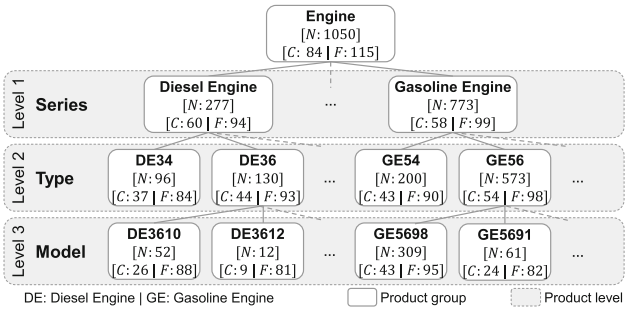


Fig. 1 Product hierarchy and numbers of samples N, classes C, and features F for the product groups in $\mathcal{X}$

2.2 C2: Heterogeneous feature space

The samples in $\mathcal{X}$ describe a variety of product variants with different technical specifications. The manufacturing domain groups products with similar technical characteristics into product groups and organizes them in a *product hierarchy* [1]. Figure 1 shows a typical product hierarchy for the engines of our use case. The first hierarchy level differentiates engines according to their *series*, e.g., whether they are diesel or gasoline engines (*DE*, *GE*). Level 2 divides them into *engine types*. For instance, group $GE54$ comprises four-cylinder gasoline engines, and $GE56$ six-cylinder gasolines. The bottom level describes *engine models* that further specify the components of an engine, e.g., which kind of injector it has. This product variety increases the *heterogeneity in the feature space* and leads to the following issues.

(C2.1) Missing features: Some features f_k are only measured for certain product variants. For example, the test bench delivers one particular feature for each cylinder of an engine. So, product variants in group $GE54$ comprise four features for cylinders, while six-cylinder variants in group $GE56$ comprise two additional features. The data structure of sample set $\mathcal{X}$ contains a column for each of the overall 115 features. Hence, a sample has six columns for cylinders, whereas two of these columns do not contain any value (i.e., "NA") for four-cylinder variants. This issue leads to several missing feature values in the whole set $\mathcal{X}$. In our use case, 17% of all values in $\mathcal{X}$ are "NA" values.

To train a classifier, the missing values must be *imputed* or *removed*. We then either create artificial values for some features f_k that have no technical causality to a particular product variant, or we remove features that may characterize a specific class c_i. For group $GE54$ with four cylinders, imputing values means that we generate artificial features f_k for the cylinders No. 5 and No. 6 that are however physically not available. By removing the features for cylinders No. 5 and No. 6, we lose information for group $GE56$ with six cylinders.

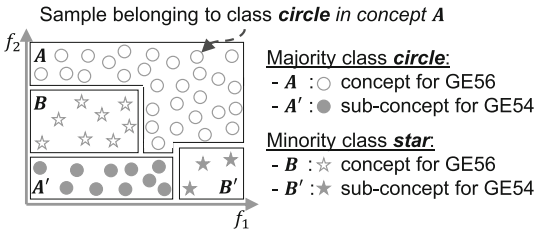


Fig. 2 Illustration of analytical challenge *sub-concepts* (C2.2) with two classes *circle* and *star*. Rectangles define the actual (sub-)concepts, i. e., the decision rules

(C2.2) Sub-concepts: For different product variants, the same class c_i may be defined by the same features, but with different value ranges. Hence, classification algorithms may have to learn multiple concepts for a single class. This even reduces the number of samples that is covered by each concept. This challenge is called *sub-concepts* in literature [17,42]. Figure 2 shows an artificial data set with two example features and two classes. For example, the figure shows concept B for group $GE56$ and sub-concept B' for group $GE54$. B and B' describe patterns for class star with the same features f_1 and f_2. However, B is characterized by low f_1 and medium f_2 values, while samples of B' have high f_1 and low f_2 values. Here, sub-concept B' is only represented by three minority samples of class star. This lack of samples may cause a learning algorithm to neglect sub-concept B'. The feature ranges for B' are then assigned to the wrong concepts A or A' of class circle.

3 Related work

Related work comprises several methods that address class imbalance (C1). Various reviews show that most methods are designed for two-class problems and are hence less effective for multi-class tasks [10,13,16,17,29,42,56]. Most solutions use *class decomposition schemes* such as *One-vs-All* (OvA) to convert a multi-class problem into several two-class problems. Afterwards, they apply two-class imbalance techniques to balance each binary sub-problem [50]. However, such decomposition schemes may reduce prediction performance if the data also contains a heterogeneous feature space (C2) [19]. The few exceptions that deal with multi-class imbalance are *cost-sensitive* techniques that consider costs as penalty of different types of misclassification [10,17,42,50]. One difficulty is that the real costs are often unknown or hard to calculate for a given problem [16,42]. Furthermore, these techniques may only address multi-class imbalance, but they are not able to additionally balance underrepresented sub-concepts (C2.2).

Other approaches use ensembles that employ *random subspace selection* to address the missing feature problem (C2.1) [33,35]. The idea is to generate multiple *base learners* $\mathcal{L}_j$ that each is trained with a random subset of features from the feature space $\mathcal{F}$. To classify a new sample x_t with missing features, only those base learners $\mathcal{L}_j$ trained with the features that are available in x_t are used. Polikar et al. [35] show that this approach has two assumptions: First, the set of features in $\mathcal{F}$ is partly *redundant*, so that the classification problem is solvable with a real subset of the features. Second, this redundancy is distributed evenly over $\mathcal{F}$. However, a heterogeneous feature space with a multiplicity of sub-concepts (C2.2) entails that these assumptions do not hold for the given sample set $\mathcal{X}$. For instance, a previous study shows that only 10 out of the initial 115 features of the sample set of our EoL test case were rated as redundant by feature selection techniques [19].

So, numerous statistical techniques exist that address single challenges in isolation. However, these techniques imply other negative effects to classification. Our key statement is that a classification approach has to consider all challenges and their mutual influences. Here, we opt for an approach that explicitly uses domain knowledge to systematically prepare the training data.

Related approaches use a pre-defined *class taxonomy* for *hierarchical classification* [39]. The assumption is that classes belonging to the same concept in the taxonomy have shared characteristics and relationships. We can then train a classifier $\mathcal{M}_l$ for each level l of the taxonomy. Based on the prediction of $\mathcal{M}_l$, we apply the next classifier $\mathcal{M}_{l+1}$ one level lower. We repeat this until we recognize the classes c_i on the leaf nodes. However, this does not solve the problem with multiple and unevenly distributed sub-concepts (C2.2). Here, we require a hierarchy that organizes the individual sub-concepts instead of the classes. This is for instance offered by a product hierarchy that organizes individual product groups and thus also their sub-concepts.

Other methods use domain knowledge for feature engineering. For instance, domain experts may specify known causal dependencies between features, which are then used for dimensionality reduction or to increase the information content in the feature space [47]. However, it is very time-consuming or even impossible for domain experts to acquire and specify all causal dependencies between features in real-world application domains. This is mainly due to the heterogeneity in the feature space (C2). This challenge often leads to a very large number of complex dependencies between features that cannot be easily distinguished by domain experts.

Snorkel lets domain experts specify labeling functions to describe causal dependencies between features and class labels [4,37]. These functions may be used to label individual samples of existing training data. This may be an appropriate approach to binary classification problems, where the number of class patterns and thus the number of required

labeling functions is small. However, this approach does not scale for multi-class problems with hundreds or even more classes [41]. Here, domain experts would have to specify a multitude of labeling functions for the high number of classes (C1) and for an even higher number of (sub-)concepts (C2.2).

Similar to our approach, constraint-based clustering [30, 48] may be used to partition training data prior to applying classification algorithms. Domain experts specify the constraints that for instance describe that two samples belong to the same or to different clusters. Constraint-based clustering algorithms then partition the data into clusters and ensure that all specified constraints are satisfied. These approaches however share the same drawback as approaches to feature engineering or to labeling functions. In complex real-world scenarios with a heterogeneous feature space (C2), domain experts have to spend a huge effort to specify a multitude of constraints to reflect all relevant feature dependencies. Often, domain experts are only able to specify a minor subset of relevant constraints. So, this approach does not scale well for complex real-world applications that exhibit both analytical challenges C1 and C2.

Hence, a more scalable approach for complex multi-class problems may not directly involve domain experts. Instead, we exploit domain knowledge that already exists in the relevant application domain. For instance, any manufacturing company possesses domain knowledge as a documentation of a company's product family, e.g., a product hierarchy [1]. We may use the clearly distinguishable product groups and their hierarchical relationships to partition the sample set into several subsets in which the negative effects of both challenges C1 and C2 are mitigated. In other domains, the necessary domain knowledge may be provided by hierarchical relationships of knowledge graphs or semantic nets, e.g., taxonomies or ontologies [36,40] (see Sect. 6.2).

4 Classification approach

We now introduce our classification approach that addresses both analytical challenges C1 and C2 together. The core idea is to exploit available domain knowledge in a training set preparation phase to partition the whole sample set $\mathcal{X}$ into several subsets (Sect. 4.1). Afterwards, in the predictive modeling, we train a classifier for each sample subset and combine the results of the classifiers (Sect. 4.2). In the following, we mainly focus on how to use product hierarchies from manufacturing as domain knowledge to partition the data. Nevertheless, we discuss in Sect. 6 that our approach is also applicable in other domains, e.g., the medical domain. The approach even works with various kinds of hierarchical domain models, even if they differ from the product hierarchy shown in Fig. 1, e.g., in terms of the number of levels and the branching factor.

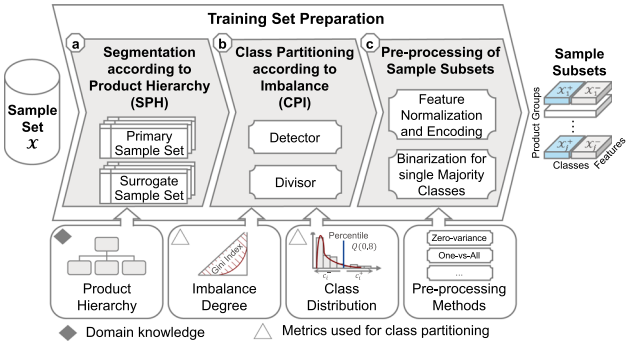


Fig. 3 Major steps of the training set preparation that exploits domain knowledge from a product hierarchy and reasonable metrics for class distributions to address the analytical challenges discussed in Sect. 2

4.1 Training set preparation

Figure 3 shows the steps of the training set preparation to partition a sample set $\mathcal{X}$. The main step (a) is the *segmentation according to product hierarchy* (SPH). SPH uses this hierarchy to divide $\mathcal{X}$ according to individual product groups. This way, we obtain sample subsets with technically similar product variants. So, the features f_k in each subset are more homogeneous, i.e., SPH addresses challenge C2. The next step (b) is a *class partitioning according to imbalance* (CPI) addressing challenge C1. Based on metrics that describe the class distributions of the sample subsets, CPI makes informed decisions on whether and how to further divide the subsets among majority and minority classes. In the last step (c), we pre-process the resulting sample subsets for the training phase.

4.1.1 Segmentation according to product hierarchy

The standard design principle is that a classifier $\mathcal{M}$ is trained on the entire sample set $\mathcal{X}$. In our approach, we however divide $\mathcal{X}$ into several subsets according to individual levels of the domain taxonomy, e.g., the product hierarchy shown in Fig. 1. For the hierarchy level "$Type$" as example, we generate subsets $\mathcal{X}_{(l,j)}$ for each product group from $DE34$ to $GE56$, where j indexes the product group on the hierarchy level l. For reasons of simplification, we use the notation $\mathcal{X}_j$ for a group j if its level l is obvious from the context.

By considering only technically similar product variants in a sample subset $\mathcal{X}_j$, we mitigate the missing feature problem (C2.1). For example, by considering the four-cylinder product group $GE54$ on its own, all samples in the relevant subset $\mathcal{X}_j$ do not contain features for cylinders No. 5 and 6 anymore. As $\mathcal{X}_j$ now comprises samples from similar product variants, we also reduce the variety in the value ranges of features f_k. This leads to a decreased number of sub-concepts (C2.2).

It is critical to select a proper hierarchy level l when performing the segmentation into sample subsets. The deeper we go into the hierarchy, the more we reduce the heterogeneity in the feature space. However, if the chosen level l is too deep, the number of samples for a particular group j may be too small to train a classifier. For example, group $DE3612$ at level "$Model$" has only twelve samples to characterize nine classes. Other groups on the same hierarchy level have enough samples though, e.g., group $GE5698$ with 309 samples. To handle these different groups at a specific hierarchy level l, we introduce *primary* and *surrogate* sample subsets.

Primary sets contain samples of product groups that are located at a deeper level in the hierarchy, where we mitigate the effects of challenge C2 to the greatest extent. In our product hierarchy shown in Fig. 1, this is level "$Model$". We may use the 309 samples of group $GE5698$ as a primary set. However, for very small groups, such as the sibling group $DE3612$, we introduce surrogate sample sets that are located at least one hierarchy level higher. For instance, group $DE36$ at level "$Type$" contains 130 samples, which is enough to train a classifier. We hence use group $DE36$ as a surrogate set to represent group $DE3612$ one level lower. This means we build a classifier with the samples of group $DE36$ and then use this classifier to predict the class for new samples that belong to group $DE3612$.

As a result, a surrogate set contains more samples, but is less specific, i.e., its feature space is a bit more heterogeneous than that of a primary set. Nevertheless, a surrogate set at a lower level than the root node is still more homogeneous than the whole sample set $\mathcal{X}$. So, surrogate sets also help to solve the problem of a heterogeneous feature space (C2), but not to the same extent as primary sets. Surrogate sets represent a kind of trade-off between the entire data set with more data but a very heterogeneous feature space and individual primary sets with a largely homogeneous feature space but less data.

To create primary and surrogate sample sets, we designed Algorithm 1 that traverses the product hierarchy from the bottom to the top. In our use case, we start the hierarchy traversal at the third level "$Model$". We denote this bottom level as $max_l = 3$. We then intend to create a primary sample set for each group j at this bottom level. Therefore, we check each group j at level $l = max_l$ to see whether the primary sample subset $\mathcal{X}_{(l,j)}$ fulfills the requirements to train a classifier $\mathcal{M}_{(l,j)}$ (see call of procedure CHECKS at line 6).

A classification algorithm requires a class to be represented by a minimum number of samples to learn a meaningful class pattern. So, we remove all classes and their samples from subset $\mathcal{X}_{(l,j)}$ that are represented by less than two samples (line 17). The rationale behind this rather low threshold is that many of our sample subsets, especially the primary subsets at the bottom level of our taxonomy, may contain several classes with a low number of samples. Hence,

Algorithm 1 SPH algorithm

```
   Input: 𝒳: Sample set, PH: Product hierarchy,
          thrInfoLoss: Threshold for loss of information,
          max_l: Maximum level of PH to start traversal
   Output: ℛ: Resulting sample subsets
1: procedure SPH(𝒳, thrInfoLoss, PH, max_l)
2:    ℛ ← ∅
      // Get number of nodes at bottom level max_l
3:    num_nodes ← PH.GET_NUMBER_NODES(max_l)
4:    for j = 1, ..., num_nodes do
         // Get subset for group j at level max_l of PH
5:       𝒳_(max_l,j) ← PH.GET_SUBSET(𝒳, max_l, j)
         // Check whether current sample subset
         // may be used to train a classifier
6:       𝒳_(l,k) ←CHECKS(𝒳, 𝒳_(max_l,j), thrInfoLoss, PH)
7:       Append Subset 𝒳_(l,k) to ℛ
8:    end for
9:    return ℛ
10: end procedure
11:
12: procedure CHECKS(𝒳, 𝒳_(l,j), thrInfoLoss, PH)
      // Return subset if we are at the root node
13:   if l == 0 then
14:      return 𝒳_(l,j)
15:   end if
16:   n ← |𝒳_(l,j)|
      // Remove classes with less than two samples
17:   𝒳_(l,j) ← 𝒳_(l,j) \ {(x, y) : #SAMPLES(y, 𝒳_(l,j)) < 2}
      // Check if only one class left in 𝒳_(l,j)
      // or if line 17 removed too many samples
18:   if #CLASSES(𝒳_(l,j)) = 1 or
19:      (n − |𝒳_(l,j)|)/n > thrInfoLoss then
         // Get index k of parent node and subset 𝒳_(l−1,k)
20:      k ← PH.PARENT_NODE(j)
21:      𝒳_(l−1,k) ← PH.GET_SUBSET(𝒳, l − 1, k)
22:      return CHECKS(𝒳, 𝒳_(l−1,k), thrInfoLoss, PH)
23:   else
         // All criteria satisfied, return current subset
24:      return 𝒳_(l,j)
25:   end if
26: end procedure
```

we use a likewise low threshold in order to preserve as many classes as possible and not lose too much information in the sample subsets. Afterwards, we check two criteria for a subset $\mathcal{X}_{(l,j)}$. The first criterion ensures that $\mathcal{X}_{(l,j)}$ contains at least two classes (line 18), so that a classification algorithm can identify decision boundaries between these classes. With the second criterion, we assure that the *loss of information* due to the previous removal of classes with too few samples is not higher than a specific threshold parameter (line 19). We check that the removal of classes does not reduce the number of samples in $\mathcal{X}_{(l,j)}$ by more than, e.g., 25%-points. Note that we discuss in Sect. 5.2.1 how we have found the value for this threshold parameter yielding the best classification accuracy. This also holds for the thresholds of Algorithm 2. In Sect. 7.4, we discuss the effects of optimiz-

ing these parameters for different kinds of data distributions of other multi-class problems.

