Info Vis

Article

Information Visualization
2023, Vol. 22(2) 115–139


Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/14738716221142005
journals.sagepub.com/home/ivi

VisRuler: Visual analytics for extracting decision rules from bagged and boosted decision trees

Angelos Chatzimparmpas[1], Rafael M. Martins[1] and Andreas Kerren[1,2]

Abstract

Bagging and boosting are two popular ensemble methods in machine learning (ML) that produce many individual decision trees. Due to the inherent ensemble characteristic of these methods, they typically outperform single decision trees or other ML models in predictive performance. However, numerous decision paths are generated for each decision tree, increasing the overall complexity of the model and hindering its use in domains that require trustworthy and explainable decisions, such as finance, social care, and health care. Thus, the interpretability of bagging and boosting algorithms—such as random forest and adaptive boosting—reduces as the number of decisions rises. In this paper, we propose a visual analytics tool that aims to assist users in extracting decisions from such ML models via a thorough visual inspection workflow that includes selecting a set of robust and diverse models (originating from different ensemble learning algorithms), choosing important features according to their global contribution, and deciding which decisions are essential for global explanation (or locally, for specific cases). The outcome is a final decision based on the class agreement of several models and the explored manual decisions exported by users. We evaluated the applicability and effectiveness of VisRuler via a use case, a usage scenario, and a user study. The evaluation revealed that most users managed to successfully use our system to explore decision rules visually, performing the proposed tasks and answering the given questions in a satisfying way.

Keywords

Decisions evaluation, rules interpretation, ensemble learning, visual analytics, visualization

Introduction

Ensemble learning (EL)[1] is a well-established area of machine learning (ML) that strives for better performance by merging the predictions from various ML models. Three prominent methods for building ensembles are [2]: bagging,[3] boosting,[4,5] and stacking.[6] Bagging requires training many decision trees on separate groups of instances and taking the average of their predictions.[3] Boosting attaches weak classifiers (e.g. decision stumps or shallow decision trees) sequentially, each improving the predictions made by the previous models.[4,5] Stacking involves fitting many base models from different algorithms on the same data set and using a metamodel to combine their results.[6] The common ground between bagging and boosting methods is that they incorporate ML algorithms that produce numerous decision trees,[7] such as random forest (RF)[8] and adaptive boosting/AdaBoost (AB),[9] respectively. The decision paths stemming from bagged or

[1]Department of Computer Science and Media Technology, Linnaeus University, Växjö, Sweden
[2]Department of Science and Technology, Linköping University, Norrköping, Sweden

Corresponding author:
Angelos Chatzimparmpas, Department of Computer Science and Media Technology, Linnaeus University, Vejdes Plats 7, Växjö, Kronoberg 351 95, Sweden.
Email: angelos.chatzimparmpas@lnu.se

boosted decision trees are the target of the visual analytics (VA) approach proposed in this paper.

The popularity of RF and AB is confirmed by their success in solving typical supervised classification problems, which constitute the majority of problems in the real world.[10,11] An in-depth study[12] that estimates the performances of 179 algorithms of various types[13] concludes that bagged decision trees of RF are better than other (types of) algorithms, such as deep learning approaches. Despite their remarkable predictive power, a crucial concern for algorithms that generate many decision trees is *interpretability*. Brieman,[14] for instance, indicates that RF models, while superb predictors, receive a low rating regarding their interpretability. As ML models can provide incorrect predictions,[15] ML experts have to check whether the model functions properly.[16] Also, domain experts in critical fields need to understand how a specific prediction has been reached in order to trust in ML.[17] For example, in medicine, a physician might not rely on a model without explanations of how and why it forms a prediction, since patient lives are at risk.[18–20] Or, in the financial domain, declined decisions for loan applicants require additional transparency with the precise justification of the outcome.[21] Although both algorithms follow the same concept of growing decision trees, their objectives differ: AB focuses on correcting misclassified training instances, while RF mainly reduces variance to achieve better generalizability. However, this fundamental goal of the former makes it susceptible to noisy cases,[22] while the latter arguably remains intact.[23] Thus, one research question that remains open is: (RQ1) Given the differences between rules obtained from bagged decision trees and those derived from boosted decision trees, how does their combination lead to potential benefits, regarding interpretability enhancement for decision making?

The interpretation of ML models typically happens either at a global or a local level.[24] Global approaches intend to explain the ML model as a whole,[25] assisting domain experts in exploring the general impact of each decision and gaining confidence in the produced predictions. On the other hand, local approaches aim to provide case-based reasoning,[26,27] allowing domain experts to review a prediction and trace its decision path in order to conclude if the decision rule, and consequently the prediction, is trustworthy.[28] Nevertheless, comparing numerous alternative decision paths without the support of an intelligent system is a time-consuming and resource-heavy procedure. For example, to scan the list of test instances rapidly and investigate specific instances of interest from multiple perspectives (e.g. outliers and borderline cases) can be crucial.[29] Thus, the whole process can benefit from a fast approach for automatic generation and semi-automatic exploration of reliable decisions with ML experts' involvement. It should also result in robust decisions, since domain experts are the most suitable for carefully examining and then manually picking sensible decisions according to their prior experience and understanding. One research question that arises from this (possibly underresearched) need for cooperation, starting from the selection of models to the extraction of insightful decisions, is: (RQ2) How can VA tools/systems support the collaboration between ML experts and domain experts?

In this paper, we present VisRuler, a VA tool that addresses the research questions described above by supporting the exploratory combination of decisions from two closely-related ML algorithms (i.e. RF and AB). VisRuler uses validation metrics for picking performant and diverse models and combines the decision paths from bagged and boosted trees to extract insightful and interpretable rules. Our contributions consist of the following:

- a visual analytic workflow for defining a methodical way of evaluating decisions (cf. Section System Overview and Use Case);
- a prototype VA tool, called VisRuler, that applies the suggested workflow with coordinated views that support the joint effort between ML experts and domain experts for extracting rules and making decisions, respectively;
- a use case and a usage scenario, applying real-world data, that validate the effectiveness of utilizing both bagged and boosted decision trees at the same time; and
- a user study that showed promising results.

The rest of this paper is organized as follows. In Section Related Work, we discuss relevant techniques for visualizing bagging and boosting decision trees, along with tree- and rule-based models and a bulk of relevant works of visual analytics systems for multi-model comparison. Section Random Forest versus Adaptive Boosting explains the core differences between bagging and boosting, and it further motivates why mixing decisions stemming from both algorithms could be beneficial for the users. In Section Target Groups, Design Goals, and Analytical Tasks, we describe the design goals and analytical tasks for comparing alternative decision rules, and we present the target groups (i.e. our stakeholders). Section System Overview and Use Case focuses on the functionalities of the tool and describes the first use case with the goal of identifying which countries have a higher happiness score index and why. Next, in Section Usage Scenario, we demonstrate the

applicability and usefulness of VisRuler with a usage scenario comprising another real-world data set focusing on loan applications, followed by Section User Study where we assess the effectiveness of VisRuler by reporting the results of a user study. Subsequently, in Section Limitations and Future Work, we discuss several limitations of our system and opportunities for future work. Finally, Section Conclusions concludes our paper.

Related work

According to a recent survey[30] that has extensively analyzed tree- and rule-based classification, several VA systems have been developed for this topic in the InfoVis and VA communities. However, most of these tools do not employ algorithms and measures (except for the accuracy metric) in order to compare model quality.[30] This section reviews prior work on the interpretation of bagged and boosted decision trees and the more general tools for tree- and rule-based visualization, comparing them with VisRuler to highlight our tool's novelty.

Interpretation of bagged decision trees

As in VisRuler, relevant works that utilize bagging methods use the RF algorithm to produce decision trees.[31–35] iForest[31] provides users with tree-related information and an overview of the involved decision paths for case-based reasoning, with the goal of revealing the model's working internals. However, iForest can be used only for binary classification, while VisRuler can be used with multi-class data sets, see Section System Overview and Use Case. Also, the feature flow, a node-link diagram, suffers from scalability issues (a challenge only partially overcome with aggregation). Our tool employs dimensionality reduction for clustering all decisions extracted by multiple models, thus enabling users to gain insights into the patterns inside a large quantities of rules. Therefore, VisRuler allows users to mine rules for both a particular class outcome and in connection to a specific case. ExMatrix[32] is another VA tool for RF interpretation that operates using a matrix-like visual representation, facilitating the analysis of a model and connecting rules to classification results. While the scalability is good, it does not cover the task of finding similarities between decisions from diverse models and algorithms. In conclusion, none of the above works have experimented with the fusion of bagged and boosted decision trees, and in particular, with visualizing both tree types in a joint *decisions space* to observe their dissimilarity, which can result in unique and undiscovered decisions. RfX[33] supports the comparison of several decision trees originating from a RF model with a dissimilarity projection and icicle plots, allowing electrical engineers to browse a single decision tree by using a node-link diagram. In contrast, VisRuler does not concentrate on a specific domain and gives attention to unique decision paths instead of trees with more scalable visual representations. Colorful trees[34] follows a botanical metaphor and demonstrates many core parameters essential to comprehend how a RF model operates. This method allows customized mappings of RF components to visual attributes, thus enabling users to determine the performance, analyze the behavior of individual trees, and understand how to tune the hyperparameters to improve performance or efficiency. However, this work is targeted toward hyperparameter tuning and does not focus on concurrently extracting and analyzing the decisions from each RF and AB model. Additionally, it is impossible to accomplish case-based reasoning with the proposed visual representation. Finally, Neto and Paulovich[35] describe the extraction and explanation of patterns in high-dimensional data sets from random decision trees, but model interpretation through the exploration of alternative decisions remains uncovered by this work (when compared to VisRuler).

Interpretation of boosted decision trees

Special attention has been given to boosted decision trees with VA tools for diagnosing the training process of boosting methods[36–38] and interpreting their decisions.[39] Closer to our work, GBMVis[39] aims to reveal the structure and properties of Gradient boosting,[40] enabling users to examine the importance of features and follow the data flow for different decisions. A node-link diagram may limit its scalability to monitor hundreds or thousands of decisions concurrently, as opposed to VisRuler. Furthermore, our novel parallel coordinates plot adaptation allows users to instantly combine rules and observe their differences to identify unique decisions. BOOSTVis[36] employs views such as a temporal confusion matrix visualization for verifying the performance changes of the model, a t-SNE[41] projection for inspecting the instances, and a node-link diagram for examining the rules. Through GBRTVis,[37] users can explore Gradient boosting[40] with a node-link diagram for the rules, the instances distribution shown in a treemap, and continuously monitoring the loss function. *VIS*TB[38] contains a redesigned temporal confusion matrix to track the per-instance prediction during the training process. It also enables the comparison of the impact of individual features over iterations. These VA systems focus on the online training of boosting methods and aim to

assist in feature selection and hyperparameter tuning. While these problems are (partially) tackled by our tool, we concentrate on interpreting the decisions from bagged and boosted decision trees and comparing them across models.

Tree- and rule-based model visualization

Existing work on single decision tree visualization has experimented with different visualization techniques, such as node-link diagrams,[42–51] treemaps,[52,53] icicle plots,[54,55] star coordinates,[56,57] and 2D scatter plot matrices.[58] These techniques do not generalize well when exploring multiple decision trees, which is VisRuler's primary design goal. Visualizing the surrogate models to approximate the behaviors of the original models, either globally or locally, is another branch of related works.[59–66] Rule-based visualizations have also been deployed for the interpretation of complex neural networks.[67–70] Nevertheless, these models differ due to the lack of inherent decisions that could be extracted directly from the bagged and boosted decision trees. The core mechanism of bagging and boosting methods is the generation of decisions based on the training data, which then experts can interpret.

Finally, multiple static visualizations and a few interactive VA tools have been developed for specific domains of research, such as medicine,[71–75] biology,[76,77] security,[78] and social sciences.[79] However, VisRuler is a model-agnostic solution that could be modified to work with various domains, depending on the given data set and the domain expert.

Visual analytics for multi-model comparison

Several VA systems exist that enable the comparison of ML models in classification problems, especially with the evaluation of predictive performance and the importance of features.[80,81] EnsembleLens[82] is a VA system working with multiple models trained into different feature subsets. The end goal is to visualize the correlation between ensemble models, algorithms, hyperparameters, and features that achieve the highest score for anomalous cases. On the contrary, VisRuler is not limited to anomaly detection problems, instead focusing on the interpretability of insightful rules extracted from two ensemble algorithms that produce tree-based decisions. Schneider et al.[83] explored the impact of comparing side-by-side the data and model spaces. They used both bagging and boosting ensembles and investigated these algorithms' influence on the data with the purpose of adding, deleting, or replacing models from the model space. We also support that this mixture of bagging and boosting models is beneficial because each algorithm can bring different information to the analysis of decision rules; for more details, please check Section Random Forest versus Adaptive Boosting. However, we abstract complex models into individual decisions that are accountable to and easily interpretable by users.[14]

EnsembleMatrix[84] and Manifold[85] are two VA tools specifically designed for model comparison. The former uses a confusion matrix representation for contrasting models. The latter produces and compares pairs of models across all data classes. We adopt a similar approach as with those tools, but instead of deciding which model was "optimal," we export all decisions that work well with a specific test instance to be examined by domain experts. Squares[86] is a visualization approach that showcases the per-class performance of a multi-class data set. It helps users prioritize their efforts by calculating common validation metrics. Similarly, Boxer[87] is a system that faciliates the interactive exploration of subsets of training and testing data while comparing the performance of multiple models on those instances. Another VA system, called QUESTO,[88] enables domain experts to control objective functions to set specific constraints and to search for an "optimal" model for this purpose. Li et al.[73] developed a VA system that allows multi-model comparison based on clinical data predictions with a special attention to feature contribution. *PipelineProfiler*[89] is a visualization tool for the exploration of several AutoML[90] pipelines comprising multiple models. StackGenVis[80] is a VA system for composing powerful and diverse stacking ensembles[6] from a pool of base models. Finally, VisEvol[81] is a VA tool that supports the interactive intervention in the evolutionary hyperparameter optimization process while exploring five alternative ML algorithms. On the one hand, we also facilitate users to assess various models. On the other hand, we focus on making the process of model-learned decision rules more transparent and justifiable with the involvement of a domain expert at specific phases (see Figure 3 and Section System Overview and Use Case).

There is also a body of literature devoted to regression problems.[91–94] For instance, BEAMES[94] contains four ML algorithms and a model sampling method, with the help of which the system generates an ordered list of models that aids the user in selecting a performant model. Although feature ranking is also covered by this tool, ML experts working with VisRuler aim at gathering several manual decisions extracted from two directly comparable algorithms that the domain expert should interpret. Finally, classification requires different handling in terms of the available algorithms and

validation metrics when compared to regression tasks, making such solutions challenging to adapt to classification problems and vice versa.

Random Forest versus Adaptive Boosting

In this section, we present a quick overview of the general algorithmic steps of (and differences between) the RF and AB algorithms, in order to familiarize potential readers with the techniques involved and to highlight the importance of our tool. For more details, please refer to the extensive literature published on the two algorithms since their seminal works.[8,9]

Suppose there are N instances in our training data set. If we consider a binary classification problem, then we will have: $x_i \in \mathbb{R}^n$, $y_i \in \{-1, 1\}$, where n is the number of features, x is the set of instances with $i = 1, 2, \ldots, N$, and y is the target variable which is either -1 or 1, designating either class $C1$ or $C2$.

Random Forest

This algorithm works in two stages. The first stage involves integrating numerous decision trees to construct the RF, and the second stage involves making predictions for each tree created in the first stage, followed by a majority voting strategy.

The algorithm starts by looping for $m = 1$ to M, where M is the *number of trees/estimators* hyperparameter. Then, a bootstrap sample $\mathbb{Z}^*$ of size N is drawn from the training data by sampling N times with replacement. Afterward, a decision tree T_m is grown with the bootstrapped data by recursively repeating the following steps for each terminal node of the tree, until specific conditions have been met (e.g. the *maximum depth* of a tree $d_{\max}$ is achieved). The first step is to form individual decision trees: n features, their number is limited by the *maximum number of features* hyperparameter, and it is selected at random from the N instances (defined in the previous paragraph). Next, the best *split point* p_v among the n is picked. Finally, the node v is split into two child nodes.

The output is an ensemble of trees $\{T_m\}_1^M$ that can be used to make predictions at a new test instance k, as follows. Let $\hat{C}_m(k)$ be the class prediction of the m th decision tree; the output will be calculated as: majority_vote$\{\hat{C}_m(k)_1^M\}$. For a test instance, the *decision path* that predicts it is followed repeatedly for every decision tree, and the test instance gets assigned to the category that wins the majority votes.

For each v in a decision tree, the p_v refers to a feature and different criteria (e.g. the *minimum samples* in each leaf of a tree $s_{\min}$ is reached) tunable through the hyperparameter settings that are used to split this node to children nodes. The Gini impurity[95] measurement is utilized in our case to pick this p_v, formulated for each v as follows: $GI(v) = \sum_{c=1}^{C} P_{vc} \cdot (1 - P_{vc})$, Figure 1. where P_{vc} is the probability of a certain classification c (in our example $C1$ and $C2$). Given an input test instance, each decision tree will fall into a root-to-leaf decision path, which leads to its prediction. Every decision path (or simply *decision*) is constructed by defining a range of minimum and maximum values for each feature extracted from RF and AB grown decision trees. Therefore, for n features, we will have $n \cdot 2$ ranges, that is, dimensions for projecting the decisions visible in Figure 1(c).

Adaptive Boosting

This algorithm fits successively many decision stumps or even decision trees to the training data, using various weights. The latter occurs only if the *maximum depth* hyperparameter is set to > 1. It begins by forecasting the original data set and weighting each observation equally. If the first decision stump's prediction is inaccurate, the observation that was mistakenly anticipated is given more weight. Because it is an iterative process with a specific improvement step controlled by the *learning rate* hyperparameter, it will continue to add decision stumps until the *number of trees/estimators* reaches a limit.

For each instance, AB calculates the weights. Each training example is given a weight to determine its importance in the training data set. When the given weights are substantial, that set of training instances is more likely to influence the training set and vice versa. All training instances will start with the same weight, defined as: $w_i = 1/N$. The weighted samples always add up to 1, and each individual weight's value will be between 0 and 1.

The algorithm starts by looping for $m = 1$ to M, fitting a model $G_m(x)$ to the training data using weights w_i. After that, AB uses the method to calculate the real effect of this classifier in categorizing the training instances: $\alpha_m = \log((1 - \text{err}_m)/\text{err}_m)$, where err_m is the total number of misclassifications for that training set divided by the training set size. Therefore, α_m represents how influential this stump will be in the final categorization. When a decision stump performs well or has no misclassifications, the error rate is 0 and the α value is relatively significant and positive. The alpha value will be 0 if the stump only classifies half correctly and half erroneously. Finally, the α would have a big negative number if the stump consistently produced misclassified data.

After plugging in the actual values of total error for each stump, the sample weights are updated. The

following formula is used to accomplish that: $w_i = w_{i-1} \cdot e^{\alpha}$. In detail, the new sample weight will be equal to the old sample weight multiplied by Euler's number, raised to α. As explained in the previous paragraph, the two cases for α are: (a) positive when the predicted and the actual output agree or (b) negative when the predicted output does not agree with the actual class (i.e. the sample is misclassified). In the first case, the sample weight is decreased from what it was before, since the algorithm already performs well. In the second case, the sample weight gets increased so that the same misclassification does not repeat in the next stump. This procedure is followed so that the stumps are dependent on their predecessors. Lastly, the output for AB is the sum of all trees: $G(x) = \pm\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$.

Differences

There are two key distinctions between bagging and boosting, which apply for the RF and AB algorithms. First of all, boosting changes the distribution of the training set adaptively based on the performance of previously constructed classifiers, whereas bagging changes the distribution of the training set stochastically. Second, boosting employs a function of a classifier's performance as a weight for voting, whereas bagging uses equal weight voting. Boosting appears to minimize both bias and variance, unlike bagging, which is primarily for variance reduction. Following the training of a decision stump, the weights of misclassified training examples are raised while the weights of correctly classified training examples are dropped in order to train the following decision stump. Thus, efforts to overcome the bias of the most recently generated weak model by concentrating greater attention on the samples that it misclassified gain traction. Because of its capacity to minimize bias, boosting works particularly effectively with high-bias but low-variance weak models. The fundamental issue with boosting appears to be poor performance in the presence of noise.[22] This is to be expected, as noisy cases are more likely to be misclassified, and their weight will rise as a result.

To sum up, for data with little noise, boosting is regarded as being stronger than bagging; nevertheless, bagging algorithms are far more resilient than boosting in noisy environments.[23] To mitigate this problem, using both algorithms to derive decisions is a novel idea we adopt. In the future, we may expand our approach to experiment with any decision-based algorithm (as in prior works[96,97]), but in this paper, we chose to combine two already widely-used algorithms (i.e. RF and AB).

Target groups, design goals, and analytical tasks

In this section, we specify the main design goals (G1–G5) upon which VisRuler (or any other VA system) should base its development regarding scenarios of extracting decision rules from bagging and boosting ensemble algorithms. Afterward, we report the corresponding analytical tasks (T1–T5) that a user should be capable of performing while he/she obtains assistance and guidance from our proposed system. One important aspect of the design of VisRuler is the need for collaboration between different experts, which we also motivate here.

Target groups

In the InfoVis/VA communities, most of the research in explainable ML focuses on assisting *ML experts and developers* in understanding, debugging, refining, and comparing ML models.[98,99] In this paper, we expand our method to involve another target group: the various *domain experts* affected by the ML progress in fields such as finance, social care, and health care. With the growing adoption of ML in different areas, domain experts with little knowledge of ML algorithms might still want (or be required) to use them to assist in their decision-making. On the one hand, their trust in such decisions could be low due to a lack of in-depth knowledge on how models are learning from the training data. On the other hand, ML experts often have little prior knowledge about the data from particular domains. Thus, the primary goal of VisRuler is to combine the best of both worlds, that is, to offer a solution that combines the above-mentioned benefits from both expert groups. More details about the collaboration between the ML and domain experts can be found in Section System Overview and Use Case.

Design goals

Our design goals originate from the analysis of the related work in Section Related Work, especially the three design goals from Zhao et al.[31] (G2 and G3) and the four questions from Ming et al.[68] (G1, G4, and G5) targeted to experts in domains such as health care, finance, security, and policymakers. Our methodology is similar to Zhao et al.[31], who reviewed 35 papers from the ML, visualization, and human-computer interaction (HCI) communities to come up with their decision goals.

G1: Bring diverging models' performance to the spotlight. A VA system must first focus on what each model has learned in general; as such, the assessment

of every model's performance (to decide which should remain under use) is a prerequisite. Models that fail to perform according to user-defined standards should not be part of the following procedure.

G2: Disclose connections between features and predictions. VA systems should expose the features' impacts on predictions and allow humans to delete needless features based on that. During training, ML models learn different mappings between input features and resulting predictions based on the inherent mathematical functions used and the setting of hyperparameters. These mappings describe model behavior and help humans comprehend RF and AB models' properties.

G3: Discover the core hidden operating processes. The underlying functioning processes of RF and AB models must also be revealed to check whether the models are working correctly and to understand why a given prediction was made. Before making a decision, humans should be able to audit the decision process of a prediction and ensure that they agree. Many RF and AB model interpretation problems can be resolved by analyzing the individual decision paths.

G4: Reason about the relationship of certain features and knowledge acquired. Unlike G2, this one concentrates on deviations in ranking features for justifying the rule-based generated knowledge. VA systems can support humans with this alignment of features and knowledge extracted through the decision rules. Since domain experts may have knowledge and ideas based on years of research and study that current ML models do not make use of, the facilitation of communication between ML models and humans is a primary design goal.

G5: Guide to unconfident and contradictory predictions. This goal emerges when a model fails to perform well on particular test instances. In the production deployment of ML models, a rule that some models are confident about may not be generalizable. Although undesirable, it is relatively common for models to produce contradictory predictions for certain difficult-to-classify test instances. As a result, VA systems must advise users on which occasions each model failed to predict correctly.

Analytical tasks

To fulfill our design goals, we have determined five analytical tasks that should be supported by our VA system (described in Section System Overview and Use Case).

T1: Compare the performance and architecture of models for selecting the most effective ones. Users should be able to compare different models with the support from various measurements (G1), as follows: (1) illustrate the performance of each model based on multiple validation metrics; (2) distil the number of false-positive and false-negative instances from the confusion matrix for every model; and (3) derive the number of decision trees and decision paths per model, to facilitate high-level comparison.

T2: Investigate the contribution of global features according to different models and algorithms. Following the preceding task, users should be guided through the process of selecting important features (G2). Thus, it is crucial to enable the comparison between per-algorithm and per-model feature importances.

T3: Explore alternative clusters of decisions for global explanation and case-based reasoning. The summarization of the decisions in a single view that combines the decisions of different algorithms and models should be accomplished to allow users to assess the influence of each decision (G3). For example, some decisions could overfit, and others could contain a mixture of instances falling in different classes. This last phenomenon increases their impurity. Users should be able to interact and explore this *decisions space*.

T4: Compare decision rules based on local feature ranking. The global features described in T2 might not be similarly important for specific decisions, hence, local feature ranking via contrastive analysis[100] could shed some light upon this task (G4). Moreover, the interpretation of rules extracted from the space of solutions (see T3) could be achieved if users are capable of investigating the values of both training and testing instances.

T5: Identify the different types of failure cases and confrontation via manual decisions. Failure to converge to a certain result due to the disagreement of the ML models should be highlighted to users (G5). For instance, if there is no uniformity in the final decision or the majority voted for the wrong result, it could be that these instances are outliers, borderline cases, or simply misclassified; being able to explore such cases is essential.

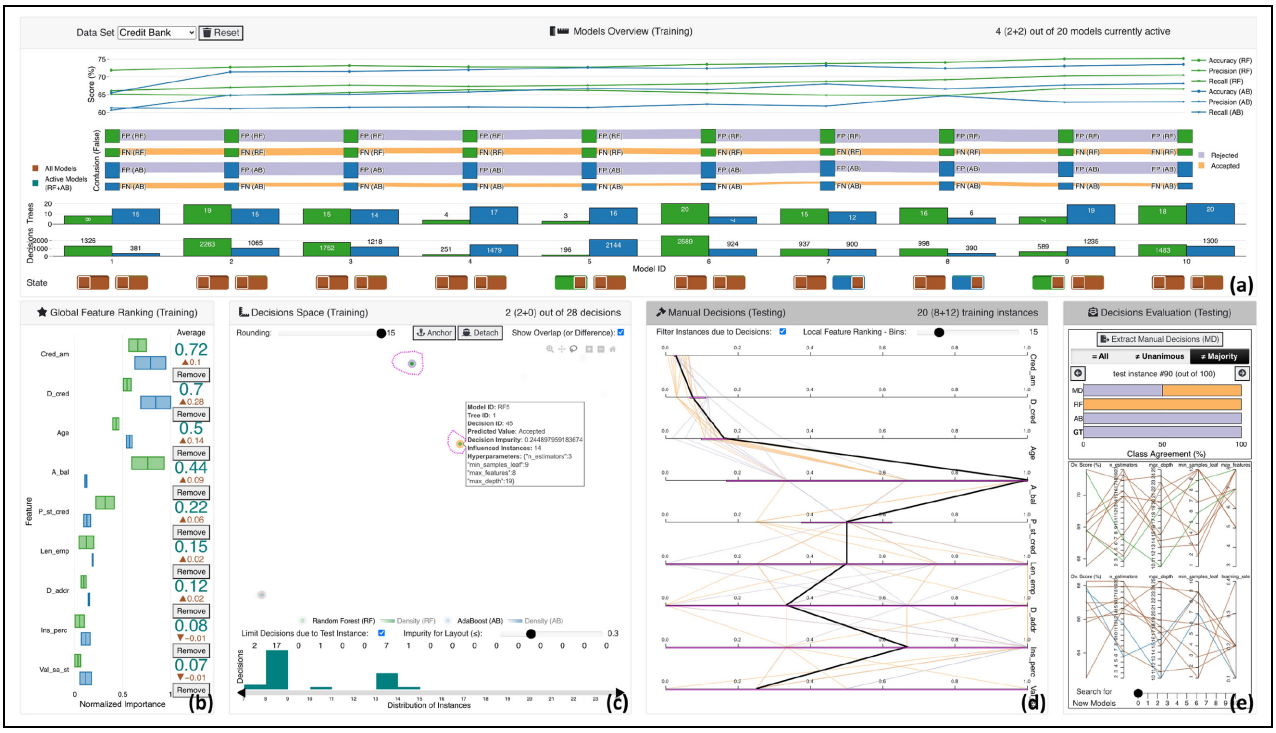

Figure 1. Extracting decision rules for manual evaluation with VisRuler: (a) panel with visual metaphors for selecting performant and diverse models, (b) box plot for feature selection according to per algorithmic importance, (c) visual embedding of computed decisions that training instances fall in due to their values, (d) vertical parallel coordinates plot that summarizes the rules with value ranges for each feature and highlights the current test instance, and (e) horizontal stacked bar chart for revealing the class agreement of each model against the manual decisions, together with the parallel coordinates plots for tuning hyperparameters and training new models.