If a subset $\mathcal{X}_{(l,j)}$ meets both criteria, it becomes a primary set. If it does not meet at least one criterion, we visit the parent node of group j one level higher (lines 20 to 22). If the sample set of this parent node meets all criteria, we consider it as a surrogate sample set for group j. Otherwise, we further traverse the hierarchy along the path to the root node. We do so until we are able to represent each group j by either its primary set at level max_l or by a surrogate set at a higher level.

Note that already this SPH reduces the class imbalance to a certain degree (C1). Figure 4 shows example distributions of the most frequent classes c_i (error codes A to G) after the segmentation into groups $GE54$ ($\mathcal{X}_1$) and $DE34$ ($\mathcal{X}_2$). If we consider both groups together, i.e., without performing the segmentation, this leads to a significant imbalance. Then, error code A is the most common class with in total 61 samples. All other classes comprise between 5 and 16 samples only. SPH allows us to consider the groups $GE54$ and $DE34$ separately. Therefore, error code A has only 10 samples in $DE34$ and is thus not a dominant class anymore for this group. So, the segmentation positively influences the class imbalance for group $DE34$.

4.1.2 Class partitioning according to imbalance

In the sample set of group $GE54$ shown in Fig. 4, however, error code A remains a majority class c_i^+ with 51 samples. This still leads to a distinct class imbalance (C1), where error codes B, C, and D may be superimposed by code A. We hence perform a class partitioning to create disjoint majority subsets $\mathcal{X}_j^+$ and minority subsets $\mathcal{X}_j^-$ for each subset $\mathcal{X}_j$ with a distinct class imbalance. For group $GE54$, we create a majority subset $\mathcal{X}_1^+$ with all samples of error code A and a minority subset $\mathcal{X}_1^-$ with all other samples (cf. Figure 4). This way, we reduce the class imbalance in subsets $\mathcal{X}_1^+$ and $\mathcal{X}_1^-$ (C1). Thereby, we ensure that learning algorithms can recognize error codes B, C, and D within $\mathcal{X}_1^-$.

The *detector* component shown in Fig. 3b uses a statistical metric to decide whether a subset $\mathcal{X}_j$ shows a distinct class imbalance and thus has to be partitioned. Then, the *divisor* component partitions $\mathcal{X}_j$ into majority and minority subsets $\mathcal{X}_j^+$ and $\mathcal{X}_j^-$. Algorithm 2 formalizes this step CPI of our approach.

Detector: There is no consensus on proper statistical metrics to determine the degree of class imbalance within data [10]. In our approach, the metric must have a normalized interval between 0 and 1, where the boundaries represent a *total balance* (0) or a *total imbalance* (1). This allows us to directly compare the results of the metric between all subsets $\mathcal{X}_j$.

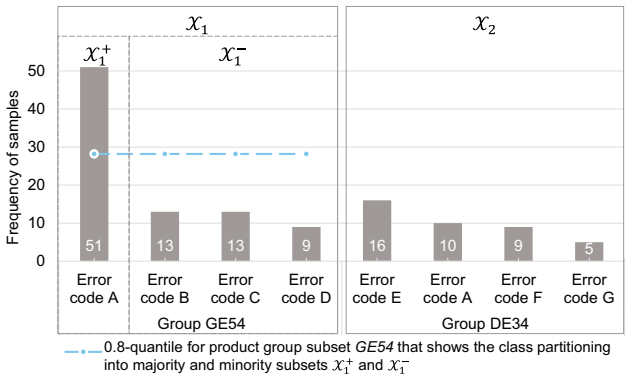


Fig. 4 Class distribution after SPH ($\mathcal{X}_1$ and $\mathcal{X}_2$) and after CPI ($\mathcal{X}_1^+$ and $\mathcal{X}_1^-$). Error codes A to G represent individual classes c_i in the sample subsets

Algorithm 2 CPI algorithm

Input: $\mathcal{R}$: Sample subsets after SPH,
thr_gini: Threshold for the Gini index,
p: Parameter value for the p-quantile
Output: $\mathcal{R}$: Resulting sample subsets after CPI

1: **procedure** CPI($\mathcal{R}$, thr_gini, p)
2: **for** $\mathcal{X}_i \in \mathcal{R}$ **do**
// Detector component
3: **if** GINI($\mathcal{X}_i$) > thr_gini **then**
// Divisor component
4: $Q_p \leftarrow$ P_QUANTILE($\mathcal{X}_i$, p)
5: $\mathcal{X}_i^- \leftarrow \{(x, y) \in \mathcal{X}_i$: #SAMPLES($y, \mathcal{X}_j$) $\leq Q_p\}$
6: $\mathcal{X}_i^+ \leftarrow \mathcal{X}_i \setminus \mathcal{X}_i^-$
7: Replace $\mathcal{X}_i$ in $\mathcal{R}$ with tuple $(\mathcal{X}_i^-, \mathcal{X}_i^+)$
8: **end if**
9: **end for**
10: **return** $\mathcal{R}$
11: **end procedure**

One of the most prominent metrics that fulfills this requirement is the *Gini coefficient* [9,15]. We calculate this coefficient on the discrete class distribution of a sample subset based on the sum formula of the Lorenz curve. The Gini coefficient of the samples in Fig. 4 is about 0.28 for group $DE34$ and about 0.48 for group $GE54$. We use a certain threshold for this coefficient to decide whether a subset $\mathcal{X}_j$ has to be partitioned or not (lines 3 and 4 in Algorithm 2). For instance, with a threshold value of 0.3, we detect a distinct class imbalance for group $GE54$. We divide the subset $\mathcal{X}_1$ in the next step into the disjoint subsets $\mathcal{X}_1^+$ and $\mathcal{X}_1^-$.

Divisor: We also use a metric that acts as a threshold to determine the *point of intersection* to partition a set $\mathcal{X}_j$. Classes with more samples than the threshold are placed in the majority set $\mathcal{X}_j^+$, and the other ones in the minority set $\mathcal{X}_j^-$. Typical examples of metrics are the *arithmetic mean* or *standard deviation*. For our approach, we have chosen the *p-quantile* $Q(p)$. The major reason is that, from the many metrics, $Q(p)$ is the only one that allows for a parameterization with p to tune it for an improved prediction performance.

The calculation of $Q(p)$ is based on the *empirical cumulative distribution function* $F(x) = p$. Thereby, x is a number of samples and p is the share of classes in a subset $\mathcal{X}_j$ that are represented by x or less samples. $Q(p)$ is calculated using the *inverse function* of $F(x)$, i.e., $Q(p) = F^{-1}(x)$. This means that those classes in $\mathcal{X}_j$ that are represented by $Q(p)$ or less samples cover a share of p of the class distribution of $\mathcal{X}_j$. The idea is that we set a value for p and calculate $Q(p)$ for each subset $\mathcal{X}_j$ with a distinct class imbalance (line 5). Then, we place all classes with $Q(p)$ or less samples in the minority subset $\mathcal{X}_j^-$ and the remaining classes in the majority subset $\mathcal{X}_j^+$ (lines 6+7). This way, all minority subsets $\mathcal{X}_j^-$ cover a share of close to p of all classes in the original subsets $\mathcal{X}_j$. This again motivates our choice for $Q(p)$, as we may influence the relative portions of majority and minority classes via p.

As indicated in Fig. 4, the 0.8-quantile ($p = 0.8$) for group $GE54$ is about 28. Statistically speaking, all classes c_i having 28 or less samples represent 80% of the class distribution of group $GE54$. We consider $Q(0.8) = 28$ as a threshold. So, error code A, which has more than 28 samples, is the only majority class c_i^+ and its samples are majority samples $\mathcal{X}_1^+$. Error codes B, C, and D are minority classes c_i^- and their samples are minority samples $\mathcal{X}_1^-$. Finally, all resulting subsets $\mathcal{X}_1^+$ and $\mathcal{X}_1^-$ are much more balanced. CPI reduces the Gini coefficient from 0.48 for the subset $\mathcal{X}_1$ to a value of 0.19 for $\mathcal{X}_1^+$ and to 0.11 for $\mathcal{X}_1^-$.

In the rest of the paper, we denote the different subsets we have for product group j as $\mathcal{X}_j^\pm$. Thus, $\mathcal{X}_j^\pm$ includes either the subset $\mathcal{X}_j$ in case no class partitioning has been performed or $\mathcal{X}_j^+$ and $\mathcal{X}_j^-$ in the other case.

4.1.3 Pre-processing of sample subsets

All subsets $\mathcal{X}_j^\pm$ must satisfy technical criteria, so that we can apply learning algorithms to them. We distinguish two tasks here, which are depicted in Fig. 3c.

Feature normalization and encoding: We remove *sparse*, *zero-* and *near-zero-variance* features, *normalize* continuous values, and perform *one-hot encoding* for categorical features.

Binarization for single majority classes: Standard classification algorithms require at least two classes to train a classifier. However, a few majority subsets, e.g., $\mathcal{X}_1^+$ shown in Fig. 4, contain only one class. We treat such cases using the OvA binarization technique [10,12]. We first add the minority subset $\mathcal{X}_j^-$ to its majority subset $\mathcal{X}_j^+$ again. Then, we re-label all samples of minority classes c_i^- to one combined "negative" class, and the single majority class c_i^+ to a "positive" class. We then use standard algorithms to train a binary classifier for this combined, re-labeled sample subset.

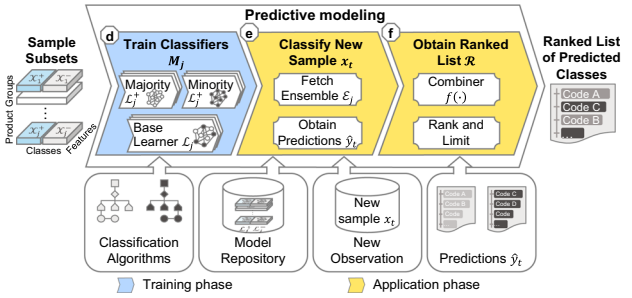


Fig. 5 Major steps of predictive modeling to train a classifier for each sample subset and to combine the results of the classifiers into a ranked list of predictions

4.2 Predictive modeling

Figure 5 shows the major steps of the predictive modeling that are applied on the sample subsets $\mathcal{X}_j^\pm$ resulting from the previous training set preparation. We first discuss how to train classifiers $\mathcal{M}_j$ for individual subsets $\mathcal{X}_j^\pm$ (Sect. 4.2.1). We then describe how to classify new observations, i.e., new samples x_t (Sect. 4.2.2), and show how to obtain a final recommendation list $\mathcal{R}$ (Sect. 4.2.3).

4.2.1 Create ensembles $\mathcal{E}_j$

We train an individual classifier $\mathcal{M}_j$ for each subset $\mathcal{X}_j^\pm$. Thereby, we are widely free in the choice of multi-class classification algorithms. One restriction is that an algorithm must be able to train *probabilistic* classifiers $\mathcal{M}_j$. Probabilistic means that $\mathcal{M}_j$ predicts a list of classes that may be ranked according to associated *confidence values* how sure the classifier is with its predictions. We recommend ensemble procedures, as the sample subsets $\mathcal{X}_j^\pm$ often contain many class labels, but a rather small number of samples. For instance, Random Forest is useful for such data characteristics because its integrated sampling method reduces the risk of overfitting [19].

When training a classifier $\mathcal{M}_j$ for a subset $\mathcal{X}_j^\pm$, we have to distinguish between two cases: (1) a subset that has been partitioned into majority and minority subsets $\mathcal{X}_j^+$ and $\mathcal{X}_j^-$, as well as (2) a subset $\mathcal{X}_j$ that has not been partitioned. In the first case, we train separate classifiers, i.e., base learners, for each of the two subsets $\mathcal{X}_j^+$ and $\mathcal{X}_j^-$. We denote $\mathcal{L}_j^+$ as a *majority base learner*, which is trained on a majority subset $\mathcal{X}_j^+$. Accordingly, we denote $\mathcal{L}_j^-$ as a *minority base learner* being trained on $\mathcal{X}_j^-$. By training separate classifiers for majority and minority classes, we ensure that learning algorithms do not ignore underrepresented minority classes.

For group $GE54$ shown in Fig. 4, we train a majority base learner $\mathcal{L}_1^+$ on subset $\mathcal{X}_1^+$, i.e., $\mathcal{L}_1^+$ is tailored to predict error code A. Furthermore, we use $\mathcal{X}_1^-$ to train a minority base learner $\mathcal{L}_1^-$, which is able to predict error codes B to D. The

resulting classifier system has to be able to predict all classes from A to D. Hence, we combine each pair of base learners $\mathcal{L}_j^+/\mathcal{L}_j^-$ to an ensemble $\mathcal{E}_j$. We store this ensemble $\mathcal{E}_j$ in a *model repository* [51] and tag it with the number j of its product group. Furthermore, we tag $\mathcal{E}_j$ with the hierarchy level l of the sample subsets $\mathcal{X}_{(l,j)}^{\pm}$. This way, we indicate whether $\mathcal{E}_j$ is a primary ensemble ($l = max_l$) or a surrogate ensemble ($l = max_l - u$, with $u \geq 1$).

In the second case, i.e., for sample sets $\mathcal{X}_j$ that have not been partitioned into majority or minority subsets, we train a base learner $\mathcal{L}_j$ on the whole set $\mathcal{X}_j$. We also store this $\mathcal{L}_j$ as an ensemble $\mathcal{E}_j$ in our model repository.

4.2.2 Classify new samples x_t

Given a new observation x_t, we first determine the group j at the lowest hierarchy level max_l to which this observation belongs. If SPH has built a primary subset for group j, we fetch the corresponding primary ensemble that is tagged with level $l = max_l$ in the model repository. Otherwise, we search for the surrogate ensemble of group j at higher levels of the hierarchy. We pass the sample x_t to the base learner(s) of the fetched ensemble $\mathcal{E}_j$ to obtain a prediction in the form of a list $\mathcal{Y}_j$. This list contains the most likely classes and ranks them according to their confidence values. In case $\mathcal{E}_j$ is composed of a majority and a minority base learner $\mathcal{L}_j^+$ and $\mathcal{L}_j^-$, we get two lists $\mathcal{Y}_j^+$ and $\mathcal{Y}_j^-$. In case $\mathcal{E}_j$ has exactly one base learner $\mathcal{L}_j$, we obtain one list $\mathcal{Y}_j$.