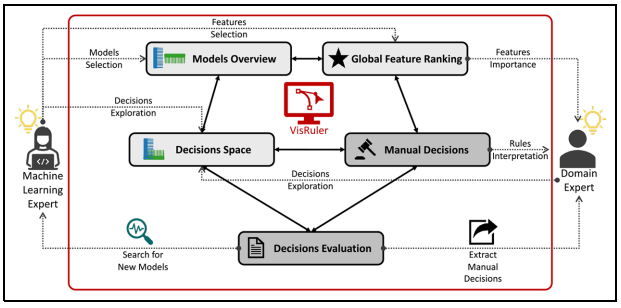

Figure 2. The VisRuler workflow allows ML experts to select performant and diverse models, choose important features, and retrain models with new hyperparameters. Domain experts can explore robust decisions, compare them to global standards, identify local decisions for a specific test instance, and extract them.

System overview and use case

Motivated by the above design goals and tasks, we have developed VisRuler. Our VA tool is written in Python and JavaScript. More technical details are available on GitHub.[101]

The tool consists of five main interactive visualization panels (Figure 1): (a) *models overview* (T1), (b) *global feature ranking* (T2), (c) *decisions space* (T3), (d) *manual decisions* (T4), and (e) *decisions evaluation* (T5). Our proposed workflow is a two-party system with the ML expert on the one side and the domain expert on the other (see Figure 2). The above-mentioned panels of our tool support the experts' collaborative effort, specifically: (i) the ML expert should select powerful and diverse models from the two separate algorithms based on their performance assessed by validation metrics (Figure 1(a)); (ii) during this phase, the ML expert should choose which features are important for the active models compared to all models (see Figure 1(b)); (iii) in the next exploration phase, both experts should examine which decisions explain the data set globally and decide upon impactful decisions for a specific test instance (cf. Figure 1(c)); (iv) in this same phase, the domain expert should interpret the manual decisions selected in order to gain insights about the models' decisions—either globally or locally—for a particular test instance (Figure 1(d)); and (v) in the final phase, the domain expert can evaluate the agreement and extract suitable manual decisions while the

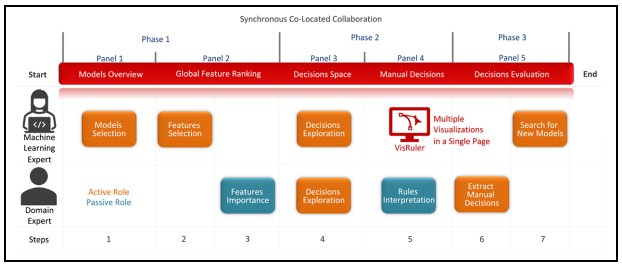

Figure 3. The VisRuler cooperation diagram illustrates how synchronous co-located collaboration typically happens between the ML expert and the domain expert. Three phases and five panels support their teamwork in a single-page tool, with the ML expert being more active (orange color) than the domain expert who receives and analyzes information (teal color). The seven linear steps taken by each user are also noted at the bottom.

ML expert should search for new models if the search did not reach a satisfactory level according to the domain expert (Figure 1(e)). Overall, this is an iterative process with a final goal to receive insightful decisions that should be interpretable for all counterparts. Details about the different views within the panels can be found below.

VisRuler incorporates a single workflow for the synchronous co-located collaboration between the ML expert and the domain expert, as depicted in Figure 3. It is a VA system that comprises multiple coordinate views arranged in a single webpage to manage the entire process without any distractions occuring due to the navigation to different tabs. The three phases and five visualization panels are already described in the previous paragraph. Both experts' usual interactions with the system can be aggregated into seven steps that are followed in the use case explained in this section and the usage scenario in Section Usage Scenario. Most of the time, the active role (cf. orange color in the diagram) is given to the ML expert who is responsible for selecting models (step 1), selecting features (step 2), exploring decisions (step 4), and finally, in step 7, search for new models (ideally based on the domain expert's feedback). On the other hand, the domain expert often has a more passive role (see teal color in the diagram) since he/she should focus on important features (step 3) and interpret the decision rules (step 5) based on the prior step. But in step 6, the domain expert becomes active because he/she has to decide which manual decisions should be extracted, for example, to be used as input to another tool, or as evidence for his/her diagnosis. The meeting point between both experts is step 4, the exploration of decisions, where the ML expert can control the multiple implemented filtering options (e.g. acceptable impurity level) while the domain expert pinpoints specific decision paths that could be interesting for a detailed manual investigation. Without their continuous communication while they operate the system and observe each other's actions, gaining deep insights would be very difficult.

The workflow of VisRuler is model-agnostic, as long as rules can be extracted from the ML algorithms. Currently, the implementation uses two popular EL methods: (1) RF and (2) AB (cf. green and blue colors in Figure 2, respectively). This choice was intentional since bagging methods work differently than boosting, as explained in Section Random Forest versus Adaptive Boosting. Furthermore, each data set is split in a stratified fashion (i.e. keeping the class balance in training/testing split) into 90% of training samples and 10% of testing samples. We also use cross-validation with 3-folds on the training set, and we scan the hyperparameter space for 10 iterations using Random search[102] in each algorithm separately. The common hyperparameters for both ML algorithms we experimented with (and their intervals) are: number of trees/estimators (2–20), maximum depth of a tree (10–25), and minimum samples in each leaf of a tree (1–10). An extra hyperparameter of RF is the maximum number of features to consider when looking for the best split $((\sqrt{number\_of\_features})–(number\_of\_features - 1))$. AB has the learning rate (0.1–0.4). It is straightforward to adapt those values through the code.

In the following subsections, we explain VisRuler by describing a use case with the *World Happiness Report 2019*[103] data set obtained from the Kaggle repository. This data set contains 156 countries (i.e. instances) ranked according to an index representing how happy the citizens of each country are. The six other variables that could be considered as features are: (1) *GDP per capita*, (2) *social support*, (3) *healthy life expectancy*, (4) *freedom to make life choices*, (5) *generosity*, and (6) *corruption perception*. Because this data set does not contain any categorical class labels, we follow the same approach as in Neto and Paulovich[35] to discretize the happiness score in three different bins. Hence, we are converting this regression problem into a multi-class classification problem.[104] Also in our case, the original variable Score becomes the target variable that our ML models should predict. In detail, the HS-Level-3 class contains 42 countries with happiness scores (HS) ranging from 6.13 to 7.76, the HS-Level-2 groups 79 countries from 4.49 to 6.13, and the HS-Level-1 class encloses 35 countries from 2.85 to 4.49.

Models overview

The exploration starts with an overview of how 10 RF and 10 AB models performed based on three validation metrics: accuracy, precision, and recall. The

models are initially sorted according to the overall score, which is the average sum of the three metrics. This choice guides users to focus mostly on the right-hand side of the line chart (as showcased in Section Use Case). Green is used for the RF algorithm, while blue is for AB. All visual representations share the same *x*-axis: the identification (ID) number of each model. The design decision to align views vertically enables us to avoid repetition and follows the best practices. The line chart in Figure 1(a) always presents the worst to best models from left to right. The *y*-axis denotes the score for each metric as a percentage, with distinct symbols used for the different metrics. The confusion plot inspired by Sankey diagrams in Figure 1(a) visually maps a confusion matrix of only false-positive and false-negative values for each model into nodes with different heights depending on the number of confused training instances. Then, they are divided into two groups reflecting the two algorithms. It also presents the confusion of all individual classes for the different instances when comparing two subsequent models, as illustrated in both Figures 1(a) and 4(a). The width of the band between two consecutive nodes indicates the increase or decrease in confusion from one model to the other sequentially, so the smaller the height of a line, the better a model's prediction compared to its predecessor or successor. The same effect applies to each node that absorbs the lines. With this plot, users can focus on the misclassified training instances that are more important for a given problem. For example, a medical doctor is typically cautious when dealing with false-negative instances since human lives may be at risk. Users can also check how many misclassified instances exist in each model and propagate from one model to another for each label class. Inspired by previous works,[36,38] we utilize a distinct visual metaphor for this plot to convey—as concisely as possible—the per class confusion for the several under examination ML models. The bar charts in Figure 1(a) showcase the two main architectural components of the bagged and boosted decisions trees, which are the *number of trees/estimators* hyperparameter and the number of decisions generated from these trees for every model mapped in the *y*-axes, respectively. These visualizations allow users to check the related hyperparameters of the individual models in a juxtaposed manner, since the number of decisions is related to the number of trees and the maximum allowed depth of each tree (i.e. *max_depth* hyperparameter). Finally, the state shown in Figure 1(a) designates which models are currently active (green or blue, respectively). In order to enable the comparison between the currently active model against all models, each icon for an active model contains a brown-colored slider thumb (Figure 1(a), legend on the left).

Global feature ranking

The box plots which aggregate per-algorithm importance (see Figure 1(b)) provide a holistic view of the performance of the models. Each pair of boxes is related to a unique feature, summarizing the active models' normalized importance per feature (from 0 to 1, i.e. worst to best). The box plots are sorted according to the average values of all active models, visible as a number in teal. The difference to all models being active is shown with arrows facing up for increase or down for decrease in per-feature importance. For both algorithms, we compute feature importance as the mean and standard deviation of accumulation of the impurity decrease within each tree. This is a measurement that can be calculated directly from RF[105] and AB[106] algorithms, cf. Section Random Forest versus Adaptive Boosting.

Decisions space

The projection-based view in Figure 1(c) is produced with the UMAP algorithm,[107] selected due to its popularity and the results of a recent quantitative survey[108] that found this algorithm the best overall among many others. In the visual embedding, decision paths are clustered based on their similarity according to their ranges for each feature, as described in Section Random Forest versus Adaptive Boosting and in the work of Zhao et al.[31] Therefore, it enables an algorithm-agnostic comparison between decisions stemming from both RF and AB models. The green color in the center of a point indicates that a decision is from RF, while blue is for AB. The outline color reflects the training instances' class based on a decision's prediction. The size maps the number of training instances that are classified by a specific decision, and the opacity encodes the impurity of each decision. Low impurity (with only a few training instances from other classes) makes the points more opaque. The positioning of the points can be used to observe if the RF and AB models produced similar rules, offering a comparison between algorithm decisions. The histogram in Figure 1(c) shows the number of decisions (*y*-axis) and the distribution of training instances in these paths (*x*-axis), and can also be used to filter the number of visible decisions in the projection-based view to avoid overfitting rules containing only a few instances (as shown in Figure 6(a)) or general rules that might not apply in problematic cases.

UMAP is initiated with variable *n_neighbors* and *min_dist* fixed to 0.1. To determine the optimal number of clusters to be visualized, DBSCAN[109] is used to compute an estimated number of core clusters from the derived decisions, which is then used to tune the *n_neighbors*, with a minimum of 2 and a maximum of 100 neighbors (the aim is to have the same magnitude in both). For the first experiment in Section Use Case, *n_neighbors* was automatically set to 20. On the other hand, in the usage scenario of Section Usage Scenario, DBSCAN estimated 477 clusters, which tuned the hyperparameter to the maximum value.

Multiple interactions are possible in this entire panel. The rounding slider (set to 15) allows users to round all decisions' range values to the desired decimal points. The comparison mode (active in Figure 1(c)) enables users to anchor groups of points and compare the selection against any other cluster. The two alternative choices are to present either the overlap or difference between the handpicked groups (shown in magenta color); the *Detach* button is for canceling this mode. Density views assist users in observing the distribution of RF against AB decisions in the projection, which is helpful if large amounts of decisions are visualized, as illustrated in Figure 5(a), projection. The *Limit Decisions due to Test Instance* checkbox alters the layout and changes global decisions' exploration to local for a particular case. Finally, a limit can be set for the acceptable impurity that is visible. If a decision is more impure than the currently chosen value, then it becomes almost transparent. As this view is tightly connected with the visualization of the following view, we proceed directly to Section Manual Decisions.

Manual decisions

The vertical PCP-like view in Figure 1(d) illustrates the range values per feature for each selected decision (comparison mode is active). The vertical polylines represent the training instances and are color-encoded based on the ground truth (GT) class. There are two options: either select to filter instances and show those that belong to the selected rules (see Figure 1(d)) or present all training instances at once (see Figure 5(c)–(e)). For example, in Figure 5(c), we see 12 identical rules that classify the training instances in the HS-Level-3 class (the red horizontal lines). The thick black polyline is the currently explorable test instance; users can compare it to the training instances of the models. All ranges for the features are normalized from 0.0 to 1.0. Scrolling is implemented when many decisions must be shown or the number of features is large. The order of the features is initially the global one, as described in Section Global Feature Ranking. When a group of points is selected using the lasso tool in the *decisions space* (DS) view, a contrastive analysis[100] is used to rank the features and highlight unique features that explain a cluster's separation from the rest. The computation works as follows: (1) break each feature into two disjoint distributions: the values inside the selected group vs. all the rest of the points; (2) discretize the two distributions of each feature into bins based on the *Local Feature Ranking - Bins* value set by the user (default is 10); (3) compute the cross-entropy[110] between the two distributions of each feature: higher values of cross-entropy suggest more unique features (i.e. the within-selection distribution is very different than the rest), while lower values suggest more common, shared features; and (4) rank the features based on step 3, with the more unique features near the top. Either with the overlap or difference setting selected as discussed in Section Decisions Space, the decision ranges bounding each feature are visualized in the vertical PCP with a magenta color in this comparison mode.

Decisions evaluation

The panel in Figure 1(e) contains interactive views that help users find outliers, borderline cases, and misclassified cases in the test set. The first main view supports extracting the *manual decisions* (MD) from the previous phase (see Section Manual Decisions). This output is stored in a JSON format where the boundaries of values per-feature are observable for every picked decision rule. This enables domain experts to reuse the hand-picked decision rules for supporting future actions, and also helps in concentrating on cases where the majority of the RF and AB models disagreed when compared to the GT, or for models that did not vote unanimously. It is also possible to go through all test instances one by one. The class agreement between RF and AB models, MD, and the GT is demonstrated via a horizontal stacked bar chart. The colors encode the different classes, and the length of each bar is the number of decisions for (1) MD, (2) RF models, (3) AB models, and (4) the GT (the latter always fills the entire bar). The second main view is used to train new models based on the *Ov. Score (%)* of each previously-trained model. The two separate standard PCPs present the active RF models in green and the active AB models in blue. The brown color is used for the inactive models.

Use case

In our use case, we observe that models with ID number 8 and above slightly outperform the rest; notably, recall in AB7 is much lower than AB8 and beyond (cf. Figure 4(a), line chart). While RF models perform consistently better than AB models, as shown in both

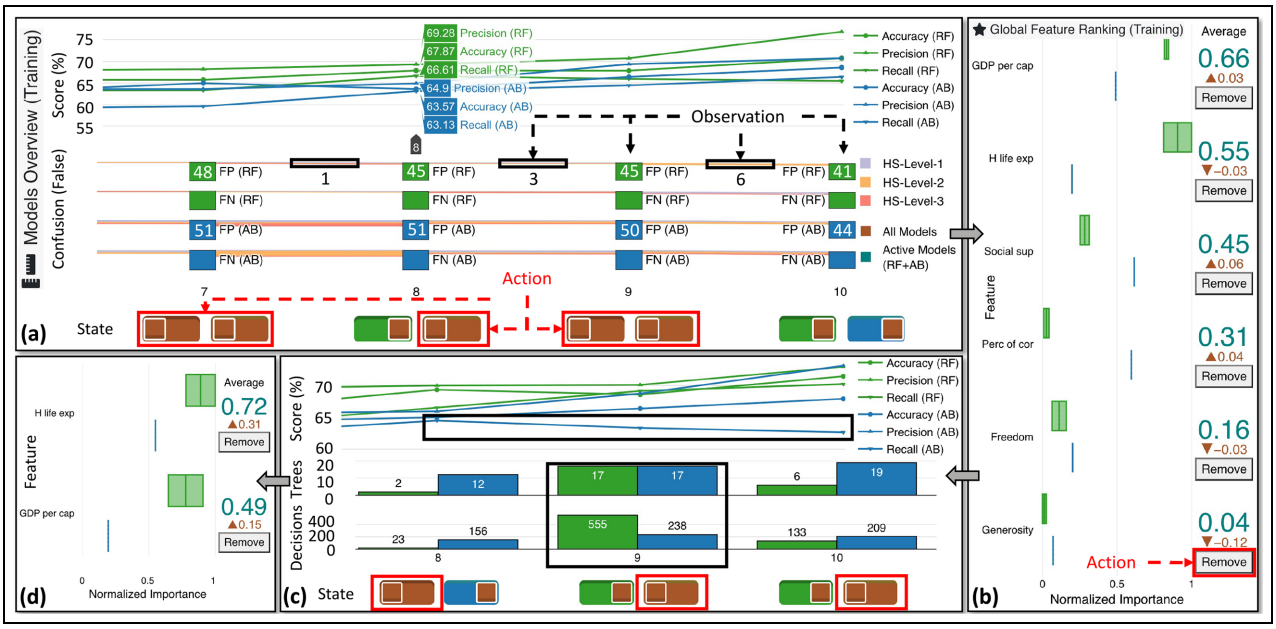

Figure 4. Exploration of ML models with VisRuler. View (a) presents the deactivation of all models except for RF8, RF10, and AB10, after consideration of their performance based on multiple metrics displayed in the visualizations. In (b), *Generosity* is the least important feature for the three active ML models and, particularly, its importance decreased while we deactivated most of the available ML models (see brown color). View (c) indicates that, after retraining with 5 of 6 original features, the new AB8 is better than the subsequent models due to the decline in recall; AB8, RF9, and RF10 remain the only active models after this step. In the box plot (d), the feature *H life exp* becomes more important by far than *GDP per cap*. Thus, these features swapped places compared to view (b).

the line chart and the confusion plot of Figure 4(a), there is an improvement in the score of AB10. Therefore, we decide to keep only this model. Furthermore, since RF8 is more reliable in training instances for the HS-Level-2 class due to false-positives being lower than the equivalent for RF9 and RF10 (Figure 4(a), confusion plot), we keep this model and RF10, that is, the top-performing model of the RF algorithm. In consequence, RF8, RF10, and AB10 are active models after selecting the corresponding states.

At this point, we want to investigate which features of the training set impacted the predictions more (see Figure 4). Interestingly, *GDP per cap*, *H life exp*, and *Social sup* are the top three features in the general ranking, as in Neto and Paulovich.[35] A surprising outcome is that, although two of the features mentioned above are still the most important for the selected RF models (all except *Social sup*), this is not true for the AB model. As seen in Figure 4(b), *Social sup*, *Perc of cor*, and *GDP per cap* are vital features for the AB algorithm in general. This pattern supports our hypothesis that different algorithms might take into account alternative features and should be combined to provide a holistic view. On the contrary, *Generosity* is unimportant for all models, specifically for the active models, since there is a −0.12 decrease in importance. Thus, we choose to remove this feature and retrain without it (cf. Figure 4(b)). For the RF algorithm (green), we pick the most performant models based on the overall score (Figure 4(c)), rightmost models). However, AB8 is better overall than the subsequent AB models due to the stable and high recall value (Figure 4(c), line chart). In a one-to-one comparison between RF9 and AB9 with the bar charts, we recognize that while they have the same *number of estimators* (i.e. 17 trees), the two models produce 555 and 238 decisions, respectively. In this case, bagged decision trees allow a higher maximum depth than the equivalent boosted decision trees. After the selection of the new models, the most important features collectively are *H life exp* with 0.72 and *GDP per cap* with 0.49, as illustrated in Figure 4(d); the opposite was valid in Figure 4(b). The new AB model considers the same features more important as the RF models. After this phase is over, AB8, RF9, and RF10 are the remaining three active models.

To investigate the global decisions based on the AB8 model we set the impurity to 0, disable limiting decisions based on the current test instance, hide the RF models, and reveal the density view of the active AB model (cf. Figure 5(a)). Most decisions are positioned in the right-hand side of the projection. Thus,

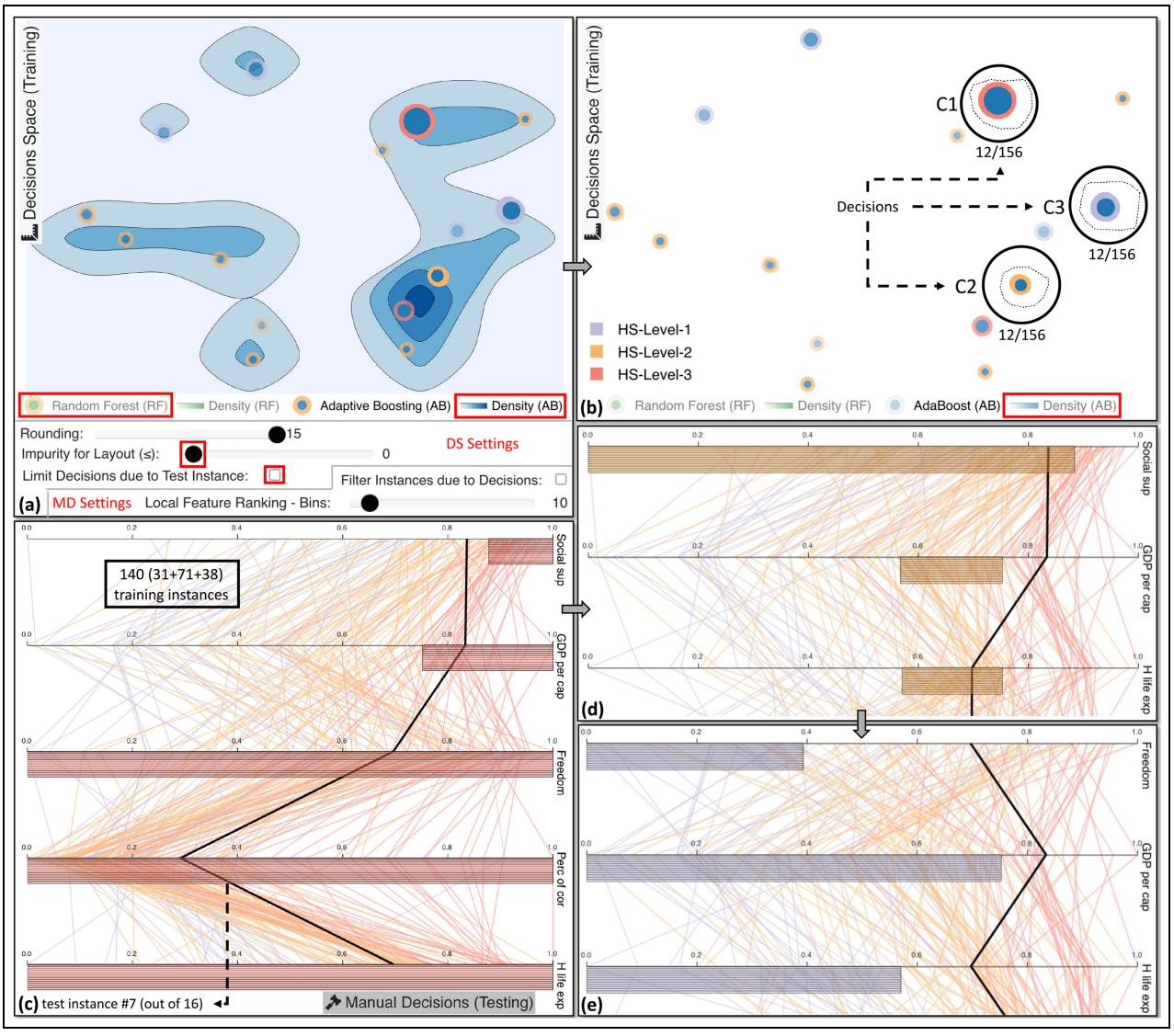

Figure 5. Examining several pure global decisions from the active AB model. In (a), we activate the density view in order to distinguish where most decisions are positioned. Note that this screenshot is composed of the *decisions space* (DS) view and the settings for the same view plus the settings for the *manual decisions* (MD) view. In (b), we select step-by-step 3 clusters of 12 identical decisions each. The decisions for C1 classify training instances only for HS-Level-3 class (as depicted in (c)). Similarly, C2 contains decisions for HS-Level-2 (visible in (d)), while C3 for the remaining class, as shown in (e). The 7th test instance, which is currently under investigation, cannot be classified by those prior decisions. However, it most likely belongs in the medium- or the high-level class.

we continue with the exploration of identical pure decisions from that region. After hiding back the Density (AB) as shown in Figure 5(b), we notice from the size of the decisions that if we analyze three core clusters (C1–C3) we can get a better understanding of global decisions. In Figure 5(c), all 140 training instances (31 + 71 + 38 spread across the classes) are observable together with the 7th test instance, which is currently under investigation because the majority of the RF and AB models disagree with the GT (not shown due to space limits, but a similar case is visible in Figure 1(e)). From Figure 5(c), we see that *Social sup* and *GDP per cap* should be very high for test instances to belong to this class. In contrast, for test instances to be in the HS-Level-2 class, they need to have a low-to-average *Social sup*, and average *GDP per cap* and *H life exp* (Figure 5(d)). Low values in the features (1) *Freedom*, (2) *GDP per cap*, and (3) *H life exp* are common for the low score in happiness countries (see Figure 5(e)), as also identified by Neto and Paulovich.[35] Regarding Saudi Arabia (the 7th test instance), it does not appear to belong to any of those

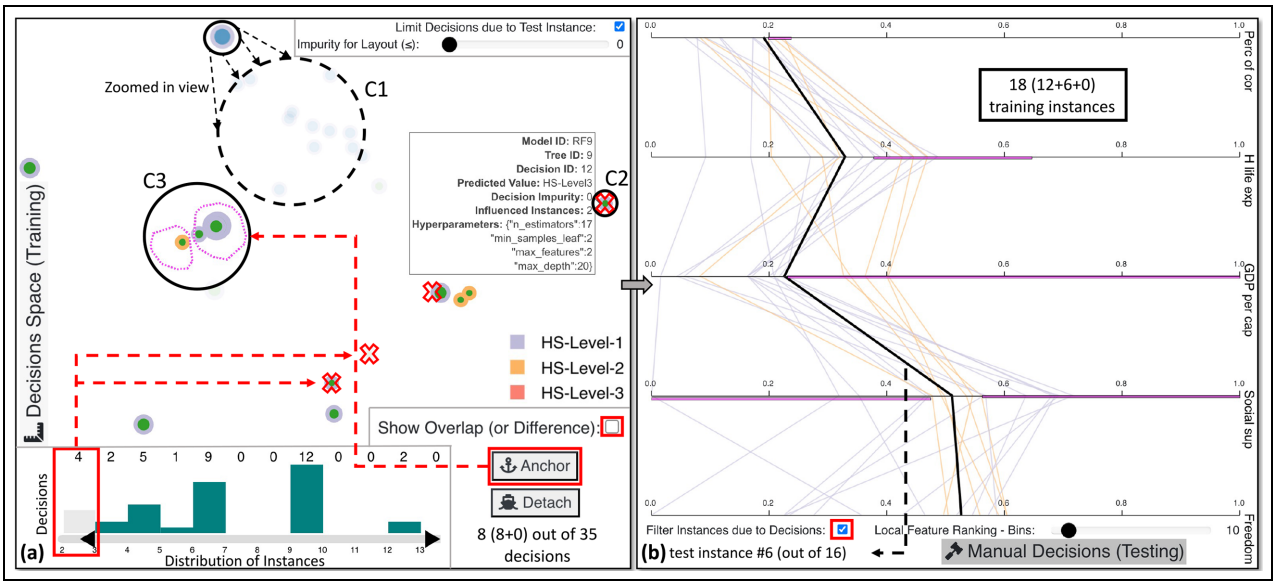

Figure 6. The local exploration of the 6th test instance with specific decisions from all the active models that apply only for this and similar test instances. (a) is a projection-based view that includes C1 with multiple impure decisions, visible only after zooming in. C2 is a decision with only two influenced training instances falling in this path, hence, we interact with the histogram below it to filter out overfitting cases with only two or three training instances. The comparison mode is enabled for C3, resulting in anchoring two subclusters of different (in terms of class) prediction decisions. In (b), the PCP highlights the differences between these previous subclusters for each feature in magenta color. This view is dynamic since the features are constantly being re-sorted based on contrastive analysis of the selected cluster against all points.

decisions, but it is far away from the values reported for the HS-Level-1 class. It has a very high *GDP per cap* to belong in the average class, but the *Social sup* is on the lower side. Despite that, *GDP per cap* is 1 out of the 2 most important features according to the analysis in Section Global Feature Ranking. Our conclusion matches the fact that it was ranked in 28th place out of the 156 countries, thus, belonging to the list of 42 countries classified as HS-Level-3.