Figure 6 shows the application phase of our classification approach, i.e., the steps (e) and (f) in Fig. 5 for an example of a new failed EoL test x_t. The underlying product is a $DE3612$ model at level $max_l = 3$ of our hierarchy. Since no primary ensemble is available for $DE3612$ in the model repository at level 3, we fetch the surrogate ensemble for $DE34$ at level 2. We pass the new sample x_t to both base learners $\mathcal{L}_1^+$ and $\mathcal{L}_1^-$. For $\mathcal{L}_1^+$, we obtain the list $\mathcal{Y}_1^+$ with one majority error code A and a confidence value of 54.0%. The list $\mathcal{Y}_1^-$ is the prediction of $\mathcal{L}_1^-$ with minority codes C, B, and D, whose confidence values are between 13.8 and 55.0%.

4.2.3 Obtain ranked list $\mathcal{R}_e$

In case of an ensemble that separately treats majority and minority classes, we need to combine both lists $\mathcal{Y}_j^+$ and $\mathcal{Y}_j^-$ into one list $\mathcal{R}$. Several reviews discuss approaches to combine votes from different base learners [12,34,38,54]. All related approaches assume that the base learners predict completely or partly the same set of classes. In our approach, the two involved base learners however predict disjoint sets of majority classes c_i^+ and minority classes c_i^-. Thus, the approaches from literature are not applicable to our ensembles. For this reason, we opt for a first approach that is easy

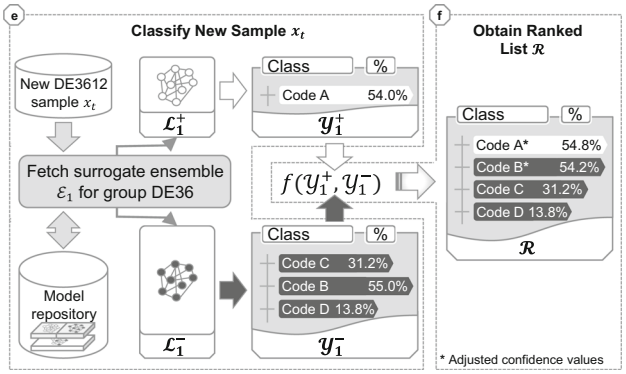


Fig. 6 Computation of a recommendation list $\mathcal{R}$ for a new sample x_t of group $DE3612$. Base learners $\mathcal{L}_1^+$ and $\mathcal{L}_1^-$ issue lists $\mathcal{Y}_1^+$ and $\mathcal{Y}_1^-$ that are merged together and ranked according to the confidence values to get $\mathcal{R}$

to implement. In most cases, the final recommendation list $\mathcal{R}$ is a union of $\mathcal{Y}_j^+$ and $\mathcal{Y}_j^-$ with unchanged confidence values. Nevertheless, we consider some special cases where we scale confidence values for certain classes. In some cases, a minority class c_i^- in $\mathcal{Y}_j^-$ has only a marginally higher confidence value than a majority class c_i^+ in $\mathcal{Y}_j^+$. Figure 6 shows such an example for the error codes A and B. However, majority classes occur much more often in the original sample set $\mathcal{X}$. Thus, we place the majority class above the minority class in the final list $\mathcal{R}$. So, we upscale the confidence value of the majority class and downscale the value of the minority class.

We consider only those cases where the difference in the confidence values of relevant classes is less than 1.5%-points. We use this low threshold, because we want to adjust the original confidence values as little as possible. This ensures that we do not distort the essential probabilistic statements of the base learners. In Fig. 6, this only applies to error codes A and B. For A, we increase the confidence value of 54.0% by the threshold value of 1.5%, i.e., by a factor of 1.015. So, we get a scaled confidence value of about 54.8%. For B, we reduce the confidence value by the same factor of 1.015, and get an adjusted value of about 54.2%. As a result, the adjusted confidence value of error code A is greater than the one of B. Finally, we rank the classes in descending order regarding the scaled confidence values to generate the final recommendation list $\mathcal{R}$.

5 Evaluation with real data of EoL testing

We have carried out an extensive evaluation of our classification approach based on its application to the real-world data of the EoL testing use case. In the following, we discuss its potential to mitigate the negative effects of analytical challenges C1 and C2 (Sect. 5.1). Afterwards, we report the results of our experimental evaluation (Sect. 5.2). Then, we

Table 1 Effect of SPH and CPI on analytical challenges: ++ positive; + partly positive; 0 not significant

	Meets analytical challenge		
	C1	C2.1	C2.2
(a) SPH	+	++	++
(b) CPI	+	0	0

illustrate the real business impact of our approach and how it improves EoL testing (Sect. 5.3).

5.1 Effects on analytical challenges

Table 1 summarizes how the two essential steps of our classification approach affect the analytical challenges. It depicts these effects separately for (a) SPH and (b) the subsequent CPI. To underpin these discussions, Table 2 reports statistical metrics that exemplify the effects on the challenges. We have collected these metrics by applying SPH and CPI to the data of our use case.

SPH splits the sample set $\mathcal{X}$ into 26 subsets $\mathcal{X}_j$. These are 21 primary subsets on level 3 of the product hierarchy and five surrogate subsets one level higher. Each subset $\mathcal{X}_j$ contains on average 54 samples and eleven classes. This reduces the mean number of samples per class from about 12 samples in $\mathcal{X}$ to now about 5 in each subset $\mathcal{X}_j$. This reduction of the number of samples per class in each subset may increase the risk of overfitting [17,28]. As discussed in Sect. 4.2.1, this may be addressed by applying ensemble techniques that are able to deal with smaller data sets, e.g., Random Forest [6,19]. However, only little is known how to tackle especially the combined effects of challenges C1 and C2. Therefore, our approach admits smaller sizes of sample subsets to mitigate the effects of C1 and C2.

SPH reduces the Gini coefficient from 55% in the sample set $\mathcal{X}$ to an average of 28% among all sample subsets $\mathcal{X}_j$. This already constitutes a significant reduction of class imbalance. A detailed analysis reveals that SPH primarily reduces the class imbalance of the 21 primary subsets to this significant degree. The five surrogate subsets have Gini coefficients between 30 and 50%. This still represents a remarkable class imbalance for these five surrogate subsets. Hence, we rate the influence of SPH on challenge C1 as partly positive.

The main advantage of SPH is apparent in its effect on challenge C2. Firstly, we reduce the number of features f_k from 115 in the sample set $\mathcal{X}$ to an average of about 82 in the subsets $\mathcal{X}_j$. Thereby, we remove those features from a subset $\mathcal{X}_j$ that are not measured for the product variants of group j. For example, reconsider the four-cylinder variants in group $GE54$. SPH removes the features for the two cylinders No. 5 and 6 that are not part of these four-cylinder variants. This significantly reduces the number of missing feature values. In our use case, about 17% of the values in the original sample set $\mathcal{X}$ were missing values. SPH reduces this to an average of about 5% in the subsets $\mathcal{X}_j$. Thus, we rate the effect of SPH on challenge C2.1 as positive.

Furthermore, we reduce the variety in value ranges of features within a subset $\mathcal{X}_j$ (C2.2). This reduces the number of concepts each classifier $\mathcal{M}_j$ has to learn. For example, reconsider the concepts shown for groups $GE54$ and $GE56$ in Fig. 2. Originally, a classifier being applicable to both product groups had to learn all four concepts A, B, A', and B'. Particularly sub-concept B' was only represented by three minority samples of all 60 samples. Learning algorithms would usually neglect these three minority samples and thus the whole sub-concept B'. After SPH, we train two separate classifiers on separate sample subsets for groups $GE54$ and $GE56$. The subset for $GE54$ only contains 13 samples to characterize sub-concepts A' and B'. So, the relative share of the three samples for B' increases in this subset. Thus, it is more likely that learning algorithms do not neglect the three minority samples and are thus able to learn a pattern for sub-concept B'. So, we rate the effect of SPH on challenge C2.2 as positive.

12 of the 26 sample subsets $\mathcal{X}_j$ resulting after SPH have a Gini coefficient higher than 30%. The subsequent CPI hence partitions these 12 subsets into majority subsets $\mathcal{X}_j^+$ and minority subsets $\mathcal{X}_j^-$. Thereby, CPI further reduces the average Gini coefficient for all resulting subsets $\mathcal{X}_j^\pm$ to about 21%. This additional reduction demonstrates a positive effect on challenge C1 of a class imbalance. Nevertheless, the reduction from 28% after SPH to now 21% is rather moderate, since CPI only partitions 12 of the 26 subsets $\mathcal{X}_j$. So, we

Table 2 Numbers of samples x_t, classes c_i, features f_k, portion of missing values ("NA"), and the Gini coefficient for the original sample set $\mathcal{X}$, as well as for the subsets $\mathcal{X}_j$ and $\mathcal{X}_j^\pm$ after SPH and after subsequent CPI

	Number of sample subsets	Average number across all subsets				
		Samples x_t	Classes c_i	Features f_k	"NA" in f_k (%)	Gini coeff (%)
Sample set $\mathcal{X}$		1050	84	115	17	55
After SPH:	26 $\mathcal{X}_j$, thereof 5 surrogates	54	11	82	5	28
After CPI:	14 $\mathcal{X}_j \wedge$ 12 $\mathcal{X}_j^\pm$	37	7	82	5	21

rate this effect on C1 as partly positive. Note that CPI does not affect the number and nature of features f_k. Hence, there is no effect on challenges C2.1 and C2.2.

5.2 Experimental evaluation

Now, we report the evaluation results. We describe the experimental set-up (Sect. 5.2.1), discuss how SPH and CPI increase classification accuracy (Sect. 5.2.2) and reduce the number of rework attempts in EoL testing (Sect. 5.2.3).

5.2.1 Experimental set-up

For details about the hardware and software set-up, we refer to a previous study [19]. Now, we focus on a description of methodological aspects of the evaluation.

Training and test set: We split the sample set $\mathcal{X}$ into a training set with 750 samples and a test set with 300 samples. We made sure that both sets contain all 84 classes and resemble the class distribution of all 1050 samples. We apply the training set preparation (Fig. 3) on the 750 samples of the training set to get the sample subsets $\mathcal{X}_j^{\pm}$. Afterwards, we apply the training phase (step d in Fig. 5) for each subset $\mathcal{X}_j^{\pm}$ to create the respective ensembles $\mathcal{E}_j$. Here, we used a fivefold cross validation on each subset $\mathcal{X}_j^{\pm}$ of the training data to find the best hyper-parameter settings for the learning algorithms [19]. We then carry out the application phase (steps e and f) for each of the 300 samples in the separate test data set to evaluate the ensembles.

Parameterization: We have carried out a grid search to find the parameterization of SPH and CPI yielding the best accuracy. SPH has one threshold parameter to limit the *loss of information* due the removal of classes with only one sample in a subset $\mathcal{X}_j$. Here, we examined values in $\{0.1, \ldots, 0.4\}$ with a step size of 0.05 to finally get the best result with 25%. For CPI, we tested seven threshold values for the Gini coefficient from 0.1 to 0.7. Moreover, we started with a value of 0.6 for p of the p-quantile and increased this parameter up to 0.9. We used a step size of 0.1 for both parameters. CPI's best parameterization for our use case data was 30% for the Gini coefficient and $p = 0.8$.

Evaluation: We report two *performance scores* that are measured by applying the classifiers $\mathcal{M}_j$ on the 300 samples of the test data set. The first score represents accuracies for several recommendation lists $\mathcal{R}_e$ of different lengths e. A correct classification means that the real class label y_t of a test sample is contained at any of the first e positions of the associated list $\mathcal{R}_e$. Accuracy at e ($A@e$) then measures the relative portion of such correct classifications among all test samples. In our use case of EoL testing, operators may work through the list $\mathcal{R}_e$, i.e., they try to repair the faulty components in the order as they are listed in $\mathcal{R}_e$. Hence, $A@e$ measures how likely it is that an operator can solve a quality issue by solely using the list $\mathcal{R}_e$, i.e., without consulting the quality engineer. Note that a large list $\mathcal{R}_e$ would usually overwhelm operators. Hence, we limit the list $\mathcal{R}_e$ to ten error codes, i.e., $e \in \{1, \ldots, 10\}$.

The second score represents the number of *rework attempts* (RA) that operators need on average to solve a quality issue by working through the list $\mathcal{R}_e$. To calculate this score, we individually consider the number of correct predictions for each position in the list $\mathcal{R}_e$. A hit at the first position means that the operator is able to solve the quality issue after one rework attempt. A hit at the second position means that s/he needs two attempts and so on. So, we respectively multiply the number of hits at a position with the ranking number. We then sum up the products and divide it by the number of all hits in $\mathcal{R}_e$ to get the score $RA@e$.

Baseline: We compare the results of our approach with the best baseline from our previous study [19]. This best baseline is applying Random Forest in combination with the feature selection technique Boruta [28] (RF+B). In addition, we evaluated a baseline from the area of sequential data analysis. Here, we opt for an ensemble of several neural networks [2]. For a new observation, the ensemble averages the prediction scores of individual neural networks to obtain the final class prediction. We refer to this baseline as *averaged Neural Network* (avNN). Figure 7 compares the results of our approach with those of the baselines RF+B and avNN.

5.2.2 Increased accuracy

In this subsection, we discuss the score $A@e$ on the left y-axis of Fig. 7. A comparison of the two baselines shows that RF+B has a higher $A@e$ than avNN for all lengths of the list $\mathcal{R}_e$. The major reason is that neural networks are usually tailored to deal with high-dimensional data, e.g., high resolution geometric data or time series data. However, our sample set $\mathcal{X}$ contains only one aggregated value for each feature, i.e., the features are aggregated over time. In addition, deep learning algorithms require plenty of data samples to train accurate neural networks. With the small number of samples in our data set, these approaches tend to overfit strongly. Here, we see that ensemble techniques such as Random Forrest may better handle these data characteristics and finally yield higher accuracies [19]. Hence, we compare our approach only with RF+B.

Our approach with SPH and CPI dominates the baseline RF+B for all $A@e$ scores. This means that the list $\mathcal{R}_e$ of our approach contains the correct faulty component more frequently compared to the list of RF+B. Thus, operators are able to solve a quality issue more often without a quality engineer by solely working through the list $\mathcal{R}_e$. The performance gains of our approach vary for individual lengths of $\mathcal{R}_e$. The lowest absolute gain is 1%-point for $\mathcal{R}_6$. Our

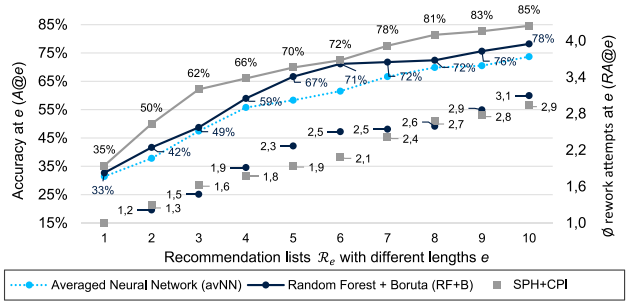


Fig. 7 Evaluation results: $A@1$ to $A@10$ and $RA@1$ to $RA@10$ for lists $\mathcal{R}_e$ with different lengths e

approach especially outperforms the baseline for shorter lists, e.g., the highest performance gain is about 13%-points for $\mathcal{R}_3$. The average gain in accuracy among all lists is about 6.3%-points.

We have also evaluated both steps SPH and CPI in isolation. This means we divided our sample set $\mathcal{X}$ once only by SPH and once only by CPI and then trained separate classifiers for each step. We found that SPH has a higher contribution to increasing $A@e$. The ten $A@e$ scores for applying only SPH exceed RF+B by an average of 2.9%-points. However, the scores for CPI are on average 2.7%-points below that baseline. Nevertheless, applying CPI after SPH even adds 3.4%-points to the 2.9%-points accuracy gain of SPH. This fact that CPI in isolation reduces accuracy, but increases it even further when applied after SPH supports our key statement: An approach focusing on either challenge C1 or C2 in isolation is not sufficient. Instead, it is much more beneficial to address both challenges at once, i.e., by applying SPH and CPI in a combined way.

5.2.3 Reduced number of rework attempts

Now, the goal is to reduce the scores $RA@e$, i.e., the average number of rework attempts operators need to solve a quality issue. As shown on the right y-axis in Fig. 7, our approach SPH+CPI leads to a high reduction of the $RA@e$ scores for the lists $\mathcal{R}_5$ and $\mathcal{R}_6$. Here, operators need on average 0.4 less rework attempts compared to the baseline RF+B. The reason is the significant higher $A@e$ scores for lower lengths e, e.g., the performance gain of 13%-points for $A@3$. A high score $A@e$ on the first positions entails that the correct class is contained more often at these first positions. Hence, it is more likely that operators solve a quality issue with a smaller number of rework attempts.