Using the tool's mechanism to detect problematic test instances, a borderline case that stands out is the sixth test instance. in Figure 6(a), we first lower the impurity threshold to focus only on completely pure decisions. This example is for a specific case, thus, *Limit Decisions due to Test Instance* is checked. C1 seems to have a zero impurity, but when zooming in, we recognize that it contains several decisions with very high impurity values. Hence, we ignore this cluster, and we move to C2 with a decision influencing only two training instances. Since this number is undoubtedly low and could overfit these two instances, we increase the lower limit of visible decisions through the bar chart at the bottom. We exempt four decisions with two or three influenced training instances spread throughout the projection with this action. An interesting insight is observable in C3: eight decisions produced by the RF models are divided, with decisions suggesting that the test instance should be classified as either HS-Level-1 or HS-Level-2. We anchor one of the subclusters and select the other for comparison by viewing the difference in the value ranges of the five features (cf. Figure 6(b)). Twelve out of the 18 training instances suggest that Guinea (i.e. the 6th instance) is similar to low-happiness countries. If *Perc of cor*, *H life exp*, and/or *GDP per cap* were slightly higher, then the outcome would have been entirely different. According to the GT ranking Guinea is in 118th place, remarkably close to the 122nd test instance which is the first country classified as low happiness.

Checking the cases where the majority of the models disagree with the GT, we stop in the 15th test instance. Figure 7(a) shows the decisions applicable for this unusual case. We use the comparison mode to select a pure cluster on the left to juxtapose it with decisions classifying countries as HS-Level-3 on the right. Anchoring these clusters of points shows us the overlap of value ranges for the different features, as depicted in Figure 7(b). Twenty-eight out of the 30 training instances are similar to this test instance and belong to the HS-Level-2 class. The ranking of the features indicates that *Perc of cor* and *H life exp* are two unique features for the selected points, with low values for the former and average values for the latter, as in Neto and Paulovich.[35] Furthermore, for the first four

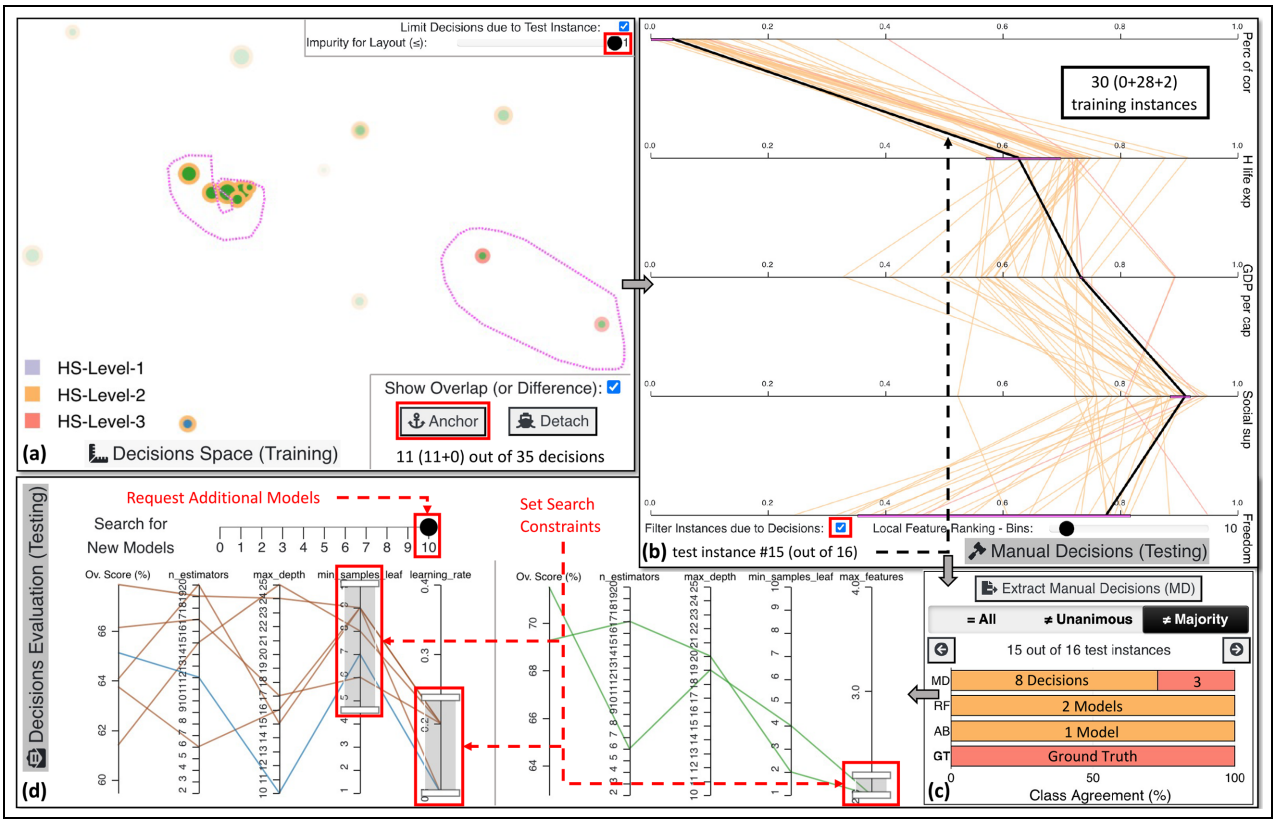

Figure 7. An outlier case exploration, the final prediction, and the training of another bunch of RF and AB models. (a) presents the anchoring of a cluster of 8 HS-Level-2 decisions to compare the overlapping rules against 3 HS-Level-3 decisions. In (b), after checking the common regions of agreement for the two clusters, we conclude that *Perc of cor* and *H life exp* are relatively low for the 15th test instance to belong in HS-Level-3 class. However, the other values for the remaining features are arguably rather high. In (c), we observe that all models voted for the average class while only the three selected manual decisions are supporting this case to be categorized as HS-Level-3 country. (d) showcases a potential search for new models by setting constraints in the hyperparameters according to the knowledge acquired from the initial training.

features, the overlap is narrow between the two selected clusters, indicating that this instance could be considered an outlier. Indeed, Figure 7(c) presents that 8 out of the 11 decisions consider this instance as HS-Level-2. All active models are wrongly predicting Trinidad and Tobago (i.e. the 15th test instance) as an average HS country. Interestingly, the three MD of the RF models classified this country as HS-Level-3.

From the analyses and the overall score of the RF and AB models, we observe that the most performant models for RF consider only two features when splitting the nodes (i.e. *max_features* hyperparameter). The PCPs in Figure 7(d) enable us to scan the internal regions of the hyperparameters' solution space for RF. As for AB, the *learning_rate* should be as low as possible for this specific data set, as seen in Figure 7(d). Also, by searching for models with high values for *min_samples_leaf*, AB models are created with complex decision trees compared to simple decision stumps, which seems to be an appropriate limitation of the hyperparameter space that could lead to better models. After all these constraints, we move the *Search for New Models* slider from 0 to 10 in Figure 7(d) to request 10 additional models for each algorithm with the hope of discovering more powerful ones. In summary, VisRuler supported the exploration of diverse decision rules extracted from two different ML algorithms and boosted the trustworthiness of the decision making process (RQ1).

Usage scenario

In this section, we describe a hypothetical usage scenario with a collaboration of a model developer (Amy, the ML expert) and a bank manager (Joe, the domain expert) who handles granting loans to customers. Joe wants to use VisRuler to improve the evaluation process of loan requests, so he asks Amy to use VisRuler

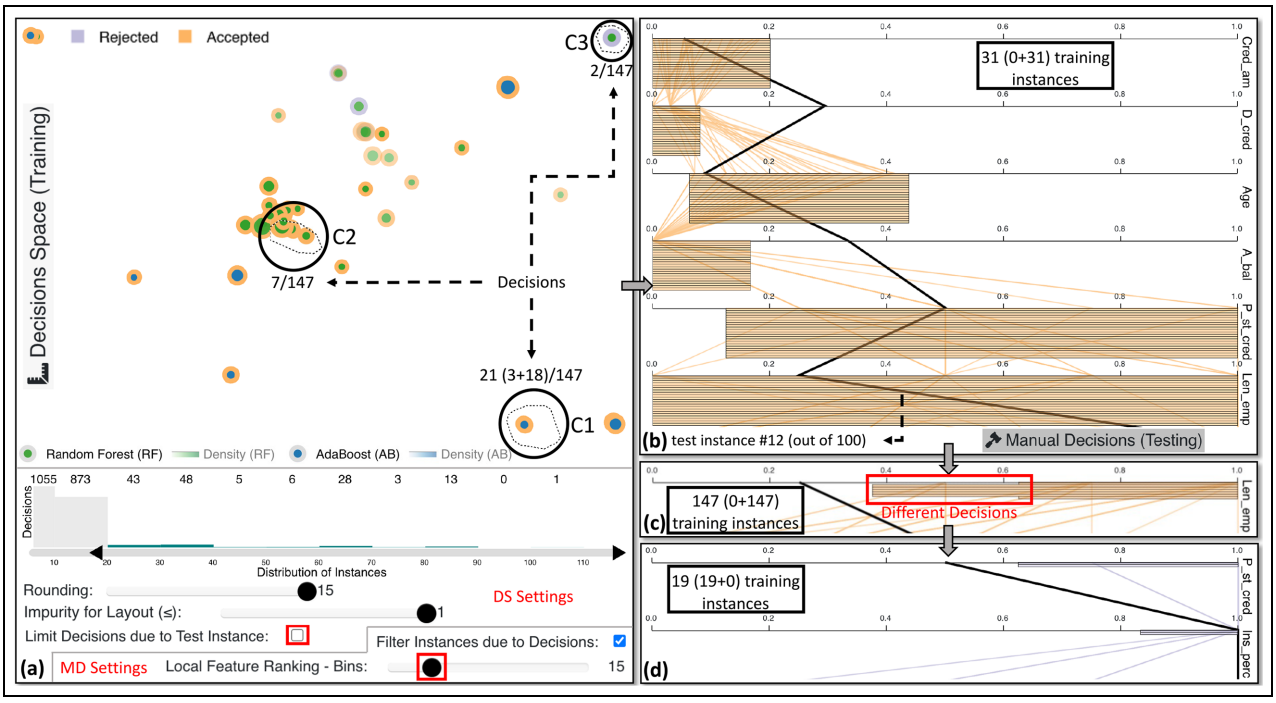

Figure 8. The exploration of clusters of decision paths from both ML algorithms. View (a) presents the selection of three clusters of global decisions that classify multiple training instances, thus, avoiding unimportant paths that might overfit. (b) provides an in-depth analysis of the decisions rules affected by C1 . In (c), *Len_emp* emerges as a unique feature that characterizes C2 with values from approximately 0.4–1.0. Finally in (d), high values in *P_st_cred* and *Ins_perc* turn over the prediction of the applicant to reject, visible via the exploration of C3.

to train ML models based on a data set collected over years of accepting or rejecting loans in the bank. The data set includes 1000 instances/customers and 9 features/customer information, with 300 rejected (purple) and 700 accepted (orange) applications. This data set is, in reality, a pre-processed version[31,32] of German Credit Data from the UCI ML repository.[13]

Exploration and selection of algorithms and models

Following the workflow in Section System Overview and Use Case, Amy loads the data set and checks the score of each model based on the three validation metrics (Figure 1(a)). For the AB algorithm, in blue, all models have a relatively low value for the recall metric, except for AB8. Also, AB7 performs very well for the Accepted class (orange), since the false-negative (FN) line reduces in height compared to all other models. Therefore, she decides to keep only AB7 and AB8. By looking at the confusion plot in Figure 4(a), Amy infers that RF5 is the model with low confusion regarding the Rejected class (purple). She is determined to use RF5 because it carries over only 104 different false-positive (FP) instances compared to RF4 with 114. The top RF models on the right-hand side also caught her attention, with RF9 and RF10 being the best options. She thinks that either of them could do the job, as they appear redundant due to similar confusion and values in both the confusion plot and the line chart (cf. Figure 1(a)). The bar charts below—which highlight the difference in the architectures of these RF models—help her to choose: with only 7 decision trees and 589 decision paths (compared to 18 and 1483), RF9 is simpler. She concludes that RF9's simplicity will make Joe's exploration of decisions more manageable later. Consequently, she deactivates RF10 and continues the feature contribution analysis with RF5, RF9, AB7, and AB8 models.