For the lists with two, three, and eight elements, the $RA@2$, $RA@3$, $RA@8$ scores of our approach are however about 0.1 higher than those of the baseline RF+B. Note again that our approach outperforms the baseline RF+B with a gain in $A@e$ between 9%-points and 13%-points with these three lists $\mathcal{R}_2$, $\mathcal{R}_3$, and $\mathcal{R}_8$. This higher $A@e$ scores mean that operators are much more likely to solve a quality issue without consulting a quality engineer. Quality engineers usually get a much higher salary than operators. Hence, the 0.1 additional rework attempts of the operator are a rather negligible price to pay, compared to the higher cost savings we achieve by consulting the quality engineer more seldom.

Note that the $RA@e$ score measures the number of rework attempts only for those cases, where the corresponding recommendation list $\mathcal{R}_e$ contains the correct faulty component. For all remaining cases, we assume that the operator consults a quality engineer to solve the quality issue. However, we cannot make a valid statement about how many additional attempts the quality engineer may then need to identify the correct faulty component. Nevertheless, we expect it to be less than four attempts. This is because the quality engineer needs on average four attempts without any data-driven approach [19] and because s/he may already exclude the false components that have been part of the list $\mathcal{R}_e$. Furthermore note that we compare our approach SPH+CPI with another data-driven baseline RF+B. In fact, the quality engineer needs roughly the same number of additional rework attempts if the list generated by SPH+CPI or the list of RF+B do not contain the correct component. Hence, it is valid to compare these two data-driven approaches SPH+CPI and RF+B with the $RA@e$ score.

5.3 Business impact to EoL testing

The results reported for $A@e$ in Fig. 7 with up to 85% are still less than typical results presented by the research community [43,49,50]. This is mainly because the data sets employed in scientific literature often do not show all characteristics of real-world data. In fact, real-world data characteristics make it hard to achieve similar levels of accuracy. We already show this in our previous study, where we tested a diverse set of methods [19]. The best combination of these methods with an accuracy of up to 78% is Random Forest with Boruta, i.e., the baseline RF+B in this paper.

We also show in the previous study that even this baseline with its accuracy of up to 78% has a positive impact on the real business of EoL testing [19]. In particular, it reduces the overall costs for reworking on defective engines. More precisely, the personal costs for a quality engineer are usually 60% higher than for an operator. These quality engineers are now only involved in cases when the correct faulty component is not part of the list $\mathcal{R}_e$, i.e., in $1 - A@e$ of all cases.

Compared to the baseline RF+B, our approach with SPH and CPI even further reduces costs for reworking on defective engines. This is mainly because it yields higher $A@e$ scores for any list $\mathcal{R}_e$. The average gain in accuracy of 6.3%-points means that operators can solve a quality issue without expensive quality engineers in 6.3%-points more cases. Furthermore, most of the lists $\mathcal{R}_e$ have lower $RA@e$ scores, so

Table 3 Summary of the discussion regarding the generality of challenges C1 and C2. The table reports (1) the most important use cases we have investigated to confirm the generality of the challenges, (2) the main formats of data sources used in these use cases, and (3+4) the major domain-specific causes for challenges C1 and C2

	Industrial value chain	Medical diagnoses
(1) Example use cases	Fault diagnosis, product quality prediction, predictive maintenance, root cause analysis, classification of products into product types	Diagnosis of rare, but dangerous or lethal diseases, e.g., different types of skin cancer
(2) Main data sources	Sensor data, text data, semi-structured JSON documents	Image data, electronic medical records, genomics databases, text data, sensor data
(3) Major cause for C1	Many error types of products or many product types that are underrepresented in data	Rare, but lethal diseases, e.g., melanoma cancers, are underrepresented in data
(4) Major cause for C2	Increasing product variety and diversity of class patterns for different product variants	Ethnic and gender bias: Symptoms and patterns for patients with darker skin color or for women are underrepresented in data

that our approach reduces the number of rework attempts. Altogether, the annual cost savings for a company may accumulate to a magnitude of up to a few millions of EURO.

6 Discussion of generality

The previous section discusses evaluation results for the particular use cases of EoL testing and its domain-specific data. In this section, we show that our the approach is generally applicable to further use cases and data. We start by first showing that the considered analytical challenges C1 and C2 are also relevant in various real-world application domains other than EoL testing (Sect. 6.1). Afterwards, we discuss the potential of applying the major steps SPH and CPI of our approach in these different application domains (Sects. 6.2 and 6.3).

6.1 Generality of challenges C1 and C2

To confirm the generality of challenges C1 and C2, we have carried out a literature review for real-world use cases of multi-class problems. Table 3 summarizes the major results. The most important finding is that challenges C1 and C2 arise in many real-world problems of various application domains. This concerns several stages of an industrial value chain, e.g., manufacturing, marketing, or sales (Sect. 6.1.1). Moreover, both challenges arise in entirely different application domains, such as medical diagnoses (Sect. 6.1.2). These real-world problems involve various kinds of data, e.g., sensor data, text data, semi-structured documents, and image data.

6.1.1 Challenges in the industrial value chain

The challenges of multi-class imbalance (C1) and heterogeneous feature space (C2) are common in the manufacturing domain. This statement is confirmed by several review articles that examine literature describing various applications of machine learning to real-world manufacturing use cases [8,27,53,55]. We have additionally examined several real-world use cases for multi-class problems that are not covered by the mentioned reviews. Gerling et al. [14] discuss how to use machine learning to predict product quality based on sensor data of individual steps in an assembly line. Thalmann et al. [45] investigate three related use cases for fault detection, fault diagnosis, and predictive maintenance. Kassner et al. [25] use text analytics to identify root causes of product quality problems related to customer warranty claims. Kiefer et al.[26] extract information from text data to suggest causes and corrective actions of machine failures in a production line.

All these authors support our argumentation regarding analytical challenges that arise from the data characteristics in manufacturing. They all mention that products and production processes may be affected by a large number of diverse error types. These error types often correspond to the classes in classification tasks. So, this usually leads to a multiplicity of imbalanced class labels (C1). Furthermore, the authors coincide that machine learning suffers from the fact that underlying data often represent diverse product variants. These product variants have different technical specifications, leading to a heterogeneous feature space and a multiplicity of overlapping class patterns (C2).

These challenges are however not only relevant in manufacturing processes. In fact, they occur in use cases across all stages of a typical industrial value chain. This is particularly the case when products with a certain technical complexity and high product variety are involved, which is common in today's industries [22,32].

Sun et al. [41] discuss such a challenging use case in the marketing and sales stage of an industrial value chain. Their goal is to classify tens of millions of products based on their textual descriptions and other product attributes stored

in JSON documents. The classes correspond to different product types, e.g., laptop computers, laptop bags, or dining chairs. They consider a multi-class problem, where each product has to be classified into exactly one of more than 5000 product types.

This use case shares our challenges in other manifestations. The authors report that existing solutions to machine learning suffer from a high multi-class imbalance in textual product data (C1) [41]. Some product types are significantly underrepresented, which may cause learning algorithms to ignore them. In the product segment "Home & Garden" for instance, thousands of products belong to the majority types "area rugs" or "stools". However, many minority product types exist that occur in very few data samples, e.g., "oil lamps".

Moreover, each product type may be divided into several sub-types. This increases the diversity of the product portfolio and the heterogeneity in the feature space (C2). Sun et al. report that data samples of individual sub-types do not contain all features (C2.1) [41]. In addition, the concepts and patterns for a certain class may differ among these product sub-types (C2.2).

6.1.2 Challenges for data-driven medical diagnoses

Another domain where both challenges C1 and C2 occur together comprises data-driven medical diagnoses. Chan et al. [7] review 70 articles discussing various use cases for machine learning in dermatology. Most of them apply learning algorithms to image data of patients to detect and diagnose different types of skin cancer. A few make use of electronic medical records, genomics databases, textual data from insurance claims, or personalized sensor data of mobile devices. Examples of skin cancer types are various melanoma cancers or different kinds of non-melanoma cancers, e.g., basal cell carcinoma or actinic keratoses. So, this reflects multi-class problems, where the classes correspond to different cancer types. According to Chan et al., most of the articles report on complex data characteristics that make machine learning a hard problem [7].

In general, less than 10% of patients have a variant of melanoma skin cancer. So, these types of cancer are usually underrepresented in available image data compared to the different non-melanoma cancers. This leads to a high degree of multi-class imbalance (C1). Here, learning algorithms and resulting classifiers are often biased towards the frequent kinds of non-melanoma skin cancers. So, they usually provide a low accuracy for the rare melanoma cancers [7]. However, these melanoma cancers are malignant and may form metastases. They are thus more lethal than other types of cancer and therefore have to be treated more fairly by classifiers.

In addition, Chan et al. found that classification performance is highly affected by the patients' ethnicity, in particular by the skin color [7]. This corresponds to the challenge of a heterogeneous feature space (C2). More precisely, the features and class patterns for certain cancer types depend strongly on the color of the cancerous skin areas in the image data. However, the color and contrast of a particular skin cancer may vary significantly between different colors of the surrounding healthy skin [7]. Even for a single cancer type, learning algorithms need to differentiate various class patterns and sub-concepts for different skin colors.

Moreover, almost all 70 articles reviewed by Chan et al. focus on image data of light-skinned Caucasian patients from Europe or Northern America [7]. In fact, very few image data are available for patients with darker skin from Africa or the Americas. So, the class patterns and sub-concepts are extremely underrepresented in image data especially for people with darker skin. This means that a data-driven classifier is often not able to correctly predict the type of skin cancer for patients of this ethnicity, i.e., it leads to an ethnic bias.

Note that such multi-class problems with complex data characteristics are not only relevant for skin cancer diagnosis. Multi-class imbalance (C1) is a common challenge for medical diagnoses, as several dangerous or even lethal diseases exist that affect only few patients. For instance, diseases such as cystic fibrosis only have an incidence of about one in 15 000 humans [23]. So, very few diagnostic data is available for patients suffering from such rare diseases. In a data-driven classification, rare diseases represent seldom classes that are underrepresented in available training data related to more common, but comparatively harmless diseases.

A further cause of a heterogeneous feature space (C2) may be gender bias in data [44]. An issue discussed in medical research is that symptoms for certain lethal diseases differ significantly between men and women. According to Baggio et al. [5], this holds for various problems related to cardiovascular diseases, oncology, liver diseases, and osteoporosis. For instance, common symptoms of a heart attack for men are severe chest pain, while women are more likely to experience fatigue, dizziness, or stomach pain [52]. In a data-driven approach, algorithms may use patients' diagnostic data to learn patterns for typical symptoms of heart attacks. Here, the patterns to detect a heart attack hence differ significantly between women and men. As diagnostic data for women are less available than for men [52], algorithms usually mainly learn the patterns for men, but consider those for women less important.

6.2 Generality of SPH

In this section, we discuss constraints the domain knowledge must fulfill so that SPH is applicable to the data of other use cases (Sect. 6.2.1). Furthermore, we discuss whether

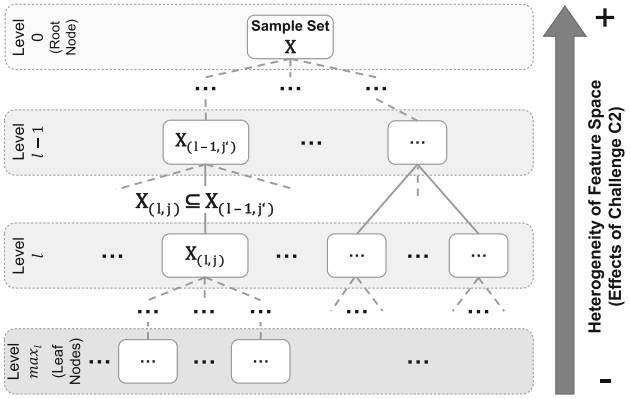


Fig. 8 Abstract hierarchical domain model illustrating the constraints SPH requires for such domain models

these constraints are met for domain knowledge that is available in different use cases of the industrial value chain and of data-driven medical diagnoses (Sect. 6.2.2). The major contribution of this discussion is that researchers and practitioners of other domains see under which circumstances they can apply SPH to their data.

6.2.1 Constraints for domain knowledge

SPH requires domain knowledge to be represented as a hierarchical model that is organized as a tree structure. Figure 8 illustrates the major constraints of SPH based on an abstract hierarchy:

1. The single root node at the top level $l = 0$ encompasses the entire sample set $\mathcal{X}$.
2. Each child node $(l,\ j)$ on a level $l \geq 1$ below the root node contains a subset of the data of its parent node $(l-1,\ j')$ one level above, i.e., $\mathcal{X}_{(l,j)} \subseteq \mathcal{X}_{(l-1,j')}$.
3. All leaf nodes are located at the same bottom level $l = max_l$.
4. For SPH to increase classification accuracy, the hierarchical domain model has to allow for segmenting a sample set $\mathcal{X}$ into subsets that show a more homogeneous feature space. More precisely, the effects of challenge C2 have to be less severe in the subset $\mathcal{X}_{(l,j)}$ of a child node than in the subset $\mathcal{X}_{(l-1,j')}$ of its parent node. This also means that these challenges are least severe in all leaf nodes at level max_l, for which SPH generates primary sample subsets.

The hierarchy of the EoL test case shown in Fig. 1 meets all four constraints. For other use cases, the first three constraints may be easily verified by checking the hierarchical structure of the domain model and the subset relationships between child and parent nodes. Verifying the fourth constraint requires ways to quantify the effects of the individual challenges C2.1 and C2.2 in the sample subsets $\mathcal{X}_{(l,j)}$ across different levels l. The effect of challenge C2.1 may be quantified by calculating the shares of missing feature values in subsets $\mathcal{X}_{(l,j)}$. So, we may check whether this share decreases along the path from the root node to the leaf nodes. However, literature does not comprise any metrics to quantify the effects of challenge C2.2. At first glance, C2.2 might be characterized by the number and diversity of concepts that exist in a subset $\mathcal{X}_{(l,j)}$. However, we usually do not know a priori any concept that exists in the sample subsets. This calls for further research to identify indicators that may at least estimate the effects of challenge C2.2 in real-world data sets.

6.2.2 Domain knowledge in different domains

Usual ways to model domain-specific concepts and their relationships are knowledge graphs and semantic nets, e.g., taxonomies or ontologies [36,40]. Such models commonly organize the entities of a domain, amongst others, via hierarchical relationships of superordinate and subordinate concepts. These hierarchical relationships typically fulfill the constraints introduced in the previous section. In particular, subordinate concepts, i.e., child nodes in the hierarchy, are more specific than their associated superordinate concepts, i.e., the parent nodes [40]. This also means that domain-specific data of any subordinate child concept usually shows a more homogeneous feature space than the data of its superordinate parent concept. So, the effects of challenge C2 decreases along the hierarchy path from a root concept to the leaf concepts. Hence, knowledge graphs or semantic nets usually offer adequate domain knowledge to partition the data via SPH.

As mentioned in Sect. 3, any manufacturing company has a documentation of its product family [1]. This documentation is often structured as a kind of hierarchical semantic net, e.g., as a product taxonomy or thesaurus that is suitable for SPH. In particular, it explicitly describes and structures the involved variety and diversity of products. This way, it significantly helps to address these major causes of challenge C2 in use cases across the industrial value chain. So, SPH may be applied to many industrial multi-class problems.