Examining the global contribution of features

After this new selection of models, Amy observes in Figure 1(b) that most features (except for the last two) are more important now than in the initial state. *Ins_perc* and *Val_sa_st* importances drop only by 0.01, implying these features are stable. She suggests Joe to keep all features for now and explore the differences through the decision rules later on. Another interesting insight is that *A_bal* is the most important feature

for the RF models, while the AB models prefer *D_cred* (see Figure 1(b)). This could indicate that mixing models' decisions from different algorithms is beneficial.

Explanations through global decision rules

Joe starts his exploration by examining the global decision rules that can help him make accurate decisions for specific cases in the future. He focuses on the 12th test instance, which is a customer application reviewed by a colleague, Silvia (cf. usage scenario by Neto and Paulovich[32]). First, he unchecks *limiting the decisions due to the test instance*, as illustrated in Figure 8(a). At this point, Amy identifies several decisions that classify only fewer than 20 customers; she thinks: "these are not so generic after all." Indeed, the larger the number of instances classified by one rule, the more generic and important it is (if the impurity is low). Consequently, they decide to increase the lower boundary of decisions, filtering out 1928 decisions (see Figure 8(a), bar chart). After the update, Joe focuses on the UMAP[108] projection. He observes multiple groups of points that could be worthy of further investigation. He selects a couple of samples from different areas, for example, C1 with 3 RF and 18 AB decisions. Another cluster with seven decisions is C2 that solely predicts accepted loan applications. On the contrary, C3 contains two pure decisions (due to high opacity) that produce rules which reject loans. Joe increases the *discretization of local feature ranking from 10 to 15 bins* to raise the sensitivity of difference between decision rule ranges, and he *filters the instances due to the decisions* to observe clearer trends. From Figure 8(b), Joe recognizes that C1 decisions are all identical, having the same ranges for every feature. Also, he understands that low *credited amount* (*Cred_am*) and short *duration of credit* (*D_cred*) are essential factors for accepting a loan application. *Account balance* is also vital because all loans are accepted when there is no account (*A_bal* being 0). Figure 8(c) reveals another intriguing pattern, that is, *the length of current employment* should be average to extremely high (from approximately 0.4 or 0.6 and above, shown in the red box) for applications to get accepted. In contrast, Figure 8(d) presents that if *payment status of previous credit* (*P_st_cred*) and *installment per cent* (*Ins_perc*) are relatively high, the applications were rejected. The 12th customer has an account without any balance, and the *D_cred* is relatively high, which flips the prediction toward rejection. Luckily, Silvia also provided an adequate justification to the customer.[32]

Extracting manual decisions through local investigations

At this point Joe knows and understands the main decision rules, but a new customer arrives. Focusing on the decisions for this case (i.e. 90th test instance), he sets impurity to less than 0.3 (cf. Figure 1(c), slider) to make impure decisions more transparent. Two fairly pure decisions from RF5 (visible due to hovering) and RF9 contradict each other. Joe uses the comparison mode, anchors 1 out of the 2 decisions, and selects the other with the lasso tool. The comparison in Figure 1(d) designates that eight similar customers' applications were rejected while 12 were accepted. The small overlap in *Cred_am*, *D_cred*, and *Age* suggest that this is a borderline case. *Cred_am* seems a bit arbitrary for the training data since only a small amount of applications in-between accepted applications were rejected, see Figure 4(d), feature on top. However, a clear insight is that if *D_cred* was lower, the application should have been accepted, while the opposite effect is true if the *duration of credit* increases. Unexpectedly, RF models vote for accepting this loan application while AB models reject (cf. Figure 1(e), top view). Besides that, the manual decisions are also in-between the two classes, which further enhances Joe's assumption that this is a borderline case. As AB models propose rejection and RF9 produces a decision for rejecting this application, he follows these recommendations. Nonetheless, Joe asks Amy to search and train new performant ML models (see next paragraph).

Tuning the search for bagged and boosted decision trees

Amy sees two possibilities of improvement for the RF in Figure 1(e), bottom view. One is to limit the *max_features* to 7 because it produces the two best models so far (visible by following the lines at the very top in *Ov. Score (%)*). The second strategy is to pick 3 and 4 for the same hyperparameter to explore an entirely new space of currently unexplored models since there is no existing line. Basically, she believes it is better to try both strategies in two separate runs. As for the AB, she reasons that selecting 0.1 and 0.2 for the *learning_rate* is a wise choice. Although it may take more time to retrain the AB models, they probably will be more powerful than with the other setting due to historical data. She performs the above actions, and finally, another cycle of exploration is unfolded for both experts. To summarize, our VA system not only helps users to reason about concrete cases, but also is capable of assisting ML experts and domain experts in enhancing their overall understanding due to their collaboration throughout the entire process (RQ2).

Table 1. User study: Shortened questions (Qs), goals (Gs), the ground truth, and results. There are five multiple choice questions, each with four possible answers. The goals and the ground truth (GT) can be found in Section Target Groups, Design Goals: goals and Section System Overview and Use Case, respectively. The results are computed as: number of correct answers/total number of participants.

Question (1, 2, 3, 4, & 5)	Goal	GT	Result
If you must remove a feature, which one will it be?	G2	4(b)	12/12
Which models present stable and high performance based on all validation metrics?	G1	4(c)	9/12
How many test instances are in conflict and require human intervention (that you can identify)?	G5	7(c)	9/12
Which features play a vital role for the classification of instances in the HS-Level-3 class?	G3 & G4	5(b)	12/12
Which feature's value is low and not contributing to the 15th instance's classification as HS-Level-3?	G3 & G4	7(b)	11/12

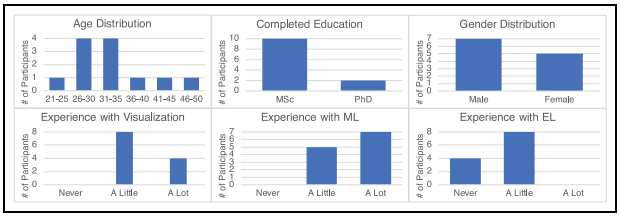

Figure 9. General information on the participants of our user study.

User study

We conducted a user study to evaluate our tool's effectiveness in supporting decision-making. As in prior works,[32,68] we created five questions (Qs) that cover VisRuler's different views, focusing on appraising the goals described in Section Target Groups, Design Goals, and Analytical Tasks with the use case outlined in Section System Overview and Use Case as the GT (see Table 1). Note that this user study was based on a slightly different arrangement of visualizations for the *models overview* panel, but all the rest of the visual representations and in-depth functionalities remained the same. In particular during the study, the line chart and the confusion plot were not aligned with the bar charts below, and the legends of this panel were positioned at the very top instead of beside the visualizations (as in Figure 1(a)).

Demographics

Figure 9 contains general information about the attendees of the user study. Seven male and five female volunteers aged 23–49 (mean: ≈33) participated in our study, all with at least an MSc degree (and two PhDs). None of them knew the data set used, and no colorblindness issues were reported. Four of the participants were highly knowledgeable in visualization and seven in ML, while the rest had limited knowledge regarding all aspects. Additionally, four of them had never worked with any EL method. All participants were researchers that have to frequently work with ML and tune ML models for various data tasks, such as image classification and natural language processing.

Methodology and instructions

Initially, participants watched an ≈18-min video tutorial about bagging and boosting concepts, VisRuler's goals, and how to work with our tool to analyze decision paths, using the Iris data set.[111] The participants experimented for 5 min with Iris, which concludes the required training for utilizing the capabilities of our tool. Then, they proceeded to use the data set described in Section System Overview and Use Case. They were asked to answer five questions (cf. Table 1 for a summary). The original document containing the questions and instructions as shown by the attendees is available in the supplemental material accompanying this paper. Finally, the participants were requested to provide qualitative feedback via the ICE-T questionnaire.[112]

Question-related results

The completion time it took the users to respond to each question and their answers to the multiple choice questions are shown in Figure 10. After the initial setting shown in Figure 4(a), all participants decided to exclude *Generosity* in Q1 (Answer: d, Q1), which happened in 2.03 min on average. For Q2, nine participants followed our GT (Answer: a, Q2), as described in Figure 4(c). The remaining attendees selected AB10 instead of AB8 (Answer: b, Q2). This action led to five test instances in conflict (Answer: b, Q3) compared to 3 (Answer: d, Q3) in our analysis (Figure 7(c) presents a single case). This result could be a strong indication that our approach is essential for making such decisions. To respond in Q2 and Q3, participants took 4.04 and 2.58 min on average, respectively. The most time-consuming question was Q4 with an average response time of 6.15 min (but with very accurate results (Answer: d, Q4), see Figure 5(b)). The average

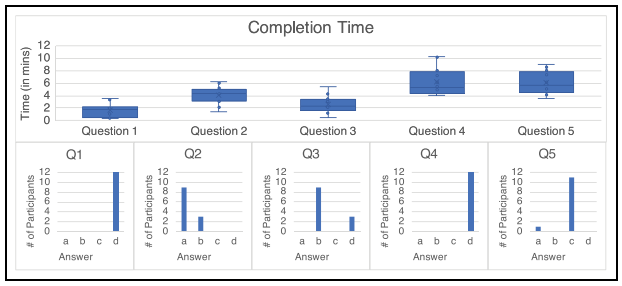

Figure 10. The question-related results of the user experiment. The top row presents the *completion time* for every question of the study separately, and the bottom row comprises the histograms of the participants' answers in all questions.

time taken for Q5 was 6.07 min, with all correct answers (Answer: c, Q5) except for one (Answer: a, Q5). The participant that responded incorrectly chose *Freedom* because it was at the bottom of the vertical PCP (cf. Figure 7(b)).

Qualitative results

In Table 2, the mean scores of the ICE-T components[113] for each participant are displayed along with the two-tailed 95% confidence interval (CIs) per component ($t^* = 2.201$, $N = 12$). Higher values in green indicate good results, as opposed to red. VisRuler has received a few 7.0 scores, and most are at least 6.0 and above (the lowest score is 4.67). Essence, Insight, and Time received large scores which means users found our tool competent in portraying decisions, guiding users to come up with fundamental questions, and performing these discoveries quickly. Confidence was lower, with a mean value of 5.87. However, this value still makes VisRuler a reliable and trustworthy VA tool according to Wall et al.[112]

Limitations and future work

In this section, we discuss several limitations of our VA system that could be regarded as improvement ideas for the future.

Generalization

Since VisRuler mainly focuses on the exploration of decision paths, our approach is applicable to any tree-based algorithm. If we follow the same methodology, RF could be changed to extremely randomized trees[113] (known as extra trees) or other bagging algorithms; while instead of AB, gradient boosting[114,115] or any boosting algorithm can be employed. An extension of our approach could be to include new supervised ML methods such as rule-based algorithms, or even association rule learning.[116] However, such algorithms consist of two parts: the antecedent (IF) and the consequent (THEN), which are complicated because there could be multiple if statements that bind many times the values using the same features. Therefore, a single decision path is non-trivial to be extracted, as in our case. One potential enhancement would be to support rule-based algorithms to encode these reoccuring information, which implies an order of consecutive events.

As shown in the paper, VisRuler works with both binary and multi-class classification problems. Since humans may have problems in perceiving more than 10 categorical colors[117] at the same time, VisRuler utilizes all of them as well as possible. Nonetheless, a limitation is the extensive (but unavoidable) use of color that might hinder our tool from operating with more than a few classes. Therefore, applying the one-versus-rest strategy is one possible idea to solve multi-class classification problems with several classes. This strategy translates to either designating one class as the

Table 2. Analyzed results from the ICE-T feedback.[113]

Components	Insight	Time	Essence	Confidence	Average
Participant 1	6.63	6.80	6.50	7.00	6.73
Participant 12	6.75	6.40	7.00	6.75	6.73
Participant 11	7.00	6.80	7.00	5.50	6.58
Participant 8	6.88	6.80	6.75	5.25	6.42
Participant 9	6.00	7.00	6.50	6.00	6.38
Participant 3	6.88	6.60	6.50	5.25	6.31
Participant 10	6.25	6.60	6.25	6.00	6.28
Participant 6	6.50	6.00	6.50	6.00	6.25
Participant 5	6.38	6.00	6.50	6.00	6.22
Participant 2	5.63	6.20	5.75	6.00	5.89
Participant 7	5.63	5.00	5.75	6.00	5.59
Participant 4	5.75	5.20	6.25	4.67	5.47
95% C.I.	**6.35 ± 0.32**	**6.28 ± 0.41**	**6.44 ± 0.25**	**5.87 ± 0.41**	**6.24 ± 0.26**

Legend: 1 2 3 4 5 6 7

positive class and all others as the negative class or choosing classes and gradually examining them.

At last, although VisRuler concentrates on the interpretation of rules and extraction of manual decisions driven by the experienced domain experts, it can also be used by ML experts to tune the hyperparameters of models and eventually debug RF and AB models. We acknowledge that VisRuler takes premature steps toward this direction, but this concept appears an exciting research opportunity for the future.

Scalability

Similar to many VA tools/systems,[118] a major challenge we considered when designing VisRuler is scalability. In the *models overview* panel, the number of trained models is limited to 20 RF and AB models in total (or 10 for each algorithm). However, in the backend users can set different hyperparameters and perform hyperparameter search with various automatic hyperparameter tuning approaches[119,120] or VA tools for this same goal.[81,121] VisRuler utilizes Random search for this purpose due to several benefits identified by Bergstra and Bengio.[102] The end result is to visualize several robust and diverse models in our tool, which can be deemed an adequate number of models. Also, the two PCPs for training new RF and AB models allow users to explore more models progressively, especially since the provided hyperparameters' ranges are also easily modifiable through the code.