For instance, Sun et al. [41] discuss a product taxonomy[1] that is suitable for SPH. This taxonomy organizes products and their data into three hierarchical levels: (1) product categories, (2) product sub-categories, and (3) product sub-types. As discussed in Sect. 6.1.1, product sub-types at the bottom level of the taxonomy show the least degree of product variety. So, SPH may partition a sample set $\mathcal{X}$ into several primary subsets according to these product sub-types. This way, it decreases the number and diversity of class patterns

[1] WalmartLabs: Taxonomy API: https://developer.walmartlabs.com/docs/read/Taxonomy_API

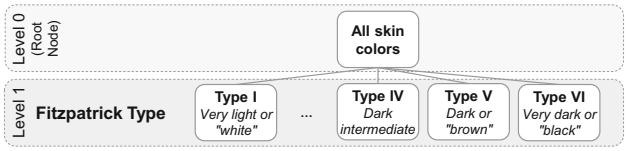


Fig. 9 Flat hierarchy categorizing skin colors based on the Fitzpatrick scale [11]

in the resulting subsets and thus reduces the negative effects of challenge C2. Only if a subset for a product sub-type contains too few samples, SPH moves up the product taxonomy to generate surrogate subsets at higher levels of product subcategories or product categories.

In the medical use case of diagnosing cancer types (see Sect. 6.1.2), the major cause for challenge C2 is that people with specific skin colors are underrepresented in data. Chan et al. interpret this ethnic bias as an *algorithm bias* [7]. Their statement is that machine learning algorithms have to be adapted to make them inclusive of ethnicity and skin color. However, we argue that the actual cause of this ethnic bias is the problematic characteristics of available training data. In these data, the class patterns of patients with darker skin color are superimposed by patterns of Cacausian patients. So, this ethnic bias rather corresponds to an *aggregation* or *representation bias* in data [31,44]. This likewise holds for the gender bias present in other problems of medical diagnoses [5,52]. To address these kinds of bias, we need to select and prepare the training data appropriately. Here, step SPH of our approach comes into play to partition training data and learn specific classifiers according to different skin colors or gender. This offers the potential to reduce bias in data and thus to increase fairness in machine learning.

Domain knowledge that SPH may use has to categorize different kinds of skin color. Jablonski discusses various categorization schemes for human skin color [24]. The most prominent one is the Fitzpatrick scale [11], which is still widely accepted in the dermatology domain [24]. It organizes skin colors in six different types based on the response of the skin to ultraviolet light.

Figure 9 shows a hierarchical taxonomy organizing types of skin color according to the Fitzpatrick scale. This taxonomy fulfills all constraints introduced in the previous section. Level 0 comprises one root node containing data of all skin colors. Level 1 partitions all data into one subset for each of the Fitzpatrick types I to VI [11]. Finally, the data in each subset at level 1 is more homogeneous regarding the feature space, i.e., the effects of challenge C2 are less severe than in the whole sample set $\mathcal{X}$. The reason is that class patterns for certain cancers are better distinguishable if we consider only training data for a specific type of skin color [7].

This example shows that SPH is also applicable if the domain knowledge is represented by a flat hierarchy with only one level below the root node. SPH then generates primary sample subsets for this bottom level 1. The difference to a non-flat hierarchy is that SPH uses the whole sample set $\mathcal{X}$ in case it decides to use a surrogate set for a specific color type. This may be addressed by adding an intermediate level to the taxonomy. Nodes at this new level may for instance build groups for light-skinned Caucasian patients or for patients with darker skin from Africa or the Americas.

6.3 Generality of CPI

The next step CPI of our approach requires metrics to identify and quantify distinct imbalances between classes in sample subsets $\mathcal{X}_j$. Here, we apply the Gini coefficient and the p-quantile. These two metrics are generic enough to be applicable to almost any kind of discrete or continuous class distribution. So, we may employ CPI to any multi-class classification problem regardless of the given use case or its domain.

Nevertheless, CPI requires a proper parameterization to increase classification accuracy. The first parameter of CPI is the *threshold of the Gini coefficient* to identify a distinct class imbalance in a subset $\mathcal{X}_j$. CPI's second parameter is *p of the p-quantile* to differentiate minority and majority classes. Both parameters of CPI have to be optimized together with the parameter of SPH. This parameter of SPH is the threshold to limit the *loss of information* due to the removal of classes with only one sample in a subset $\mathcal{X}_j$. For the EoL test case, we have tuned these three parameters and the resulting accuracy via a grid search. However, an exhaustive grid search is obviously too time-consuming when applying our approach to further use cases.

Finding the best parameter setting for both SPH and CPI is however a complex *multi-criteria* optimization problem. In fact, the parameters highly influence each other, so that it is hard to find parameter combinations that are close to the optimum. In many real-world use cases, it is even a *multi-objective* optimization problem. For instance, data-driven classification in EoL testing, has three major objectives. Besides increasing accuracy ($A@e$) and reducing the number of rework attempts ($RA@e$), the most important goal is to reduce monetary costs of EoL testing (Sect. 5.3). These objectives often compete with each other. For instance, a recent study shows that increasing accuracy does not necessarily reduce monetary costs [20]. In scenarios of medical diagnoses, sensitivity and specificity of data-driven classification must be weighted against the consequences of a particular diagnosis and accompanying medical treatment for the patient [7]. Altogether, these multi-criteria and multi-objective properties make optimization a very hard problem. In Sect. 7.4, we shed more light on this problem by analyzing various parameterizations for different kinds of data distributions.

7 Evaluation with synthetic data

Now, we discuss the results of evaluating our approach with synthetically generated data. We report how we generated these synthetic sample sets (Sect. 7.1). Next, we discuss the effects SPH and CPI have on statistics of these sample sets that reflect challenges C1 and C2 (Sect. 7.2). Subsequently, we discuss the major experimental results, e.g., in terms of accuracy (Sect. 7.3). We then detail on the effects of optimizing the parameters of SPH and CPI (Sect. 7.4). Thereupon, we focus on a more in-depth analysis, e.g., with respect to the improvements of SPH and CPI for individual groups in our hierarchical domain model (Sect. 7.5). This is followed by a discussion of the runtime efficiency and scalability of our approach with respect to bigger data sizes (Sect. 7.6). Finally, we summarize the most important findings of this evaluation (Sect. 7.7).

7.1 Synthetic data generation

As discussed in Sect. 6.1, challenges C1 and C2 are prevalent in the data of numerous real-world multi-class problems. However, we cannot use these data to evaluate our approach, especially since they are not publicly available. Data of use cases across the industrial value chain constitute intellectual property of the respective companies. These companies are therefore generally unwilling to share their data. Use cases related to medical diagnoses consider sensitive data of patients, which must not be disclosed due to privacy reasons.

We also examined the data sets of publicly available repositories, i.e., openML,[2] KEEL [3], Kaggle,[3] and the UCI ML Repository.[4] Remind that the data of the EoL test case is a multi-class data set with 1050 samples, 84 classes, 115 features, 17% missing feature values, and a multi-class imbalance with a Gini coefficient of 55%. However, none of the around 3500 data sets in the above-mentioned repositories comes even close to sharing these characteristics and thus both challenges C1 and C2. All repositories together only offer 40 data sets with more than 10 classes and with at least a moderate multi-class imbalance, i.e., a Gini coefficient greater than 30% (C1). Moreover, none of these data sets consider a heterogeneous feature space (C2) to an extent as found in the real-world use cases discussed in Sect. 6.1. This can be seen, for instance, in the share of missing feature values, which is less than 10% for all the above-mentioned 40 multi-class data sets.

As conclusion, the only way to get additional data sets for our evaluation is to generate synthetic data. We have made our data generator [46], the data we used for our evaluation, and the implementation of our approach publicly available on GitHub.[5] For the process of data generation, we first defined a hierarchy model that is similar to the one shown in Fig. 1 in terms of the number of levels and the number and distribution of groups on each level. Then, we used this hierarchy model to generate five synthetic sample sets that differ in specific data characteristics. Here, we followed a bottom-up procedure: We first manually defined the number of samples, features, and classes for each group at the bottom-level of the hierarchy. Thereby, we followed a similar distribution as for the EoL data. Afterwards, we used an existing data generator from sklearn[6] to generate the feature values with their corresponding class labels and with varying class distributions.

Table 4 Major data characteristics that differ among the five synthetically generated sample sets

Sample set	Avg. # of classes with single samples in a group	Gini coefficient (%)
Very bal	2.41	32
Balanced	2.79	42
Medium	4.55	51
Imbalanced	4.79	54
Very imbal	5.31	57

For the differences among the five synthetic sample sets, we focused on a minimal set of data characteristics that have the greatest impact on the extent to which SPH and CPI influence the final classification accuracy. Varying too many, possibly interdependent data characteristics, could lead to many unexpected side effects when applying SPH and CPI. Then, we would risk that the effects on, e.g., the accuracy could no longer be explained. Each of the five sample sets has 1050 samples, 84 classes, and 100 features with 24% missing feature values. The main difference among them is that we adapted the class distribution in each individual group in the hierarchy model to be either more balanced or more imbalanced. We hence denote the sample sets as *very balanced*, *balanced*, *medium*, *imbalanced*, and *very imbalanced*. Table 4 summarizes basic characteristics that vary among the sample sets. These specific differences in data characteristics allow us to examine the major influencing factors of SPH and CPI as follows:

– **SPH:** The more imbalanced sample sets contain more classes that have only one sample in individual groups at the bottom level of the hierarchy. Table 4 reports the average number of such classes across all groups. SPH

[2] openML: https://www.openml.org/

[3] Kaggle data sets: https://www.kaggle.com/datasets

[4] UCI ML Repository: https://archive.ics.uci.edu/ml

[5] Prototypical Implementation: https://github.com/IPVS-AS/SPHandCPI

[6] Sklearn data generator: https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_classification

first removes these classes with single samples from the primary subsets $\mathcal{X}_j$. So, SPH also removes more classes from these primary subsets the more imbalanced the overall sample set $\mathcal{X}$. Then, SPH also decides more often to choose surrogate subsets at higher levels of the hierarchy. This way, we can evaluate our approach with varying numbers of chosen surrogate subsets. As surrogate subsets are less specific, the expectation is that the higher their number, the lower the gain in accuracy of SPH.

– **CPI:** Another important factor for the resulting accuracy is the number of sample subsets $\mathcal{X}_j$ that CPI splits into minority and majority subsets $\mathcal{X}_j^{\pm}$. Thereby, the Gini coefficient indicates the probability how many sample subsets $\mathcal{X}_j$ are split by CPI. We hence examine sample sets, where the Gini coefficients differ between 32% and 57% (see Table 4). This value range of the Gini coefficient is common for those data sets in the above-mentioned repositories, e.g., openML, which have a comparable number of samples and a multi-class imbalance.

Note that, for our evaluation, we deliberately generated sample sets with a rather low size of only 1050 samples, but with comparably high numbers of features and classes. The major reason is that such data characteristics are common in many real-world industrial multi-class problems [25,26,41,55]. A small sample size may increase the risk of overfitting [17,28]. Nevertheless, as discussed in Sects. 4.2.1 and 5.1, we address this issue by applying the ensemble technique Random Forest [6] as classification algorithm in our evaluation. Ensemble techniques are able to reduce the risk of overfitting in case of smaller data sets [19]. In addition, Hirsch et al. show [19] that a small sample size usually amplifies the negative effects of a multi-class imbalance (C1) and a heterogeneous feature space (C2) on the classification accuracy. This motivates to use this small sample size in order to assess whether SPH and CPI are then still able to address challenges C1 and C2.

7.2 Effects of SPH and CPI on challenges C1 and C2

We use a similar experimental setup as described in Sect. 5.2.1, e.g., we use the same train and test split and performance scores. We carried out a grid search to find the best parameterization of SPH and CPI for each of the five sample sets individually. For each parameter, we defined a reasonable grid on the basis of the best parameter values for the EoL test case. For the maximum information loss of SPH, we examined values in $\{0.1, \ldots, 0.4\}$. Regarding CPI, we examined values in $\{0.2, \ldots, 0.4\}$ for the threshold of the Gini coefficient and values in $\{0.7, \ldots 0.9\}$ for the p-quantile. We used a step size of 0.05 for each parameter. In the following sections, we focus on the respectively best parameterization found by this grid search. We first discuss detailed statistics that highlight the effects of SPH and CPI on challenges C1 and C2 in the synthetic sample sets.

Table 5 shows these statistics for the five sample sets. In its second column, the table shows in (a) how many sample subsets SPH generates and how many of them correspond to surrogate subsets. Furthermore, this column indicates in (b) how many of SPH's subsets the next step CPI does not split ($\mathcal{X}_j$), and how many it splits into minority and majority subsets $\mathcal{X}_j^{\pm}$. The next columns show average numbers of samples x_t, classes c_i, and features f_k, the shares of missing feature values "NA", as well as Gini coefficients among all sample subsets after applying (a) SPH and (b) SPH+CPI.

The statistics prove the tendency discussed in the previous section that the more imbalanced the data, the more surrogate sets SPH generates. Note that it generates between 8 and 18 surrogate sets for our synthetic sample sets, while it is 5 for the EoL test case. The reason is that the bottom groups of the hierarchy of all synthetic sample sets contain more classes with only one sample than in case of the EoL data. So, we can evaluate whether this higher number of surrogate sets has a negative effect on the final classification accuracy.

Since SPH generates more surrogate sets, the mean numbers of samples x_t and classes c_i in each subset $\mathcal{X}_j$ are also higher for the synthetic data than for the EoL data (cf. Tables 5 and 2). The mean number of samples after SPH (cf. lines (a) in Table 5) is now between 78 and 138 per subset, while the number of classes is between 16 and 22. Nevertheless, the number of samples per class is about 5 to 6 in each subset $\mathcal{X}_j$, which in turn is similar to the EoL data. Moreover, SPH has a positive effect on challenge C1 of a multi-class imbalance. It reduces the Gini coefficients of all five sample sets. As shown in Table 4, the original Gini coefficients of these sample sets are between 32 and 57%. SPH reduces these to values between 17 and 42% (see Table 5).

Again, the major advantage of SPH is its positive effect on challenge C2 of a heterogeneous feature space. SPH for instance reduces the number of features for the five generated sample sets from 100 to a mean number of about 82 to 84 in the subsets $\mathcal{X}_j$. Thereby, it removes most of the features that contain missing values in the sample sets. This leads to a significant reduction of the share of missing values ("NA") from originally 24% to at most 9% for the resulting subsets $\mathcal{X}_j$.

We have also investigated to which extent primary sets and surrogate sets individually contribute to addressing the problem of a heterogeneous feature space. For instance, the 14 primary sets SPH generates for the medium sample set contain on average 80 features with a share of missing values of only 5.2%. This leads to a significantly positive effect on challenge C2. The remaining 12 surrogate sets contain on average 86 features with about 11.7% of missing values. So, they are more heterogeneous than the primary sets. Nevertheless, the 12 surrogates are still much more homogeneous than

Table 5 Detailed statistics of the five generated sample sets that illustrate the effects of SPH and CPI on challenges C1 and C2. The values are averages among all sample subsets after applying SPH and CPI. Here, $\mathcal{X}_j$ describe the resulting subsets after applying SPH, and $\mathcal{X}_j^{\pm}$ describe the subsets that are further partitioned with CPI

Sample set	Number of sample subsets after (a) SPH and (b) SPH+CPI	Average number across all subsets				
		Samples x_t	Classes c_i	Features f_k	"NA" in f_k (%)	Gini coeff (%)
Very balanced	(a) 26 $\mathcal{X}_j$, thereof 8 surrogates	78	16	82	7	17
	(b) 23 $\mathcal{X}_j \wedge 3\ \mathcal{X}_j^{\pm}$	68	14	82	7	14
Balanced	(a) 26 $\mathcal{X}_j$, thereof 9 surrogates	84	17	82	7	23
	(b) 22 $\mathcal{X}_j \wedge 4\ \mathcal{X}_j^{\pm}$	74	15	82	7	20
Medium	(a) 26 $\mathcal{X}_j$, thereof 12 surrogates	101	17	82	8	33
	(b) 19 $\mathcal{X}_j \wedge 7\ \mathcal{X}_j^{\pm}$	86	15	82	8	25
Imbalanced	(a) 26 $\mathcal{X}_j$, thereof 14 surrogates	113	18	83	8	37
	(b) 16 $\mathcal{X}_j \wedge 10\ \mathcal{X}_j^{\pm}$	85	14	83	8	25
Very imbalanced	(a) 26 $\mathcal{X}_j$, thereof 18 surrogates	138	22	84	9	42
	(b) 7 $\mathcal{X}_j \wedge 19\ \mathcal{X}_j^{\pm}$	85	13	84	9	22

the whole sample set and thus also contribute to addressing challenge C2. In fact, the 11,7% of missing values constitutes a reduction by half of the 24% of the original medium sample set. While the 14 primary sets contain on average 27 samples, the surrogates comprise about 228 and thus many more samples. Altogether, this confirms our statement in Sect. 4.1.1 that surrogates represent an appropriate trade-off between the entire sample set with 1050 samples but a very heterogeneous feature space and individual primary sets with a largely homogeneous feature space but fewer samples.

CPI splits more sample subsets $\mathcal{X}_j$ into minority and majority subsets $\mathcal{X}_j^{\pm}$ the more imbalanced the class distributions of the five sample sets are. As the number of splits for the balanced and very balanced sample sets is only 3 and 4, CPI only moderately reduces the Gini coefficients for each of them by 3%-points. For the other sample sets, CPI splits much more of SPH's $\mathcal{X}_j$, which leads to a likewise higher reduction of the Gini coefficients. Especially for the very imbalanced sample set, CPI splits 19 subsets and reduces the Gini coefficient to the highest degree, i.e., from 42% after SPH to 22% after CPI. This confirms our main intuition that the positive effect of CPI is stronger the more imbalanced the class distribution of the original sample set.

7.3 Classifier accuracy for varying class distributions

Now, we discuss the results of evaluating our approach with the five synthetic sample sets w. r. t. the classification accuracy. Here, we focus on the baseline RF+B, as it clearly outperforms avNN. This is due to the same reasons as discussed in Sect. 5.2.2: Our sample sets do not contain high-resolution geometric data or time series data and their size is too small, so that neural networks tend to overfit very strongly. In the following, we discuss the $A@e$ scores with $e \in \{1, \ldots, 10\}$ for the baseline RF+B and for applying both SPH and CPI. We also present the $Avg\ A@e$ score, i.e., the average of the single $A@e$ scores among all $e \in \{1, \ldots, 10\}$.

Medium sample set: We start with the discussion for the medium sample set (cf. Fig. 10) because the $A@e$ scores of our approach SPH+CPI and of the baseline RF+B show a similar behavior for this sample set as those reported in Fig. 7 for the EoL test case. In particular, SPH+CPI again dominates RF+B, i.e., our approach achieves better results for all lengths e of the recommendation list $\mathcal{R}_e$. On average among all $e \in \{1, \ldots, 10\}$, SPH+CPI achieves 73.1% accuracy, while RF+B only yields 65,4%. Hence, our approach leads to an average performance gain of 7,7%-points. This is comparable to the gain achieved for the EoL test case.

Moreover, SPH+CPI again outperforms the baseline for shorter lists $\mathcal{R}_e$, e.g., the highest gain in accuracy is about 11%-points for $\mathcal{R}_2$. These performance gains for shorter lists are important for many real-world applications. In the EoL test case, for instance, they increase the probability that oper-

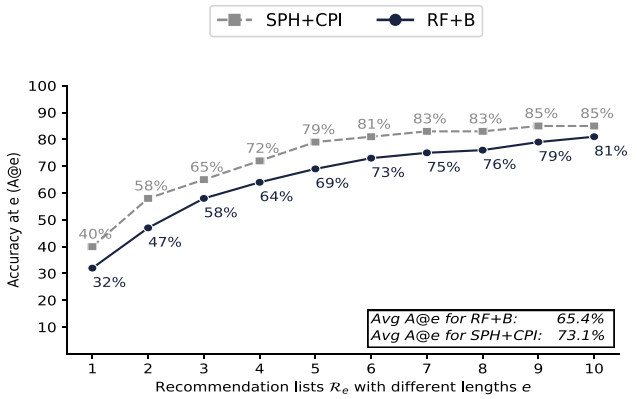


Fig. 10 Evaluation results for the medium sample set. The accuracy is shown as $A@1$ to $A@10$ for recommendation lists $\mathcal{R}_e$ with different lengths $e \in \{1, \ldots, 10\}$

Table 6 $RA@10$ scores for RF+B and SPH+CPI

Sample set	RA@10	
	RF+B	SPH+CPI
Very balanced	5.35	**4.10**
Balanced	3.50	**3.22**
Medium	2.96	**2.43**
Imbalanced	2.75	**2.50**
Very imbalanced	2.12	**1.80**

Bold values are the respectively best results

ators may solve a quality issue with fewer rework attempts (see Sect. 5.2.3). To illustrate this, Table 6 shows the $RA@10$ scores yielded by RF+B and SPH+CPI for all five sample sets. This $RA@10$ score represents the average number of rework attempts that operators need when working through the whole list $\mathcal{R}_{10}$ with 10 elements. Note again that the $RA@10$ score is lower the higher the hit rate on the first positions in a list $\mathcal{R}_e$ (see Sect. 5.2).

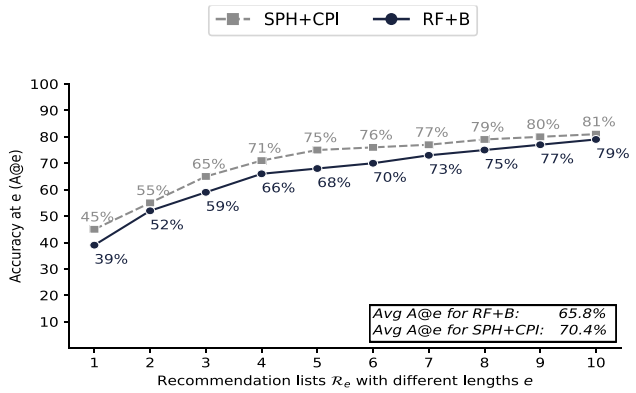
For the medium sample set, our approach reduces the $RA@10$ score by 0.53 points. This leads to significant cost reductions for reworking on defective engines. As discussed in Sect. 5.3, a company using our approach may save costs in a magnitude of several millions of EURO per year. The significant reduction of the $RA@10$ score also entails practical benefits for use cases of data-driven medical diagnoses (see Sect. 6.1.2). Here, the idea is that a physician likewise works through the recommendations of the list $\mathcal{R}_e$. However, given that each recommendation list delivers only a moderate accuracy, the physician performs further diagnostics for each prediction of the list to verify or falsify it. Nevertheless, any further diagnostics may be associated with inconveniences or even adverse effects for the patient. Here, a lower $RA@10$ score leads to a reduction of the average number of such inconvenient further diagnostics. For instance, reducing the $RA@10$ score by 0.53 for the medium sample set entails that we need to carry out one fewer diagnostics for every second patient.

We have again evaluated to which extent SPH and CPI contribute to the overall performance gain. On average, the accuracy of applying SPH in isolation is 72.7%. which is only about 0.4%-points less than for applying SPH+CPI together. So, the performance gain of CPI is smaller compared to the EoL test case, where CPI adds 3.4%-points to the gain in accuracy of SPH. Nevertheless, we discuss in Sect. 7.5 in more detail that CPI still increases the $A@e$ score for the samples of specific groups of our hierarchical domain model.

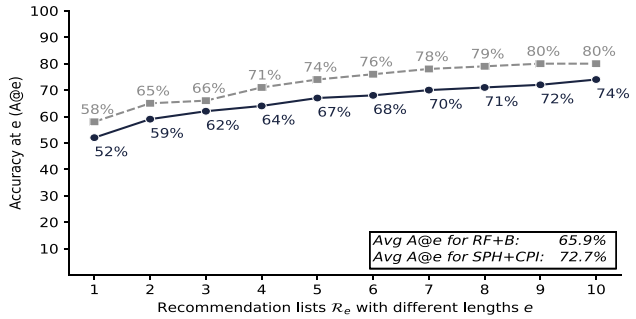
Finally, we assess to which extend SPH and CPI increase the accuracy for majority and minority classes separately. Thereby, we treat a class as majority class if it is represented by more than the median number of samples per class, while minority classes are represented by less than this median. This way, we may investigate whether our approach is indeed able to address multi-class imbalance. The goal is to increase the accuracy of especially the underrepresented minority classes, while not reducing that of majority classes. The baseline RF+B yields an $Avg\ A@e$ score of 76.0% for majority classes and 28.5% for minority classes. SPH increases accuracy for majority classes to 79.8%, i.e., by 3.8%-points. For minority classes, the gain is even much higher with 18.3%-points, i.e., SPH here yields 46.8% accuracy. CPI does not change the accuracy of majority classes, but even adds additional 1.8%-points to that of minority classes, so that the final $Avg\ A@e$ score for them is 48.6%. So, our approach SPH+CPI reaches its goal to especially and significantly increase accuracy for minority classes.

Imbalanced and very imbalanced sample sets: For the more imbalanced sample sets, the accuracies at lower positions, e.g., $A@1$, increase for both the baseline RF+B and for our approach (see Fig. 11). The reason is that the more imbalanced sample sets contain more samples of majority classes. It is hence more likely that especially these majority classes are predicted correctly. Majority classes are usually predicted on the first positions in a list $\mathcal{R}_e$. So, this also increases the accuracy at these first positions. Yet, for longer lists ($e > 2$), most of the $A@e$ scores are even less compared to the sample set of the medium class distribution. Here, we see the opposite effect for minority classes. More imbalanced sample sets contain less samples of minority classes that are then usually predicted less accurately at higher positions of a recommendation list.

Overall, SPH+CPI again increases accuracy compared to the baseline RF+B for any length of the list $\mathcal{R}_e$. The average performance gain for the imbalanced sample set is about 4.6%-points, while it is about 6.8%-points for the very imbalanced set. Note that again primarily SPH contributes to this increase in accuracy to a similar extent as for the medium sample set. Likewise, our approach is able to address multi-class imbalance, i.e., SPH+CPI especially increases accuracy

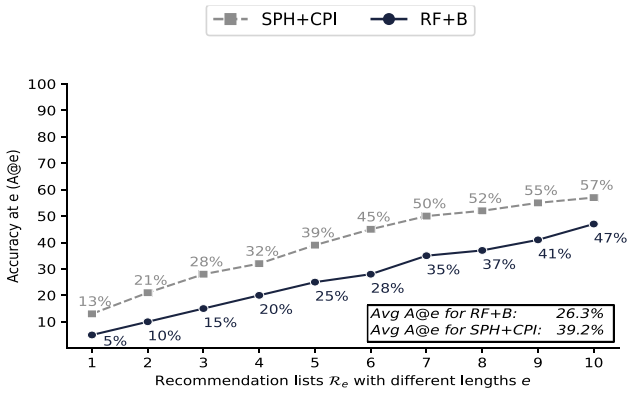


(a) Imbalanced Sample set

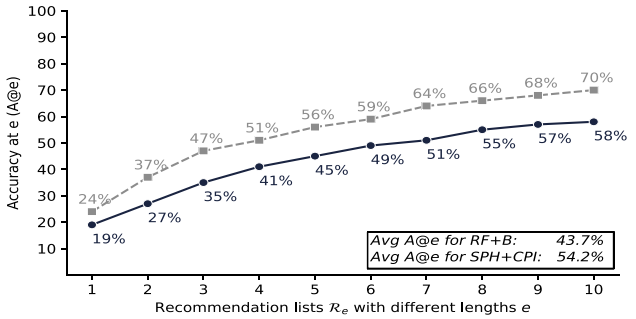


(b) Very Imbalanced Sample set

Fig. 11 Evaluation results for imbalanced sample sets



(a) Very Balanced Sample set



(b) Balanced Sample set

Fig. 12 Evaluation results for balanced sample sets

for minority classes, while not reducing that of majority classes. For the imbalanced sample set, the $Avg\ A@e$ score of majority classes increases by 3.9%-points from 77.0% of RF+B to 80.9% of SPH+CPI. Again, the gain in accuracy is much higher for minority classes with 9.5%-points, i.e., from 27.8 to 37.3%. The gain in accuracy for the very imbalanced sample set are 2.8%-points for majority and 11.5%-points for minority classes.

Yet, the increase in average accuracy achieved by both SPH and CPI is less than for the medium sample set. The reason is that we built more surrogate subsets and thus less primary subsets for the more imbalanced sample sets (cf. Table 5). A classifier trained on a primary subset is more specialized to the samples of the relevant group than a classifier trained on a surrogate. Hence, the accuracy for the surrogates is less than for the primary subsets. This confirms our expectation stated in Sect. 7.1 that SPH yields a gain in accuracy, but that this gain is lower if it chooses more surrogates.

Similar to the increase in the accuracy, the $RA@10$ scores decrease by 0.25 or 0.32 points, respectively. This again entails practical benefits for real-world use cases. It for instance reduces the number of rework attempts in EoL tests or the number of inconvenient further diagnostics of patients in medical use cases.

Balanced and very balanced sample sets: For the more balanced sample sets, we have the contrary effect that the $A@e$ scores for both RF+B and SPH+CPI are lower compared to the medium sample set (cf. Fig. 12). The reason is that the majority classes are now represented less frequently in the sample sets. As a result, the classifiers predict majority classes with a lower accuracy. This especially decreases the accuracy scores for lower positions of a recommendation list.

Nevertheless, SPH+CPI still outperforms the baseline RF+B with average accuracy gains of 12.9%-points for the very balanced and 10.5%-points for the balanced sample set. These are even higher accuracy gains than for the medium and especially for the imbalanced sample sets. The reason for this is that SPH generates fewer surrogate subsets for the balanced sample sets (cf. Table 5). This means that we observe the opposite effect, i.e., the higher number of primary subsets leads to a higher gain in accuracy.

In particular for cases where the baseline RF+B achieves such small accuracy scores, the significant accuracy gains of SPH+CPI are to be considered very important. For the very balanced sample set as example, SPH+CPI increases $A@e$ scores by a factor between 2.6 and 1.9 for the first three positions of the recommendation lists ($e \in \{1, \ldots, 3\}$). This way, SPH+CPI decreases the $RA@10$ score especially for

Table 7 Average accuracy $A@e$ and $RA@10$ scores for the five synthetic sample sets with two different parameterizations of SPH and CPI. *Optimized* parameters result from an individually performed grid parameter search, while *default* parameters correspond to those yielding the best accuracy scores for the data of the EoL test case

Sample set	Max. information loss		Gini		p value		$AvgA@e$ (%)		$RA@10$	
	Optimized	Default	Optimized	Default	Optimized	Default	Optimized	Default	Optimized	Default
Very balanced	**0.35**	0.25	0.30	0.30	**0.70**	0.80	**39.2**	39.0	**4.10**	4.24
Balanced	**0.30**	0.25	**0.35**	0.30	0.80	0.80	**54.2**	53.9	**3.22**	3.30
Medium	**0.35**	0.25	**0.40**	0.30	**0.90**	0.80	**73.1**	71.1	**2.43**	2.67
Imbalanced	**0.35**	0.25	**0.40**	0.30	**0.85**	0.80	**70.4**	69.4	**2.50**	**2.50**
Very imbalanced	**0.30**	0.25	**0.40**	0.30	0.80	0.80	**72.7**	71.6	**1.80**	1.90

The bold values indicate the best scores obtained for each sample set

the very balanced sample set by the largest amount, i.e., by 1.25 points.

For the balanced sample set, SPH again contributes most to the accuracy gain with 9.9 of the overall 10.5%-points. However, we make a different observation for the very balanced sample set. Here, the gain of 12.9%-points is achieved by SPH alone, i.e., CPI neither increases nor reduces accuracy. Note that the intended purpose of CPI is to further increase accuracy especially for sample subsets with a high class imbalance, i.e., a high Gini coefficient. The subsets SPH generates for the very balanced sample set, however, show a small average Gini coefficient of 17% (see Table 5). So, it is evident that CPI does not further increase accuracy for these subsets of the very balanced sample set.

In a similar way, the gain in accuracy is roughly the same for majority and minority classes, i.e., our approach does not significantly favor any of these classes in case of these balanced sample sets with low Gini coefficients. For the balanced sample set, SPH+CPI increases the $Avg\ A@e$ score of majority classes by 9.9%-points and that of minority classes by 10.9%-points. The gain in accuracy for the very balanced sample set are 12%-points and 14.3%-points, respectively.