In the *decisions space* view, circles may overlap when the number is large, as with any dimensionality reduction technique presented in a scatter plot. Although VisRuler can visualize thousands of decision paths (illustrated by the usage scenario in Section Usage Scenario), the cluttering of the low-dimensional embedding could be considered as an intrinsic difficulty. To address this issue, we first adopt a filtering approach to remove irrelevant or even overfitting decisions (e.g. see Figure 6(a)), and second we enable users to partially hide out impure decisions (cf. Figure 5(a)). Users may also utilize interactions like panning and zooming to focus on certain regions of circles. A potential future idea is to enhance the current scatter plot with approaches designed to reduce overlapping.[122,123]

In the *manual decisions* view while trying to get an overview, the vertical PCP may be challenging to interpret because it requires users to scroll through a list of decisions that expands by the number of features. Despite that, we have implemented multiple layout treatments (e.g. filtering out decisions visible in Figure 6(b)) and interaction possibilities (e.g. the comparison mode for juxtaposing groups of decisions shown in Figure 7(b)) for users to partially overcome these challenges. Furthermore, the most common scenario is to explore regions of decisions paths and focus on specific test instances which by default drastically limits the number of decision rules. In summary, the benefits of this tweaked visualization are many, since users can directly compare a test case with training instances for the various rules applicable and all features of a data set. The vertical PCP can help domain experts to externalize their domain knowledge because it serves as a root for a discussion between experts and the general public. One potential update is to try out alternative PCP designs that could boost the scalability of this view, such as the proposal from Wu et al.[124]

Efficiency

The performance of VisRuler could pose problems if numerous models are simultaneously active and produce too many decisions. Indeed, the excessive computational time required for exploring thousands of decisions paths along with the initial training of those ML algorithms can be a root cause for further troubles. Using distributed computation processes on performant cloud servers can be one solution for scaling VisRuler to enormous data sets. In the future, we believe that the improvement in high-performance hardware as well as progressive VA approaches[125,126] will also benefit VisRuler.

The use case, usage scenario, and user study were performed on a MacBook Pro 2019 with a 2.6 GHz (6-Core) Intel Core i7 CPU, an AMD Radeon Pro 5300M 4 GB GPU, 16 GB of DDR4 RAM at 2667 Mhz, running macOS Monterey, and with Chrome (version 99) as the browser. The system can perform interactively after the model training stage is over, which may take a few minutes for the data sets used in the use case of Section System Overview and Use Case and the usage scenario of Section Usage Scenario. However, we cache the results to speed-up drastically the future executions for the same data sets.

Complexity

Compared to several VA tools/systems,[99] VisRuler is no exception in terms of the high cognitive load that could overwhelm users. Despite that, the proposed workflow of VisRuler (see Figure 2 and read Section System Overview and Use Case) is mainly linear. Furthermore, the participants of our user study (cf. Section User Study) correctly performed most of the provided tasks after a specific training period, which is indicative of the gradual learning curve of our tool. However, in a future iteration of the tool, we plan to implement a hiding functionality for the *models overview* panel after the initial phase, involving mainly an

ML expert selecting powerful and diverse models, is over. Another incremental improvement could be to increase the size of the symbols for the three validation metrics present in the line chart of Figure 1.

Evaluation

While we already conducted a task-based user study with 12 participants that tested the applicability and effectiveness of VisRuler, additional review sessions with experts could help us to validate our tool further. However, as illustrated in Figure 3, our VA system is designed to be operated with a single workflow for two experts that most of the time are set apart and work independently. The prior knowledge and expertise of each group of experts is useful in specific steps of the collaboration schema, especially since they meet only in step 4, related to the *decisions space* exploration. A threat in this case is the overconfidence effect and overinterpretation of the models' capabilities by both domain-specific and ML experts, especially in noisy data scenarios. Despite that, we believe our first user study was an appropriate choice of method to understand preliminarily if VisRuler R is usable and effective. In the future, we could further evaluate the particular designs of this multi-component system with both ML and domain experts.

Conclusions

We presented VisRuler, a VA tool that allows users to explore diverse rules extracted from bagged and boosted decision trees to reach a consensus about a final decision for each individual case. The multiple coordinated views facilitate the selection of diverse and performant models, the characterization of per-feature contribution, the management of multiple decisions, the analysis of global decisions, and support case-based reasoning. A use case and a usage scenario with real-world data sets emphasize the necessity for combining similar, but yet, different algorithms and the importance of transparency in critical domains. We also validate the usability and efficacy of VisRuler via a user study. Finally, we describe a series of limitations of our VA system with an ultimate goal to extract future directions for our work.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was partially supported through the ELLIIT environment for strategic research in Sweden.

ORCID iDs

Angelos Chatzimparmpas https://orcid.org/0000-0002-9079-2376
Andreas Kerren https://orcid.org/0000-0002-0519-2537

Supplemental material

Supplemental material for this article is available online.

References

1. Zhou ZH. *Ensemble learning*. Boston, MA: Springer, 2009.
2. Sagi O and Rokach L. Ensemble learning: a survey. *WIREs Data Mining and Knowledge Discovery* 2018; 8(4): e1249.
3. Breiman L. Stacked regressions. *Mach Learn* 1996; 24(1): 49–64.
4. Freund Y and Schapire RE. Experiments with a new boosting algorithm. In: *Proceedings of the 13th International Conference on Machine Learning* (ed Saitta L), Bari, 3–6 July 1996, pp.148–156. Burlington, MA: Morgan Kaufmann Publishers Inc.
5. Schapire RE. The strength of weak learnability. *Mach Learn* 1990; 5(2): 197–227.
6. Wolpert DH. Stacked generalization. *Neural Netw* 1992; 5(2): 241–259.
7. Kingsford C and Salzberg SL. What are decision trees? *Nat Biotechnol* 2008; 26(9): 1011–1013.
8. Breiman L. Random forests. *Mach Learn* 2001; 45(1): 5–32.
9. Freund Y, Schapire R and Abe N. A short introduction to boosting. *J Japan Soc Artif Intell* 1999; 14(5): 771–780.
10. Opitz D and Maclin R. Popular ensemble methods: an empirical study. *J Artif Intell Res* 1999; 11(1): 169–198.
11. Wyner AJ, Olson M, Bleich J, et al. Explaining the success of adaboost and random forests as interpolating classifiers. *J Mach Learn Res* 2017; 18(1): 1558–1590.
12. Fernandez-Delgado M, Cernadas E, Barro S, et al. Do we need hundreds of classifiers to solve real world classification problems? *J Mach Learn Res* 2014; 15(1): 3133–3181.
13. Dua D and Graff C. UCI machine learning repository, http://archive.ics.uci.edu/ml (2017, accessed 12 November 2022).
14. Breiman L. Statistical modeling: the two cultures (with comments and a rejoinder by the author). *Stat Sci* 2001; 16(3): 199–231.
15. Caruana R, Lou Y, Gehrke J, et al. Intelligible models for healthcare: predicting pneumonia risk and hospital 30 day readmission. In: *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp.1721–1730. New York, NY: ACM.
16. Tam GK, Kothari V and Chen M. An analysis of machine- and human-analytics in classification. *IEEE Trans Vis Comput Graph* 2017; 23(1): 71–80.

17. Zhou J and Chen F. 2D transparency space—bring domain users and machine learning experts together. In: Zhou J and Chen F (eds.) *Human and machine learning*. Springer, 2018, pp.3–19.
18. Ribeiro MT, Singh S and Guestrin C. "Why should i trust you?": explaining the predictions of any classifier. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp.1135–1144. New York, NY: ACM.
19. Hastie T, Tibshirani R and Friedman J. *The elements of statistical learning*. 2nd ed (Springer Series in Statistics). New York, NY: Springer New York Inc, 2001.
20. Lakkaraju H, Bach SH and Leskovec J. Interpretable decision sets: a joint framework for description and prediction. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp.1675–1684. New York, NY: ACM.
21. Sachan S, Yang JB, Xu DL, et al. An explainable AI decision support-system to automate loan underwriting. *Expert Syst Appl* 2020; 144: 113100.
22. Bauer E and Kohavi R. An empirical comparison of voting classification algorithms: bagging, boosting, and variants. *Mach Learn* 1999; 36(1/2): 105–139.
23. Kotsiantis SB and Pintelas PE. Combining bagging and boosting (2007, accessed 12 November 2022).
24. Kopitar L, Cilar L, Kocbek P, et al. Local vs. global interpretability of machine learning models in type 2 diabetes mellitus screening. In: Marcos M, Juarez JM, Lenz R, et al. (eds.) *Artificial Intelligence in medicine: knowledge representation and transparent and explainable systems*. Cham: Springer International Publishing, 2019, pp.108–119.
25. Lipton ZC. The mythos of model interpretability: in machine learning, the concept of interpretability is both important and slippery. *Queue* 2018; 16(3): 31–57.
26. Du M, Liu N and Hu X. Techniques for interpretable machine learning. *Commun. ACM* 2019; 63(1): 68–77.
27. Carvalho DV, Pereira EM and Cardoso JS. Machine learning interpretability: a survey on methods and metrics. *Electronics* 2019; 8(8): 832.
28. Weller A. Transparency: motivations and challenges. In: Samek W, Montavon G, Vedaldi A, et al. (eds.) *Explainable AI: interpreting, explaining and visualizing deep learning*. Cham, Switzerland: Springer, 2019, pp.23–40.
29. Kim B, Rudin C and Shah J. The Bayesian case model: a generative approach for case-based reasoning and prototype classification. *Adv Neural Inform Process Syst* 2014; 2: 1952–1960.
30. Streeb D, Metz Y, Schlegel U, et al. Task-based visual interactive modeling: decision trees and rule-based classifiers. *IEEE Trans Vis Comput Graph* 2022; 28: 3307–3323.
31. Zhao X, Wu Y, Lee DL, et al. IForest: interpreting random forests via visual analytics. *IEEE Trans Vis Comput Graph* 2019; 25(1): 407–416.
32. Neto MP and Paulovich FV. Explainable matrix - visualization for global and local interpretability of random forest classification ensembles. *IEEE Trans Vis Comput Graph* 2021; 27(2): 1427–1437.
33. Eirich J, Münch M, Jäckle D, et al. RfX: a design study for the interactive exploration of a random forest to enhance testing procedures for electrical engines. *Comput Graph Forum* 2022; 41: 302–315.
34. Nsch RH, Wiesner P, Wendler S, et al. Colorful trees: visualizing random forests for analysis and interpretation. In: *2019 IEEE winter conference on applications of computer vision (WACV)*, Waikoloa, HI, USA, 7–11 January 2019, pp.294–302. New York, NY: IEEE.
35. Neto MP and Paulovich FV. Multivariate data explanation by jumping emerging patterns visualization. *IEEE Trans Vis Comput Graph*. Epub ahead of print 21 November 2022. DOI: 10.1109/TVCG.2022.3223529.
36. Liu S, Xiao J, Liu J, et al. Visual diagnosis of tree boosting methods. *IEEE Trans Vis Comput Graph* 2018; 24(1): 163–173.
37. Huang Y, Liu Y, Li C, et al. GBRTVis: online analysis of gradient boosting regression tree. *J Vis* 2019; 22(1): 125–140.
38. Wang J, Zhang W, Wang L, et al. Investigating the evolution of tree boosting models with visual analytics. In: *2021 IEEE 14th Pacific visualization symposium (PacificVis)*, Tianjin, China, 19–21 April 2021, pp.186–195. New York, NY: IEEE.
39. Xia Y, Cheng K, Cheng Z, et al. GBMVis: Visual analytics for interpreting gradient boosting machine. In: Luo Y (ed.) *Cooperative design, visualization, and engineering*. Cham: Springer International Publishing, 2021, pp. 63–72.
40. Friedman JH. Greedy function approximation: a gradient boosting machine. *Ann Stat* 2001; 29: 1189–1232.
41. van der Maaten L and Hinton G. Visualizing data using t-SNE. *J Mach Learn Res* 2008; 9: 2579–2605.
42. van den Elzen S and van Wijk JJ. Baobabview: interactive construction and analysis of decision trees. In: *2011 IEEE conference on visual analytics science and technology (VAST)*, Providence, RI, USA, 23–28 October 2011, pp.151–160. New York, NY: IEEE.
43. Nguyen T, Ho T and Shimodaira H. A visualization tool for interactive learning of large decision trees. In: *Proceedings 12th IEEE international conference on tools with artificial intelligence (ICTAI)*, Vancouver, BC, Canada, 15 November 2000, pp.28–35. New York, NY: IEEE.
44. Lee T, Johnson J and Cheng S. An interactive machine learning framework. *arXiv:1610.05463*, 2016.
45. Cavallo M and Demiralp C. Clustrophile 2: guided visual clustering analysis. *IEEE Trans Vis Comput Graph* 2019; 25(1): 267–276.
46. Barlow T and Neville P. Case study: visualization for decision tree analysis in data mining. In: *IEEE symposium on information visualization (INFOVIS)*, San Diego, CA, USA, 22–23 October 2001, pp.149–152. New York, NY: IEEE.
47. Phillips ND, Neth H, Woike JK, et al. FFTrees: a toolbox to create, visualize, and evaluate fast-and-frugal decision trees. *Judgm Decis Mak* 2017; 12(4): 344–368.
48. Bremm S, von Landesberger T, Heß M, et al. Interactive visual comparison of multiple trees. In: *2011 IEEE conference on visual analytics science and technology (VAST)*, Providence, RI, USA, 23–28 October 2011, pp.31–40. New York, NY: IEEE.