7.4 Optimization of SPH and CPI parameters for varying class distributions

In this section, we analyze the influence of the parameters of SPH and CPI for the five synthetic sample sets. Thereby, we verify if the multi-criteria parameter optimization is worth its computational effort (see Sect. 6.3). Table 7 summarizes how the results differ for the five sample sets between two parameterizations of SPH and CPI. First, the *optimized* parameters in Table 7 report the parameter values yielding the best accuracy scores via our grid search. The optimization goal was to maximize the average accuracy among all positions e of a list $\mathcal{R}_e$. Table 7 shows this as the $AvgA@e$ scores, while it also presents the $RA@10$ scores. The second parameterization is the one that obtained the best result for the original sample set of the EoL test case (see Sect. 5.2.1). Table 7 denotes them as the *default* parameters. Note that, for any of the five sample sets, the results for $AvgA@e$ and $RA@10$ yielded by these default parameters are still better then the results of the baseline RF+B. Our goal is to examine whether the parameters for the EoL data provide a generally applicable heuristic for other sample sets, so that we may avoid the multi-criteria optimization.

For the more balanced sample sets, the differences in $AvgA@e$ between the optimized and default parameters are only marginally, i.e., 0.2%-points for the very balanced and 0.3 %-points for the balanced sample set. The $RA@10$ scores differ only with 0.14 and 0.08 points. These improvements can be seen as negligible for most real-world use cases. Hence, the effort of optimizing the parameters of SPH and CPI is usually not rewarding for sample sets with more balanced class distributions.

At first glance, the optimization seems to be more rewarding the more imbalanced the data is. For the two more imbalanced sample sets, the improvements in accuracy are 1.0 and 1.1%-points. However, the gains in the $RA@10$ score are only 0.1 points for the very imbalanced and even 0.0 for the imbalanced sample set. The reason is that gains in the $RA@10$ score are mainly achieved with higher hit rates at first positions in a recommendation list $\mathcal{R}_e$ (see Sect. 5.2). Yet, the accuracy gains for the two imbalanced sample sets are mainly achieved in the upper positions of a list $\mathcal{R}_e$. Altogether, the gains for such imbalanced sample sets are usually negligible as well, so that a parameter optimization does not pay off its computational effort.

For the medium sample set, however, the optimization increases the $AvgA@e$ score by 2.0%-points and the $RA@10$ score by 0.24 points. This may be beneficial for a limited set of use cases, where such comparatively low gains in both scores are crucial. For instance, this holds for data-driven medical diagnoses, where it results in a comparatively less number of further diagnostics and thus in fewer adverse effects for the patients.

In summary, the default parameters we found for the EoL data already achieve good results for many different sample

Table 8 Accuracy $A@1$, average $A@e$ ($Avg\ A@e$), and $RA@10$ score differentiated between groups containing less than or equal to the median number of samples per group and those containing more than this median

Groups with…	$A@1$ (%)			Avg $A@e$ (%)			$RA@10$		
	RF+B	SPH	SPH+CPI	RF+B	SPH	SPH+CPI	RF+B	SPH	SPH+CPI
$\leq$ 22 samples	21.3	**36.2**	**36.2**	46.8	**69.3**	68.7	3.67	**2.66**	2.69
$\geq$ 22 samples	34.8	36.8	**41.1**	69.8	73.4	**74.0**	2.61	2.15	**2.09**

Bold values are the respectively best results

sets. So, we can usually avoid the overhead for optimizing the parameters of SPH and CPI. Only in rare cases, a sophisticated parameter optimization may be worthwhile. This is for instance the case if the data shows similar characteristics as our medium sample set.

7.5 Detailed analysis of SPH and CPI improvements

Now, we discuss the improvements obtained with SPH and SPH+CPI compared to the baseline RF+B in more detail. To this end, we separate and compare the results for two different sets of the 26 groups at the bottom level of our hierarchy model. We distinguish between groups (1) containing less than or equal to the median number of samples per group and (2) those containing more than this median. The rationale behind this is that our approach, in particular SPH, aims at improving the accuracy especially for the first set of groups that are underrepresented in the whole sample set $\mathcal{X}$. This way, SPH may for instance address the increasing product variety in use cases across the industrial value chain, as well as the ethnic or gender bias present in use cases of data-driven medical diagnoses (see Sect. 6.1). For the sake of clarity, we focus on the medium sample set in this section. Here, the median number of samples per group is 22. Table 8 shows the results of $A@1$, $AvgA@e$, and $RA@10$ for groups with $\leq$ 22 samples and groups with $>$ 22 samples. Note that the results for the other sample sets show similar trends.

SPH especially increases the accuracy for smaller groups with $\leq$ 22 samples in comparison to the baseline RF+B. It increases the $A@1$ value by 14.9%-points, while the increase in the $AvgA@e$ score is even higher with 22.5%-points. This also leads to a reduction of the $RA@10$ score by 1.01 points, which corresponds to one fewer rework attempt for each underrepresented group. So, SPH achieves remarkably better results for smaller groups with less than 22 samples. This shows that SPH is especially beneficial for groups that are underrepresented in data. For groups with $>$ 22 samples, SPH also increases the $AvgA@e$ score by 3.6%-points, and it reduces the $RA@10$ score by 0.46. So, this step of our approach also entails benefits for such bigger groups.

The subsequent step CPI changes the scores $A@1$, $AvgA@e$, and $RA@10$ only negligibly for the smaller groups. The strengths of CPI come apparent for bigger groups with more than 22 samples. Here, it further increases the accuracy scores of SPH by 0.6%-points for $AvgA@e$ and even by 4.3%-points for $A@1$. The improvement occurs mainly due to one single group that contains 309 samples and thus by far the most samples. Five classes of this group have more than the median number of 22 samples. Table 9 shows a detailed view on the results of the $A@1$ score for these five classes after applying only SPH and SPH+CPI. The table also indicates how CPI partitions these classes into majority classes $\mathcal{X}_j^+$ and minority classes $\mathcal{X}_j^-$.

Class A, which occurs most often with 95 samples, is predicted very accurately after only applying SPH with an $A@1$ score of 97.4%. The other four frequent classes with 33 to 47 samples are however predicted very inaccurately. Also applying CPI to the subset of this group yields a slightly smaller accuracy for class A, but it significantly increases the accuracy of the other classes. The reason is that CPI generates a separate sample subset for the most frequent class A. Then, it applies the OvA binarization strategy as described in Sect. 4.1.3. The resulting subset thus has one positive class with 95 samples for the single majority class A and one negative class with the remaining 214 samples of the whole subset. So, this negative class occurs more often than the positive class, so that the original majority class A becomes a minority class in this new subset.

In addition, CPI generates a minority subset for all remaining classes, i.e., also for classes B to E. In this minority subset, these four classes are no longer underrepresented by class A. Hence, there is a significant increase in accuracy for classes B to E. This even leads to an increase of the $A@1$ score for the whole group by 11%-points, i.e., to 55% in comparison to 44% of SPH. Altogether, CPI is especially worthwhile for groups with a large number of samples, where a single majority class accounts for a big share of all samples.

7.6 Runtime efficiency and scalability

With SPH and CPI, our approach introduces additional steps to a data preparation and model training pipeline. Furthermore, it changes the model training phase, as we train several classifiers $\mathcal{M}_j$ for individual sample subsets $\mathcal{X}_j$ resulting from SPH and CPI. We now discuss the effects of these changes on the runtime efficiency. This also includes a discussion of the trade-off between a possible runtime overhead and the gain in accuracy of SPH and CPI compared to the baseline RF. Furthermore, we investigate the scalability of our approach by evaluating both the runtime efficiency and

Table 9 Accuracy results for the five most frequent classes of the group containing in total 309 and thus the most samples. We also show how CPI partitions the classes into majority $\mathcal{X}_j^+$ and minority classes $\mathcal{X}_j^-$

CPI partition	Class	Occurrence	A@1	
			SPH	SPH+CPI
$\mathcal{X}_j^+$	A	95	**97.4**	86.8
$\mathcal{X}_j^-$	B	47	31.3	**68.8**
	C	45	7.1	**50.0**
	D	35	0.0	**8.0**
	E	33	7.7	**23.1**
	..	..	..	..
$A@1$ for the whole group:			44.0	**55.0**

Bold values are the respectively best

the classification performance with bigger data sets of up to one million samples. The intention is to asses an important aspect of the generality of our approach, i.e., whether it is still efficient in use cases with higher data sizes.

Table 10 reports the results regarding runtime efficiency and scalability of our approach. Here, we used the medium sample set with its 1050 samples and employed our data generator [46] to generate sample sets with the same data distribution, but with sizes of 10 000, 100 000, 500 000, and 1 000 000 samples. The results for the data distributions of the other four synthetic sample sets again show similar trends. We have carried out all measurements on a virtual machine with Ubuntu 22.04 as operating system, 32 GB RAM, and 16 virtual CPU cores. We parameterized SPH and CPI using the default parameters as reported in Table 7, i.e., 0.25% for the maximum information loss of SPH, 0.3% for the Gini threshold of CPI and 0.8 as p value. Similar to the discussion in Sect. 7.4, the evaluation outcomes and trends do not significantly differ with another parameterization that is optimized towards accuracy.

For each of the sample sets with different data sizes, we used the same split ratio to divide it into a training set and a test set as explained in Sect. 5.2.1. The third column in Table 10 shows the respective numbers of samples used as training set for the RF baseline. For instance, we used 750 training samples for the data set with a total number of 1050 samples and 7142 training samples for the data set with 10 000 samples. The remaining 300 or 2858 samples constitute the test set. For our approach SPH+CPI, we used the same train/test split and number of training samples. However, SPH+CPI further splits the training set into 26 subsets $\mathcal{X}_j$ and then applies the further data preprocessing and model training on each subset individually. Hence, the fourth column of the table indicates for SPH+CPI the average number of samples in these subsets $\mathcal{X}_j$, e.g., 65 or 145 samples for the data sets with in total 1050 or 10 000 samples. Columns 5 to 7 show the results for the performance scores $AvgA@e$,

Table 10 Evaluation results of the baseline RF and our approach SPH+CPI for data sets with varying and higher numbers of samples. The table shows the total number of samples, the number of samples used as training set (for RF), the average size of the sample subsets $\mathcal{X}_j$ (for SPH+CPI), the values of the performance scores $AvgA@e$, $A@1$ and $RA@10$, as well as the overall runtime of the data pipeline and the specific runtime of model training

# samples total	Method	# samples in training set	Avg # samples in subsets $\mathcal{X}_j$	Performance scores			Runtime (in seconds)	
				$AvgA@e$ (%)	$A@1$ (%)	$RA@10$	Overall	Model training
1050	RF	750	–	65.4	31.3	2.89	**0.36**	**0.36**
	SPH+CPI	–	65	**73.1**	**33.7**	**2.65**	2.46	1.68
10 000	RF	7142	–	73.6	39.7	2.74	**3.44**	3.42
	SPH+CPI	–	145	**77.3**	**43.7**	**2.65**	3.52	**2.18**
100 000	RF	71 428	–	76.6	39.7	2.69	37.69	37.54
	SPH+CPI	–	1428	**81.5**	**50.0**	**2.33**	**20.04**	**13.15**
500 000	RF	357 142	–	77.4	38.8	2.64	197.52	196.68
	SPH+CPI	–	7142	**83.9**	**53.2**	**2.16**	**123.57**	**70.18**
1 000 000	RF	714 285	–	77.3%	38.8%	2.66	405.72	404.19
	SPH+CPI	–	14 285	**84.5**	**54.2**	**2.11**	**307.29**	**149.77**

Bold values are the respectively best results

$A@1$ and $RA@10$ of the baseline RF and of our approach SPH+SPI. The eighth column shows the overall runtime, i.e., the sum for both data preprocessing—including SPH and CPI in our approach—and for training the classifiers $\mathcal{M}_j$. We also separately report the runtime of model training in column 9 to investigate how our approach especially affects this model training.

The results reported in Table 10 show that our approach SPH+CPI outperforms the baseline RF for all sample sets. For all data sizes from 1050 to 1 000 000 samples, we achieve similar gains in the performance scores as reported in the subsections above. For instance, the gain in $AvgA@e$ even increases from the data set with 10 000 samples to that with 1 000 000 samples in a range from 3.7 to 7.2%-points. A similar trend may be observed for the other scores $A@1$ and $RA@10$.

For the smaller data set with 1050 samples, both the overall runtime and that of model training are higher for SPH+CPI than for the baseline RF. Nevertheless, the overall runtime overhead is only about two seconds, which can be seen as negligible. In fact, such a small runtime overhead is acceptable in exchange for the significant improvements in the performance scores, e.g., an increase of 7.7%-points in $AvgA@e$.

While our approach SPH+CPI indeed introduces an additional overhead to data preprocessing, it significantly reduces the runtime of model training for all bigger sample sets with 10 000 or more samples. In most cases, this even leads to a reduction of the overall runtime. The runtime of model training is mainly determined by the computational complexity of the Random Forest algorithm. The lower bound of this complexity is in $\mathcal{O}(n * log(n) * f * v)$, where n is the number of training samples, f is the number of features in the training data set, $\mathcal{O}(n * log(n) * f)$ is the lower bound of the complexity to build one decision tree, and v is the number of decision trees Random Forest builds [6]. So, the runtime of model training grows faster than linearly with the input data size n. The baseline RF applies the Random Forest algorithm to the whole training data set, e.g., the input data size n is 714 285 samples in case of the biggest data set with in total 1 000 000 samples. This leads to a high runtime for model training, which even accounts for by far the largest share of the overall runtime of the baseline RF.

In contrast, our approach SPH+CPI subdivides the whole training data set into several sample subsets $\mathcal{X}_j$. It then applies the Random Forest algorithm on each of these much smaller subsets. On the one hand, our approach hence applies the Random Forest algorithm more often than the baseline, e.g., 26 times for each of the 26 sample subsets $\mathcal{X}_j$ of our data sets. On the other hand, SPH+CPI significantly reduces the number of training samples to which the Random Forest algorithm is applied each time. In case of the biggest data set, e.g., it is on average applied to only 14 285 samples of each $\mathcal{X}_j$, which constitutes a reduction of the average input data size n by a factor of about 50. As the runtime grows faster than linearly with the input size n, i.e., in $\mathcal{O}(n * log(n))$, this significant reduction of the average input size n leads to likewise significant reduction of the runtime of model training and finally also of the overall runtime. Altogether, this again proves the generality of our approach, as it leads to an increase in classification performance and at the same time reduces the runtime in use cases with bigger data sets.

Note that we used an implementation of our approach that carries out CPI, the data preprocessing and model training sequentially for each sample subset $\mathcal{X}_j$ that SPH delivers. Actually, we could even further reduce the runtime of our approach SPH+CPI, as it allows for an easy parallelization of all steps after SPH for the individual sample subsets. However, we deliberately chose the sequential implementation to assume a worst case scenario for SPH+CPI.

7.7 Evaluation summary

In summary, our extensive evaluation yields the following major findings:

- SPH and CPI are suitable to mitigate the negative effects of both challenges C1 and C2 in sample sets with different data and class distributions (see Sect. 7.2). SPH for instance significantly reduces the heterogeneity in the feature space, e.g, the share of missing feature values in all our synthetically generated sample sets (see Table 5). The benefit of CPI is apparent especially for sample sets with a higher multi-class imbalance. Here, it is able to further reduce the Gini coefficient by up to 20%-points.
- Our approach applying both SPH and CPI together outperforms the baseline RF+B for any of the class distributions of the five synthetic sample sets (see Sect. 7.3). It increases the average classification accuracy by values between 4.6 and 12.9%-points.
- In addition, our approach reduces the $RA@e$ scores by up to 1.25. This entails considerable practical benefits for real-world use cases. It for instance reduces the number of ineffective rework attempts in EoL testing or the number of further medical diagnostics that may have adverse effects for a patient.
- With our grid search, we verify that the multi-criteria optimization of the parameters of both steps SPH and CPI is usually not worth its computational effort (Sect. 7.4). In fact, the parameter values that yielded the best result for the EoL data already offer a good heuristic for the synthetic sample sets.
- Our detailed analysis in Sect. 7.5 reveals that SPH serves its intended purpose: It achieves significantly better classification accuracies and $RA@10$ scores especially for smaller groups that are underrepresented in the sample sets. So, it helps to reduce representation bias in data [31],

e.g., ethnic or gender bias in data-driven medical diagnoses.
- The next step CPI has its strength for bigger groups in imbalanced data. This especially holds for groups where a single majority class accounts for a big share of the samples. Here, CPI significantly increases accuracy for all other classes that have previously been superimposed by the single majority class.
- Our evaluation results with bigger data sets (Sect. 7.6) reinforce the generality of our approach. In fact, SPH+CPI significantly increase classification performance and at the same time reduce the runtime of a data preparation and model training pipeline even for use cases with very high data sizes.

8 Conclusion

The main contribution of this paper is an approach that exploits domain knowledge to systematically prepare training data for multi-class classification. The domain knowledge may be provided by hierarchical relationships in semantic nets, e.g., in taxonomies or ontologies. Thereby, our approach partitions the training data into several sample subsets in order to address two of the most important analytical challenges in real-world classification problems: multi-class imbalance and heterogeneous feature space.

This is confirmed by our evaluation, where we first applied our approach on real-world manufacturing data and used a product hierarchy as domain knowledge. To prove the generality of our approach, we conducted additional measurements with several synthetically generated data sets that represent different data distributions found in other application domains. In any of these evaluations, our approach dominates the baseline solutions and achieves significant increases in classification accuracy. Moreover, it entails considerable practical benefits in real-world use cases. For instance, it reduces the number of rework attempts in EoL testing or the number of further inconvenient diagnostics in medical use cases.

Our approach requires a hierarchical taxonomy structure to build homogeneous sample subsets. For use cases where such a taxonomy is not yet defined, we are going to investigate how to build adequate sample subsets based on clustering techniques, especially hierarchical or constraint-based clustering. In addition, we discussed our results for the Random Forest algorithm, as this ensemble technique yielded the best baseline results with our data characteristics. In future, we are going to evaluate our approach and its two steps SPH and CPI for other learning algorithms and data characteristics, e.g., for neural networks dealing with high-resolution time series data and bigger sample sets. Finally, our approach focuses on multi-class classifications and thus presumes to have labeled data. Future work may hence investigate whether a data partitioning based on domain knowledge may help to prepare unlabeled data, e.g., for unsupervised data analytics.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

1. Agard, B., Kusiak, A.: Data-mining-based methodology for the design of product families. Int. J. Prod. Res. **42**(15), 2955–2969 (2004). https://doi.org/10.1080/00207540410001691929
2. Akhand, M.A.H., Murase, K.: Neural network ensemble training by sequential interaction. In: Proceedings of the 17th International Conference on Artificial Neural Networks, LNCS, pp. 98–108. Springer, Porto, Portugal (2007). https://doi.org/10.1007/978-3-540-74690-4_11
3. Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S.: KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. Multiple-Valued Logic Soft Comput. **17**(2–3), 255–287 (2011)
4. Bach, S.H., Rodriguez, D., Liu, Y., Luo, C., Shao, H., Xia, C., Sen, S., Ratner, A., Hancock, B., Alborzi, H., Kuchhal, R., Ré, C., Malkin, R.: Snorkel Drybell: A case study in deploying weak supervision at industrial scale. In: Proceedings of the 2019 International Conference on Management of Data (SIGMOD), pp. 362–375. Amsterdam, The Netherlands (2019). https://doi.org/10.1145/3299869.3314036
5. Baggio, G., Corsini, A., Floreani, A., Giannini, S., Zagonel, V.: Gender medicine: a task for the third millennium. Clin Chem Lab Med **51**(4), 713–727 (2013). https://doi.org/10.1515/cclm-2012-0849
6. Breiman, L.: Random forests. Mach Learn **45**(1), 5–32 (2001). https://doi.org/10.1023/A:1010933404324
7. Chan, S., Reddy, V., Myers, B., Thibodeaux, Q., Brownstone, N., Liao, W.: Machine learning in dermatology: current applications, opportunities, and limitations. Dermatol Therapy **10**(3), 365–386 (2020). https://doi.org/10.1007/s13555-020-00372-0
8. Cheng, Y., Chen, K., Sun, H., Zhang, Y., Tao, F.: Data and knowledge mining with big data towards smart production. J. Ind. Inf. Integr. **9**, 66 (2017). https://doi.org/10.1016/j.jii.2017.08.001
9. Cowell, F.: Measuring Inequality, 3rd edn. Oxford Academic (2011). https://doi.org/10.1093/acprof:osobl/9780199594030.001.0001
10. Fernández, A., López, V., Galar, M., del Jesus, M.J., Herrera, F.: Analysing the classification of imbalanced data-sets with multiple classes: binarization techniques and ad-hoc approaches. Knowl. Based Syst. **42**, 97–110 (2013). https://doi.org/10.1016/j.knosys.2013.01.018

11. Fitzpatrick, T.B.: The validity and practicality of sun-reactive skin types I through VI. Arch. Dermatol. **124**(6), 869–871 (1988). https://doi.org/10.1001/archderm.1988.01670060015008
12. Galar, M., Fernández, A., Barrenechea, E., Bustince, H., Herrera, F.: An overview of ensemble methods for binary classifiers in multi-class problems: experimental study on one-vs-one and one-vs-all schemes. Pattern Recognit. **44**(8), 1761–1776 (2011). https://doi.org/10.1016/j.patcog.2011.01.017
13. Galar, M., Fernández, A., Barrenechea, E., Bustince, H., Herrera, F.: A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. IEEE Trans. Syst. Man Cybern. C Appl. Rev. **42**(4), 463–484 (2012). https://doi.org/10.1109/TSMCC.2011.2161285
14. Gerling, A., Schreier, U., Hess, A., Saleh, A., Ziekow, H., Ould Abdeslam, D.: A reference process model for machine learning aided production quality management. In: Proceedings of the 22nd International Conference on Enterprise Information Systems (ICEIS 2020), pp. 515–523. Prague, Czechia (2020). https://doi.org/10.5220/0009379705150523
15. Gini, C.: Measurement of inequality of incomes. Econ J **31**(121), 124–126 (1921). https://doi.org/10.2307/2223319
16. Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., Bing, G.: Learning from class-imbalanced data: review of methods and applications. Expert Syst. Appl. **73**, 220–239 (2017). https://doi.org/10.1016/j.eswa.2016.12.035
17. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. **21**(9), 1263–1284 (2009). https://doi.org/10.1109/TKDE.2008.239
18. Hirsch, V., Reimann, P., Kirn, O., Mitschang, B.: Analytical approach to support fault diagnosis and quality control in end-of-line testing. Procedia CIRP **72**, 1333–1338 (2018). https://doi.org/10.1016/j.procir.2018.03.024
19. Hirsch, V., Reimann, P., Mitschang, B.: Data-driven fault diagnosis in end-of-line testing of complex products. In: Proceedings of the 6th IEEE International Conference on Data Science and Advanced Analytics (DSAA), pp. 492–503. IEEE (2019). https://doi.org/10.1109/DSAA.2019.00064
20. Hirsch, V., Reimann, P., Mitschang, B.: Approach to incorporate cost aspects into the ordering of a data-driven recommendation list for end-of-line testing. Procedia CIRP **74**, 747–752 (2020). https://doi.org/10.1016/j.procir.2020.03.026
21. Hirsch, V., Reimann, P., Mitschang, B.: Exploiting domain knowledge to address multi-class imbalance and a heterogeneous feature space in classification tasks for manufacturing data. PVLDB **13**(12), 3258–3271 (2020). https://doi.org/10.14778/3415478.3415549
22. Hu, S., Zhu, X., Wang, H., Koren, Y.: Product variety and manufacturing complexity in assembly systems and supply chains. CIRP Ann. **57**(1), 45–48 (2008). https://doi.org/10.1016/j.cirp.2008.03.138
23. Humphreys, G.: Coming together to combat rare diseases. Bull. World Health Organ. **90**(6), 401–476 (2012). https://doi.org/10.2471/BLT.12.020612
24. Jablonski, N.: The evolution of human skin and skin color. Ann. Rev. Anthropol. **33**, 585–623 (2004). https://doi.org/10.1146/annurev.anthro.33.070203.143955
25. Kassner, L., Mitschang, B.: Exploring text classification for messy data: an industry use case for domain-specific analytics technology. In: Proceedings of the 19th International Conference on Extending Database Technology (EDBT), pp. 491–502. Bordeaux, France (2016). https://doi.org/10.5441/002/edbt.2016.47
26. Kiefer, C., Reimann, P., Mitschang, B.: A hybrid information extraction approach exploiting structured data within a text mining process. In: Proceedings of the 18th Conference on Datenbanksysteme für Business, Technologie und Web (BTW), pp. 149–168. Rostock, Germany (2019). https://doi.org/10.18420/btw2019-10
27. Köksal, G., Batmaz, I., Testik, M.C.: A review of data mining applications for quality improvement in manufacturing industry. Expert Syst Appl. **38**(10), 13448–13467 (2011). https://doi.org/10.1016/j.eswa.2011.04.063
28. Kursa, M.B., Rudnicki, W.R.: Feature selection with the Boruta package. J. Stat. Softw. **36**(11), 66 (2010). https://doi.org/10.18637/jss.v036.i11
29. Leevy, J.L., Khoshgoftaar, T.M., Bauder, R.A., Seliya, N.: A survey on addressing high-class imbalance in big data. J. Big Data **5**(42), 66 (2018). https://doi.org/10.1186/s40537-018-0151-6
30. Liu, Y., Jin, R., Jain, A.: BoostCluster: boosting clustering by pairwise constraints. In: Proceedings of the 13th International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 450–459. San Jose, CA, USA (2007). https://doi.org/10.1145/1281192.1281242
31. Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., Galstyan, A.: A survey on bias and fairness in machine learning. ACM Comput. Surv. **54**, 66 (2021). https://doi.org/10.1145/3457607
32. Mehrpouya, M., Dehghanghadikolaei, A., Fotovvati, B., Vosooghnia, A., Emamian, S.S., Gisario, A.: The potential of additive manufacturing in the smart factory industrial 4.0: a review. Appl. Sci. **9**(18), 66 (2019). https://doi.org/10.3390/app9183865
33. Nanni, L., Lumini, A., Brahnam, S.: A classifier ensemble approach for the missing feature problem. Artif. Intell. Med. **55**(1), 37–50 (2012). https://doi.org/10.1016/j.artmed.2011.11.006
34. Polikar, R.: Ensemble based systems in decision making. IEEE Circuits Syst. Mag. **6**(3), 21–45 (2006). https://doi.org/10.1109/MCAS.2006.1688199
35. Polikar, R., DePasquale, J., Syed Mohammed, H., Brown, G., Kuncheva, L.I.: Learn++.MF: a random subspace approach for the missing feature problem. Pattern Recognit. **43**(11), 3817–3832 (2010). https://doi.org/10.1016/j.patcog.2010.05.028
36. Quillian, R.: Word concepts. A theory and simulation of some basic semantic capabilities. Behav. Sci. **12**, 410–430 (1967). https://doi.org/10.1002/bs.3830120511
37. Ratner, A., Bach, S.H., Ehrenberg, H., Fries, J., Wu, S., Ré, C.: Snorkel: rapid training data creation with weak supervision. VLDB J. **29**, 709–730 (2020). https://doi.org/10.1007/s00778-019-00552-1
38. Rokach, L.: Ensemble-based classifiers. Artif. Intell. Rev. **33**(1–2), 1–39 (2010). https://doi.org/10.1007/s10462-009-9124-7
39. Silla, C.N., Freitas, A.A.: A survey of hierarchical classification across different application domains. Data Min Knowl Discov **22**(1–2), 31–72 (2011). https://doi.org/10.1007/s10618-010-0175-9
40. Sowa, J.F.: Principles of Semantic Networks. Explorations in the Representation of Knowledge. Representation and Reasoning. Morgan Kaufmann (1991)
41. Sun, C., Rampalli, N., Yang, F., Doan, A.: Chimera: large-scale classification using machine learning, rules, and crowdsourcing. PVLDB **7**(13), 1529–1540 (2014). https://doi.org/10.14778/2733004.2733024
42. Sun, Y., Wong, A., Kamel, M.: Classification of imbalanced data: a review. Int. J. Pattern Recognit. Artif. Intell. **23**(04), 687–719 (2009). https://doi.org/10.1142/S0218001409007326
43. Sun, Z., Song, Q., Zhu, X., Sun, H., Xu, B., Zhou, Y.: A novel ensemble method for classifying imbalanced data. Pattern Recognit. **48**(5), 1623–1637 (2015). https://doi.org/10.1016/j.patcog.2014.11.014
44. Suresh, H., Guttag, J.: A framework for understanding sources of harm throughout the machine learning life cycle. In: Proceedings of the 1st ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization (EAAMO) (2021). https://doi.org/10.1145/3465416.3483305
45. Thalmann, S., Gursch, H.G., Suschnigg, J., Gashi, M., Ennsbrunner, H., Fuchs, A.K., Schreck, T., Mutlu, B., Mangler, J., Kappl, G.,

Huemer, C., Lindstaedt, S.: Cognitive decision support for industrial product life cycles: a position paper. In: Proceedings of the 11th International Conference on Advanced Cognitive Technologies and Applications (COGNITIVE). IARIA, Venice, Italy (2019)
46. Treder-Tschechlov, D., Reimann, P., Schwarz, H., Mitschang, B.: Approach to synthetic data generation for imbalanced multi-class problems with heterogeneous groups. In: Proceedings of the 20th Conference on Datenbanksysteme für Business, Technologie und Web (BTW). Dresden, Germany (2023)
47. Verron, S., Li, J., Tiplica, T.: Fault detection and isolation of faults in a multivariate process with Bayesian network. J. Process Control **20**(8), 902–911 (2010). https://doi.org/10.1016/j.jprocont.2010.06.001
48. Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S.: Constrained K-means clustering with background knowledge. In: Proceedings of the 18th International Conference on Machine Learning (ICML), pp. 577–584. Williamstown, MA, USA (2001)
49. Wang, S., Minku, L.L., Yao, X.: A systematic study of online class imbalance learning with concept drift. IEEE Trans. Neural Netw. Learn. Syst. **29**(10), 4802–4821 (2018). https://doi.org/10.1109/TNNLS.2017.2771290
50. Wang, S., Yao, X.: Multiclass imbalance problems: analysis and potential solutions. IEEE Trans. Syst. Man Cybernet. B Cybernet. **42**(4), 1119–1130 (2012). https://doi.org/10.1109/TSMCB.2012.2187280
51. Weber, C., Hirmer, P., Reimann, P.: A model management platform for industry 4.0— enabling management of machine learning models in manufacturing environments. In: Proceedings of the 23rd International Conference on Business Information Systems (BIS), pp. 403–417 (2020). https://doi.org/10.1007/978-3-030-53337-3_30
52. Whitley, H.P., Smith, W.D.: Sex-based differences in medications for heart failure. The Lancet **394**(10205), 1210–1212 (2019). https://doi.org/10.1016/S0140-6736(19)31812-4
53. Wilhelm, Y., Schreier, U., Reimann, P., Mitschang, B., Ziekow, H.: Data science approaches to quality control in manufacturing: a review of problems, challenges and architecture. In: Proceedings of the 14th Symposium on Service-Oriented Computing (SummerSOC), Communications in Computer and Information Science (CCIS), pp. 45–65. Springer (2020). https://doi.org/10.1007/978-3-030-64846-6_4
54. Woźniak, M., Graña, M., Corchado, E.: A survey of multiple classifier systems as hybrid systems. Inf. Fusion **16**, 3–17 (2014). https://doi.org/10.1016/j.inffus.2013.04.006
55. Wuest, T., Weimer, D., Irgens, C., Thoben, K.D.: Machine learning in manufacturing: advantages, challenges, and applications. Prod. Manuf. Res. **4**(1), 23–45 (2016). https://doi.org/10.1080/21693277.2016.1192517
56. Zhou, Z.H., Liu, X.Y.: On multi-class cost-sensitive learning. In: Proceedings of the 21st National Conference on Artificial Intelligence—Vol. 1 (AAAI'06), pp. 567–572. AAAI Press, Boston, MA, USA (2006)