49. Song H, Curran EP and Sterritt R. Multiple foci visualisation of large hierarchies with FlexTree. *Inf Vis* 2004; 3(1): 19–35.
50. Munzner T, Guimbretière F, Tasiran S, et al. TreeJuxtaposer: scalable tree comparison using focus + context with guaranteed visibility. *ACM Trans Graph* 2003; 22(3): 453–462.
51. Behrisch M, Korkmaz F, Shao L, et al. Feedback-driven interactive exploration of large multidimensional data supported by visual classifier. In: *2014 IEEE conference on visual analytics science and technology (VAST)*, Paris, France, 25–31 October 2014, pp.43–52. New York, NY: IEEE.
52. Muhlbacher T, Linhardt L, Moller T, et al. TreePOD: sensitivity-aware selection of Pareto-optimal decision trees. *IEEE Trans Vis Comput Graph* 2018; 24(1): 174–183.
53. Guerra-Gómez J, Pack ML, Plaisant C, et al. Visualizing change over time using dynamic hierarchies: TreeVersity2 and the StemView. *IEEE Trans Vis Comput Graph* 2013; 19(12): 2566–2575.
54. Padua L, Schulze H, Matković K, et al. Interactive exploration of parameter space in data mining: comprehending the predictive quality of large decision tree collections. *Comput Graph* 2014; 41: 99–113.
55. Ankerst M, Ester M and Kriegel HP. Towards an effective cooperation of the user and the computer for classification. *In: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2000, pp.179–188. New York, NY: ACM.
56. Teoh ST and Ma KL. Starclass: interactive visual classification using star coordinates. In: *Proceedings of the 3rd SIAM international conference on data mining*, pp.178–185. Philadelphia, PA: SIAM.
57. Teoh ST and Ma KL. Paintingclass: interactive construction, visualization and exploration of decision trees. In: *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2003, pp.667–672. New York, NY: ACM.
58. Do TN. Towards simple, easy to understand, an interactive decision tree algorithm. *Technical Report 06–01, College of Information Technology, Cantho University, Cantho, Vietnam*, 2007.
59. Yuan J, Barr B, Overton K, et al. Visual exploration of machine learning model behavior with hierarchical surrogate rule sets. *ArXiv e-prints* 2022; 1–18.
60. Cao F and Brown ET. DRIL: descriptive rules by interactive learning. In: *2020 IEEE visualization conference (VIS)*, Salt Lake City, UT, USA, 25–30 October 2020, pp.256–260. New York, NY: IEEE.
61. Di Castro F and Bertini E. Surrogate decision tree visualization interpreting and visualizing black-box classification models with surrogate decision tree. *CEUR Workshop Proc* 2019; 2327.
62. Han J and Cercone N. Ruleviz: a model for visualizing knowledge discovery process. In: *Proceedings of the sixth ACM SIGKDD international conference on knowledge discovery and data mining*, 2000, pp.244–253. New York, NY: ACM.
63. Deng J and Brown ET. RISSAD: rule-based interactive semi-supervised anomaly detection. In: *Proceedings of EuroVis 2021 short papers*. The Eurographics Association.
64. Ware M, Frank E, Holmes G, et al. Interactive machine learning: letting users build classifiers. *Int J Hum Comput Stud* 2001; 55(3): 281–292.
65. Yuan J, Nov O and Bertini E. An exploration and validation of visual factors in understanding classification rule sets. In: *2021 IEEE visualization conference (VIS)*, New Orleans, LA, USA, 24–29 October 2021. New York, NY: IEEE.
66. Eisemann M, Albuquerque G and Magnor M. A nested hierarchy of localized scatterplots. *In: 2014 27th SIBGRAPI conference on graphics, patterns and images*, Rio de Janeiro, Brazil, 26–30 August 2014, pp.80–86. New York, NY: IEEE.
67. Marcílio-Jr WE, Eler DM, Paulovich FV, et al. ExplorerTree: a focus + context exploration approach for 2D embeddings. *Big Data Res* 2021; 25: 100239.
68. Ming Y, Qu H and Bertini E. RuleMatrix: visualizing and understanding classifiers with rules. *IEEE Trans Vis Comput Graph* 2019; 25(1): 342–352.
69. Jia S, Lin P, Li Z, et al. Visualizing surrogate decision trees of convolutional neural networks. *J Vis* 2020; 23(1): 141–156.
70. Thomas LV, Deng J and Brown ET. Facetrules: discovering and describing related groups. In: *2021 IEEE workshop on machine learning from user interactions (MLUI)*, New Orleans, LA, USA, 24–25 October 2021, pp.21–26. New York, NY: IEEE.
71. Hummelen R, Fernandes AD, Macklaim JM, et al. Deep sequencing of the vaginal microbiota of women with HIV. *PLoS One* 2010; 5(8): e12078–e12079.
72. Viros A, Fridlyand J, Bauer J, et al. Improving melanoma classification by integrating genetic and morphologic features. *PLoS Med* 2008; 5(6): e120.
73. Li Y, Fujiwara T, Choi YK, et al. A visual analytics system for multi-model comparison on clinical data predictions. *Vis Inform* 2020; 4(2): 122–131.
74. Niemann U, Völzke H, Kühn JP, et al. Learning and inspecting classification rules from longitudinal epidemiological data to identify predictive features on hepatic steatosis. *Expert Syst Appl* 2014; 41(11): 5405–5415.
75. Carlson JM, Brumme ZL, Rousseau CM, et al. Phylogenetic dependency networks: inferring patterns of CTL escape and codon covariation in HIV-1 Gag. *PLoS Comput Biol* 2008; 4(11): e1000225.
76. Abramov D, Otto J, Dubey M, et al. Rulevis: constructing patterns and rules for rule-based models. In: *2019 IEEE visualization conference (VIS)*, 2019, pp.191–195. New York, NY: IEEE.
77. Sydow JF, Lipsmeier F, Larraillet V, et al. Structure-based prediction of asparagine and aspartate degradation sites in antibody variable regions. *PLoS One* 2014; 9(6): e100736.
78. Aupetit M, Zhauniarovich Y, Vasiliadis G, et al. Visualization of actionable knowledge to mitigate DRDoS attacks. In: *2016 IEEE symposium on visualization for cyber security (VizSec)*, 2016, pp.1–8. New York, NY: IEEE.

79. Moussaïd M, Kämmer JE, Analytis PP, et al. Social influence and the collective dynamics of opinion formation. *PLoS One* 2013; 8(11): e78433.
80. Chatzimparmpas A, Martins RM, Kucher K, et al. StackGen-Vis: alignment of data, algorithms, and models for stacking ensemble learning using performance metrics. *IEEE Trans Vis Comput Graph* 2021; 27: 1547–1557.
81. Chatzimparmpas A, Martins RM, Kucher K, et al. VisEvol: visual analytics to support hyperparameter search through evolutionary optimization. *Comput Graph Forum* 2021; 40(3): 201–214.
82. Xu K, Xia M, Mu X, et al. EnsembleLens: ensemble-based visual exploration of anomaly detection algorithms with multidimensional data. *IEEE Trans Vis Comput Graph* 2019; 25(1): 109–119.
83. Schneider B, Jackle D, Stoffel F, et al. Integrating data and model space in ensemble learning by visual analytics. *IEEE Trans Big Data* 2021; 7: 483–496.
84. Talbot J, Lee B, Kapoor A, et al. Ensemblematrix: interactive visualization to support machine learning with multiple classifiers. In: *Proceedings of the SIGCHI conference on human factors in computing systems*, 2009, pp.1283–1292. New York, NY: ACM.
85. Zhang J, Wang Y, Molino P, et al. Manifold: a model-agnostic framework for interpretation and diagnosis of machine learning models. *IEEE Trans Vis Comput Graph* 2019; 25(1): 364–373.
86. Ren D, Amershi S, Lee B, et al. Squares: supporting interactive performance analysis for multiclass classifiers. *IEEE Trans Vis Comput Graph* 2017; 23(1): 61–70.
87. Gleicher M, Barve A, Yu X, et al. Boxer: interactive comparison of classifier results. *Comput Graph Forum* 2020; 39(3): 181–193.
88. Das S, Xu S, Gleicher M, et al. Questo: interactive construction of objective functions for classification tasks. *Comput Graph Forum* 2020; 39(3): 153–165.
89. Ono JP, Castelo S, Lopez R, et al. *PipelineProfiler*: a visual analytics tool for the exploration of AutoML pipelines. *IEEE Trans Vis Comput Graph* 2021; 27(2): 390–400.
90. AutoML. Google cloud AutoML, https://cloud.google.com/automl/ (accessed 12 November 2022).
91. Mühlbacher T and Piringer H. A partition-based framework for building and validating regression models. *IEEE Trans Vis Comput Graph* 2013; 19(12): 1962–1971.
92. Sehgal G, Rawat M, Gupta B, et al. Visual predictive analytics using iFuseML. In: *Proceedings of the EuroVis workshop on visual analytics*, 2018, pp.13–17. Eindhoven, The Netherlands: The Eurographics Association.
93. Zhao K, Ward MO, Rundensteiner EA, et al. LoVis: local pattern visualization for model refinement. *Comput Graph Forum* 2014; 33(3): 331–340.
94. Das S, Cashman D, Chang R, et al. BEAMES: interactive multi-model steering, selection, and inspection for regression tasks. *IEEE Comput Graph Appl* 2019; 39(5): 20–32.
95. Genuer R, Poggi JM and Tuleau-Malot C. Variable selection using random forests. *Pattern Recognit Lett* 2010; 31(14): 2225–2236.
96. Kotsiantis SB and Kanellopoulos D. Combining bagging, boosting and dagging for classification problems. In: Apolloni B, Howlett RJ and Jain L (eds) *Knowledge-based intelligent information and engineering systems*. Berlin, Heidelberg: Springer, 2007, pp.493–500.
97. Kotsiantis S. Combining bagging, boosting, rotation forest and random subspace methods. *Artif Intell Rev* 2011; 35(3): 223–240.
98. Chatzimparmpas A, Martins RM, Jusufi I, et al. A survey of surveys on the use of visualization for interpreting machine learning models. *Inf Vis* 2020; 19(3): 207–233.
99. Chatzimparmpas A, Martins RM, Jusufi I, et al. The state of the art in enhancing trust in machine learning models with the use of visualizations. *Comput Graph Forum* 2020; 39(3): 713–756.
100. Zou JY, Hsu DJ, Parkes DC, et al. Contrastive learning using spectral methods. *Adv Neural Inf Process Syst* 2013; 26: 2238–2246.
101. Angelos Chatzimparmpas. VisRuler code, https://github.com/angeloschatzimparmpas/VisRuler (2022, accessed 12 November 2022)
102. Bergstra J and Bengio Y. Random search for hyper-parameter optimization. *J Mach Learn Res* 2012; 13: 281–305.
103. John FH, Richard L and Jeffrey DS. *World happiness report 2019*. New York, NY: Sustainable Development Solutions Network, 2019.
104. Salman R and Kecman V. Regression as classification. In: *Proceedings of the 2012 IEEE Southeastcon*, Orlando, FL, USA, 15–18 March 2012, pp.1–6. New York, NY: IEEE.
105. Rogers J and Gunn S. Identifying feature relevance using a random forest. In: *Proceedings of the 2005 international conference on subspace, latent structure and feature selection*, 2005, pp.173–184. Berlin, Heidelberg: Springer.
106. Wang R. Adaboost for feature selection, classification and its relation with SVM, a review. *Phys Procedia* 2012; 25: 800–807.
107. McInnes L, Healy J and Melville J. UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv:1802.03426*, 2018.
108. Espadoto M, Martins RM, Kerren A, et al. Toward a quantitative survey of dimension reduction techniques. *IEEE Trans Vis Comput Graph* 2021; 27(3): 2153–2173.
109. Ester M, Kriegel HP, Sander J, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In: *Proceedings of the second international conference on knowledge discovery and data mining*, 1996, pp.226–231. Washington, DC: AAAI Press.
110. Mannor S, Peleg D and Rubinstein R. The cross entropy method for classification. In: *Proceedings of the 22nd international conference on machine learning*, 2005, pp.561–568. New York, NY: ACM.
111. Fisher RA. The use of multiple measurements in taxonomic problems. *Ann Eugen* 1936; 7(2): 179–188.
112. Wall E, Agnihotri M, Matzen L, et al. A heuristic approach to value-driven evaluation of visualizations. *IEEE Trans Vis Comput Graph* 2019; 25(1): 491–500.

113. Geurts P, Ernst D and Wehenkel L. Extremely randomized trees. *Mach Learn* 2006; 63(1): 3–42.
114. Ke G, Meng Q, Finley T, et al. LightGBM: a highly efficient gradient boosting decision tree. In: *Proceedings of the 31st international conference on neural information processing systems*, 2017, pp.3149–3157. Red Hook, NY: Curran Associates Inc.
115. Chen T and Guestrin C. XGBoost: a scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp.785–794. New York, NY: ACM.
116. Song C. Research of association rule algorithm based on data mining. In: *2016 IEEE international conference on big data analysis (ICBDA)*, Hangzhou, China, 12–14 March 2016, pp.1–4. New York, NY: IEEE.
117. Ware C. *Information visualization: perception for design*. Burlington, MA: Morgan Kaufmann, 2019.
118. Robertson G, Ebert D, Eick S, et al. Scale and complexity in visual analytics. *Inf Vis* 2009; 8(4): 247–253.
119. Claesen M and De Moor B. Hyperparameter search in machine learning. In: *Proceedings of the MIC*.
120. Claesen M, Simm J, Popovic D, et al. Easy hyperparameter search using optunity. *arXiv:1412-1114*, 2014.
121. Li T, Convertino G, Wang W, et al. Hypertuner: visual analytics for hyperparameter tuning by professionals. In: *Proceedings of the machine learning from user interaction for visualization and analytics workshop at IEEE VIS*, 2018.
122. Mayorga A and Gleicher M. Splatterplots: overcoming overdraw in scatter plots. *IEEE Trans Vis Comput Graph* 2013; 19(9): 1526–1538.
123. Hilasaca GM, Marcilio WE Jr and Eler DM. Overlap removal of dimensionality reduction scatterplot layouts. *arXiv:1903-06262*, 2019.
124. Wu HY, Niibe Y, Watanabe K, et al. Making many-to-many parallel coordinate plots scalable by asymmetric biclustering. In: *2017 IEEE Pacific visualization symposium (PacificVis)*, Seoul, South Korea, 18–21 April 2017, pp.305–309. New York, NY: IEEE.
125. Stolper CD, Perer A and Gotz D. Progressive visual analytics: user-driven visual exploration of in-progress analytics. *IEEE Trans Vis Comput Graph* 2014; 20(12): 1653–1662.
126. Turkay C, Pezzotti N, Binnig C, et al. Progressive data science: potential and challenges. *arXiv:1812.08032*, 2018.